Global Information Assurance Certification Paper

GIAC Certification

GCUX Practical Assignment
Version 1.8

Securing MySQL server on FreeBSD 4.5

Prepared by: Jason Lam

Introduction

GIAC Enterprises is an e-business which deals in the online sale of fortune cookies. Recently GIAC Enterprises decided to revise network perimeter security and implement a centralized database for storing business data (fortune cookies sayings). The database administrator decided to implement a MySQL server for the task. The database stores all GIAC Enterprises' fortune cookies saying and is a very critical part of business assets.
GIAC's database administrator understand the importance of properly securing a server against external and internal attack, so the internal Unix security administrator was consulted to build a secure server platform for the database to run on.

Description of the System

The system hosting the MySQL server will be running FreeBSD 4.5 release. This OS was chosen because all system administrators onsite are familiar with it, there is an internal standard procedure of secure installation and maintenance of the OS. This ensures no administrator will compromise the security of the system due to lack of knowledge to handle the platform.
With to the criticality of the database server, GIAC Enterprises decided to utilize a two stages change implementation system. All changes done by the database administrator will first be tested on the simulated platform, which has identical setup as the production server. When the changes are proven to be proper, they are finally committed to the production database server. Under this scheme, two identical platforms are needed to be setup exactly the same but for the purpose of this discussion, only one platform's setup procedure will be shown.

Brand:	Clone
Spec:	Intel P3-1Ghz, 512M RAM, 3Com Network card On-board RAID controller, mirrored 80G IDE harddisk Adaptec 29160, Sony SDX-400C tape drive
OS version:	FreeBSD 4.5-RELEASE
IP Filter version:	3.4.22
Tripwire version:	2.3.1 build 2
Stunnel version:	3.22
MySQL version:	3.23.49
SSH version:	OpenSSH 2.9 FreeBSD localisations 20020307

GIAC Enterprises' network denfense perimeter were designed and implemented by a GCFW (Refer to http://www.giac.org/practical/Jason_Lam_GCFW.pdf) which consists of multi-layered firewalls, VPNs and IDSs. The database server will be located in a network segment dedicated for internal servers (Refer to Appendix I). This network segment is protected by an internal IP Filter firewall running on FreeBSD. The internal firewall not only filter all traffic going to the internal server segment but also segregate the internal network into production, sales and development segment and ensure that only legitimate traffic will pass through the segments. There is also an external firewall

to protect all internal platforms, all incoming Internet connection can only arrive at the DMZ and do not transverse to internal segment of the network.

The MySQL server will act as an internal database for all GIAC Enterprises' fortune cookie sayings, all sayings will be stored and maintained on the server. The database only serves requests coming from production and development network segments, any request not from these hosts should be ignored and logged. Initial database size is about 5GB and is expected to grow to 8GB within one year. The performance requirement is minimal, less than 5 transactions per seconds are expected. Management responsibility will be shared amongst the database administrator and the Unix security administrator.

IP Filter will be installed on the server as a layered protection mechanism. The internal firewall should have filtered all of the unauthorized traffic, the IP Filter is there to ensure that any of the harmful traffic evaded through the internal firewall will be blocked and any of the un-intended listening services on the server hosts will not be exposed.

Stunnel will also be installed as an encryption handler for MySQL database transactions between the server to client, it will be used to create end to end SSL tunnels to protect against eavesdropping on the database traffic.

Tripwire will also be present on the server as a integrity checking mechanism, notifying the administrator at the first sight of unknown system files changes.

Risk Analysis

Fortune cookie sayings are the major business asset of GIAC Enterprises. All information stored in the database are critical and confidential. This database is expected to be running non-stop during normal business hours and occasionally outside of business hours (staff overtime). Any disruption of services or unavailability of information on the database during these hours can cause lost of business and damage in reputation for GIAC Enterprises.

GIAC Enterprises' management highly regards security in the organization, however, security and business needs are always a compromise. GIAC Enterprises cannot allow security to stop business operations nor can security be a major resource drawback to the organization. The management team is a firm believer in practical security and they understand that any networked system will bear certain risk. The management team is also aware that the more secure the system, the more resource it takes to build it. GIAC is a relatively new startup company that could not afford any major spending in security (third party audit and source code audit are out of the question). In this database server scenario, the management hopes to secure the database server with existing or freely available resources and minimize the installation and maintenance time required, yet providing sufficient level of security to conduct business operation without significant risk to the data on the database and if compromise is unavoidable, the database server and the data should be restorable and accessible within short period of time and lost of information in such event should be kept to within two days worth of data.

The primary security concerns and threat for the database server are:

Mis-configuration by the database administrator

MySQL server has its own authentication mechanism and it does not depend on the authentication of the OS system. Since the database administrator takes full responsibility in the MySQL authentication scheme, there are risks involved in the privileges granting process done by the database administrator, any accidental or unintended granting of extra privileges to users could lead to data leak, lost or even leak of world readable files on the server (if "file" access are given to the attacker).

Fortunately, the database administrator in GIAC Enterprises is very proficient in MySQL access privileges granting system and is aware of the consequences of "too much" access to the database. Under the cooperation of the security administrator and database administrator, a policy has been set for access granting procedure in order to minimize unnecessary privileges being granted to user.

Eavesdropping on database traffic

In a normal MySQL version 3 installation, all remote database queries and results are sent as non-encrypted binary format. Any attacker able to eavesdrop on the network traffic would be able to determine the content. Although GIAC uses switches instead of hubs in the network architecture which should eliminate most sniffing activities, but this is not a foolproof solution. A mis-configured VLAN port or attacker using overflow technique could sometimes still let attacker monitor traffic on the network. As mentioned earlier, all data stored in the database are confidential, leak of such information even to internal unauthorized personnel is a still a considerable threat.

To mitigate this threat, stunnel is used to create an end to end encrypted tunnel to each of the database clients (workstations). With the tunnel, the queries and results are encrypted with strong encryption to protect the data transfer on the LAN.

Internal compromise through vulnerable services

GIAC Enterprises are very careful when it comes to hiring, security screening and background checking are performed to every job candidate, however, the chance of internal compromise still exists. Internal attacker would most likely be staff that has access to the network, and they would attempt to exploit known vulnerability on the system to get elevated privileged access to information on database.

The internal firewall and the firewall software running on the database host generally stops most un-authorized access from the internal network but it does not stop exploit of vulnerabilities on accessible server ports. Usually, when an attacker plan an attack, they will do some reconnaissance to confirm the target's IP and OS for ensure the success of an attack. The firewall should log such internal reconnaissance attempt, leading to early detection of possible internal attack.

Attack from external source is not totally impossible but relatively unlikely. The attacker would have to penetrate the external firewall which block all incoming connections to the internal hosts (even to internal firewall) and also the internal firewall which is also set to block all incoming connections. In the event that these two firewalls fail to filter the attack, the database host itself should block the connection attempt with its own firewall software and TCP Wrapper.

Physical access to disable the machine

Physical security of the database host is very important, as data can be much easier to be destroy or stolen if the attacker has physical access to the computer. GIAC Enterprises is aware of this and recently built a secure room to house all servers in the organization.
Other physical factor such as power and environment control can also create a denial of service scenario for the machine. All UPS and HVAC should be properly maintained and secure to avoid any interruption of service.

Information lost due to hardware or software failure

All hardware and software can fail at some point of time. The data on databases hosts are critical and should not be jeopardized by a single hardware or software failure, therefore, backup and redundancy should be incorporated into the database server host's design and configuration. The components such as harddisk and power supply all have redundancy. In the event that a non-redundant component failed, a nightly backup is available to bring the database server up (using the test platform's hardware) with minimal data lost.

Step by step Guide

During the installation and configuration of the database server host, the text editor "vi" is used extensively. It is a complex and powerful text editor that comes default with most Unix platform. For detailed usage instruction, refer to the following web site,
http://www.msn.fullfeed.com/faq/vi.help
http://ecn.www.ecn.purdue.edu/ECN/Documents/VI/
http://www.cs.wustl.edu/~jxh/vi.html

FreeBSD also includes a simple text editor and can be executed by "ee" command. To use "ee" instead of vi, replace all the commands from "vi file" to "ee file".

1. Physical site preparation

1.1 Evaluate the location

GIAC Enterprises' building has a key card access control system. Anyone entering the building will either have a key card or check in with security and have a guest key card assigned. The database server will be housed in a restricted room in the office, only administrators' keycard will be able to gain access to it.

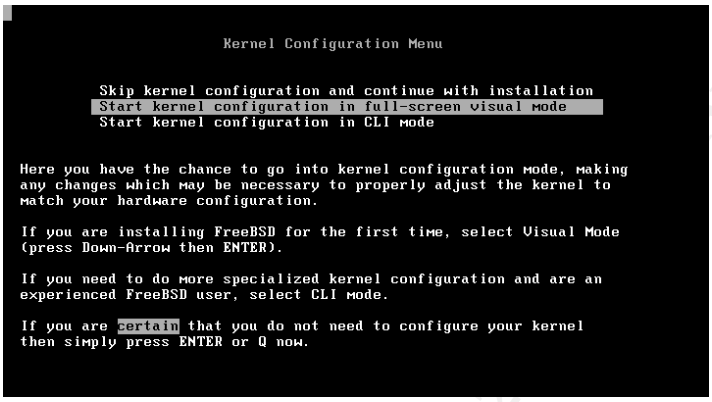
The rack that will house the database server has a glass door with locks on it, the door is always shut and locked to maintain security physical security to the server. Any direct access to the server is obviously very risky, all locks should be checked and physical security audit should be conducted before the installation of the server.

UPS and HVAC systems are regularly tested and checked to ensure flawless power supply and stable environment for the computer racks.

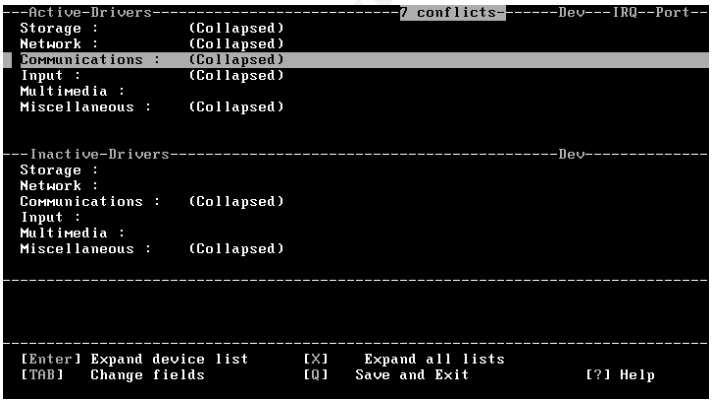
2. Base OS Installation

2.1 Installing OS from CD

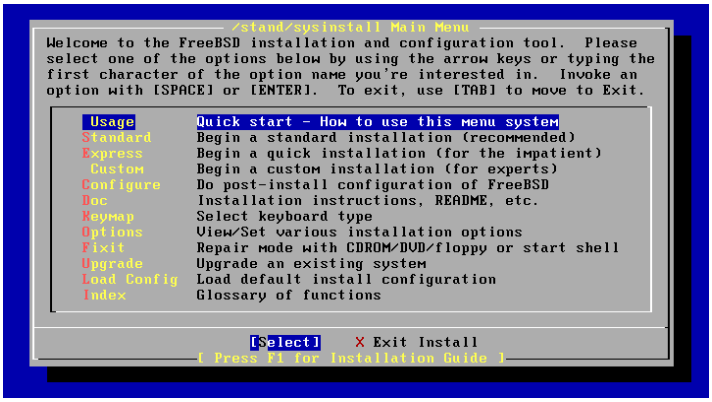
1. Ensure that the network cable is not plugged in, network access is not needed at this stage. Power on with FreeBSD 4.5 CD in the CD-ROM. Select the full-screen visual mode



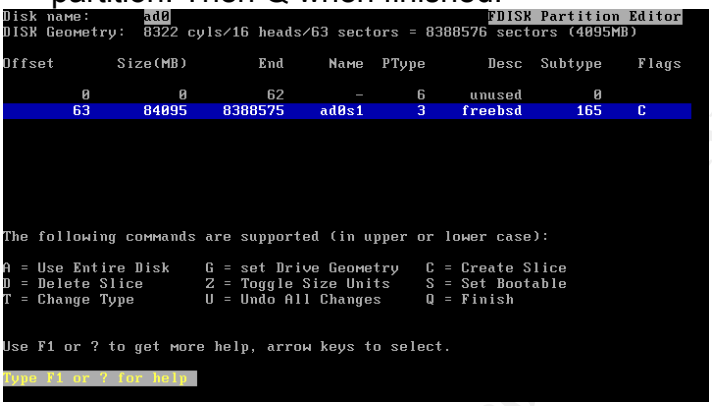
2. With the database server host hardware, most of the hardware driver are not needed (but they can still be safely left there). Press Q for save and exit after the unnecessary drivers are unloaded.



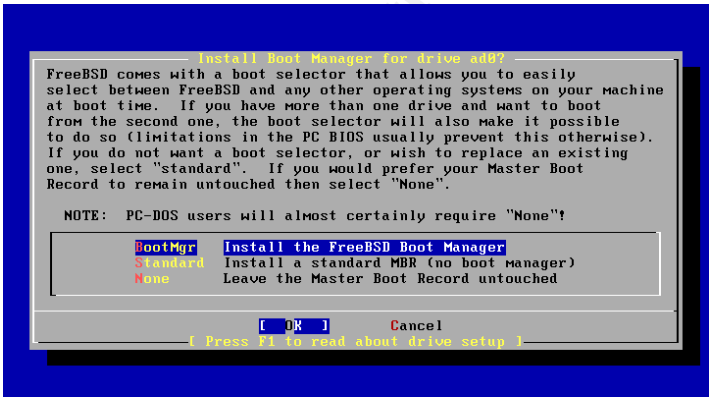
3. Select “Standard” from the main menu.



4. Select the size for the partition, for the database host, all off the drive space on the drive array will be used (press Z to toggle to size, it makes it easier to see the actual size). Press A to use the whole disk and then press S to make this a bootable partition. Then Q when finished.

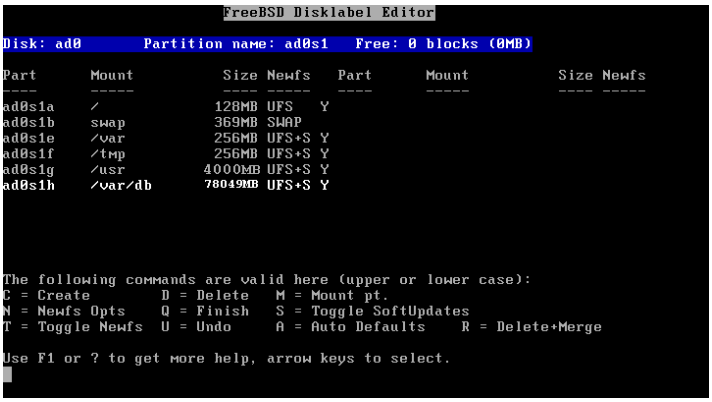


5. Choose Boot manager since FreeBSD is the only OS on the machine. Failure to do so may render the computer unbootable after the installation.

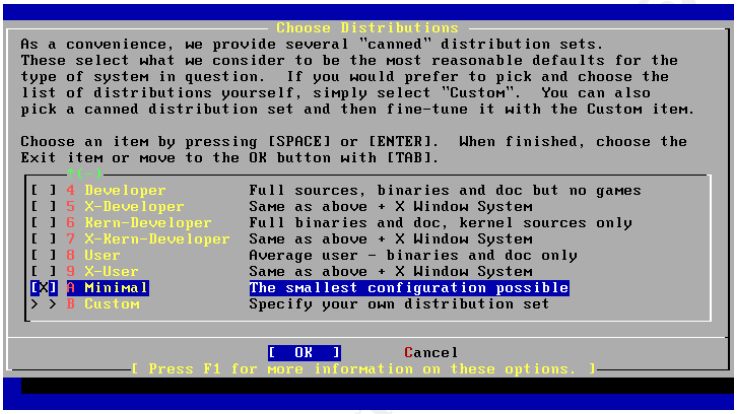


6. Use auto assign for the disk space. The program will automatically assign most of the space to /usr. Since the database will physically reside at /var/db, most of the disk space will be assigned to that file system. Move cursor to the /usr and delete the partition. Create /usr again with 4G which is more than sufficient for most purposes and assign all other space to /var/db which is the database. /usr will be used to install most of the binaries and programs that do not come with FreeBSD

and should have sufficient amount of storage. The /var/db file system will host the MySQL database, so most drive spaces are assigned to it. When this is done, press Q.



7. When prompted for distribution sets, select Minimal config (move cursor to minimal and press space) and then select “Custom”. Most of the other components are not useful for database host and installing them will only increase the size of the OS and increase the burden to maintain them. At the “Custom” menu, select “man”, this is manual for all binaries, very important component for all administrators.

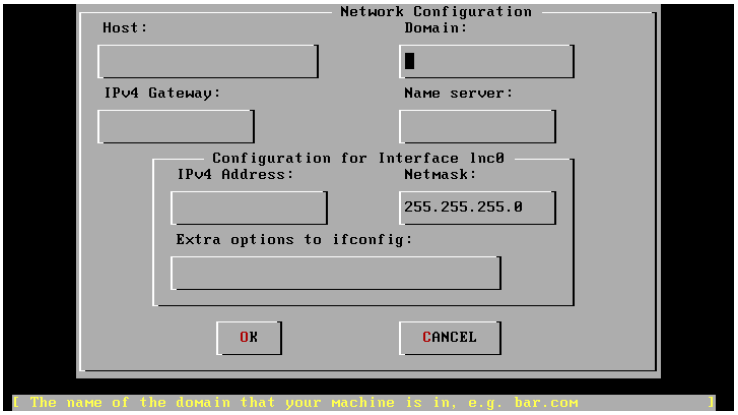


8. When asked for configuration for Ethernet adapter, answer YES and select the “xl0” network card (which is the 3Com network card).

9. When prompted for IPv6 configuration, answer NO. GIAC Enterprises’ network does not run IPv6.

10. When prompted for DHCP configuration, answer NO.

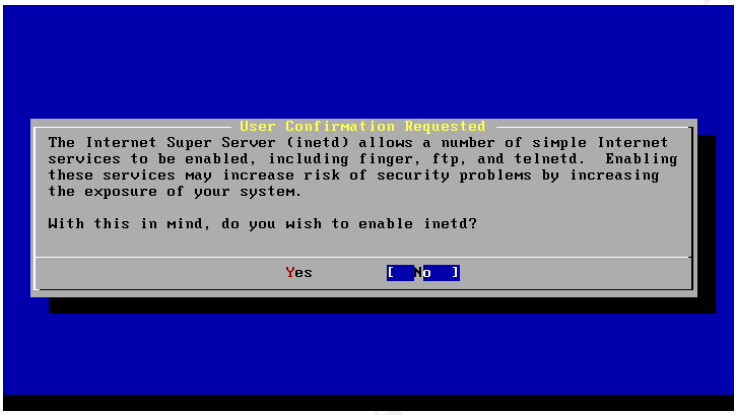
11. Then the IPv4 configuration screen will come up, this database host will have a temporary internal IP address for installation (it will be connected to a specially firewalled network segment for installation).



12. When prompted for whether to bring up the network interface now, answer NO. (The network cable is not even in place, there is no point of bringing up the interface).

13. When prompted for this host being a gateway, answer NO (There is no need for packet forwarding on this machine).

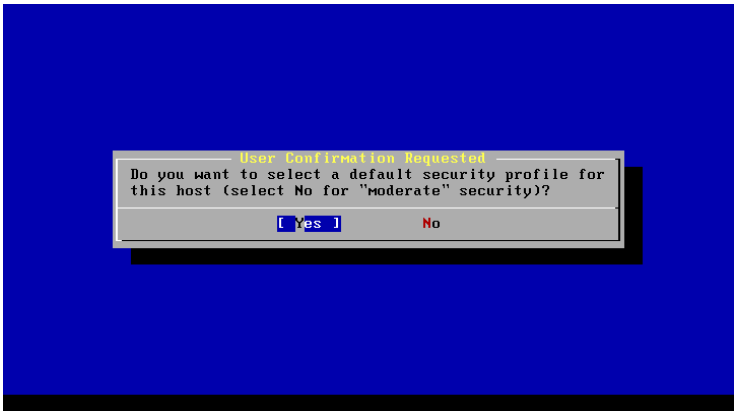
14. Then the prompt for configuring inetd and simple Internet services will come up, answer YES.



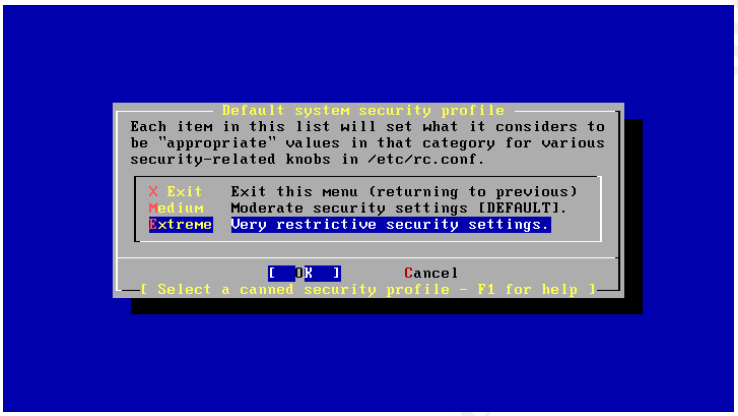
15. When asked whether to enable inetd, answer NO. There is no service on the database server host that will run with inetd.

16. When asked whether this host is an anonymous FTP server, NFS server and NFS client, answer NO. These are all very high risk server software and are not needed for the database server operation.

17. At prompt for choosing security profile, answer YES.



18. This will allow a choice of security profile which can make the OS much more secure. Choose Extreme in selection for security profile. The extreme security profile default to not start with sshd and sendmail (unlike in medium) and it also set the kernel securelevel to 2, which requires single user mode for a lot of system modification (will slow down and/or deter hacking attempts).

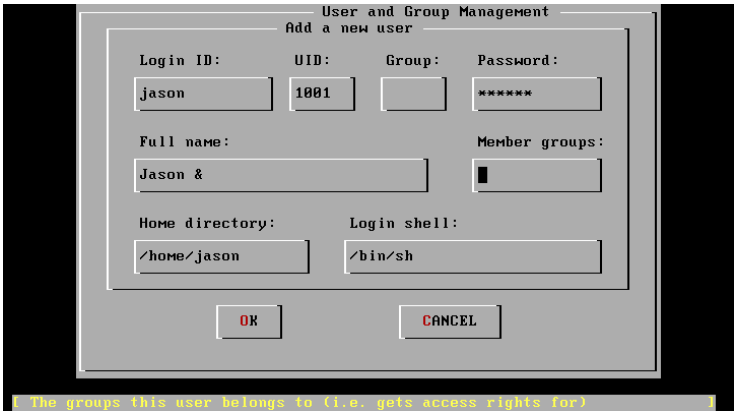


19. When prompted for time zone, select YES and choose the correct time zone.

20. Answer NO to Linux Compatibility.

21. When prompted for packages installation, answer NO. The packages in the CD may not be most up to date and may require immediate update. Software installation will be done at a later stage.

22. Answer YES to adding user to system, then add a normal user to the system. For this installation, user jason is added to the system.



23. When prompted for root password, enter a strong password.

24. When offered to view the options again, answer YES and review all options before exiting the installation and reboot.

3. Securing the base OS

3.1 Login to the system

The first step to modify any configuration is to login to the system. At the login prompt, login as the root user

```
FreeBSD/i386 (machinename) (ttyv0)

Login: root
Password: <Type the password for root user>
```

The first thing to do is to elevate the normal user's group, making the normal user able to become root user later.

```
# vi /etc/group
```
and edit the first line (after the # comments) which starts with wheel,
```
wheel:*:0:root
```
becomes
```
wheel:*:0:root,jason
```
where jason is the normal user that was created at installation time.

As a normal user, FreeBSD only allow minimal privileges. No changes to the system configuration files are allowed. For the purpose of securing the system, configuration would have to be changed. Unix systems have a superuser account for the purpose of system configuration and maintenance. The name of this account is "root", it is the God mode account, a root user can do anything to the system. Given this amount of privileges, it is considered to be dangerous to login using this account for day to day operation because the consequence of a simple mistake is just too great, therefore, it is much safer to operate or "use" the system with a normal user account and only become the "root" user as need arises. To allow this to happen the user must be a member of wheel group (group 0) which is where user "jason" was assigned above.

Logoff as root user, type
```
# exit
```
and the screen will return to login prompt.

3.2 Login with normal user

Login as normal user

```
FreeBSD/i386 (machinename) (ttyv0)

Login: jason
Password: <Type the password for this user>
```

The user should be logged in the system (provided that the username and password combination is correct).

FreeBSD provides “su” utility to substitute the user’s identity, it can also be used to let a normal user to become root and then back to the normal user again after the “root” required operation is done, as shown in the following example:

```
# su
Password: <Type in root account password here>
# <Execute the commands that require root access>
# exit
# <become the original user again>
```

In normal operation, always use the normal user account, only “su” as needed and immediately switch back to normal user (by typing “`exit`”) after the required operation has finished.
At this point, the system needs to be configured with root privileges, so execute “su” to become root.

3.3 Change root’s default umask

Root user created file are usually related to system configuration and should be kept as secret, therefore, root user’s default file permission should be set to root access only. This should save the root user a lot of time trying to change the permission of each created file.

Edit the .cshrc file
```
# vi /root/.cshrc
```
Change umask line to
```
umask 077
```

This will make all root created file to be defaulted to root access only.

3.3 Edit rc.conf, altering system configuration

According to the man page (`man rc.conf`) of rc.conf, "The file rc.conf contains descriptive information about the local host name, configuration details for any potential network interfaces and which services should be started up at system initial boot time." [1] rc.conf is an important file for system configuration, especially true in BSD system architecture where each service does not start in its individual scripts (like Sys V and Linux).

To edit the rc.conf file
```
# vi /etc/rc.conf
```
Note that in the file there are many lines of configuration already. Most of the lines are direct consequences of the options chosen at installation time, such as IP and gateway addresses.

Out of box, FreeBSD is defaulted to accept log entries from other hosts (syslog) and a network port (UDP/514) is opened listening to incoming log files. There is no authentication to this logging capability, anyone with access to that port could potentially write to the log file on the database server host. With enough log entries, the disk drive may fill up and lead the denial of service (though it is hard to dump that much data without being noticed). Also, the opened syslog port on the host may be a possible entry point for attacker to take advantage of a vulnerability (if there is one) of syslog and attack the host. It would be a wise idea to shutdown this network logging capabilities.

In rc.conf file, adding a line
```
syslogd_flags="-s"
```
will disable the message logging from remote hosts (in next reboot).

Tips:	*If not logging to any network host, "-ss" may be used, this will eliminate any network connections with syslogd*

Normally, when an attacker plans an attack on a host they might want to perform reconnaissance on the target host, some of these actions may include portscan, which means trying every port on a system to see what ports are actually open. The opened ports may provide information of weakness for the attacker (such as vulnerable service running on the open port). FreeBSD provides an option to log all these connection attempts to un-opened ports.

By adding the line
```
log_in_vain="YES"
```
to rc.conf, all attempts to closed ports will be logged.

Tips:	*This could produce a lot of log entries and may fill up logging disk. Use with caution.*

Similarly, when an attacker perform reconnaissance on a system, the knowledge of which OS platform is running would be of great value. Sometimes, an attacker would send out of spec packets to a host and by the information returned, it would be able to determine the type of OS running (this usually teams up with port scan to produce more information). To avoid unnecessary information leaking to the attacker,

[1] FreeBSD File Formats Manual [man rc.conf]

```
tcp_drop_synfin="YES"
```
should be added to rc.conf, this will effectively tell the system to ignore all the TCP packets with SYN and FIN flag set. Notice that the kernel has to be set with "TCP_DROP_SYNFIN" to activate this option (see below – kernel section).

Tips:	*This option would break RFC compliance. Do not use this on a web server.*

ICMP specification allows a type of packets for the router to tell a sending host the most optimized route to another host. In GIAC's network, routing is done statically and there is only one router out to other parts of the network, this type of packets should not exist under normal condition. If these packets are on the network, it should either be a network component error (maybe mis-configuration) or a local LAN attack. The danger of these packets lies in the redirection, if traffic is redirected to a hostile local LAN host, that host would be able to eavesdrop on all traffic to and from the server.

To disable the effect of such packet and to log them, add
```
icmp_drop_redirect="YES"
icmp_log_redirect="YES"
```
in the rc.conf file.

3.4 Edit sysctl.conf file

In rc.conf configuration, "log_in_vain" option was selected to log all the abnormal connection. This is only a logging feature and does not eliminate the threat of information leaked.

Normally, when a host send a SYN packet indicating the intention to establish a connection, the receiving host would either send a SYN+ACK packet back to continue the connection or send an RST packet to notify that the port is not listening. By monitoring whether SYN+ACK or RST packet is in the reply, the attacker would be able to map out the opened ports on a host. On the other hand, if the target host does not send back anything, the sending host would wait till timeout before trying another port which would slow down the scanning process. FreeBSD has an option to disable sending back the RST packet for unopened ports.

Edit the sysctl.conf by typing
```
# vi /etc/sysctl.conf
```
this file should only contain comments. After the comments, add the following lines
```
net.inet.tcp.blackhole=2
net.inet.udp.blackhole=1
```

Tips:	*This is not a replacement for a firewall (packet filter). It should be used in conjunction with a packet filter and possibly be a failsafe mechanism for the firewall. With a proper firewall configured, any log produced by log_in_vain should be a warning that firewall is failing or misconfigured.*

3.5 Securing password encryption

Since the database server has its own user authientication system, the OS level authientication is limited to personnel administrating this computer. With such minimal amount of user, it is easier to do local authientication than network user authientication.
Since local authientication requires the storing encrypted password file on local host. This means if a user somehow acquired this password file, brute force attack (with a a password cracker) on the encrypted string may produce the original password which could lead to system compromised.
FreeBSD comes default with MD5 one way password encryption, which is considered to be more secure than DES encryption method used by a lot of older Unix systems. In addition to MD5, FreeBSD also support blowfish encryption, which is even more secure than MD5. Having a stronger encryption scheme on the password file would mean longer time for password cracker to acquire the password and thus slow down the attack (more time and more chances for detection).

To enable blowfish encryption :
```
# vi /etc/login.conf
```
Editor should show the login.conf file for edit, in default section, there is a line "`:passwd_format=md5:\`" change md5 to blf, so it becomes "`:passwd_format=blf:\`"

To apply the changes made to the file,
```
# cap_mkdb /etc/login.conf
```

At this point, the system is aware of the preferred encryption method (blowfish – blf) but since the password is one way encrypted, the system cannot automatically change the encrypted strings in the password file to the new encryption format. Manual password change is required in order for the encrypted string on the password file to be in new format. Since there are only two users on the system at this point, the changes should be easy.

```
# passwd root
Changing local password for root.
New password:  <Enter password here>
Retype new password: <Enter password again here>
passwd: updating the database...
passwd: done
```
Execute again for the user account that was created earlier.
```
# passwd jason
```
and enter the password for the normal user account.

4. OS Update

To build a secure server, it is essential that the computer is free from known vulnerability. After the securing the base OS, the database server should get updated to ensure it is free from known vulnerability.

One of the biggest strength of FreeBSD is the ability to update the whole OS from source code, this makes updating the OS an easy task. It can save FreeBSD administrator a lot of time tracking down bugfixes and patches.

4.1 Prepare secure network segment

There are a lot of tools in FreeBSD that will assist in the update via source code and some of them will be used for building the database server. The main concern about updating from source is to keep the source code updated. Since source code usually consist of many files and the size of source code are usually quite big, to download all sources everytime the system needs an update would be a waste of resource. FreeBSD employ CVSup program as one of the methods to distribute source code efficiently. The user only download updated source file, leaving all the un-updated files alone.

The problem with updating source via the network while building this database server is the risk involved in putting the database host on an Internet accessible network. Without proper update to get rid of known vulnerabilities, the database server is very susceptible to attack from Internet. Luckily, GIAC Enterprise has a firewall dedicated just for installation purpose.

The installation firewall machine is running Linux with Netfilter (Iptables) as the firewall software. Since GIAC Enterprise is running NAT for all internal machines and only allow incoming connections to the server in DMZ machines, all internal machines (including the installation firewall) should not have any incoming connections from the Internet. (due to the stateful rule on the primary firewall) The duty of the installation firewall is to further enhance the rule of the primary firewall against attacks from Internet (layered protection) as well as protecting the database host against attacks from internal host at installation time. The database host will be protected by rules in Appendix II.

After the firewall rules are in place, plug a crossover network cable from the database server to the installation firewall's internal interface.

4.2 Verify the DNS

Since FreeBSD updates are going to be downloaded onto the machine and certain programs are being run as root, it is absolutely essential to make sure the files that are installed are not a trojanized version.
At current time, FreeBSD does not provide digital signing or checksum of package files for verification, that means the user has to be responsible to somehow verify the source authenticity of the package file (or any files they install for that matter). For source code, the installing person has the choice of examining the source code and verify that the source is authentic and is safe to install, however this will take a lot of time if every packages are to be examined. This definitely does not make any business sense for GIAC Enterprises (unless they want to spend 300 hours to install a machine). GIAC's management decided that download of packages from listed FreeBSD mirrors (without digital signature or checksumming) is an acceptable installation method and the risk factor involved is acceptable (this is also the most popular method to update packages).
The mirror site listing in FreeBSD handbook shows the host name of the download site but it is up to the local DNS server (as configured at installation time) to resolves the name to an IP address. If an attacker somehow poisoned the local DNS server and redirect the installing computer to an CVSup server having a trojanized version of FreeBSD, then the results can be catastrophic. To reduce the risk of such occurrences, DNS server should be audited all the time. Another quick check could be verifying the local DNS resolve results with another DNS on the Internet.

Let's say the download site to be used is ftp2.freebsd.org,

```
# nslookup ftp2.freebsd.org
Server:  ns1.giacenterprise.com
Address:  x.y.z.11

Name:    ftp2.freebsd.org
Address:  130.94.149.162
```

The local DNS server resolved ftp2.freebsd.org to 130.94.149.162. To verify this, www.samspade.org provides a lookup service on the web. Type in ftp2.freebsd.org and the "Do Stuff" button, then the resolution is performed. When the results is done, it should show "ftp2.freebsd.org resolves to 130.94.149.162" which matches the local DNS results. This test is definitely not foolproof, if the site has a distributed host setup (using round-robin DNS) or recently had the IP address migrated might make the IP address shows up different. In that case, either find a mirror site which has both matching IP addresses or fully audit the DNS again (which should be done regularly anyways).

Tips: *The quick check should be used to supplement the DNS audit, NOT to replace it.*
There is still always some risk to download updates and patches from the internet, md5 and pgp signature helps a lot but nothing is totally bulletproof (if you verify against an already fake checksum value or a fake key). Always exercise caution and download only from known trusted source only.
If the risk of not having checksum for binary packages is too great, then FreeBSD might not be the best OS for you. (FreeBSD team is considering to provide pgp signature in future versions)

4.3 Download CVSup for updating

CVSup software will be used to update the source code, however, the installation media of FreeBSD 4.5 RELEASE does not come with a cvsup package that can be run without the GUI library (X Window is not installed on this database server). The workaround can be manually download and re-compile cvsup with parameter "-DWITHOUT X" or to download the cvsup package compiled without the GUI library from the FreeBSD distribution site.

FreeBSD has a tool for automatic retrieval and installation of a package from remote site, but the URL location of the actual package has to be known. The page http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/mirrors-ftp.html lists the known public FreeBSD mirror sites. The closest site to GIAC Enterprises location is ftp2.freebsd.org, manually ftp there and locate the cvsup package. In this case, the location (URL) of the required file is ftp://ftp2.freebsd.org/pub/FreeBSD/ports/i386/packages-4-stable/net/cvsup-without-gui-16.1f.tgz. The packages-4-stable in the URL path is correct although the actual OS is 4.5-RELEASE, packages only have one branch on the ftp sites (the latest).

To install this package into the database server, execute

```
# pkg_add -v ftp://ftp2.freebsd.org/pub/FreeBSD/ports/i386/packages-4-stable/net/cvsup-without-gui-16.1f.tgz
```

The –v parameter is for the pkg_add program to produce more text information regarding the installation. When pkg_add finishes, CVSup is installed on the machine.

4.4 Create configuration file for CVSup

CVSup can update all the source code provided by FreeBSD or just download and update ones that the user specify. All of these is controlled by a configuration file provided at run time.

To configure CVSup,
`# cp /usr/share/examples/cvsup/cvs-supfile /root`
This command copies the supplied CVSup sample configuration to /root.

`# vi /root/cvs-supfile`
will edit the configuration.

Change the first configuration line from "*default host=CHANGE_THIS.FreeBSD.org" to "`*default host=cvs3.freebsd.org`". Host is the configuration parameter that is being changed, the value for the parameter is cvs3.freebsd.org which is the CVSup mirror site for source tree synchronization (FreeBSD handbook has a list of CVS mirror sites at http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/cvsup.html#CVSUP-MIRRORS, pick the one closest to current physical location if possible) Again, the IP address can be verified with another DNS server on the Internet to prevent DNS poisoning (see above).

Change the second line from "*default base=/usr" to "`*default base=/usr/local/etc/cvsup`". This specifies the working directory for CVSup program. Note that this directory does not currently exist in place, it should be created after the configuration of this file.

Change the third line from "*default prefix=/home/ncvs" to "`*default prefix=/usr`". This specifies the directory of the downloaded source code file.

Add a line "`*default tag=RELENG_4_5`". This line tells CVSup client which version of source code to update from. There are many different source code trees on the CVSup server, each of them carry different version and branches of FreeBSD. The tag "RELENG_4_5" means the 4.5-RELEASE branch with all security patches and fixes. For more information on the available tags, refer to http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/cvs-tags.html.

Next section is the "Main Source Tree", in this configuration, all source except a few collections are omitted (will be covered below). The first configuration line defaulted to "`src-all`", this instructs CVSup to download all source tree collections. The following lines in the sections which are commented out are for selection of individual collection.

The next section configure the ports collections is part of BSD systems' strength, most of today's widely used available software packages has been ported to the FreeBSD system. The port tree allows a FreeBSD system constantly having access to the latest version of many different software as long as the port tree is up to date.

The port tree contains many different collections, eg. Games, office tools. Since the database server only requires packages from some collections, there are a lot of collections which would not be needed on this machine. To avoid downloading un-needed packages, only the useful collections will be synchronized.

The packages that should be included in the CVSup updates should be as follows,

```
ports-archivers tag=.
ports-databases tag=.
ports-devel tag=.
ports-net tag=.
ports-security tag=.
ports-sysutils tag=.
```

Note that the tag=. is to override the default version to download, this is needed because there is only one port tree (CURRENT, which is reprensented by ".") available.

The final part of the CVSup configuration file is documentation, it is always best to have the latest documentation on the machine itself. So, leave the uncommented "`doc-all`" line alone. After verifying the configuration, save the file.

Create directory "`/usr/local/etc/cvsup`" to satisfy the CVSup configuration, by running

```
# mkdir /usr/local/etc/cvsup
```

Earlier in this section, CVSup client is configured to update all source tree, however, this may not be needed, as there is no real need for games or documentation in other languages other than English on this machine. In order to conserve bandwidth, the unnecessary source synchronization should be disabled. To do this, edit /usr/local/etc/cvsup/sup/refuse by executing

```
# vi /usr/local/etc/cvsup/sup/refuse
```

Add the following lines to the file,

```
doc/es*
doc/ja*
doc/ru*
doc/zh*
doc/fr*
ports/chinese*
ports/german*
ports/japanese*
ports/korean*
ports/russian*
ports/vietnamese*
www/es*
www/ja*
www/ru*
www/zh*
data/es*
data/ja*
data/ru*
data/zh*
www/data/es*
www/data/ja*
```

```
www/data/ru*
www/data/zh*
src/share/doc/es*
src/share/doc/ja*
src/share/doc/ru*
src/share/doc/zh* [2]
src/games
```
and then save the file, this will avoid CVSup client to update those directories.

By executing as root user
```
# /usr/local/bin/cvsup -g -L 2 /root/cvs-supfile
```
the source tree and ports tree will be synchronized with the FreeBSD distribution server.

4.6 Compile the source

Now that the source code is updated, the FreeBSD base system is ready for rebuilt which should get rid of all known vulnerabilities in the base system.
To compile the base system, execute
```
# cd /usr/src
# make -j4 buildworld
```

This process takes quite a bit of time, on the database server platform, it may take more than an hour for this procedure to finish.

4.7 Re-compile the kernel

The kernel "is responsible for managing memory, enforcing security controls, networking, disk access, and much more."[3] It is necessary to re-compile the kernel for some security features in the OS to work.

The first step to re-compile the kernel is to configure it,
```
# cd /usr/src/sys/i386/conf
# cp GENERIC DATAKERN
# vi DATAKERN
```

where GENERIC is a sample configuration file.

While the DATAKERN file is a copy of the sample config file, it already contained some kernel options. Most of the unnecessary hardware device options in the kernel are to be disabled, this is more for performance reason. The sample config file does not include all possible options, refer to /usr/src/sys/i386/conf/LINT for details. For security of the host, the following options are to be added to the config file.

The "IPFILTER" option, as its name implies, provides IP Filter software firewall support. The "IPFILTER_LOG" option allows IP Filter software log to syslog services running on the machine. The "IPFILTER_DEFAULT_BLOCK" option allows the IP Filter software to

2 How to refuse stuff using cvsup, DVL Software Ltd. [http://www.freebsddiary.org/refuse.php]

[3] FreeBSD Handbook, chapter 9 http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig.html

base the rules on an explicit allow mode, all the traffic that are not specified to allow will be dropped.

```
options         IPFILTER                ipfilter support
options         IPFILTER_LOG            ipfilter logging
options         IPFILTER_DEFAULT_BLOCK  block all packets by default
```

As discussed above in the securing base OS section, the "TCP_DROP_SYNFIN" option has to be present in the kernel for it to work. The "ICMP_BANDLIM" option enables ICMP response bandwidth limiting[4] which can protect the machine from DOS attack as well as reducing the chances of being used as a DOS relay.

```
options         TCP_DROP_SYNFIN         #drop TCP packets with SYN+FIN
options         ICMP_BANDLIM

options         SC_DISABLE_DDBKEY       # disable `debug' key
options         SC_DISABLE_REBOOT       # disable reboot key sequence
```

The option "SCDISABLE_DDBKEY" and "SC_DISABLE_REBOOT" are for the physical security of the console. These options avoid anyone having access to the physical console accidentally reboot or gain unnecessary information on the machine.

After editing the config file, it is ready to be compiled,

```
# cd /usr/src
# make buildkernel KERNCONF=DATAKERN
```

The kernel is ready to be installed at this point.

```
# make installkernel KERNCONF=MYKERNEL
```

will install the kernel.

To activate the kernel, a reboot is necessary. As root user, type the command

```
# reboot
```

The system will reboot at this stage. When the reboot is finished, the system is running on the new kernel.

> *Tips:* *If the system failed to boot, you can revert to the original kernel and retry different options. Refer to http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/kernelconfig-trouble.html#KERNELCONFIG-NOBOOT for information on this.*

4.7 Install compiled binaries

The source tree was compiled earlier, now that the kernel are compiled and installed, it is ready for the binaries to be installed. For this to happen, the system has to be in single user mode,

```
# shutdown now
```

After the system gets into single user mode,

```
# cd /usr/src
```

[4] LINT file v 1.749.2.96, FreeBSD 4.5-RELEASE

```
# make installworld
```

This will install all the updated binaries compiled from source downloaded by CVSup, and the system is then ready to be rebooted.

```
# reboot
```

5. More securing OS

The login prompt should appear on the screen. Login as normal user and then “su” to root.

5.1 Change permission of root directory

There should not be any normal user other than system administrators logging onto the system, but just in case this happens, the root directory should be properly protected. This avoid unnecessary monitoring by other users on the system.

```
# chmod 700 /root
```

5.2 Change permission of suid and sgid binaries

Suid binaries allows a user to execute the program as a different user (usually root). Sgid works in similar fashion and allows the user to become another group. Some of the suid binaries are badly implemented and are easily exploited, they could easily lead to local user compromising the machine through these suid and sgid binaries. The best practice is to use suid and sgid binaries only if necessary and disallow the use of the unnecessary ones.

To find all suid binaries on a machine,

```
# find / -perm –4000
```

To find all sgid binaries on a machine,

```
# find / -perm –2000
```

Some binaries are almost for sure never to be touched, for those binaries, permission 000 should be given, this will disallow any read, write and execute from any user. For the binaries that are only useful to root, permission 500 should be given and should be owned by root, it would only allow root to execute and read them.

The following binaries’ permission should be set to 000,

```
/usr/bin/cu
/usr/bin/uucp
/usr/bin/uuname
/usr/bin/uustat
/usr/bin/uux
/usr/bin/at
/usr/bin/atq
/usr/bin/atrm
/usr/bin/batch
/usr/bin/ypchpass
/usr/bin/ypchfn
```

```
/usr/bin/ypchsh
/usr/bin/keyinfo
/usr/bin/keyinit
/usr/bin/lock
/usr/bin/yppasswd
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/lpq
/usr/bin/lpr
/usr/bin/lprm
/usr/libexec/sendmail/sendmail
/usr/libexec/uucp/uucico
/usr/libexec/uucp/uuxqt
/usr/sbin/mrinfo
/usr/sbin/mtrace
/usr/sbin/sliplogin
/usr/sbin/timedc
/usr/sbin/traceroute6
/usr/sbin/ppp
/usr/sbin/pppd
/bin/rcp
/sbin/ping6
```

To set the above binaries to permission 000, for each binary,

```
# chmod 000 [binary path/name]
```

The following binaries' permission should be set to 500.

```
/usr/bin/crontab
/sbin/shutdown
```

To set the above binaries to permission 500, for each binary,

```
# chmod 500 [binary path/name]
```

5.3 Mount drives

Some local directory should never encounter certain type of files, such as suid. Such type of files running on directory such as /tmp is certainly abnormal and should be stopped. Fortunately, FreeBSD allows mount options that will limit the type of operation on a mount point. Even though no user should be allowed to log into database host system to execute anything or run any files, a precaution measure is always helpful.

To set the allowed operation on a mount point,

```
# vi /etc/fstab
```

For /tmp's option, change to `rw,nosuid,noexec,nodev`
For /usr's option, change to `nodev,rw`
For /var's option, change to `nodev,nosuid,noexec,rw`

The "nodev" option disallow files to be a device, avoid unnecessary access to hardware devices . The "nosuid" option disallow files on the specified file system to run as suid binaries. The "noexec" option disallow execution of any files on the specified file system. By limiting the capabilities of files in different file system, normal user's ability on the system is reduced. Even if an attacker gained access as a normal user on the system, it would be harder to exploit local vulnerabilities.

6. Server Installation and configuration

6.1 Configure IP Filter

To build a layered secure environment, a firewall on the database server host protecting itself from attack is necessary. IP Filter is a very popular choice amongst Unix systems, it provides stateful inspection capabilities to filter packets.

While compiling the kernel, IP Filter options were already set, the default to block option is also set. At this point of time, the database hosts' network connection to outside or even to itself (loopback interface) is blocked by IP Filter. To make the database host useable, certain change has to be made to allow the host to provide service.

Every firewall needs rules or instruction to operate, IP Filter default to reading a file in /etc called ipf.rules. For the database server host, access should be given on absolute necessity basis.

```
# vi /etc/ipf.rules
```
add the rules
[rules with explanation in Appendix III]
With the provided rules, the firewall provides only necessary access to its user and to the OS itself.

After the rule file is in place, the rules have to be activated. It is also a good idea to make the database host implement the firewall rules at startup time, to do so,
```
# vi /etc/rc.conf
```
add the following lines at the end of the file
```
ipfilter_enable="YES"
ipmon_enable="YES"
```

The first line instruct the host to startup with IP Filter. The second line tells the ipmon to startup as well, this is the logging program for IP Filter, it enables firewall logging to syslog.

6.2 Read and change syslog

GIAC Enterprises has a centralized syslog server for all servers logging. All servers on the network logs locally as well as to the centralized logging server.

To setup appropriate logging for the database host,

```
# vi /etc/syslog.conf
```

Add the following line,
```
*.*                                             @x.y.z.5
```
where x.y.z.5 is the IP address of the syslog host on the network. This would send a copy of all local log to the remote syslog server, so all local events would be logged centrally. Even if an intruder changed the local log entries, the remote entry would not be changed, giving administrator more chances to trace the attacker.

6.3 Install and configure NTP

Logging of events accurately is important when it comes to incidents handling and forensics. Time plays a critical role in correlation of events that happened, therefore, it is essential to keep a fairly accurate system clock. FreeBSD comes with xntpd which can synchronize the clock with a central ntp server as well as automatically compensate for the time drift of the system clock, keeping the clock on the machine accurate enough for most purposes.

To configure xntpd, create the configuration file,

```
# vi /etc/ntp.conf
```

Add the line

```
server x.y.z.1
server x.y.z.2
driftfile /etc/ntp.drift
```

x.y.z.1 and 2 are the local NTP servers. The driftfile is strictly for ntp's to keep track of clock drift.

To make xntpd starts at the next reboot, edit rc.conf

```
# vi /etc/rc.conf
```

Add the line

```
xntpd_enable="YES"
```

Upon next reboot, the system should start xntp automatically and will synchronize the clock automatically with central time server on the network.

6.4 Install and configure MySQL

The easiest way to install and maintain software on FreeBSD is through the use of ports. MySQL, being a popular database package, is in the FreeBSD port. To install MySQL via ports,

```
# cd /usr/ports/databases
# ls
```

The directory listings should show quite a few software packages related to mysql. The package of interest here is mysql323-server.

```
# cd mysql323-server
# make
```

The make process will automatically download the software source from a known source and then check md5 signature of the downloaded file for integrity (and to a lesser extend authenticity) of the file. Then the compiling of the MySQL server software will start. During the compilation of MySQL software, some other packages (such as libtool)would also be needed to satisfy dependency, the "make" process will automatically takes care of all the download, compile and installation of related software.

After the "make" process is done, do a

```
# make install
```

to install the software. At the install stage, the "make" process will notice the dependency of mysql-server to mysql-client and will automatically download and install the client.

After the installation process is done, a script is added to `/usr/local/etc/rc.d/mysql-server.sh` which will start the MySQL server at the next reboot.

Since the current installation task is to provide a secure MySQL setup for the database administrator to work on, individual user will not be created (that's the work of db admin). But an administrator account on the MySQL server will be created so the MySQL admin can remotely control the MySQL server with full privilege to only the database and not the system.

MySQL server uses option file to specify and control server's operation and parameters. There is a need to set a global option file to specify some security option to properly protect the database server.

The default global option file is /etc/my.cnf

```
# vi /etc/my.cnf
```

Add the following lines,

```
[mysqld]
bind-address=127.0.0.1
port = 3306
skip-name-resolve
log
safe-show-database
```

The option "skip-name-resolve" is used to skip DNS resolve in authentication, all host name will have to be in IP, this can converse some bandwidth as well as potentially avoid the effect of DNS poisoning.

The "safe-show-database" option limits the output of "show database" to only show databases, which that specific user have some sort of privileges, this avoids user to acquire un-necessary knowledge about different databases on the server.
MySQL also supports another option "local-infile=0" which disallow any user to import file into the database from local filesystem. This option is not used because the database administrator would have to import some files such as logs for review from time to time. But the database administrator has also been told not to give out file privileges to user, so normal user cannot import files.
MySQL server is set to listen to localhost only (127.0.0.1) so direct un-encrypted conncection can be eliminated.

MySQL installation from the port tree include some dummy account in the database authentication, some of these accounts are using user and host based authentication and may not be very secure so eliminating useless and password-less account is important.
To change password, login to the database

```
# mysql
```

At the database prompt,

```
mysql > \u mysql;
```

This is to change to the mysql database, and then look at the content of the user file.

```
mysql > SELECT * FROM user;
```

Two account from localhost and two more from hostname are listed, it is a good idea to delete the hostname account,

```
mysql > DELETE FROM user WHERE Host = 'dbhost';
```

This SQL statement will eliminate two useless account. There are two more account left, one is root@localhost and the other one is @localhost which is a wildcard account which should be deleted.

```
mysql > DELETE FROM user WHERE User = '';
```

Only root@localhost account is left at this point, the password for this account should be set.

```
mysql > SET PASSWORD FOR 'root'@'localhost'=PASSWORD('password');
```

Out of box, MySQL also provides a test database called "test", to get rid of it,

```
mysql > DROP DATABASE test;
```

The database administrator need a full access account on the database to administrate it. To do this,

```
INSERT INTO user VALUES
('localhost','db_admin',PASSWORD('password'),'Y','Y','Y','Y','Y','Y','
Y','Y','Y','Y','Y','Y','Y','Y');

mysql> FLUSH PRIVILEGES;
```

6.5 Install and configure stunnel

To avoid local eavesdropping on database traffic, stunnel is used to create an encrypted tunnel to transport the database transaction. Note that this is a solution for MySQL 3 only, starting from version 4, MySQL support SSL connection directly.

To install stunnel,

```
# cd /usr/ports/security/stunnel
# make
```

At the end of the make process, an RSA key pair will be generated. The user will be prompted for information to generate this RSA key. After the key is generated, the make process is finished, the software is ready for installation.

```
# make install
```

should install stunnel in place.

stunnel automatically loaded a sample configuration file, it can be modified so that stunnel can be started automatically at next reboot.

```
# mv stunnel.sh.sample stunnel.sh
# vu stunnel.sh
```

Delete the lines

```
${STUNNEL} -d 993 -r localhost:imap -p /usr/local/etc/stunnel.pem
${STUNNEL} -d 995 -r localhost:pop3 -p /usr/local/etc/stunnel.pem
```

so these un-necessary ports won't be opened.

Add the necessary port for MySQL

```
${STUNNEL} -d 3307 -r localhost:3306 -p /usr/local/etc/stunnel.pem
```

This will allow encrypted connection from other hosts to port 3306 to be redirected to local port 3306 which the MySQL server is listening.

The stunnel software was compiled with TCP wrapper support that means the access will have to be given by TCP wrapper for stunnel port to accept connections. (see below for modifying instructions). GIAC's database host only allow local LAN connections, so TCP wrapper is also configured as such.

6.6 Configure OpenSSH

OpenSSH is used for remote administration and transfer of backup files. FreeBSD comes standard with OpenSSH, so no installation is required.

To modify the sshd configuration,
```
# vi /etc/ssh/sshd.conf
```

Uncomment the Protocol 2,1 line and change it to read [5]
```
Protocol 2
```
This will disable SSH version 1 which may not be as secure.

Modify the X11Forwarding line to
```
X11Forwarding no
```
There is no need for this host to forward the X11 traffic anywhere.

Add line
```
AllowUsers jason
```
Where jason is the normal user on this machine, this restrict SSH to only allow this user to login. When superuser privileges is needed, the normal user can "su" to root user.

SSH service, like stunnel is governed by TCP Wrapper which restrict and control access to specified host. Changes to the configuration file is needed to restrict SSH access to the host from only one control workstation. (stunnel's access is also configured here)

```
# vi /etc/hosts.allow
```
comment out ALL : ALL : allow (by putting a "#" at the beginning of the line) and add the following lines right after the file's description.
```
sshd : x.y.z.15 : allow
localhost.3307 : x.y.z.1/255.255.255.0 : allow
ALL : ALL : deny
```

Comment out all the other active configuration in the file.

> *Tips: hosts.deny on FreeBSD is deprecated. All TCP wrappers entries has to be in hosts.allow. The TCP wrapper default to allow all services which is not secure in most situation.*

6.6 Install and configure Tripwire

[5] Aeonflux, *FreeBSD Security How-To, Chapter One,* http://www.daemonnews.org/200108/security-howto.html

Tripwire is a file integrity checking tool, it is used as a local intrusion detection utility. A tripwire database is created after all installation is done to ensure a “clean” checksum of the system for future audit.

To install Tripwire (and other required dependent tools)

```
# cd /usr/ports/security/tripwire/
# make
```

When the make process finishes, install Tripwire by
make install
The user will have to agree to the license agreement and confirm the path of the installation. Then the user will be prompted for site passphase and local passphase, strong password should be provided. Tripwire then install and initialize a standard database which contains standard files in FreeBSD installation.

Since there are a few components that database server host does not have, Tripwire will complains about missing file and finishes the installation. Also, the standard configuration does not include e-mail notification in case of events. The problem can be corrected by manually modifying the standard database

```
# vi /usr/local/etc/tripwire/twpol.txt
<change according to Appendix IV>
```

After the configuration file is modified, Tripwire needs to recognize this new database,

```
# /usr/local/sbin/tripwire --update-policy -Z low
/usr/local/etc/tripwire/twpol.txt
```

This will update the policy database.

And then the database can be initialized with signature update to it.

```
# /usr/local/sbin/tripwire --init
```

This will initialize the database with the current file checksum.

Ongoing Maintenance

Installation and configuration is only the first step to a secure database server, maintenance is equally important. Maintenance procedure involves many areas of system administrations.

Subscribe to mailing list

Intelligence can be both the weapon and the shield. Attackers and administrators are always on a match to “hack and defend” with the attackers. Administrators need quick and accurate access to information regarding any possible vulnerability on the database hosts for effective patches and fix to get ahead of the attackers. One of the best source of such security information is usually available directly from the software developer. FreeBSD and MySQL both have mailing list setup for announcement, these are very low traffic list only for important information.

FreeBSD security related announcement mailing list:
freebsd-security-notifications@freebsd.org

freebsd-announce@freebsd.org
Refer to: http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/eresources.html

MySQL announcement list
announce-subscribe@lists.mysql.com
Refer to: http://www.mysql.com/documentation/lists.html

Usually, FreeBSD announcement list will also include vulnerabilities for any software in the port tree, but this is not the most reliable and timely source of information. Bugtraq mailing list usually has the latest bug or vulnerabilities covered or discussed.

Bugtraq mailing list
Refer to: http://online.securityfocus.com/cgi-bin/subscribe.pl

FreeBSD also has a webpage dedicated to the OS security at
http://www.freebsd.org/security/index.html

OS and software update

Updating is an essential part of maintaining a secure system, it is also the way to get rid of vulnerabilities on a system. As discussed above in the system installation stage, FreeBSD can utilize CVSup utility in order to update the source code of the system as well as the port tree. It is essential that the system has the most up to date source and port tree for the system re-compile to be effective.

A cron job can be setup so the host will automatically update the source and port tree every night. Due to the effective CVSup utility, only the source updated on that day will be downloaded.

To setup a cron job to automatically update the source and port tree at night,

```
# vi /etc/crontab
```
Add the following line
```
0 5 * * *       root    /usr/local/bin/cvsup -g -L 2 /root/cvs-supfile
```

This will activate CVSup every day at 5 AM to update the source and port tree. This way, if the administrator received a notice from FreeBSD's mailing list regarding a vulnerability in FreeBSD OS, the new source can be downloaded very quickly (due to minimal difference within 24 hours) and then re-compile to get rid of new vulnerability.

For ports, the procedure is similar, if a vulnerability of a software package on the database server is known and the update is available on the port tree. Ensure the port tree is synchronized by running CVSup again and then upgrade the specific port. There is however a little trick here at the port update, since port are compiled from source and may require dependency on other software, the intention of upgrading one port may mean upgrading a few depended port as well. There is a tool that can take care of this dependency problem by upgrade all the depended port, this is the feature of

"portupgrade", it is a utility in a port tree "`/usr/ports/sysutils`". When using portupgrade, just type "`portupgrade -rv <name of package to update>`".

Sometimes, a software package is updated for features and not for security reasons, such update are not announced in the security announcement lists. To examine the installed packages having newer version available, the command "`pkg_version`" would provide the packages status (whether a new version is available) provided that the port tree is up to date. "pkg_version" should be run every month to ensure that all software packages has all the latest features.

Notice that this is a database server host, where security and stability matters the most and features can be less important. When a change or update to the MySQL package is noticed (not security related), it may not be the best idea to immediately update to the latest version on the production server. With GIAC's two stage change implementation setup, the latest version should be put on the testing machine for stability testing. After proven stable, the latest version can then be installed on the production machine.

Tips:	*During source recompile and update (build installworld), some of the disabled suid binaries are being reset to original suid state. It might be a wise idea to write a script to change all the unnecessary suid binaries back to disabled state instead of doing it manually.*

Tripwire integrity check

In the installation and configuration stage, tripwire is configured with a policy and an initial database. This gives the baseline configuration for future audit. Tripwire's database should be checked regularly to ensure the integrity of the files. To automate this procedure, add the following line in /etc/crontab

```
1 5 * * *       root    /usr/local/sbin/tripwire -m c -M
```

This will make the database check runs at 5:01 AM everyday. After the check is done, a e-mail is sent out to the defined e-mail address in the config file. System administrators should monitor this e-mail every day and check for every files that changed and possibly update the database due to legitimate file change.

While Tripwire may seem to be quite sufficient for automatic integrity checking, but there is a fundamental problem in the whole integrity checking scenario – even the checking tool is in the same environment as the files being checked. If a machine is compromised, the attacker could simply alter the tripwire binaries and make it e-mail very normal report with the green flag on everyday, making the administrator to have a false sense of security.

It is wise to have a second set of checking tool isolated from the environment to check the Tripwire binary from time to time to ensure the integrity of the whole system. FreeBSD comes with "md5" utility in the "/sbin" directory, most of the binaries in "/bin" and "/sbin" are compiled statically meaning they can be run without any system libraries installed. The best way to create an isolated environment is to burn all of "/bin" and "/sbin" onto a CD right after installation of OS and run "md5" against tripwire binaries and databases (located in "/usr/local/etc/tripwire") for a baseline signature and record

the signature for later comparison. Other tools on the CD can also provide a trustworthy auditing toolkit (although a few more tools, such as lsof would be even better).

Backup

There are two concurrent backup strategies on this database server. One is by the database administrator which backup the data in the database remotely. Another backup routine is done locally to the tape drive and it focuses on system configurations and binaries.

The database administrator connects remotely from the backup workstation to the MySQL server to dump the database every night (with a script) and then backup this dump to a tape drive.
MySQL client has the "mysqldump" command which facilitate this operation, this backup will also utilize SSL tunnel that every host have to use in order to connect to the MySQL server so data transfer is secure. The backup station is located at a remote locked room and the backup tapes is stored securely in the same room. To dump the remote database, the database administrator's script execute,

```
mysqldump -C -h 127.0.0.1 -p 3306 -u backupuser -p<password> DATABASE
[table] > <backup file path and name>
```
[6]

For the system configuration backup, since the database server's configuration is relatively static and changes very little over time, backup should be done on a weekly basis. Backup can be done with the command,

```
# tar cpf /dev/sa0 --directory --exclude=proc --exclude=mnt --
exclude=dev/sa0 --exclude=var/db
```

This command uses the tar utility to backup all files except MySQL database's data to the tape.

For audit purposes, five sets of tape should be used and rotate in a round robin fashion for system backup which would provide one week worth of backup (for database and one month for system) in with file history correlation. Due to the criticality of the database server, both the database content backup and the system configuration backup are manually done for an extra time every month, this set of special backup is sent in a locked box to a remote safe for two year storage, this is done to avoid site damage (flood, earthquake or fire) leading to unrecoverable damage to all local computer systems and tapes.

Log monitoring and others

The database server host not only log locally but also to the central logging facility, all network devices and servers will also log to the central logging host. Log processing and monitoring will also happen on the central logging location because of the easiness in correlation of events. Swatch is installed on the central logging location to monitor the log and provide immediate notification through either pager or e-mail depending on the

[6] Benson, Comments on http://www.mysql.com/doc/m/y/mysqldump.html

log entry. It is already set to look for patterns like “su”, “panic”, “reboot” and “refuse connect”. A daily regular manual log check is performed by administrators on site. All logs are kept in compressed format for 1 year in case there is such correlation needs.

GIAC Administrators also rely on FreeBSD’s own reporting utility (“periodic”) that runs as often as daily to provide valuable information about the status and events of the database server. By default, “periodic” is set to report to root (by mail), each day this utility compiles a status report consisting of important information such as “Disk Status”, “Uptime”, “login failures” which could help to verify the findings of log reading.

Password policy

Since most of the security mechanism on the database server are password based, having a good password policy is an important step to securing and maintaining the system. All staff and administrator should be taught about password choosing techniques and the importance of protecting one’s password. Every password on the database server should be changed every 2 months. This should reduce the threat of password cracking and enhance overall password security level.

Check your configuration

Even the best administrators make mistakes, vigorous checking and verification is the key to reduce errors. Each individual procedure mentioned in the installation guide should be checked and verified. Five checking procedures are demonstrated here.

Testing for Blowfish password

In installation procedure 3.5, blowfish password encryption was implemented in order to make offline cracking of password file much more computation intensive which possibly slows down the attempt.

This section will test whether blowfish password encryption is implemented properly and that all passwords are properly encrypted with blowfish encryption.

Actual encrypted password strings are located in /etc/master.passwd file, this file is only readable by root. To check the password strings, login or su to root user and then view the master.passwd file,

```
# more /etc/master.passwd
```

This should display the password file listings with encrypted strings, like the following line,

```
root:$2a$04$jvGHrnlamuWN//ANgMAVe.jrgnke0oZvPHPvuN4L98QtzJuYBVZvW:0:0:
::0:0:Root User &:/root:/bin/csh
```

Each line (like the one above) in the password file represent a user, each element of the user is separated by ":", the first column is the user name, the second is the encrypted password strings. This string is definitely much longer than most other typical Unix password strings encrypted with DES or MD5, but it does not provide enough information to prove that it is blowfish encrypted string.

From the text in the FreeBSD Handbook, "Passwords starting with `$2$` are encrypted with the Blowfish hash function."[7] Comparing this fact to the actual file, check all the lines of passwords text and make sure that all the passwords are encrypted with blowfish (starts with `$2$`). If any user's password is not in blowfish format, change the password of that user to ensure that user's password is updated and encrypted with blowfish.

Tripwire test

Tripwire is installed and will be used as a file integrity checking tool. The check for configuration of Tripwire is rather simple, since Tripwire should report any changes of the monitored files, those files are to be manipulated to trigger a test alarm.

There is a rule that monitors all files in /usr/sbin,

```
/usr/sbin                   -> $(SEC_CRIT) (recurse = true) ;
```

any changes should be reported to the designated E-mail address. To manually test this rule, a new file ("test") is added to "/usr/sbin" (use text editor) and then Tripwire is manually run.

```
# /usr/local/bin/tripwire -m c -M
```

and then check for the e-mail that arrives at the designated mailbox. This test should verify that the rule as well as the e-mail notification are working properly. Examine the e-mail,

```
---------------------------------------------------------------------------
Rule Name: System Administration Programs (/usr/sbin)
Severity Level: 100
---------------------------------------------------------------------------

Added:
"/usr/sbin/test"

Modified:
"/usr/sbin"
```

The addition of the file triggered an alarm, this shows that the rule is working as planned. Delete /usr/sbin/test file and move on to the next test.

There is also another rule to monitor any file changes in "/etc",

```
/etc                        -> $(SEC_CRIT) (recurse = true) ;
```

Similar to the above test, the my.cnf file is edited, one extra empty line is added to the end of the file, which should not affect any functions of MySQL but would change the signature of the file.

Run Tripwire in checking mode again to verify the integrity of the system, after the e-mail arrive, examine it and look for

[7] FreeBSD Handbook, Chapter 10 Security, http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/crypt.html

```
---------------------------------------------------------------------------
Rule Name: /etc (/etc)
Severity Level: 100
---------------------------------------------------------------------------

Modified:
"/etc/my.cnf"
```

which shows Tripwire picking up the change in the my.cnf file and notify the user by e-mail.

These two tests shows Tripwire to be performing properly and should be aware of any changes in the system files and will notify the user by e-mail about such occurrences.

Network traffic eavesdropping

Since MySQL 3.23 will be running on the server and this version of MySQL server does not directly support any encryption, stunnel is used to create SSL tunnel for encrypting database queries and results to avoid any information leak even if someone can eavesdrop on the network traffic.

To test whether the encryption is working, actual traffic on the network should be monitored. Tcpdump is a tool for packet capture and monitor, it is the tool used for this test.

It is hard to determine whether traffic is encrypted just by looking at single stream of traffic, so a comparison test should be setup for both encrypted and un-encrypted traffic. In an isolated network segment or a direct cross over network connection, revert the MySQL server to listen on its temporary installation IP address,

```
# vi /etc/my.cnf
```

Change

```
bind-address=127.0.0.1
```

to

```
bind-address=192.168.30.2
```

where 192.168.30.2 is the local temporary IP address.

Restart MySQL server by

```
# /usr/local/etc/rc.d/mysql-server.sh stop
# /usr/local/etc/rc.d/mysql-server.sh start
```

Another MySQL client host (192.168.30.3) will initial a connection to the MySQL server, but before this can succeed, setup the prilivilages on the server so this client can connect. At the same time, execute “tcpdump –X port 3306” on the server to listen in on the MySQL port, login from the client and execute “\u mysql” to switch database. On the server’s tcpdump console, network traffic should be capture which should look like

```
16:19:07.923864 192.168.30.3.1040 > 192.168.30.2.3306: P 2010871901:2010871911(10) ack
2681740259 win 33304 <nop,nop,timestamp 22901954 633583974> (DF)
0x0000   4500 003e d2e8 4000 4006 0000 xxxx yyyy        E..>..@.@.......
0x0010   xxxx zzzz 0410 0cea 77db 785d 9fd8 1be3        ........w.x]....
0x0020   8018 8218 abcd 0000 0101 080a 015d 74c2        .............]t.
0x0030   25c3 b966 0600 0000 026d 7973 716c              %..f.....mysql

16:19:07.925505 192.168.30.2.3306 > 192.168.30.3.1040: P 10:136(126) ack 29 win 32120
<nop,nop,timestamp 633584961 22901954> (DF)
```

```
0x0000   4500 00b2 5ad3 4000 3f06 9d32 xxxx yyyy        E...Z.@.?..2....
0x0010   xxxx zzzz 0cea 0410 9fd8 1bec 77db 787a        ............w.xz
0x0020   8018 7d78 b8b0 0000 0101 080a 25c3 bd41        ..}x........%..A
0x0030   015d 74c2 0100 0001 0114 0000 0200 0844        .]t............D
0x0040   6174 6162 6173 6503 4000 0001 fe03 0100        atabase.@.......
0x0050   1f01
```

By examining the packets' content carefully, some plain text information about the database is revealed. MySQL transfer most of the results in a propietry binary format, but since it is not encrypted, decoding it and acquiring the original data is rather simple.

After examining the un-encrypted queries and results transfers, the encrypted traffic should be verified. Revert the my.cnf file back to the original status and ensure that the stunnel daemon is listening (by using command "netstat –an" and observe that port 3307 is listening). Connect from the client to the server and execute the same command as before. Again, using tcpdump ("tcpdump –X port 3307") to monitor the traffic.

```
17:02:31.328900  192.168.30.3.1053  >  192.168.30.2.3307:  P  1:89(88)  ack  1  win  33304
<nop,nop,timestamp 23162290 633845296> (DF)
0x0000   4500 008c d8f6 4000 4006 0000 xxxx yyyy        E.....@.@.......
0x0010   xxxx zzzz 041d 0ceb f893 3cb6 2552 d2e7        ..........<.%R..
0x0020   8018 8218 ce6b ac41 0701 080a 0161 6db2        .....k.......am.
0x0030   25c7 b630 1623 e83b 53f1 86d1 4f03 713c        %..0.....S...O..<
0x0040   cb11 e7ca e984 7711 2c0b f6b7 5ed3 da0b        ......w.,...^...
0x0050   b665
```

All packets are similar to the one above, bit and bytes are all random without any pattern or plain text. Also checking the log to ensure that an SSL tunnel has been created.

```
host stunnel[24764]: 192.168.30.2.3307 connected from 192.168.30.3:1913
host stunnel[24764]: Connection closed: 5417 bytes sent to SSL, 2249 bytes sent to
socket
```

This proves that encryption tunnel was created to encrypt the MySQL traffic between the server and the client host.

Portscan (Reconnaissance attempt)

Almost all serious server systems nowadays are tested with portscans before they are put into serious production. Portscans are usually used to find un-intended serving ports on the platform. On the database server scenario, portscan can be used to determine the logging of reconnaissance attempts is working correctly as well as no un-intended ports are opened.

In procedure 3.3, option "log_in_vain" is activated to log any connection attempts to non-listening port at syslog. However, there is also a local firewall software blocking all the unauthorized connection attempts, if there are any leak of packets by the firewall, the "log_in_vain" option should respond by logging the attempt to syslog. To test whether the connection attempts are handled or logged properly, Nmap is used as a portscanning tool for testing.

A testing host is located on the same network segment as the database server host, without the isolation by any firewall host. A Nmap scanning attempt is initiated from the test host to the database server.

```
# nmap -P0 -p 1-65535 -sS 192.168.30.2
```

The command will start a scan on the database server host, searching for any listening services on the host.

When the scan finishes which should take very long (because of the firewall), the results should be similar to below,

```
Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
All 65534 scanned ports on  (192.168.30.2) are: closed

Nmap run completed -- 1 IP address (1 host up) scanned in 384 seconds
```

There is no opening ports on the database server due to the firewall blocking access except to specific network segments (the test host's IP does not match any of those). So no ports are shown as open.

The log should show all the unsuccessful connection attempts done by the portscan. To verify this, execute

```
# tail -n 400 /var/log/messages | more
```

this should show many unsuccessful connection attempts, such as

```
host ipmon[48]: 19:48:37.051270 xl0 @0:7 b testhostip,54757 -> 192.168.2.68,9959 PR
tcp len 20 40 -S IN
host ipmon[48]: 19:48:37.051467 xl0 @0:7 b testhostip,54757 -> 192.168.2.68,9845 PR
tcp len 20 40 -S IN
```

which proves that the unsuccessful attempts are logged. Also make sure that there are no leaked "log_in_vain" log in the log file by executing

```
# grep "Connection attempt to" /var/log/messages
```

If there are no such entries corresponding to the portscan, the firewall is not leaking any packets.

This test can also be furthered to include remote logging capabilities checking. In procedure 6.2, local syslog server is configured to forward a copy of all syslog entries to a remote syslog server. All the local entries (including the portscan related entries) are also logged on remote syslog server. To verify that the logging is established and in working order, view the log at the syslog server and check that all local log entries are also present in the syslog server.

In the above test, no ports are opened because the firewall is blocking access to any ports on the server. More test is needed to verify that at the production and development segments, the access is appropriate and only the required port are served. Nmap is again used with a host located in production segment, the results should show port 3307 is open which proves that only the port required is opened.

```
# nmap -P0 -p 1-65535 -sS 192.168.2.68
```

Nmap will start to port scan the target host for possible opened port. The results of this scan should look like the following,

```
Starting nmap V. 2.54BETA22 ( www.insecure.org/nmap/ )
Interesting ports on  (192.168.2.68):
(The 47 ports scanned but not shown below are in state: filtered)
Port       State       Service
3307/tcp     open        unknown


Nmap run completed -- 1 IP address (1 host up) scanned in 8 seconds
```

The Nmap result on this test shows the only port opened for the hosts in production and development segment is TCP port 3307 which is the SSL tunnel port for MySQL, this is exactly as intended.

Netcat server – test for IP Filter

IP Filter is installed on the database server as a last line of defense against internal and external attacker. In the last test (port scan), this local firewall has proved to have blocked access to SSL encrypted MySQL server port from unauthorized network segment.

Another concern that may arises is IP Filter's capability to block actual open ports on localhost. The rule base of IP Filter is set to allow a few services and deny all others, this should defend the scenario that some ports are opened due to accidental running of some server programs. In such event, the firewall should still block all traffic not specified as allowed. To test the firewall for such events, a utility called netcat is used to create the necessary service on the server.

The netcat utility is in the FreeBSD ports (/usr/ports/net/netcat), so installation procedure is same as most other software installed from the port tree (refer to installation section).
After the installation, the following command is executed

```
# echo CONNETED | nc -l -i  -p 80
```

which will start a listening server at port 80 and listen to any incoming connections not blocked by the firewall.

To test whether the ports are actually opened, the portscan tests with Nmap are repeated from both location with the above command running, the results should show that port 80 is not listening which proves that the firewall is blocking access to unauthorized opened port on the server.

Works Cited:

"MySQL Reference Manual for version 4.0.2-alpha" http://www.mysql.com/documentation/mysql/full/, http://www.mysql.com/doc/m/y/mysqldump.html

aeonflux. "FreeBSD Security How-To, Chapter One". http://www.mysql.com/documentation/mysql/full/

Delves, "Fluid", Markus. "Armoring FreeBSD" http://www.daemonnews.org/200102/armoring.html

"FreeBSD Handbook", http://www.freebsd.org/doc/en_US.ISO8859-1/books/handbook/

Silver, Mark. "A basic guide to securing FreeBSD 4.x-STABLE" http://draenor.org/securebsd/secure.txt

Walden, James. "Examples-Mysql encryption with stunnel 3.14" http://www.stunnel.org/examples/mysql.html

Schlacter, Marty. "How to Build a FreeBSD-STABLE Firewall with IPFILTER" http://www.schlacter.net:8500/public/FreeBSD-STABLE_and_IPFILTER.html

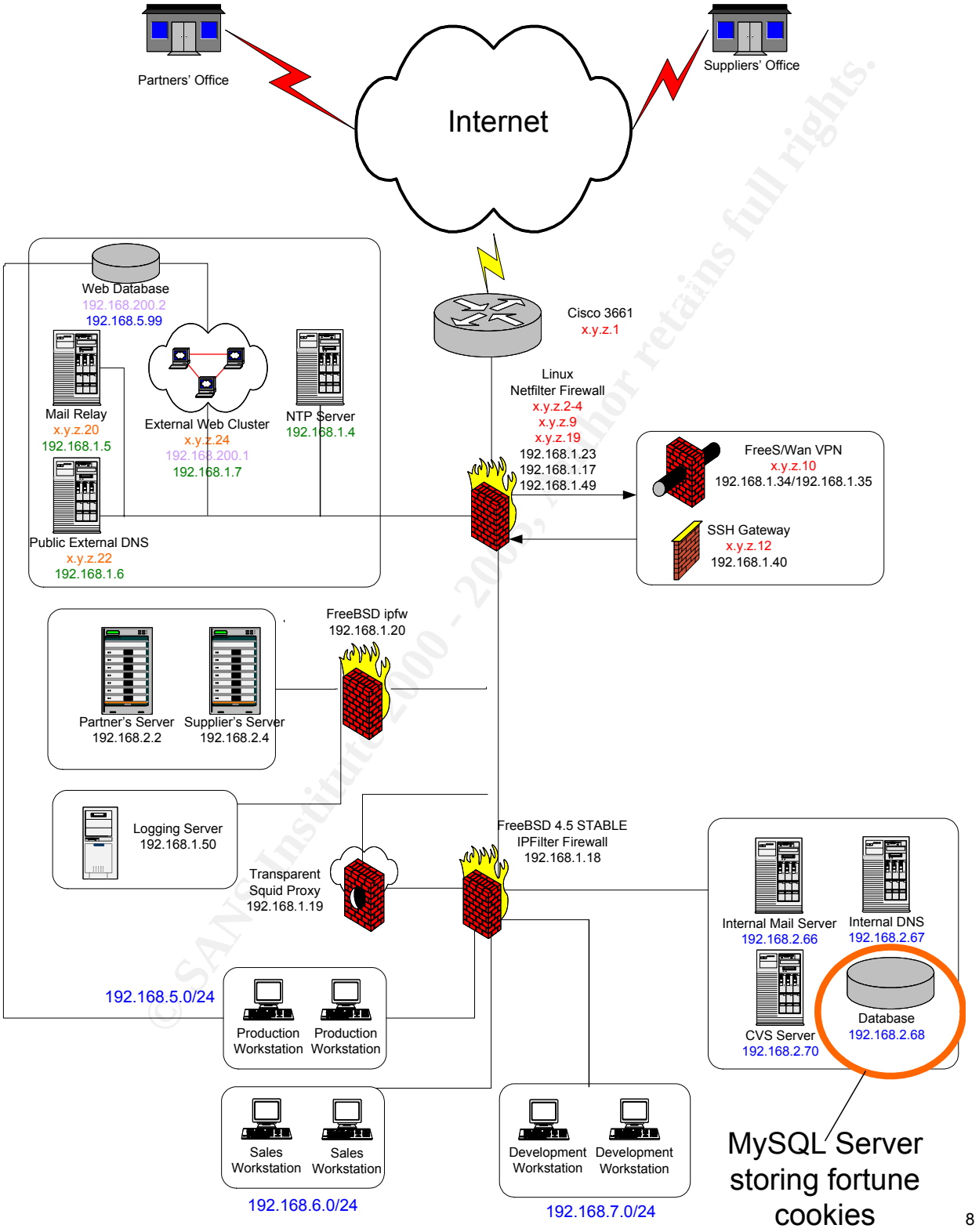
"How to refuse stuff using cvsup" , DVL Software Ltd. http://www.freebsddiary.org/refuse.php

Lam, Jason. "GCFW practical assignment", http://www.giac.org/practical/Jason_Lam_GCFW.pdf

LINT file v 1.749.2.96, FreeBSD 4.5-RELEASE

FreeBSD man pages

Appendix I - Network structure and location of MySQL server



[8]

[8] Lam, Jason. "GCFW practical assignment", http://www.giac.org/practical/Jason_Lam_GCFW.pdf

Appendix II - Rules to protect the database host

eth1 is the interface connected to the internal interface,
eth0 is the interface connected to the external interface.

Allow the database server host to connect to ftp server

```
$IPTABLES -A FORWARD -p TCP -i eth1 -o eth0 -s $INSTALL_HOST --dport
21 -j ACCEPT
```

Allow the database server host to connect to CVSup server

```
$IPTABLES -A FORWARD -p TCP -i eth1 -o eth0 -s $INSTALL_HOST --dport
5999 -j ACCEPT
```

Allow the database server host to connect to local DNS server

```
$IPTABLES -A FORWARD -p UDP -i eth1 -o eth0 -s $INSTALL_HOST -d
$INTERNAL_DNS --dport 53 -j ACCEPT
```

Allow any existing connections already initiated by the database server to pass

```
$IPTABLES -A FORWARD -p TCP -i eth0 -o eth1 -d $INSTALL_HOST -m state
-state ESTABLISHED,RELATED -j ACCEPT
```

Appendix III – IP Filter rules to protect the host

Block all fragmented packets (abnormal on GIAC Enterprises LAN)

```
block in log quick all with frag
```

Get rid of all short IP fragments (too small for valid comparison)[9]

```
block in log quick all with short
```

Block and log any packets with options set in them[10]

```
block in log quick all with ipopts
```

Allow local LAN MySQL connection to the database host

```
pass in quick on xl0 proto tcp from 192.168.5.0/24 to 192.168.2.68
port = 3307 keep state
pass in quick on xl0 proto tcp from 192.168.7.0/24 to 192.168.2.68
port = 3307 keep state
```

Allow the database server to log to remote syslog server

```
pass out quick on xl0 proto udp from 192.168.2.68 to x.y.z.4 port =
514
```

Allow the server to CVSup with CVS server on the Internet

```
pass out quick on xl0 proto tcp from 192.168.2.68 to any port = 5999
keep state
```

Allow the server to resolve hostname (DNS host = x.y.z.3)

```
pass out quick on xl0 proto udp from 192.168.2.68 to x.y.z.3 port = 53
keep state
```

Allow the server to synchronize time with ntp server (x.y.z.1, x.y.z.2)

[9] IP Filter Examples, http://coombs.anu.edu.au/ipfilter/examples.html
[10] IP Filter Examples, http://coombs.anu.edu.au/ipfilter/examples.html

© SANS Institute 2000 - 2005 Author retains full rights.

```
pass out quick on xl0 proto udp from 192.168.2.68 to x.y.z.1 port =
123 keep state
pass out quick on xl0 proto udp from 192.168.2.68 to x.y.z.2 port =
123 keep state
```

Allow the administrators (database admin included) to SSH to this host

```
pass in quick on xl0 proto tcp from 192.168.8.8/26 to 192.168.2.68
port = 22 keep state
```

Allow all connection from and to localhost (this is relatively low risk)

```
pass in quick on lo0 all
pass out quick on lo0 all
```

All other traffic will be dropped (by default)

Appendix IV – Tripwire configuration file

```
#
#                       Policy file for FreeBSD
#
# $FreeBSD: ports/security/tripwire/files/twpol.txt,v 1.2 2002/03/04 16:55:21 cy Exp $


#
# This is the example Tripwire Policy file.  It is intended as a place to
# start creating your own custom Tripwire Policy file.  Referring to it as
# well as the Tripwire Policy Guide should give you enough information to
# make a good custom Tripwire Policy file that better covers your
# configuration and security needs.  A text version of this policy file is
# called twpol.txt.
#
# Note that this file is tuned to an install of FreeBSD using
# buildworld.  If run unmodified, this file should create no errors on
# database creation, or violations on a subsiquent integrity check.
# However it is impossible for there to be one policy file for all machines,
# so this existing one errs on the side of security.  Your FreeBSD
# configuration will most likey differ from the one our policy file was
# tuned to, and will therefore require some editing of the default
# Tripwire Policy file.
#
# The example policy file is best run with 'Loose Directory Checking'
# enabled. Set LOOSEDIRECTORYCHECKING=TRUE in the Tripwire Configuration
# file.
#
# Email support is not included and must be added to this file.
# Add the 'emailto=' to the rule directive section of each rule (add a comma
# after the 'severity=' line and add an 'emailto=' and include the email
# addresses you want the violation reports to go to).  Addresses are
# semi-colon delimited.
#



#
# Global Variable Definitions
#
# These are defined at install time by the installation script.  You may
# Manually edit these if you are using this file directly and not from the
# installation script itself.
#

@@section GLOBAL
TWDOCS="/usr/local/share/doc/tripwire";
TWBIN="/usr/local/sbin";
TWPOL="/usr/local/etc/tripwire";
TWDB="/var/db/tripwire";
TWSKEY="/usr/local/etc/tripwire";
TWLKEY="/usr/local/etc/tripwire";
TWREPORT="/var/db/tripwire/report";
HOSTNAME=hostname;

@@section FS
SEC_CRIT      = $(IgnoreNone)-SHa ;  # Critical files that cannot change
SEC_SUID      = $(IgnoreNone)-SHa ;  # Binaries with the SUID or SGID flags set
```

```
SEC_BIN       = $(ReadOnly) ;         # Binaries that should not change
SEC_CONFIG    = $(Dynamic) ;          # Config files that are changed infrequently but accessed
often
SEC_TTY     = $(Dynamic)-ugp ;        # Tty files that change ownership at login
SEC_LOG       = $(Growing) ;          # Files that grow, but that should never change ownership
SEC_INVARIANT = +tpug ;               # Directories that should never change permission or
ownership
SIG_LOW       = 33 ;                  # Non-critical files that are of minimal security impact
SIG_MED       = 66 ;                  # Non-critical files that are of significant security impact
SIG_HI        = 100 ;                 # Critical files that are significant points of
vulnerability


# Tripwire Binaries
(
  rulename = "Tripwire Binaries",
  severity = $(SIG_HI), emailto=root
)
{
  $(TWBIN)/siggen                       -> $(SEC_BIN) ;
  $(TWBIN)/tripwire                     -> $(SEC_BIN) ;
  $(TWBIN)/twadmin                      -> $(SEC_BIN) ;
  $(TWBIN)/twprint                      -> $(SEC_BIN) ;
}

# Tripwire Data Files - Configuration Files, Policy Files, Keys, Reports, Databases
(
  rulename = "Tripwire Data Files",
  severity = $(SIG_HI), emailto=root
)
{
  # NOTE: We remove the inode attribute because when Tripwire creates a backup,
  # it does so by renaming the old file and creating a new one (which will
  # have a new inode number).  Inode is left turned on for keys, which shouldn't
  # ever change.

  # NOTE: The first integrity check triggers this rule and each integrity check
  # afterward triggers this rule until a database update is run, since the
  # database file does not exist before that point.

  $(TWDB)                               -> $(SEC_CONFIG) -i ;
  $(TWPOL)/tw.pol                       -> $(SEC_BIN) -i ;
  $(TWPOL)/tw.cfg                       -> $(SEC_BIN) -i ;
  $(TWPOL)/twcfg.txt                    -> $(SEC_BIN) ;
  $(TWPOL)/twpol.txt                    -> $(SEC_BIN) ;
  $(TWLKEY)/$(HOSTNAME)-local.key       -> $(SEC_BIN) ;
  $(TWSKEY)/site.key                    -> $(SEC_BIN) ;

  #don't scan the individual reports
  $(TWREPORT)                           -> $(SEC_CONFIG) (recurse=0) ;
}

# Commonly accessed directories that should remain static with regards to owner and group
(
  rulename = "Invariant Directories",
  severity = $(SIG_MED), emailto=root
)
{
  /                                     -> $(SEC_INVARIANT) (recurse = false) ;
  /home                                 -> $(SEC_INVARIANT) (recurse = false) ;
}

#
# First, root's "home"
#

(
  rulename = "Root's home",
  severity = $(SIG_HI)
)
{
  # /.rhosts                          -> $(SEC_CRIT) ;
  /.profile                           -> $(SEC_CRIT) ;
  /.cshrc                             -> $(SEC_CRIT) ;
  /.login                             -> $(SEC_CRIT) ;
  # /.exrc                            -> $(SEC_CRIT) ;
  # /.logout                          -> $(SEC_CRIT) ;
  # /.forward                         -> $(SEC_CRIT) ;
  /root                               -> $(SEC_CRIT) (recurse = true) ;
  !/root/.history ;
  !/root/.bash_history ;
  # !/root/.lsof_SYSTEM_NAME ;        # Uncomment if lsof is installed
}
```

```
#
# FreeBSD Kernel
#

(
  rulename = "FreeBSD Kernel",
  severity = $(SIG_HI), emailto=root
)
{
  /kernel                             -> $(SEC_CRIT) ;
  /kernel.old                         -> $(SEC_CRIT) ;
  /kernel.GENERIC                     -> $(SEC_CRIT) ;
}


#
# FreeBSD Modules
#

(
  rulename = "FreeBSD Modules",
  severity = $(SIG_HI), emailto=root
)
{
  /modules                            -> $(SEC_CRIT) (recurse = true) ;
  /modules.old                        -> $(SEC_CRIT) (recurse = true) ;
  # /lkm                              -> $(SEC_CRIT) (recurse = true) ; # uncomment if using lkm
kld
}


#
# System Administration Programs
#

(
  rulename = "System Administration Programs",
  severity = $(SIG_HI), emailto=root
)
{
  /sbin                               -> $(SEC_CRIT) (recurse = true) ;
  /usr/sbin                           -> $(SEC_CRIT) (recurse = true) ;
}


#
# User Utilities
#

(
  rulename = "User Utilities",
  severity = $(SIG_HI), emailto=root
)
{
  /bin                                -> $(SEC_CRIT) (recurse = true) ;
  /usr/bin                            -> $(SEC_CRIT) (recurse = true) ;
}


#
# /dev
#

(
  rulename = "/dev",
  severity = $(SIG_HI), emailto=root
)
{
  /dev                                -> $(Device) (recurse = true) ;
  !/dev/vga ;
  !/dev/dri ;
  /dev/console                        -> $(SEC_TTY) ;
  /dev/ttyv0                          -> $(SEC_TTY) ;
  /dev/ttyv1                          -> $(SEC_TTY) ;
  /dev/ttyv2                          -> $(SEC_TTY) ;
  /dev/ttyv3                          -> $(SEC_TTY) ;
  /dev/ttyv4                          -> $(SEC_TTY) ;
  /dev/ttyv5                          -> $(SEC_TTY) ;
  /dev/ttyv6                          -> $(SEC_TTY) ;
  /dev/ttyv7                          -> $(SEC_TTY) ;
  /dev/ttyp0                          -> $(SEC_TTY) ;
  /dev/ttyp1                          -> $(SEC_TTY) ;
  /dev/ttyp2                          -> $(SEC_TTY) ;
```

```
  /dev/ttyp3                            -> $(SEC_TTY) ;
  /dev/ttyp4                            -> $(SEC_TTY) ;
  /dev/ttyp5                            -> $(SEC_TTY) ;
  /dev/ttyp6                            -> $(SEC_TTY) ;
  /dev/ttyp7                            -> $(SEC_TTY) ;
  /dev/ttyp8                            -> $(SEC_TTY) ;
  /dev/ttyp9                            -> $(SEC_TTY) ;
  /dev/ttypa                            -> $(SEC_TTY) ;
  /dev/ttypb                            -> $(SEC_TTY) ;
  /dev/ttypc                            -> $(SEC_TTY) ;
  /dev/ttypd                            -> $(SEC_TTY) ;
  /dev/ttype                            -> $(SEC_TTY) ;
  /dev/ttypf                            -> $(SEC_TTY) ;
  /dev/ttypg                            -> $(SEC_TTY) ;
  /dev/ttyph                            -> $(SEC_TTY) ;
  /dev/ttypi                            -> $(SEC_TTY) ;
  /dev/ttypj                            -> $(SEC_TTY) ;
  /dev/ttypl                            -> $(SEC_TTY) ;
  /dev/ttypm                            -> $(SEC_TTY) ;
  /dev/ttypn                            -> $(SEC_TTY) ;
  /dev/ttypo                            -> $(SEC_TTY) ;
  /dev/ttypp                            -> $(SEC_TTY) ;
  /dev/ttypq                            -> $(SEC_TTY) ;
  /dev/ttypr                            -> $(SEC_TTY) ;
  /dev/ttyps                            -> $(SEC_TTY) ;
  /dev/ttypt                            -> $(SEC_TTY) ;
  /dev/ttypu                            -> $(SEC_TTY) ;
  /dev/ttypv                            -> $(SEC_TTY) ;
  /dev/cuaa0                            -> $(SEC_TTY) ;        # modem
}


#
# /etc
#

(
  rulename = "/etc",
  severity = $(SIG_HI), emailto=root
)
{
  /etc                                  -> $(SEC_CRIT) (recurse = true) ;
  # /etc/mail/aliases                   -> $(SEC_CONFIG) ;
  /etc/dumpdates                        -> $(SEC_CONFIG) ;
  /etc/motd                             -> $(SEC_CONFIG) ;
  !/etc/ppp/connect-errors ;
  /etc/skeykeys                                 -> $(SEC_CONFIG) ;
  # Uncomment the following 4 lines if your password file does not change
  # /etc/passwd                                 -> $(SEC_CONFIG) ;
  # /etc/master.passwd                  -> $(SEC_CONFIG) ;
  # /etc/pwd.db                                 -> $(SEC_CONFIG) ;
  # /etc/spwd.db                        -> $(SEC_CONFIG) ;
}


#
# Copatibility (Linux)
#

(
  rulename = "Linux Compatibility",
  severity = $(SIG_HI), emailto=root
)
{
  /compat                               -> $(SEC_CRIT) (recurse = true) ;
#
# Uncomment the following if Linux compatibility is used.  Replace
# HOSTNAME1 and HOSTNAME2 with the hosts that have Linux emulation port
# installed.
#
#@@ifhost HOSTNAME1 || HOSTNAME2
#  /compat/linux/etc                   -> $(SEC_INVARIANT) (recurse = false) ;
#  /compat/linux/etc/X11                       -> $(SEC_CONFIG) (recurse = true) ;
#  /compat/linux/etc/pam.d             -> $(SEC_CONFIG) (recurse = true) ;
#  /compat/linux/etc/profile.d                 -> $(SEC_CONFIG) (recurse = true) ;
#  /compat/linux/etc/real              -> $(SEC_CONFIG) (recurse = true) ;
#  /compat/linux/etc/bashrc            -> $(SEC_CONFIG) ;
#  /compat/linux/etc/csh.login                 -> $(SEC_CONFIG) ;
#  /compat/linux/etc/host.conf                 -> $(SEC_CONFIG) ;
#  /compat/linux/etc/hosts.allow               -> $(SEC_CONFIG) ;
#  /compat/linux/etc/hosts.deny                -> $(SEC_CONFIG) ;
#  /compat/linux/etc/info-dir          -> $(SEC_CONFIG) ;
#  /compat/linux/etc/inputrc           -> $(SEC_CONFIG) ;
#  /compat/linux/etc/ld.so.conf                -> $(SEC_CONFIG) ;
```

```
#  /compat/linux/etc/nsswitch.conf    -> $(SEC_CONFIG) ;
#  /compat/linux/etc/profile          -> $(SEC_CONFIG) ;
#  /compat/linux/etc/redhat-release   -> $(SEC_CONFIG) ;
#  /compat/linux/etc/rpc                     -> $(SEC_CONFIG) ;
#  /compat/linux/etc/securetty               -> $(SEC_CONFIG) ;
#  /compat/linux/etc/shells           -> $(SEC_CONFIG) ;
#  /compat/linux/etc/termcap          -> $(SEC_CONFIG) ;
#  /compat/linux/etc/yp.conf          -> $(SEC_CONFIG) ;
#  !/compat/linux/etc/ld.so.cache ;
#  !/compat/linux/var/spool/mail ;
#@@endif
}


#
# Libraries, include files, and other system files
#

(
  rulename = "Libraries, include files, and other system files",
  severity = $(SIG_HI), emailto=root
)
{
  /usr/include                       -> $(SEC_CRIT) (recurse = true) ;
  /usr/lib                           -> $(SEC_CRIT) (recurse = true) ;
  /usr/libdata                       -> $(SEC_CRIT) (recurse = true) ;
  /usr/libexec                       -> $(SEC_CRIT) (recurse = true) ;
  /usr/share                         -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man                     -> $(SEC_CONFIG) ;
  !/usr/share/man/whatis ;
  !/usr/share/man/.glimpse_filenames ;
  !/usr/share/man/.glimpse_filenames_index ;
  !/usr/share/man/.glimpse_filetimes ;
  !/usr/share/man/.glimpse_filters ;
  !/usr/share/man/.glimpse_index ;
  !/usr/share/man/.glimpse_messages ;
  !/usr/share/man/.glimpse_partitions ;
  !/usr/share/man/.glimpse_statistics ;
  !/usr/share/man/.glimpse_turbo ;
  /usr/share/man/man1                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man2                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man3                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man4                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man5                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man6                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man7                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man8                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/man9                -> $(SEC_CRIT) (recurse = true) ;
  /usr/share/man/mann                -> $(SEC_CRIT) (recurse = true) ;
  ! /usr/share/man/cat1 ;
  ! /usr/share/man/cat2 ;
  ! /usr/share/man/cat3 ;
  ! /usr/share/man/cat4 ;
  ! /usr/share/man/cat5 ;
  ! /usr/share/man/cat6 ;
  ! /usr/share/man/cat7 ;
  ! /usr/share/man/cat8 ;
  ! /usr/share/man/cat9 ;
  ! /usr/share/man/catl ;
  ! /usr/share/man/catn ;
  /usr/share/perl/man                -> $(SEC_CONFIG) ;
  !/usr/share/perl/man/whatis ;
  !/usr/share/perl/man/.glimpse_filenames ;
  !/usr/share/perl/man/.glimpse_filenames_index ;
  !/usr/share/perl/man/.glimpse_filetimes ;
  !/usr/share/perl/man/.glimpse_filters ;
  !/usr/share/perl/man/.glimpse_index ;
  !/usr/share/perl/man/.glimpse_messages ;
  !/usr/share/perl/man/.glimpse_partitions ;
  !/usr/share/perl/man/.glimpse_statistics ;
  !/usr/share/perl/man/.glimpse_turbo ;
  /usr/share/perl/man/man3           -> $(SEC_CRIT) (recurse = true) ;
  ! /usr/share/perl/man/cat3 ;
  /usr/local/lib/perl5/5.00503/man   -> $(SEC_CONFIG) ;
  ! /usr/local/lib/perl5/5.00503/man/whatis ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_filters ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_filetimes ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_messages ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_statistics ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_index ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_turbo ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_partitions ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_filenames ;
  ! /usr/local/lib/perl5/5.00503/man/.glimpse_filenames_index ;
```

```
  /usr/local/lib/perl5/5.00503/man/man3             -> $(SEC_CRIT) (recurse = true) ;
  ! /usr/local/lib/perl5/5.00503/man/cat3 ;
}


#
# X11R6
#

#(
#  rulename = "X11R6",
#  severity = $(SIG_HI)
#)
#{
#  /usr/X11R6                          -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/lib/X11/xdm              -> $(SEC_CONFIG) (recurse = true) ;
#  !/usr/X11R6/lib/X11/xdm/xdm-errors ;
#  !/usr/X11R6/lib/X11/xdm/authdir/authfiles ;
#  !/usr/X11R6/lib/X11/xdm/xdm-pid ;
#  /usr/X11R6/lib/X11/xkb/compiled     -> $(SEC_CONFIG) (recurse = true) ;
#  /usr/X11R6/man                      -> $(SEC_CONFIG) ;
#  !/usr/X11R6/man/whatis ;
#  !/usr/X11R6/man/.glimpse_filenames ;
#  !/usr/X11R6/man/.glimpse_filenames_index ;
#  !/usr/X11R6/man/.glimpse_filetimes ;
#  !/usr/X11R6/man/.glimpse_filters ;
#  !/usr/X11R6/man/.glimpse_index ;
#  !/usr/X11R6/man/.glimpse_messages ;
#  !/usr/X11R6/man/.glimpse_partitions ;
#  !/usr/X11R6/man/.glimpse_statistics ;
#  !/usr/X11R6/man/.glimpse_turbo ;
#  /usr/X11R6/man/man1                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man2                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man3                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man4                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man5                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man6                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man7                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man8                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/man9                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/manl                 -> $(SEC_CRIT) (recurse = true) ;
#  /usr/X11R6/man/mann                 -> $(SEC_CRIT) (recurse = true) ;
#  ! /usr/X11R6/man/cat1 ;
#  ! /usr/X11R6/man/cat2 ;
#  ! /usr/X11R6/man/cat3 ;
#  ! /usr/X11R6/man/cat4 ;
#  ! /usr/X11R6/man/cat5 ;
#  ! /usr/X11R6/man/cat6 ;
#  ! /usr/X11R6/man/cat7 ;
#  ! /usr/X11R6/man/cat8 ;
#  ! /usr/X11R6/man/cat9 ;
#  ! /usr/X11R6/man/catl ;
#  ! /usr/X11R6/man/catn ;
#}

#
# sources
#

(
  rulename = "Sources",
  severity = $(SIG_HI), emailto=root
)
{
  /usr/src                            -> $(SEC_CRIT) (recurse = true) ;
  /usr/src/sys/compile                -> $(SEC_CONFIG) (recurse = false) ;
}

#
# NIS
#

#(
#  rulename = "NIS",
#  severity = $(SIG_HI)
#)
#{
#  /var/yp                             -> $(SEC_CRIT) (recurse = true) ;
#  !/var/yp/binding ;
#}

#
# Temporary directories
#
```

```
(
  rulename = "Temporary directories",
  recurse = false,
  severity = $(SIG_LOW), emailto=root
)
{
  /usr/tmp                              -> $(SEC_INVARIANT) ;
  /var/tmp                              -> $(SEC_INVARIANT) ;
  /var/preserve                         -> $(SEC_INVARIANT) ;
  /tmp                                  -> $(SEC_INVARIANT) ;
}

#
# Local files
#

(
  rulename = "Local files",
  severity = $(SIG_MED), emailto=root
)
{
  /usr/local/bin                     -> $(SEC_BIN) (recurse = true) ;
  /usr/local/sbin                    -> $(SEC_BIN) (recurse = true) ;
  /usr/local/etc                     -> $(SEC_BIN) (recurse = true) ;
  /usr/local/lib                     -> $(SEC_BIN) (recurse = true ) ;
  /usr/local/libexec                 -> $(SEC_BIN) (recurse = true ) ;
  /usr/local/share                   -> $(SEC_BIN) (recurse = true ) ;
  /usr/local/man                     -> $(SEC_CONFIG) ;
  !/usr/local/man/whatis ;
  !/usr/local/man/.glimpse_filenames ;
  !/usr/local/man/.glimpse_filenames_index ;
  !/usr/local/man/.glimpse_filetimes ;
  !/usr/local/man/.glimpse_filters ;
  !/usr/local/man/.glimpse_index ;
  !/usr/local/man/.glimpse_messages ;
  !/usr/local/man/.glimpse_partitions ;
  !/usr/local/man/.glimpse_statistics ;
  !/usr/local/man/.glimpse_turbo ;
  /usr/local/man/man1                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man2                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man3                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man4                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man5                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man6                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man7                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man8                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/man9                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/manl                -> $(SEC_CRIT) (recurse = true) ;
  /usr/local/man/mann                -> $(SEC_CRIT) (recurse = true) ;
  ! /usr/local/man/cat1 ;
  ! /usr/local/man/cat2 ;
  ! /usr/local/man/cat3 ;
  ! /usr/local/man/cat4 ;
  ! /usr/local/man/cat5 ;
  ! /usr/local/man/cat6 ;
  ! /usr/local/man/cat7 ;
  ! /usr/local/man/cat8 ;
  ! /usr/local/man/cat9 ;
  ! /usr/local/man/catl ;
  ! /usr/local/man/catn ;
#  /usr/local/krb5                    -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man                -> $(SEC_CONFIG) ;
#  !/usr/local/krb5/man/whatis ;
#  !/usr/local/krb5/man/.glimpse_filenames ;
#  !/usr/local/krb5/man/.glimpse_filenames_index ;
#  !/usr/local/krb5/man/.glimpse_filetimes ;
#  !/usr/local/krb5/man/.glimpse_filters ;
#  !/usr/local/krb5/man/.glimpse_index ;
#  !/usr/local/krb5/man/.glimpse_messages ;
#  !/usr/local/krb5/man/.glimpse_partitions ;
#  !/usr/local/krb5/man/.glimpse_statistics ;
#  !/usr/local/krb5/man/.glimpse_turbo ;
#  /usr/local/krb5/man/man1                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man2                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man3                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man4                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man5                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man6                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man7                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man8                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/man9                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/manl                -> $(SEC_CRIT) (recurse = true) ;
#  /usr/local/krb5/man/mann                -> $(SEC_CRIT) (recurse = true) ;
#  ! /usr/local/krb5/man/cat1 ;
```

```
#  ! /usr/local/krb5/man/cat2 ;
#  ! /usr/local/krb5/man/cat3 ;
#  ! /usr/local/krb5/man/cat4 ;
#  ! /usr/local/krb5/man/cat5 ;
#  ! /usr/local/krb5/man/cat6 ;
#  ! /usr/local/krb5/man/cat7 ;
#  ! /usr/local/krb5/man/cat8 ;
#  ! /usr/local/krb5/man/cat9 ;
#  ! /usr/local/krb5/man/catl ;
#  ! /usr/local/krb5/man/catn ;
#  /usr/local/www                    -> $(SEC_CONFIG) (recurse = true) ;
}


(
  rulename = "Security Control",
  severity = $(SIG_HI), emailto=root
)
{
  /etc/group                          -> $(SEC_CRIT) ;
  /etc/crontab                        -> $(SEC_CRIT) ;
}

#=============================================================================
#
# Copyright 2000 Tripwire, Inc. Tripwire is a registered trademark of Tripwire,
# Inc. in the United States and other countries. All rights reserved.
#
# FreeBSD is a registered trademark of the FreeBSD Project Inc.
#
# UNIX is a registered trademark of The Open Group.
#
#=============================================================================
#
# Permission is granted to make and distribute verbatim copies of this document
# provided the copyright notice and this permission notice are preserved on all
# copies.
#
# Permission is granted to copy and distribute modified versions of this
# document under the conditions for verbatim copying, provided that the entire
# resulting derived work is distributed under the terms of a permission notice
# identical to this one.
#
# Permission is granted to copy and distribute translations of this document
# into another language, under the above conditions for modified versions,
# except that this permission notice may be stated in a translation approved by
# Tripwire, Inc.
#
# DCM
```