Global Information Assurance Certification Paper

Name: Adam Crooke

Version: 1.9
Title: Securing Unix Step by Step - Securing a Solaris Web Server

Contents

Introduction

This paper details the installation, configuration and maintenance of an Internet web server. The paper discusses the platform (both hardware and software) followed by a risk analysis identifying areas of risk for this type of server. Following the identified risks: a step-by-step guide to installing, configuring, and most importantly securing the operating system and the web server software based upon the identified risks is presented.

Once the system has been secured it is necessary to keep the server up to date by fixing holes and vulnerabilities as they are found – this is described in the maintenance section as well as log checking and other regular maintenance tasks that need to be performed to keep the system secure.

The final section of the report verifies that the work preceding it is successful, and the system is secure to known vulnerabilities and hacking exploits. This is validated by such tools as Nessus and manual checking (e.g. try to write files, checking logs etc.)

1 System Description

The system used (see specification below) could be described as a typical system in both hardware and software terms, and could easily be found as a web server (both external and internal) within many corporate networks

1.1 Hardware

The hardware for the web server is:

Manufacturer	SUN (http://www.sun.com)
CPU	Netra T-1 (UltraSPARC-IIe 500MHz)
RAM	256MB
HDD	18GB

1.2 Software

The web server operating system is Solaris 2.6, which is still commonly used throughout organisations illustrating its popularity, and was chosen exactly for this purpose.

The web server software that is featured in this paper is Apache 1.3 (the latest stable release at time of writing). The reason for choosing Apache is to give a good indication of a typical web server on the Internet. According to the Netcraft[1] website: as of March 2003 - 63% of web sites on the Internet run the Apache web server software. This figure not only demonstrates its popularity with web administrators and designers, but also makes it a prominent target for hackers. Because of the visibility of Apache on the Internet - there are very well documented hacks and utilities for compromising Apache web servers, and as such it is vitally important to secure and keep the web server up to date with patches and fixes.

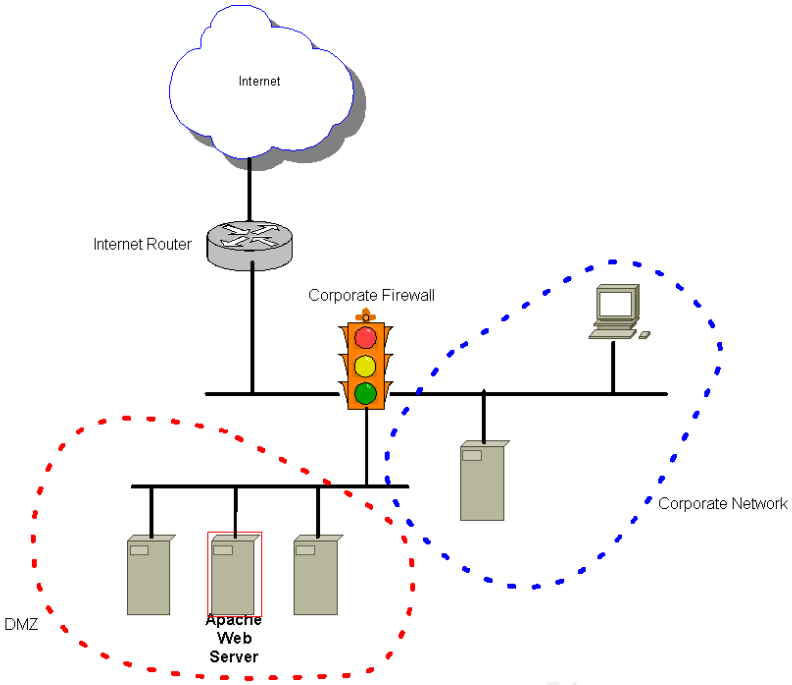
A summary of the software running on the web server:

Operating System	Solaris 2.6 (http://wwws.sun.com/software/solaris/sparc/index.html)
Web Server	Apache 1.3.27 (http://httpd.apache.org/download.cgi)

1.3 Topology

The network topology shown on the following page is a common one found throughout networks with a presence on the Internet (e.g. Public website, Email etc.). The Internet connection is presented through the border router that is connected on one side to an ISP and to a firewall on the other. The internal network is then split into two – one is the DMZ containing servers that provide services to the Internet such as our web server. The other part of the network (subnetted with VLAN's for example) is the internal LAN containing file servers, internal applications, and printers for example.

[1] Netcraft "March 2003 Web Server Survey"



1.4 Final System State

The final system is designed to be a publicly available (accessible from the public Internet) web server situated within the DMZ of a company. It is this function that the risk analysis and configuration check will be measured against. The web server will only listen to requests on the http port (80) and will be administered remotely by using Secure Shell (SSH).

2 Risk Analysis

When asking a layman what the Internet is, they may tell you that it is a collection of web pages. That is true; although web pages are a subset of the Internet - they are by far the biggest, alongside email. There are several core services that make up the majority of Internet traffic, web services is one of these core services, it is also, crucially one of the most tangible. Companies rapidly realise that they can use the Internet to increase their revenues and market share while providing an advertising and information portal 24 hours a day, 365 days a year.

It is important to note that these web services require the use of some of the other core Internet services (such as DNS) and as such these are equally important from a security perspective. However, for the purposes of this paper, only web services are examined.

The web server is accessible from two networks – this means that the threats from each network can and will be different. The first threat (and often the most under-estimated) is from the internal network. Internal threats can be that much more severe because attackers (such as disgruntled employees) may know company procedures and organisation, and use this to their advantage when attempting to compromise systems. The threats from the inside include all those from the outside and some more (due to administration from the inside) it is therefore an equal if not higher risk threat than an outside attack, and can produce the same result (e.g. web site defacement).

Because the server is vulnerable to the same inside attacks (and some) as it is to the outside the risk analysis breaks these down and deals with the outside threats first.

2.1 Risks from the Internet:

The web server is behind a fire-walled Internet connection, and as such initial access will only be granted to outsiders on TCP port 80. This helps to severely limit the vulnerability of the web server, but administrators should not be complacent. Many administrators believe because they are protected by a firewall that security is unnecessary or simply an afterthought. Many cases have proved that a firewall should not be the only method of protection. While firewalls block many connection attempts and attacks, they can only be as good as the rule base, and as a company running a web server at least one port needs to be open - already this is a risk. Therefore firewalls should form a part of the overall security strategy and this document details another important part of the overall strategy.

The only process open to the Internet is “httpd” (the Apache web server daemon) listening on port 80. This is necessary because it is the port that web browsers (by default) connect to, to retrieve web pages. The primary threat here is that there is some kind of vulnerability with the daemon (such as buffer overflow attacks). Just because Microsoft’s IIS has had wide publicity for

being inherently insecure this does not detract from the fact that despite its deserved reputation – Apache is not without its security issues. One such example of this (and a good example of how a firewall would not prevent this attack) is the recent chunk handling vulnerability[2]. It briefly describes an attack by sending certain HTTP requests that would allow an attacker to execute arbitrary code. New vulnerabilities are being discovered all the time and the only way to combat these threats is to keep the operating system and the Apache software up to date by patching and upgrading in a timely manner.

According to the SANS/FBI Top 20 Vulnerabilities: Apache is second in the top ten list of Unix vulnerabilities - one of the reasons is the attack such as those described above on the daemon itself. Another reason is the fact that Apache is feature rich and these services are simply left at default therefore they open a security hole that is simply not necessary. For example the SSL module that comes with Apache now has had reported security problems[3], if SSL is not necessary, as it is not in out case, then it should be simply turned off.

Apache offers popular services such as PHP and CGI. There have been many reported problems with CGI – certainly the hardest one to combat is simply bad programming. The SANS/FBI Top 20 Vulnerabilities website states "A hardened Apache configuration can still yield unauthorized access to data if CGI scripts are not themselves verified or database access controls not properly set". A poorly written CGI script can allow attackers to execute arbitrary code or even take control of the whole machine.

A further risk of any server accessible from the Internet is a denial of service attack. If a company running a web server sells their products on their website – a denial of service attack could cost millions in lost revenue. Protection from denial of service attacks is difficult but the risk can be minimised with good network design as well as well-configured firewalls. Not only this later in the document TCP parameters are modified to try to mitigate the risk of this type of attack.

2.2 Inside Risks

As well as the reported risks above, because of the necessity of remote internal administration of the server, the following risks are also apparent from an internal perspective.

For remote administration the most secure method for maintenance is a secure shell. A default installation of Solaris would feature services such as Telnet which means passwords are crossing the network in clear text. Using free software such as sniffers can mean an attacker can simply "sniff" the wire and grab root passwords. It is this reason that SSH is used for remote administration. The fact that this port is listening and a daemon runs on it is a

[2] CERT Advisory CA-2002-17
[3] CERT Advisory CA-2002-27

potential risk. There have been many documented attacks on SSH and although measures will be taken to ensure the security of SSH, it is again very important to ensure proper configuration and patching is performed. SSH should also only be available to administrators who need to access the box to perform maintenance and configuration.

Password access to the box brings risks. Good password enforcement is essential to mitigating the risk of using passwords. Many people set their password as their name, birth date, wife's name or something just as obvious. An attacker could then gain access to the box and launch further attacks. If the firewall and router were configured correctly in our network for anti-spoofing, then this risk should not be apparent from outsiders spoofing an internal IP address. Access should then only be granted to web server administrators who need to maintain and configure the server. Not only this but these administrators should be educated on good password management and a clear policy should be in place outlining the use of passwords.

Backup and staging is important for a critical server such as a web server. Whether the server is attacked or an innocent configuration change renders the system insecure or unusable it is important to establish backup routines, this is a risk apparent to all systems and is a risk that should be identified no matter the application being used.

Another risk with any server is physical access. If the box is not secured physically then an attacker could simply remove the hard drive and access all the data on the disk at their leisure. Adequate physical measures need to be taken to mitigate the risk of physical compromise of the server. The only people who need to physically access the server are a limited number of server administrators.

Each risk that has been identified in this section will be addresses by configuration steps in the next section.

3 Step by step guide

This section details the step-by-step procedures to building and configuring the secure web server.

3.1 Caveats

All configuration steps preformed in this, and subsequent sections are performed as root via the console, and as such care should be taken when performing the following steps.

There is no need to reboot after every step – only reboot when the steps explicitly mention. Failure to do so can result in errors during and after reboot.

The web server contains static web pages – it does not serve dynamic pages, nor does it offer ssl capabilities.

3.2 Operating System Installation

The operating system was installed using CD's obtained from Sun Microsystems. It is important to ascertain the integrity of these CD's to ensure that they have not been tampered with. For example, when downloading any operating system, updates, patches, or applications ensure the checksum is the same; there have been several distributions of programs that have been Trojan Horses.

Before turning on the machine ensure that the only cables plugged in is the power and the console cable (unless you wish to use a keyboard and monitor directly connected to the box). Do not plug a network cable into the NIC. Boot the machine from the Solaris CD-ROM disk 1. To do this I was given a machine without knowing the root password so I began to boot the machine and issued a break (STOP-A on a Solaris keyboard) to bring the system to the ok prompt. I then issued the command:

ok boot cdrom

This command rebooted the machine and started a boot from the Solaris 8 CD. After a short while the installation program will ask for language settings and the type of terminal being used. After these steps the installation program will display a short introduction – press F2 to continue.

Next the program needs to identify the system. It will ask if the system is networked – answer yes, although remember not to plug the network card into the network yet. Answer the following questions regarding the network as below:

Use DHCP	No
Primary Network Interface	eri0
Hostname	acmeweb (although any hostname can be used)

IP Address	192.168.100.10
System part of a subnet	Yes
Netmask	255.255.255.0
Enable Ipv6	No

The user should then be presented with a summary of the networking information, which should like below:

> Confirm Information

> Confirm the following information. If it is correct, press F2;
to change any information, press F4.

Networked: Yes
Use DHCP: No
Primary network interface: eri0
Host name: acmeweb
IP address: 192.168.100.10
System part of a subnet: Yes
Netmask: 255.255.255.0
Enable IPv6: No

F2_Continue F4_Change F6_Help

Press F2. When prompted for security – select standard UNIX security. After confirmation, the install program will prompt for naming – select only DNS and press F2. To continue through the naming process use the value below:

Domain Name	acme.com (although anything can be used for this demonstration.
DNS IP	192.168.100.11
DNS Search List	None

The user should then be presented with a summary of the networking information, which should like below:

> Confirm Information

> Confirm the following information. If it is correct, press F 2;
to change any information, press F4.

Name service: DNS
Domain name: acme.com
Server address(es): 192.168.100.11

F2_Continue F4_Change F6_Help

Press F2. After a moment a naming error message will appear:

Unable to find an address entry for acmeweb with the specified DNS configuration.

This is because the network card is not attached. Simply select No when prompted to enter new name service information.

After entering timezone information and time the next screen will appear.

```
_ Solaris Interactive Installation _____________________________

  This system is upgradable, so there are two ways to install the Solaris
  software.

  The Upgrade option updates the Solaris software to the new release, saving
  as many modifications to the previous version of Solaris software as
  possible.  Back up the system before using the Upgrade option.

  The Initial option overwrites the system disks with the new version of
  Solaris software.  This option allows you to preserve any existing file
  systems.  Back up any modifications made to the previous version of Solaris
  software before starting the Initial option.

  After you select an option and complete the tasks that follow, a summary of
  your actions will be displayed.  If you want to install the system with a
  Flash archive, select Initial.

_____ _____________________________________________________________

    F2_Upgrade    F4_Initial    F5_Exit    F6_Help
```

This message is displayed because there was a previous version of the operating system installed on the disk. We want to overwrite this with a new version of the operating system because we will then be sure exactly what is, and is not installed on the system. To install a new version of the software select F4 Initial. The next screen will prompt for installation type, select F2 Standard. After selecting the necessary geographic region(s) the installation program will prompt for the type of software to install.

An important step when considering security for a server is what it is being used for and therefore which services will need to run. A best practice is to install a minimum amount of software and services and then build upon them to enable the server to perform its job. To build our server securely we need to select just core system support.

```
> Select Software
_____________________________ _____________________________

  Select the Solaris software to install on the system.

  NOTE: After selecting a software group, you can add or remove software by
  customizing it. However, this requires understanding of software
  dependencies and how Solaris software is packaged. The software groups
  displaying 64-bit contain 64-bit support.

    [ ]  Entire Distribution plus OEM support 64 -bit  1280.00 MB
    [ ]  Entire Distribution 64 -bit ..... 1245.00 MB
    [ ]  Developer System Support 64-bit ..... 1193.00 MB
    [ ]  End User System Support 64 -bit ..... 771.00 MB
    [X]  Core System Support 64 -bit ..... 270.00 MB (F4 to Customi
```

ze)

F2_Continue F3_Go Back F4_Customize F5_Exit F6_Help

It is possible to customise the packages installed further but for educational purposes we will turn off any more unwanted services once the base operating system has been installed. Choose “Core System Support” and press F2.

The next step in the installation asks if existing data on the drives needs to be kept – as stated previously this is because there was an existing operating system on the servers already. We do not wish to keep this data and so select F2 Continue. The next step is to configure the layout of the file systems, the installer program can automatically set these, however we want to set these manually to give us granular control over the operating system and the file system usage. Choose F4 to customise. The screenshot below denotes how the disk was partitioned:

> Customize Disk: c1t0d0

Boot Device: c1t0d0s0

Entry:		*Recommended: MB*	*Minimum: MB*
Slice	*Mount Point*	*Size (MB)*	
0	*/*	*1000*	
1	*swap*	*501*	
2	*/usr*	*4001*	
3	*/opt*	*2001*	
4	*/var*	*5001*	
5	*/export/home*	*4762*	
6		*0*	
7		*0*	

Capacity: 17269 MB
Allocated: 17266 MB
Rounding Error: 3 MB
Free: 0 MB

F2_OK F4_Options F5_Cancel F6_Help

The swap file is 500MB so it is twice the size of the amount of RAM in the machine. The /*var* partition needs to be quite big because it is the dynamic partition where all the logs will be stored and we want enough space left on the drive for logs to be written otherwise the machine may crash. The home directory partition is large because this is where the website pages will be stored.

Choose F2 to continue. The installation program will then ask if you want to mount remote file systems. We do not want to do this as using services such as NFS can present security holes – the server does not require these services so we should not turn them on. After choosing continue, a summary screen will appear, it should look similar to this:

> Profile

The information shown below is your profile for installing Solaris software.
It reflects the choices you've made on previous screens.

==========

Installation Option: Initial
Boot Device: c1t0d0
Client Services: None

Software: Solaris 8, Core System Support 64 -bit

File System and Disk Layout: / *c1t0d0s0 1000 MB*
swa p *c1t0d0s1 501 MB*
/usr *c1t0d0s2 4001 MB*
/opt *c1t0d0s3 2001 MB*
/var *c1t0d0s4 5001 MB*
/export/home *c1t0d0s5 4762 MB*

F2_Continue F4_Change F5_Exit F6_Help

After choosing F2 to continue, accept auto reboot and press F2 to begin the installation. After the installation has finished the system will reboot. Depending on the speed of the machine the installation may take some time.

The first step after login with the root account via the console is to set a password. This should be done carefully as the root account has full access to the system. To do this I executed the command:

*# **passwd***
passwd: Changing password for root
New password:
Re-enter new password:
passwd (SYSTEM): passwd successfully changed for root
#

3.3 Backup Important Configuration Files

Before making any changes to the system it is important to back up system files beforehand. To do this execute the following at the command line:

***for file in /etc/ftpusers /etc/hosts.equiv /etc/inittab ***
***/etc/issue /etc/.login /etc/motd /etc/pam.conf ***
***/etc/passwd /etc/profile /etc/rmmount.conf ***
***/etc/shadow /etc/shells /etc/syslog.conf /etc/system ***
***/etc/vfstab /etc/default/cron /etc/default/ftpd ***
***/etc/default/inetinit /etc/default/init ***
***/etc/default/login /etc/default/sendmail ***
***/etc/default/telnetd /etc/inet/inetd.conf ***
***/etc/dfs/dfstab /etc/ssh/ssh*_config /.rhosts ***

***/.shosts /etc/cron.d/*.allow /etc/cron.d/*.deny ***
***/etc/dt/config/Xaccess /etc/dt/config/Xservers ***
***/etc/dt/config/*/sys.resources ***
/etc/dt/config/*/Xresources; do
[-f $file] && cp $file $file-preGCUX
done

This script was adapted from the Solaris Benchmark Document.

3.4 Additional Packages

There are several additional packages, which were not installed by default from the base install. The packages that need installing are:

SMCbinut	(Utilities for the GCC compiler)
SUNWntpr	NTP, (Root)
SUNWntpu	NTP, (Usr)
SUNWtoolx	programming Tools (64-bit)
SUNWgzip	GNU Zip compression utility
SUNWzlib	The Zip compression library
SUNWpl5u	Perl

To install these packages the Solaris CD 1 was inserted and the following command executed to mount the cdrom:

mount -r -F hsfs /dev/dsk/c0t0d0s0 /mnt

Execute the “pkgadd –d” command on each package (the gzip utlitiy is on the second CD. To do this unmount the first CD, eject and insert the second CD and mount it as before.

The GNU C compiler was also installed in readiness for Apache installation later in the document. The most recent version can be downloaded from www.sunfreeware.com. This was installed by downloading to the “/tmp” directory and “untarring” the file:

tar –xvf gcc*

Again the package was installed by executing the command:

pkgadd –d gcc*

3.5 Operating System Patches

As vulnerabilities, holes, and bugs are discovered in the operating system and its packages a patch is released. Therefore to ensure that each application is up to date it is necessary to patch the system. The most recent patch cluster was downloaded from:
ftp://sunsolve.sun.com/pub/patches/8_Recommended.zip.

After moving the file to /tmp the following commands were executed to install the patch cluster:

cd /tmp
unzip -qq 8_Recommended.zip
cd 8_Recommended
./install_cluster –q

The "–q" switches mean quiet which suppress the masses of information and prompts that are generated when unzipping and installing the file. The status that is displayed should look like this:

Patch cluster install script for Solaris 8 Recommended

Determining if sufficient save space exists...
Sufficient save space exists, continuing...
Installing patches located in /tmp/8_Recommended
Using patch_order file for patch installation sequence
Installing 110380-04...
Installing 110934-13...

Each patch number will appear as it is being installed. Some patch installations will generate errors. In this case some patches return with error code 8 – this means that the patch being installed applies to an operating system package, which is not installed on the machine. Some patches will return with error code 2, which means that the patch is already installed on the system.

A good way to ensure the system is up to date is to install and configure Sun's Solaris Patch Manager (http://www.sun.com/service/support/sw_only/patchmanager.html), which automatically checks for updated patches and downloads them accordingly. This saves the administrative burden and ensures that the system is constantly up to date.

To complete the patch installation a reboot is required.

3.6 Third party secure applications

3.6.1 Open SSH

Open SSH was downloaded from http://www.sunfreeware.com/openssh.html. This package includes all dependencies (e.g. OpenSSL, more information obtaining the required dependant packages and installing them are on http://www.sunfreeware.com/openssh26-7.html). After uncompressing the package it was installed with the command below:

pkgadd -d Open* all

SSH can use privilege separation which means should root access be needed a separate process is spawned – this can help prevent privilege escalation by containing corruption to an unprivileged process. To do this, perform these steps:

```
mkdir /var/empty
chown root:sys /var/empty
chmod 755 /var/empty
groupadd sshd
useradd -g sshd -c 'sshd privsep' -d /var/empty -s /bin/false sshd
```

We can now use TCP Wrappers to control network access (using the hosts files – see next section). We add the network address of the management LAN or internal network (whichever network the box will be administered from). To do this execute the command:

```
echo sshd:ALL > /etc/hosts.deny
echo sshd: 192.168.10.0/255.255.255.0, 192.168.11.0/255.255.255.0 >  /etc/hosts.allow
```

This means that only machines with IP addresses from our local network can access this web server via SSH.

Every SSH server has a key that is used during the exchange process to setup an encrypted tunnel between the SSH server and client. To do this execute the following commands:

```
ssh-keygen -t rsa1 -f /usr/local/etc/ssh_host_key -N ""
ssh-keygen -t dsa -f /usr/local/etc/ssh_host_dsa_key -N ""
ssh-keygen -t rsa -f /usr/local/etc/ssh_host_rsa_key -N ""
```

After this step has been performed the startup script needs to be configured to allow the daemon to run each time the server is turned on. The following script is taken from (http://www.sunfreeware.com/openssh26-7.html) this script was put in the file “/etc/init.d/sshd.server”:

```
#!/bin/sh

case "$1" in
'start')
        if [ -x /usr/local/sbin/sshd ]; then
                echo "Starting the secure shell daemon"
                /usr/local/sbin/sshd &
        fi
        ;;

'stop')
        echo "Stopping the secure shell daemon "
        pkill -TERM sshd
        ;;
*)
        echo "Usage: /etc/init.d/sshd { start | stop }"
        ;;
esac
exit 0
```

The permissions and script needs linking to the correct run-level. To do this execute the following commands:

```
chown root /etc/init.d/sshd
```

chgrp sys /etc/init.d/sshd
chmod 555 /etc/init.d/sshd
ln -s /etc/init.d/sshd /etc/rc2.d/S98sshd

This completes the installation of SSH. After a reboot the SSH daemon will be running.

3.6.2 TCP Wrappers

TCP Wrappers was downloaded from: ftp://ftp.sunfreeware.com/pub/freeware/tcp_wrappers_ipv6-7.6-sol8-sparc-local.gz.

To unzip the file the utility gunzip or similar is required, this is not installed with the core install of Solaris so it can either be installed (download from the sunfreeware.com ftp site) or can be uncompressed on another machine, e.g. Windows using WinZip.

The package was installed by executing the following command:

pkgadd –d tcp_wrappers.1-7.6-sol8-sparc-local all

After installation is complete configuration is done via two files. The "hosts.allow" file contains all networks that are allowed to connect to a daemon specified in "inetd.conf" and the "hosts.deny" file contains hosts that should not be allowed to connect to a particular daemon. All other hosts that do not match the rules are allowed.

"hosts.allow" file:

ALL: 192.168.10.0/255.255.255.0, 192.168.11.0/255.255.255.0

These networks are the internal networks that the box is administered from.

"hosts.deny" file:

ALL: ALL

This denies all other traffic to any daemon specified in "inetd.conf".

Later in the document the "inetd.conf" file will be disabled and so there will be no inet daemons running for TCP Wrappers to protect, however this is a useful exercise should "inetd.conf" file be later reactivated. To allow TCP wrappers to protect each daemon "inetd.conf" needs to be reconfigured. Each time a connection is made the inet daemon checks the configuration and spawns the necessary daemon depending on the port number of the connection. An example line from the "inetd.conf" is:

ftp stream tcp6 nowait root /usr/sbin/in.ftpd in.ftpd

Each line needs to be modified to spawn TCP Wrappers as opposed to the particular net daemon. This checks the access and will then spawn the

necessary daemon (such as FTP in this case). The new TCP Wrappers enabled “inetd.conf” file would look like this:

ftp stream tcp6 nowait root /usr/local/bin/tcpd /usr/sbin/in.ftpd

3.6.3 Other third party software

There is many other types of software to help with securing our Solaris box. A few of note are mentioned here. Tripwire (www.tripwire.com) is a program that creates a snapshot of the system (particularly the essential configuration files) and then periodically looks for unauthorised changes in these files. It does this by maintaining a database of hashes of each file that it is configured to “look after”.

Nessus (www.nessus.org) is a security auditing software and is a very powerful freeware product. This will be used in section 6 – Configuration Check.

3.7 Prevent Remote Root logins

The ability of remote users to connect to the machine as root is a security problem. As another layer of security we should prevent root logins from across the network. While users would still be able to issue an “su” command to gain super user, the auditing and logging would still show the original user that logged in, and would enable an audit trail for forensic purposes should it be necessary. Remote root logins are prevented by default and since this is a fresh install this will still be the case. To check the following line should be commented in the “/etc/default/login” file:

#CONSOLE=/dev/console

3.8 Limit number of login retries

The ability for an attacker to brute force attack the system is a popular way to break into a system. To negate the risk of this set the number of failed login retires to three as opposed to the default five. To do this make sure the” /etc/default/login” file has a line containing the following:

RETIRES=3

This means that the connection will drop after three failed login attempts and the attacker will have to reconnect. This does not prevent brute force attacks but it can slow the attacker down considerably.

3.9 Operating System Hardening

3.9.1 inetd

The “inetd.conf” file contains many services that are not needed for most machines; they are definitely not needed for this web server.

Services such as finger and echo are not needed; even services such as FTP and telnet are not necessary for our box because we are using SSH to administer it. The fact that this is a web server means that the only services we need to run is SSH and Apache (port 80).

The “inetd.conf” file now contains the configuration for all the services to run through TCP Wrappers but every line is commented, so the server now no longer responds to any services normally run through the inet daemon. As one final precaution the “inetd.conf” file is copied to “inetd.conf.original” and a new blank file placed as “inetd.conf”. This enables the administrator to make a quick check of the “inetd.conf” file periodically to ensure no services have been re-enabled.

mv inetd.conf inetd.conf.original
touch inetd.conf
chown root:sys inetd.conf
chmod 444 inetd.conf

As a precautionary measure the last two commands ensure that “inetd.conf” file has the correct permissions.

Since we are not using the inet daemon we can prevent it being started at boot time. This is in the configuration script called S72inetsvc in the run level 2 directory (rc2.d). The very last line starts the inet daemon – simply comment this out or remove the line completely:

#/usr/sbin/inetd -s &

3.9.2 Disabling Modem login

We do not require any logins through the modems or terminals, only the console and SSH logins should be allowed. To prevent a login by these other means, remove the line:

sc:234:respawn:/usr/lib/saf/sac -t 300

This line is in the “inittab” file located in the /etc directory.

3.9.3 Unneeded Boot Services

Just like the default “inetd.conf” file, there are many services, which start at boot time that are not needed for the particular application of this server.

3.9.3.1 Syslog

The syslog daemon is inherently insecure because there is no authentication built in. As a result the machine as currently configured is susceptible to attacks form the internal network. An attacker sending traffic to the syslog daemon listening on UDP port 514 can fill the logs on the disk or launch a denial of service. Since this server is not a log server we can disable the daemon from listening on the network.

The script to start the syslog deamon is called syslog within the /etc/inet.d directory. It is linked by default to /etc/rc2.d/S74syslog. The section that starts the syslog daemon within the script contains the line:

/usr/sbin/syslogd >/dev/msglog 2>&1 &

We still require the syslog daemon to run (for logging purposes) but we do not want it to listen to requests from the network. By running the daemon with the "-t" flag means the syslog daemon will no longer listen on the network. The line should now look like this:

/usr/sbin/syslogd –t >/dev/msglog 2>&1 &

3.9.3.2 Sendmail

Even by just installing the core system – the sendmail daemon is still configured to run at boot time. There is no need for the server to listen on port 25 to receive mail or even for sendmail to run at all. There maybe times when it is necessary to run sendmail so users of the machine or processes can send emails, in which case proper due diligence should be paid to configuring sendmail securely and keeping it up to date. To turn off sendmail at boot time simply rename the linked start-up script called S88sendmail. The following command will accomplish this:

mv /etc/rc2.d/S88sendmail /etc/rc2.d/.NOS88sendmail

The naming convention of disabled scripts used here is a popular one. By putting a dot as a prefix to the filename hides it from a regular "ls" directory listing. By then appending the word "NO" shows that we have disabled this service.

3.9.3.3 NFS

By using NFS a Solaris server can be a jumpstart or network boot server. Although we will be turning of NFS completely because we do not require it, it is prudent to disable tftp should NFS need to be turned on again. We need to make sure that should an administrator turn some services back on, it does not open holes for attackers and the functionality must be turned on manually making the administrator think exactly what is needed.

Because this server will not be a boot server we need to disable tftp. To do this edit the "nfs.server" script located in /etc/init.d. The following lines should appear:

```
# If /tftpboot exists become a boot server

    if [ -d /tftpboot ]; then
        /usr/sbin/in.rarpd  -a
        /usr/sbin/rpc.bootparamd
    fi

    # If /rplboot exists become a boot server for x86 clients

    if [ -d /rplboot ]; then
```

```
        /usr/sbin/rpld  -a
        if [ !  -d /tftpboot ]; then
                /usr/sbin/in.rarpd  -a
                /usr/sbin/rpc.bootparamd
        fi
fi
```

By commenting out these lines prevents a tftp server from being run – even though the necessary directories are not present (/tftpboot and /rplboot) it, and NFS will be turned off. Again it adds an extra layer of protection and makes the administrator think about the services needed before turning them all on again.

3.9.3.4 Other miscellaneous unneeded boot services

There are many other services that are not needed, obviously, depending on the type of server will depend on the services that can be turned off, and so for this web server the following services can and should be disabled:

LDAP Client
RPC (very insecure do disable if not necessary)
Autoinstall Jumpstart
Cache Filesystem
NFS Client
Automatic Mounting of CD-ROM's
Preservation of editor sessions
NFS Server

The script below is an efficient way to turn off al these unneeded boot services by following the naming convention mentioned in section 4.5.3.2.

The services above correspond to the order of the scripts below respectively.

```
cd /etc/rc2.d
for file in S71ldap.client S71rpc S72autoinstall S73cachefs.daemon  \
S73nfs.client S74autofs S80PRESERVE S93cacheos.finish  \
; do
[ -s $file ] && mv $file .NO$file
done

mv /etc/rc3.d/S15nfs.server /etc/rc3.d/.NOS15nfs.server
```

Because Solaris was installed with just the core operating system there are not as many services to turn off as there would be if the default end user system were installed.

3.10 Create Warning Banners

By setting a banner when users login or boot the machine means two security issues can be solved. The first may help in prosecuting malicious users of the machine (if they are warned they will be prosecuted it may be easier to bring a judgement in court) and the second is that it hides system version information that can be invaluable to an attacker. All they would have to do is connect to

the machine and they already have enough information to restrict their attacks to known vulnerabilities of this particular version of Solaris.

A good warning banner should include the name of the organisation that owns the system and the fact that usage is monitored and using the system means the user consents to monitoring. For this example web server the banner could be something like this:

This system is owned and operated by Acme Company. Use is restricted to Acme Co. authorised users who must comply with the Electronic Communications Policy. Usage is monitored; unauthorised use will be prosecuted.

To add these banners to the active services, perform the following commands. Firstly configure the banner boot up eeprom banner:

```
eeprom oem-banner="This system is owned and operated by Acme /
Company. Use is restricted  to Acme Co. authorised users who must comply /
with the Electronic Communications Policy. Usage is monitored; /
unauthorised use will be prosecuted."
eeprom oem-banner\?=true
```

The other two files that need to configured are “/etc/issue” and “/etc/motd”. Adding the banner will mean that it is visible when users logon to the system. To do this execute these commands:

```
cd /etc
echo " This system is owned and operated by Acme Company. Use is restricted to /
Acme Co authorised users who must comply with the Electronic Communications /
Policy. Usage is / monitored; unauthorised use will be prosecuted." >motd
echo "This system is owned and operated by Acme Company. Use is restricted to
Acme Co / authorised users who must comply with the Electronic Communications
Policy. Usage is / monitored; unauthorised use will be  prosecuted." >issue
```

This ensures that when a user connects to the box (via SSH) they will be presented with the warning banner.

It is prudent to configure the telnet and ftp daemons to display the warning banner. Even though they are currently turned off an administrator may choose to turn them on again at a late date and so the banner will already be configured. To do this – execute the commands below:

```
cd /etc
echo “BANNER=\" This system is owned and operated by Acme Company. Use is /
restricted to Acme Co authorised users who must comply with the Electronic /
Communications Policy. / Usage is monitored; unauthorised use will be /
prosecuted.\\\n\\\n\"” > default/telnetd
echo “BANNER=\" This system is owned and operated by Acme Company. Use is /
restricted to Acme Co authorised users who must comply with the Electronic /
Communications Policy. / Usage is monitored; unauthorised use will be /
prosecuted.\\\n\\\n\"” > default/ftpd
```

Finally – ensure the files have the correct permissions.

```
chown root:sys /etc/motd
chown root:root /etc/issue
```

chown root:sys /etc/default/telnetd /etc/default/ftpd
chmod 644 /etc/motd /etc/issue
chmod 444 /etc/default/telnetd /etc/default/ftpd

3.11 Kernel Configuration

There are certain parameters of the kernel that can be configured to enable better security.

3.11.1 Core Dumps

Should the system crash the core dump file can take up a lot of disk space and can contain sensitive information such as passwords in memory. To disable the core dumps this line was added to the system specification file: /etc/system:

set sys:coredumpsize = 0

3.11.2 Stack Protection

A well-known and well-used form of attack is the buffer overflow attack (mentioned previously in section 3). "Enabling stack protection prevents certain classes of buffer overflow attacks and is a significant security enhancement"[4]. These lines were added to the system specification file:

set noexec_user_stack = 1
set noexec_user_stack_log = 1

3.12 Network Parameters

3.12.1 Network Parameters

Various network parameters can be used to enhance security against various denial of service attacks. This script – called "networkconfig" is executed at boot time and sets the parameters, each line is explained below. This script was created with reference from Network Environment Network Settings for Security document. Using "ndd" which is used to set TCP kernel parameters.

cat <<END_CONFIG >/etc/init.d/networkconfig
#!/sbin/sh
ndd -set /dev/ip ip_forwarding 0
ndd -set /dev/ip ip_forward_src_routed 0
ndd -set /dev/tcp tcp_rev_src_routes 0
ndd -set /dev/ip ip_send_redirects 0
ndd -set /dev/ip ip_forward_directed_broadcasts 0
ndd -set /dev/ip ip_strict_dst_multihoming 1
ndd -set /dev/tcp tcp_conn_req_max_q0 4096
ndd -set /dev/tcp tcp_ip_abort_cinterval 60000
ndd -set /dev/ip ip_respond_to_timestamp 0
ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0

[4] Solaris Benchmark from CISecurity

ndd -set /dev/ip ip_respond_to_address_mask_broadcast 0
ndd -set /dev/arp arp_cleanup_interval 60000
ndd -set /dev/ip ip_ire_arp_interval 60000
ndd -set /dev/ip ip_ignore_redirect 1
END_CONFIG
chown root:root /etc/init.d/networkconfig
chmod 744 /etc/init.d/networkconfig
ln -s /etc/init.d/netconfig /etc/rc2.d/S69networkconfig

Parameter explanations:

ndd -set /dev/ip ip_forwarding 0
This turns off IP forwarding which prevents packets traversing the Ethernet interfaces. This could provide a base for attackers to launch attacks on other systems.

ip_forward_src_routed
This parameter means that if IP forwarding were enabled (for a router or firewall for instance) it would not forward source routed packets. Source routed packets are packets that determine their own route through the network rather than relying on decisions made by individual routers. While we have disabled IP forwarding on this box, this parameter is added for extra security should the administrator turn IP forwarding back on.

ndd -set /dev/tcp tcp_rev_src_routes 0
This parameter sets to ignore source routing (as above) but for reverse routes on incoming TCP packets.

ndd -set /dev/ip ip_send_redirects 0
The only network equipment that should send redirect packets are routers – this machine will not be a router and so it can be turned off.

ndd -set /dev/ip ip_forward_directed_broadcasts 0
This line prevents forwarding of directed broadcasts when IP forwarding is enabled. A directed broadcast is a broadcast directed to a remote network from another remote host. This can be used to mount an attack and probe of remote systems.

ndd -set /dev/ip ip_strict_dst_multihoming 1
This tells the kernel to ignore packets sent to an interface from which it did not arrive. This prevents an attacker from creating packets destined for networks only connected to servers which do not forward packets.

ndd -set /dev/tcp tcp_conn_req_max_q0 4096
Solaris has two TCP queues, one for successful connection attempts and another for connection attempt that have not yet completed the three-way handshake. This is to prevent a SYN flood attack but the queue may be too small for a production web server. This increases the queue size for our web server.

ndd -set /dev/tcp tcp_ip_abort_cinterval 6 0000
Decreases connection time out value to one minute this makes it more difficult for an attacker to mount a SYN attack.

ndd -set /dev/ip ip_respond_to_timestamp 0
ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0
ndd -set /dev/ip ip_respond_to_addr ess_mask_broadcast 0
These timestamps are used to synchronise clocks. While our server will synchronise with an NTP time source (see later section) it is not necessary to respond to requests. It is for this reason that it is disabled.

ndd -set /dev/arp arp_cleanup_interval 60000
ndd -set /dev/ip ip_ire_arp_interval 60000
These two parameters set the timeout for arp entries in the arp cache and the routing table.

ndd -set /dev/ip ip_ignore_redirect 1
This parameter ensures the box will ignore ICMP redirect requests, which will mean an attacker could not inject bogus routes.

3.12.2 Sequencing
A poorly implemented TCP sequence number algorithm can mean the attacker can easily hijack a TCP session. By changing the algorithm used to a better-randomised one this cuts down the chance of a TCP hijack attack. To do this change the number from 1 to 2 in the "inetinit" file in /etc/default directory. It should read:

TCP_STRONG_ISS=2

3.13 Logging
Logging is not only an important part of forensics in discovering how an attack was made and what damage was done but it is also a good preventative measure as well as a useful way to keep track of system activity for management reports for example.

3.13.1 Log failed login attempts
Multiple failed login attempts may indicate someone is trying to hack into the system by using methods such as brute force password attack. To enable audit of failed logins create a file called "loginlog" in /var/adm. Set the relevant permissions. Set the "SYSLOG_FAILED_LOGINS parameter to 0 which will ensure every failed login attempt is logged. The command line execution should look like this:

touch /var/adm/loginlog
chown root:sys /var/adm/loginlog
chmod 600 /var/adm/loginlog

Then ensure that the failed logins parameter is set correctly in the "login" file in /etc/default directory.

3.13.2 Log AUTH messages

Messages of type AUTH in Solaris contain information that may be valuable (failed logins for example!). To enable this on the web server the following line was added to the "syslog.conf" file in /etc directory:

auth.info /var/log/authlog

Ensure that the relevant file exists and has the correct permissions in place:

touch /var/log/authlog
chown root:sys /var/log/authlog
chmod 600 /var/log/authlog

3.13.3 Log "inetd" connections

Again, while "inetd" has been disabled it is good practice to configure syslog to log this information should the administrator enable "inetd" at a later point. To do this - create the file "connlog" in /var/log directory and add the following line to "syslog.conf" and set the correct permission on the newly created file:

daemon.debug /var/log/connlog

touch /var/log/connlog
chown root:root /var/log/connlog
chmod 600 /var/log/connlog

3.13.4 Cron Logging

The cron log will show every job that is executed on the system and should, like all the logs be reviewed regularly. This feature was turned on by default on the web server installation but a check of the file permissions ensured that it was only accessible by the necessary logins and processes:

chown root:sys cron
chmod 444 cron

3.13.5 Root file system logging

An attacker can corrupt the root file system as a way of gaining access to the system. To limit the chances of root file system corruption simply add the option "remount,logging" to the root file system mount options in the "/etc/vfstab" file

3.13.6 System accounting

System accounting is useful to ascertain if any rogue processes are altering the normal baseline operations of the web server (CPU utilisation for example). This script was adapted from the CIS Solaris Benchmark document:

cat <<END_SYSTEM_ACC >/etc/init.d/newperf
#!/sbin/sh

```
/usr/bin/su sys -c \
"/usr/lib/sa/sadc /var/adm/sa/sa \`date +%d\`"
END_SYSTEM_ACC
chown root:sys /etc/init.d/newperf
chmod 744 /etc/init.d/newperf
rm -f /etc/rc2.d/S21perf
ln -s /etc/init.d/newperf /etc/rc2.d/S21perf
/usr/bin/su sys -c crontab <<END_ENTRIES
0,20,40 * * * * /usr/lib/sa/sa1
45 23 * * * /usr/lib/sa/sa2 -s 0:00 -e 23:59 -i 1200 -A
END_ENTRIES
```

This script gathers data every twenty minutes and logs it to a nightly report file enabling a regular check will ensure there is no evidence of a rogue process on the web server.

3.13.7 Kernel Auditing

The Basic Security Module supplied with Solaris allows auditing (to a binary format) of events and "This is one of the most powerful security features that Solaris provides out of the box, yet it is probably the least understood and least used." [5]

BSM is enabled on the web server box by executing the following commands:

/etc/security/bsmconv
This script is used to enable the Basic Securi ty Module (BSM).
Shall we continue with the conversion now? [y/n] ***y***
bsmconv: INFO: checking startup file.
bsmconv: INFO: move aside /etc/rc2.d/S92volmgt.
bsmconv: INFO: turning on audit module.
bsmconv: INFO: initializing device allocation files.

The Basic Security Module is ready.
If there were any errors, please fix them now.
Configure BSM by editing files located in /etc/security.
Reboot this system now to come up with BSM enabled.

After the script has run ensure that the "auditcontrol" file in the /etc/security directory looks like this:

```
dir:/var/audit
flags:lo,ad,ex,fm,-fw,-fc,-fd,na
naflags:lo,ad,ex,fm,-fw,-fc,-fd,nt
minfree:20
```

The flags are codes for auditing events, e.g. "lo" denotes login events. "Min free" means that a warning is generated when there is only 20% left for logging space.

By default only the command is logged, to log all command arguments for security purposes execute the command:

[5] SecurityFocus: FOCUS on Sun: Solaris BSM Auditing

auditconfig -setpolicy +argv

To switch the log files every hour to enable better auditing execute the following script:

```
cd /var/spool/cron/crontabs
crontab -l >root.temp
echo '0 * * * * /usr/sbin/audit  -n' >>root.temp
crontab root.temp
rm -f root.temp
```

3.14 File/Directory Permissions

It is vitally important to ensure correct permissions are set on directory – it is always prudent to grant least privilege where possible to limit the scope and fallout should an attacker attempt to gain privileged escalation.

3.14.1 Mount necessary file systems read-only

Some file systems on our web server will not need to be writable for most of the time or even at all. Therefore to decrease the chance of an attack we can mount some of the file systems read only. From the partitions that were created we cannot obviously mount the root as read only, nor can we mount /var as read only because this is where our logs will be stored and it will be written to constantly. /export/home mount point also needs to be writable because this is where the web site and home directories will be located. It will be necessary to remount the /opt file system as read/write when installing Apache (see later). The remaining file systems: /usr and /opt can be mounted read only at boot time, to do this add “ro” as a mount option under the “vfstab” file.

3.14.2 “nosuid” file system mounting

The remaining file systems that need to be mounted read/write should be mounted with the “nosuid” option in the mount options of the “vfstab” file. User programs that run with the “setuid” bit set are potentially very dangerous because they have the ability to run as another user with those privileges.

This is accomplished by simply adding “nosuid” to the mount options as we did with “ro” in the previous section. The file systems therefore that we add “nosuid” to are: /var and /export/home. We cannot add this to root file system because no device files can operate with “nosuid” enabled and the /dev directory is under the root file system.

3.14.3 Check log file permissions

The importance of securing the log file is obvious – they become useless if they are compromised by an attack. Their forensic and pre-emptive value

become useless. Therefore firstly ensure the following log files have the root:sys permission:

/var/log/syslog
/var/log/authlog
/var/log/loginlog

Ensure the following have root:root permission:

/var/cron/log
/var/adm/messages

Ensure the following have write and execute permissions on the group owner:

/var/log/syslog /var/adm/messages

Ensure the following have read, write and execute permissions on the group owner:
/var/log/authlog
/var/adm/loginlog
/var/cron/log

Execute the following commands to ensure the security on each file:
chown root:bin utmpx
chown adm:adm wtmpx
chmod 644 utmpx wtmpx
*chown sys:sys /var/adm/sa/**
*chmod go-wx /var/adm/sa/**

3.14.4 Fix Modes

Fix Modes is a program that sets appropriate permissions and ownerships on files throughout the file system. It should be run after packages are added to the system to ensure all permissions are as they should be (we will run this after Apache is installed). Fix modes can be downloaded from: http://www.sun.com/blueprints/tools/FixModes.tar.Z.

3.15 Access and Authentication

3.15.1 Restrict shells

A user can run any program as a shell (because at the moment the file "/etc/shells" does not exist), which opens a potential security hole. To restrict users to certain shells – create a file called "shells" in /etc directory and add the following entries:

/sbin/sh
/bin/sh
/usr/bin/sh
/bin/csh
/usr/bin/csh
/bin/ksh
/usr/bin/ksh

3.15.2 Remove "rhosts"

Rhosts implements authentication based on the IP address or host name of the connecting machine, which can easily be spoofed. Like the rest of the "r" commands they are useful but implement very weak or no security whatsoever. To disable these remove the following two lines from /etc/pam.conf:

rlogin auth sufficient pam_rhosts_auth.so.1
rsh auth sufficient pam_rhosts_auth.so.1

3.15.3 Cron/AT Restriction

The Cron and AT jobs run as different users and can consequently run with privileged access rights therefore restricting the ability to schedule "cron" jobs is advisable. To do this remove the two deny files:

rm -f cron.deny at.deny

Create two files called "cron.allow" and "cron.deny" containing only the word root in each. This means only root can now schedule jobs. It is important to ensure the correct permissions are set on the files:

chown root:root cron.allow at.allow
chmod 400 cron.allow at.allow

3.16 Empty Crontab files

Crontab files present a security problem because they are run by the "crontab" daemon, which runs with super user rights. Allowing any users the ability to read and write to these files could allow a user to escalate their privileges. To negate this risk we should removed the crontab file that are empty. In this installation the "adm" crontab file is empty and can simply be removed.

rm –f adm

The permissions on the remaining files in the crontab directory (/var/spool/cron/crontabs) should have only read permission by root.

3.17 Set prom security

As well as setting the boot banner (see earlier) setting a password on the eeprom is a security measure that can be implemented to negate the risk of a physical attack on the box. Of course, if the box is not physically locked then an attacker could simply steal the hard drives and mount them on their own machine. This will prevent an attacker from being able to boot from a cd-rom or other such device should they gain console access to the box. To do this execute the command:

eeprom security-mode=command

The system will then prompt for a password. By setting this password any command (apart from the normal multi-user boot command) at the ok or > prompt will need the password for authorisation.

Not only do we audit failed logins remotely and through the console but we also wish to log failed logins through the console on the eeprom. To do this execute the command:

eeprom security-#badlogins=0

This ensures any bad logins will be logged.

3.18 User Accounts

User accounts are the most common way for an attacker to initially gain access, whether it is through social engineering or brute force attacks it is the best way to begin to compromise a system.

3.18.1 System Accounts

There are several accounts that are supplied by default with each Solaris system and these will be the first accounts that an attacker will start with. The following script commands set the password for each of these accounts to an invalid string (this is possible because we should never be using these accounts to logon to the system):

passwd –l sys
passwd –l daemon
passwd –l adm
passwd –l bin
passwd –l lp
passwd –l nobody
passwd –l noaccess
passwd –l uucp
passwd –l nuucp
passwd –l listen
passwd –l nobody4

it is also a good idea to ensure that all the system accounts contain an invalid shell as an extra measure of security – running this script will ensure this happens:

```
for user in adm bin lp smmsp nobody noaccess uucp nuucp smtp listen nobody4; do
passmgmt -m -s /dev/null $user
done
```

This script iterates through the list of default system accounts and sets the shell via the “passmgmt” utility to a null shell.

3.18.2 Account Expirations

This very good script taken from CIS Solaris Benchmark document forces users to change their passwords every 91 days (13 weeks), and then prevent changes for seven days. Users receive warnings 28 days before their password expires.

```
logins -ox |awk -F: '($1 == "root" || $8 == "LK") { next }
{ $cmd = "passwd" }
($11 <= 0 || $11 > 91) { $cmd = $cmd " -x 91" }
($10 < 7) { $cmd = $cmd " -n 7" }
($12 < 28) { $cmd = $cmd " -w 28" }
($cmd != "passwd") { print $cmd " " $1 }' \
> /etc/CISupd_accounts
/sbin/sh /etc/CISupd_accounts
rm -f /etc/CISupd_accounts
cat <<EO_DefPass >/etc/default/passwd
MAXWEEKS=13
MINWEEKS=1
WARNWEEKS=4
PASSLENGTH=6
EO_DefPass
```

3.18.3 Set default umask

The default umask is the permission that is applied to every new file that is created. The current default umask is 022, by setting this to 077 it means any files created by users will be readable by only themselves. Should they wish to enable another user to read the file they must explicitly define it via chmod. To change the default umask value change the umask configuration line in "/etc/default/login" file to:

```
# UMASK sets the initial shell file creation mode mask.  See umask(1).
#
UMASK=077
```

Also insert the umask line at the bottom of both "/etc/profile" and "/etc/.login". The "profile" file may already specifiy a umask – if it does simply overwrite with the new value.

3.18.4 Setting mesg n

By setting mesg n will prevent users being able to communicate with each other via the write or talk command. This can help to stop social engineering attacks. Add the line "mesg n" to the "profile" and ".login" files as mentioned above.

3.19 Installing Apache

Finally it is time to install and configure Apache. Apache can be downloaded from: httpd.apache.org

Apache needs to be compiled before it can be run – most popular C language compilers can be used. For this demonstration "gcc" which is the free GNU

C++ compiler is being used. It needs to be installed and configured within the path environment before the Apache configuration can be run.

After "untarring", execute the following command to build the configuration for the Apache installation utilising only the modules that we need, for instance the web server does not require SSL, so to keep the security maxim – only install what we need:

./configure --prefix=/usr/local/secureapache –disable-module=all –server-uid=apache –server-gid=apache –enable-module=access –module=log_config –enable-module=dir –enable-module=mime –enable-module=auth

This will compile the configuration of Apache with only the modules listed above. These are the only modules required for our version of Apache to run, below is a list of the modules and a description of the features of each one:

Module Name	Description
access	Provides access based on hostname and IP address.
log_config	Enables detailed logging of connections made to the server
dir	Search and server index files
mime	Character set, encoding and MIME types.
auth	User authentication

The description of these and all other Apache modules are available at: http://httpd.apache.org/docs/mod.

The configure script above will also install the program in "/usr/local/secureapache" and for Apache to run as the user Apache. The user and group Apache needs to be created, to do this, the following command was executed:

groupadd apache
useradd -d /dev/null -g apache -s /bin/false apache

After executing the following two commands:

make
make install
chown -R root:sys /usr/local/secureapache

The web server is now compiled and installed. The next step is to limit the processes access to the system by providing a "jail". This is achieved via the "chroot" command and basically creates a mini file system that only the Apache web server runs in. Therefore this means should the web server be comprised via the Apache software an attacker can only gain access to the cut-down "jail" file system rather than the real one containing configuration files and passwords for example. The first step is to create the necessary directories that the Apache software requires to be able to run:

mkdir -p /chroot/httpd/dev
mkdir -p /chroot/httpd/etc
mkdir -p /chroot/httpd/var/run
mkdir -p /chroot/httpd/usr/lib
mkdir -p /chroot/httpd/usr/libexec
mkdir -p /chroot/httpd/usr/local/apache/bin
mkdir -p /chroot/httpd/usr/local/apache/logs
mkdir -p /chroot/httpd/usr/local/apache/conf
mkdir -p /chroot/httpd/www

There is no need to create the "/chroot" or "/chroot/httpd" for example, the "p" switch will perform this task automatically. Ensure that this is done by root because it is necessary for the directories to be owned by root.

We also need to create the null device that the Apache user has configured for the shell.

mknod /chroot/httpd/dev/false c 2 2
chown root:sys /chroot/httpd/dev/false
chmod 666 /chroot/httpd/dev/false

The "mknod" command is used to create special device files.

A definitive list now needs to be obtained of all the dependencies and linked libraries that are needed for Apache – to do this use the "ldd" command. Depending on the current configuration and modules that were compiled (if for instance this was not a new install) the output will differ from installation to installation. For this server we need to copy the following files:

cp /usr/local/apache/bin/httpd /chroot/httpd/usr/local/apache/bin/
cp /etc/hosts /chroot/httpd/etc/
cp /etc/host.conf /chroot/httpd/etc/
cp /etc/resolv.conf /chroot/httpd/etc/
cp /etc/group /chroot/httpd/etc/
cp /etc/master.passwd /chroot/httpd/etc/passwords
cp /usr/local/secureapache/conf/mime.types /chroot/httpd/usr/local/apache/conf/

It may take some time to copy all the files but after some trial and error the web server will run. Before this however, we need to remove all entries from the password and group files except "nobody" and "apache" and then build the password database:

cd /chroot/httpd/etc
pwd_mkdb -d /chroot/httpd/etc passwords
rm -rf /chroot/httpd/etc/master.passwd

Before running the server the default configuration file and a sample page need to be copied across.

cp /usr/local/secureapache/conf/httpd.conf /chroot/httpd/usr/local/apache/conf/
cp /usr/local/secureapache/htdocs/index.html.en /chroot/httpd/www/index.html

Next the document root variable needs to be changed to reflect the new path (chrooted). This is located in the main configuration file

(/chroot/httpd/usr/loca/apache/conf/httpd.conf) and should read:

```
DocumentRoot "/www"
```

Finally the web server can be started. The Apache logs can be used to check if any problems have occurred while starting Apache.

Once Apache is running the configuration file that comes as default with Apache needs to be modified to enable a more secure server (httpd.conf). the configutration file can be found in Appendix A.

As with SSH, a startup script needs to be written, adding the below to a /etc/init.d/chrootapache file will accomplish this:

```
#!/bin/sh

CHROOT=/chroot/httpd/
HTTPD=/usr/local/apache/bin/httpd
PIDFILE=/usr/local/apache/logs/httpd.pid

echo -n " apache"

case "$1" in
start)
        /usr/sbin/chroot $CHROOT $HTTPD
        ;;
stop)
        kill `cat ${CHROOT}/${PIDFILE}`
        ;;
*)
        echo ""
        echo "Usage: `basename $0` {start|stop}" >&2
        exit 64
        ;;
esac

exit 0
```

It is a good idea to run the fix modes program after installing Apache to be sure that all permissions are as they should be for the system to be secure.

Reboot the server, which will complete all the configuration steps necessary for hardening of the web server.

3.20 Physical security

An often-neglected part of securing a system is the physical access that is granted to employees and visitors. If an attacker can gain physical access to the system it becomes very easy for them to gain logical access and privilege escalation for example – all efforts at logical security become null if physical access is not up to scratch.

Because this is a critical server it needs to be housed in a suitable secure room such as a server room with good physical access procedures. A server

room will provide the necessary temperature regulation to prevent the box from overheating. The doors should be locked either by a key or by a card entry system preventing unauthorised access. The cabinet that the server resides in should also be secured with a lock and only authorised people (such as server administrators or server room technicians) should be allowed access. The machine should also be secured where possible with a lock on the case to prevent unauthorised removal of hardware (such as the hard disk) from the machine.

3.21 Power Supply

A denial of service can occur through loss of power to the box –either via a power cut to the building where the web server is physically located or by a failure in the cabling or transformer of the box itself. To negate the risk it is strongly advisable to install a UPS to cope with brown and black outs as well as power surges. The web server should ideally contain a backup power supply that is connected to a unique power block eliminating the risk or failure of cabling or transformers. This is not a common attack by hackers but power failures are surprisingly common and as such the risk is certainly high enough to warrant preventative measures such as this.

4 Ongoing Maintenance

While the web server is now operating and is as secure as it can be in the present climate, technology is ever evolving and more and more vulnerabilities are found all the time. This means it is vitally important to not undo all the work that has been done by not keeping the system up to date.

4.1 Effective backups

It is imperative that a backup is performed after the initial configuration and regularly thereafter. A backup can be performed any number of ways – although a typical way is to backup the disk to tape which can then be stored off-site in a secure place. This is useful should there be a hardware failure, but also if there was an attack directed at the web server, this could involve defacement or denial of service. The machine can quickly be restored via the backup limiting the amount of downtime on the service. Although, if the server were comprised it is necessary to find out how it was comprised and how it is vulnerable before putting the machine with the same configuration back on the network!

After a full backup has been made regular incremental backups should be performed. Although the content on the web server will change infrequently (this is not a fully fledged e-commerce site) it is important to keep a backup of all files that may have changed. This backup job could easily run every night and simply backup all files changed since the last backup and then run full backups at the end of each week using built in utilities such as “ufsdump”.

4.2 Patching

As mentioned in Section 4.4, Sun’s Patch Manager can be used to download and install all the latest patches as they are released. A typical implementation of this would be a patching server that could download all the necessary patches, and then each local server could download the patches that it needs. This guarantees centralised administration while ensuring each server running on the network remains patched in a timely manner.

This would also apply for Apache so the whole system would be covered by the patch upgrades.

4.3 Log Checking

The importance of logs has been discussed previously in this document – however, like all security practices, they become useless unless due diligence takes place. This means a regular check of the logs. The web server logs locally and so a process for administration log checking and archiving is necessary.

A good utility is log check. This is best run from a cron job and will examine the logs for suspicious behaviour. It has a number of configuration files, which

can be “tweaked” to suit the current environment. This is a better option than paying an administrator so watch logs all day. Log check is still available at: http://www.zpdee.net/~joecat/files/logcheck.tgz. Both the documentation that comes supplied with log check and common sense would suggest running the program a few times on logs to check it produces the desired results and configure accordingly.

Another great open source piece of software is Swatch (http://swatch.sourceforge.net). A now mature piece of software that can monitor almost any type of log.

4.4 Files and Permissions

Fix modes should also be run regularly to ensure that all files have the correct ownership and permissions. It is a good idea to run this after any programs are installed or patches are applied. Therefore if the patch manager is used fix-modes should also be used to ensure file integrity.

Some of the tasks that were carried out in the step-by-step section should be carried out regularly; these could be incorporated into a script run from cron. The sections that should be monitored regularly are:
4.17.1 System Accounts
4.17.2 Account Expirations
4.17.3 Set default umask
4.17.4 Set mesg n

4.5 Fix Modes

Fix modes should be run frequently, especially after patches or packages are added to the system. It will ensure that ownership and permissions are set correctly – this is why it is important to run after every addition to the system. Patches can change permissions on existing files or new ones. This should become part of the change control process.

4.6 Adequate change control process

Any change that is made to the server can potentially impact on the security of the server. It is therefore advisable to have a change control procedure in place to ensure all changes to the system are documented and authorised. This means that there is an audit trail and any changes can be easily “rolled back” should testing show that there is a security risk. After a change has been introduced section 5 and 6 of this report should again be followed. Any change to the system, whether it be program updates or configuration changes can lead to a security risk. Therefore it is necessary to immediately perform security maintenance and testing on the server.

4.7 Adequate Security Policy

The need for an adequate security policy is obvious. It should cover all standards as well as procedures for carrying out installations (such as this) and maintenance so that everybody involved knows how to carry out tasks and what is expected of them.

5 Configuration check

5.1 NMAP

Network Mapper is a free tool used to scan a remote system(s) for open ports and can even determine Operating System type. It is a highly configurable command line utility and is used by Nessus as part of its scan. Nmap is available from: http://www.insecure.org/nmap/

5.2 Nessus

One of the most useful tools for assessing vulnerabilities is Nessus. It is a very feature rich application that is available for free and should be in every security consultant's toolkit! It utilises "nmap" for port scanning and then uses a database of vulnerability signatures (in the form of plugins) to test the server. A report is generated after the scan showing the open ports and the vulnerabilities found.

Nessus was compiled and installed on a linux box on an isolated LAN. A connection was made from the nessus daemon on the linux box to the Apache server. A Windows client on another machine controlled the daemon.

The result of the port scan showed only two ports open. SSH and port 80 http. This is exactly as we would expect because they are the only services that we require to be listening. The port scan is the most important part of the scan – it shows us that we only have services running and listening on the ports that we want. The next stage is to attack those ports with known vulnerabilities – this is what Nessus does for us. Because Nessus uses a database of vulnerabilities that are constantly updated it is important to keep the database up to date. The results were as suspected. No vulnerabilities were found because the system is running the latest patched Apache, however Nessus does give valuable information as to mis-configuration that can allow an attack to be launched. It also provides valuable benchmarks for future reference.

5.3 Read only filesystem

Some of the filesystems on our web server are read only for security pruposes. To check this we need to try to write files to they system (even as root). The filesystems that were mounted read-only at boot time are:
/usr
/opt

To test that these are read only we can try to create a file within these directories as root. Execute the command:

touch /usr/test
touch /opt/test

An error should occur while creating these files – also note that because

these partitions are mounted read-only similar error messages will appear if trying to install more third-party software.

5.4 Check SUID/SGID Executables

A good script to check for unauthorised SUIG/SGID executables is one supplied from Solaris Benchmark document from CIS Security, they also offer a automated tool which could be run via “cron” monthly or weekly to alert for unauthorised SUID/SGID files. While this script is in this section to enable a check of the configuration, it should also be used in ongoing maintenance to ensure unauthorised executables do not have their SUID/SGID bit set. It is important to understand that there are some executables that may have set-UID snd set-GID shipped as standard. This script may be run through “cron”:

```
for part in `awk '($4 == "ufs" || $4 == "tmpfs")  \
{ print $3 }' /etc/vfstab`
do
find $part -xdev -type f \
\( -perm -04000 -o -perm -02000 \) -print
done
```

5.5 Invalid Shells for system accounts

All system accounts should have invalid shells – to check this execute the command:

cat /etc/passwd | grep /d ev/null

This will show all the entries that have the shell: “/dev/null”. This is an invalid shell. Each system account should have this. The output from the command is shown below. If there are any other system accounts (the ones below are the default installed when the operating system was installed earlier) or there are user accounts that should not have shell access ensure that these are updated to reflect the new changes:

```
bin:x:2:2::/usr/bin:/dev/null
adm:x:4:4:Admin:/var/adm:/dev/null
lp:x:71:8:Line Printer Admin:/usr/spool/lp:/dev/null
uucp:x:5:5:uucp Admin:/usr/lib/uucp:/dev/null
nuucp:x:9:9:uucp Admin:/var/spool/uucppublic:/dev/null
listen:x:37:4:Network Admin:/usr/net/nls:/dev/null
nobody:x:60001:60001:Nobody:/:/dev/null
noaccess:x:60002:60002:No Access User:/:/dev/ null
nobody4:x:65534:65534:SunOS 4.x Nobody:/:/dev/null
```

5.6 Check banners

Check each login banner and startup banners to ensure they are correct as per the earlier configuration instructions. This is especially important for legal reasons. Banners should appear during boottime, connection via console, and

connection through SSH.

5.7 Check SSH access

The server is configured to only allow connections to the SSH port via the local network. This can be tested by attempting to SSH to the server from an untrusted IP address and ensuring that we are unable to connect. Conversely attempt to SSH from a trusted network and ensure that a connection can be made and login successful with a valid login user account.

5.8 Check Apache

It is important to first check that the server is accessible from all networks (including the Internet) on port 80 (http). To do this, connect with a web browser to the server from the local trusted network and the internet to ensure that the front page is visible.

It is impossible to launch every type of attack on our web server. It is the latest version (and should continue to be so through regular patching) and so should be hardened to all known vulnerabilities. The removal of the default configuration file and replacement with a customised one should mean that default configuration issues and problems should not lead to security issues such as server software identification or anonymous directory browsing.

Appendix A – Chrooted Apache Config

```
================================================
# Basic settings
# ================================================
ServerType standalone
ServerRoot "/usr/local/apache"
PidFile /usr/local/apache/logs/httpd.pid
ScoreBoardFile /usr/local/ apache/logs/httpd.scoreboard
ResourceConfig /dev/null
AccessConfig /dev/null

# ================================================
# Performance settings
# ================================================
Timeout 300
KeepAlive On
MaxKeepAliveRequests 100
KeepAliveTimeout 15
MinSpareServers 5
MaxSpareServers 10
StartServers 5
MaxClients 150
MaxRequestsPerChild 0

# ================================================
# Apache's modules
# ================================================
ClearModuleList
AddModule mod_log_config.c
AddModule mod_mime.c
AddModule mod_dir.c
AddModule mod_auth.c

# ================================================
# General settings
# ================================================
Port 80
User apache
Group apache
ServerAdmin webmaster@sansgcux.exam
UseCanonicalName Off
ServerSignature Off
HostnameLookups Off
ServerTokens Prod
<IfModule mod_dir.c>
    DirectoryIndex index.html
</IfModule>
DocumentRoot "/www"

# ================================================
# Access control
# ================================================
<Directory />
    Options None
    AllowOverride None
    Order deny,allow
    Deny from all
```

```
</Directory>


# ==============================================
# MIME encoding
# ================================ =================
<IfModule mod_mime.c>
    TypesConfig /usr/local/apache/conf/mime.types
</IfModule>
DefaultType text/plain
<IfModule mod_mime.c>
    AddEncoding x-compress Z
    AddEncoding x-gzip gz tgz
    AddType application/x-tar .tgz
</IfModule>

# ==============================================
# Logs
# ==============================================
LogLevel warn
LogFormat "%h %l %u %t \"%r\" %>s %b \"%{Referer}i\" \"%{User-Agent}i\"" combined
LogFormat "%h %l %u %t \"%r\" %>s %b" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent
ErrorLog /usr/local/apache/logs/error_log
CustomLog /usr/local/apache/logs/access_log combined
```

Adapted from: http://securityfocus.org/infocus/1694

References

CERT. “CERT® Advisory CA-2002-17 Apache Web Server Chunk Handling Vulnerability”. 27 Mar. 2003. URL: http://www.cert.org/advisories/CA-2002-17.html (09 Apr. 2003).

CERT. “CERT® Advisory CA-2002-27 CERT® Advisory CA-2002-27 Apache/mod_ssl Worm”. 11 Oct. 2002. URL: http://www.cert.org/advisories/CA-2002-27.html (09 Apr. 2003).

Netcraft. “March 2003 Web Server Survey”. 25 Mar. 2003. URL: http://www.netcraft.com/Survey/Reports/200303/overallc.gif (11 Apr. 2003).

SANS/FBI. “Apache Web Server” SANS/FBI The twenty Most Critical Internet Security Vulnerabilities. 3 Mar. 2003. URL: http://www.sans.org/top20/#U2 (09 Apr. 2003).

SecurityFocus HOME Infocus: “Securing Apache: Step-by-Step”. 14 May 2003. URL: http://www.securityfocus.com/infocus/1694 (10 June 2003).

SecurityFocus HOME Infocus: “FOCUS on Sun: Solaris BSM Auditing”. URL: http://www.securityfocus.com/infocus/1362 (02 May 2003).

Sun Microsystems Inc. “Solaris Operating System for Sparc Platforms”. URL: http://wwws.sun.com/software/solaris/sparc/index.html (29 May 2003).

The Apache Software Foundation: “Download – The Apache HTTP Server Project”. http://httpd.apache.org/download.cgi (25 May 2003).

The Center for Internet Security: “Solaris Benchmark v1.2.0”, 19 Febuary 2003. URL: http://www.cisecurity.org/bench_solaris.html (12 Apr 2003).

Watson, Keith & Noordergraaf: “Solaris Operating Environment Network Settings for Security, December 2000. URL: http://www.sun.com/blueprints (02 May 2003).

Last Updated: May 27th, 2017

Upcoming Training

SANS London July 2017	**London, United Kingdom**	**Jul 03, 2017 - Jul 08, 2017**	**Live Event**
SANSFIRE 2017	**Washington, DC**	**Jul 22, 2017 - Jul 29, 2017**	**Live Event**
SANS Network Security 2017	**Las Vegas, NV**	**Sep 10, 2017 - Sep 17, 2017**	**Live Event**
SANS OnDemand	**Online**	**Anytime**	**Self Paced**
SANS SelfStudy	**Books & MP3s Only**	**Anytime**	**Self Paced**