Global Information Assurance Certification Paper

Step-by-Step Secure Configuration Guide to Create a Solaris 9 Syslog Server

Travis Hildebrand
Issue Date: 08/22/2003

Summary: The purpose of this paper is to provide a step-by-step guide for a novice system administrator to install a functional and secure central syslog server running on Solaris 9 in a corporate environment for multiple UNIX machines and other syslog-enabled network devices.

GCUX Practical v1.9 Option 1

1.0 DESCRIPTION OF THE SYSTEM

This paper is designed to allow the System Administrator (SA) to create a secure UNIX environment for a production syslog collector. The purpose of this machine is to collect syslog messages from other UNIX and network devices for log reviews. The operating system this paper helps secure is Solaris 9. The hardware to be used is SPARC hardware. Specifically, we will be installing an Ultra 30 using one 18 GB disk, and at least 512 MB of RAM. The CPU speed does not need to be extremely fast for syslog analysis, but should be at least 133MHz. Similarly, the author suggests having at least 18 GB of hard drive space for this system. A graphical terminal is not required. We will be installing the entire machine from a dumb terminal. The logs that will be collected by the system will be collected with the 'syslog' services provided from the OS vendor. This paper will ensure that the operating system the syslog service runs on is secure. For the sake of simplicity, we'll call our syslog server "*logger*."

Logger will be placed on a corporate network behind an Internet-facing firewall. Network intrusion detection devices are expected to be in place to monitor the network for attacks. The setup also assumes there is a corporate Veritas Netbackup server though which backups will be performed. Additionally, there are corporate mail servers and timeservers assumed to be on the internal corporate network.

The convention this document uses is that *italics* are generally commands or input from the user.

1.1 System End-State Description:

The system end-state is to have a secure operating system environment for the syslog service. The server name will be placed in the company's internal DNS so host resolution will work for all internal IP addresses. It will NOT be advertised on the company's external DNS server. The machine will not receive e-mail; however it will have the ability to send mail from the server. The only user IDs on the machine will be those of the System Administrators. The system will have many programs/patches installed on it to help support the syslog service. Some of the programs are as follows:

- Latest recommended patches from the OS vendor
- OpenSSL 0.9.7b (or higher)
- OpenSSH 3.6.1p1 (or higher)
- Perl 5 (or higher)
- Sudo 1.6.7p5 (or higher)
- TCP Wrappers 7.6 for services like ftp, telnet, ssh
- NTP (for time synchronization)

- SNMPD will be running for network monitoring services
- Tripwire ASR for file integrity
- Veritas Netbackup Client 4.5
- homegrown scripts like *SecurityCheck.pl* and *passwd_age.pl*
- *logcheck* (from http://sourceforge.net/projects/sentrytools)
- *logrotate* (to rotate system logs)
- *syslogd* will be running to log syslog events locally from remote UNIX servers and network devices as well as the local logs generated on *logger*.

There are very few ports expected to be open when the machine is configured securely. The only the expected ports expected to be listening are ssh (port 22), ntp (port 123), syslog (port 514), and sendmail (port 25), and four ports (13724, 13782, 13783, and 13722) for NetBackup. The NetBackup ports will only answer to the backup server because the services will be protected with TCP wrappers.

2.0 RISK ANALYSIS:

By creating a centralized syslog server, we are helping to ensure the integrity of the logs. Logs are expected to be stored on the remote (source) UNIX device or network device where technically possible. The *logger* server provides and additional copy to the original source. If a UNIX machine is breached by an attacker we will still have logs on our *logger* server to see what he/she did to break-in. This is not an all-inclusive solution, but is a "best practice" to help in the forensic analysis of a cyber crime. Additionally, a centralized log server is a good source of information when troubleshooting other random difficulties on the network.

2.1 Main Security Concerns:

2.1.1 Minimize the number of services running on the system

We will turn off all unnecessary services on the machine to reduce the risk of exploits that would compromise the integrity of the system. This includes services like telnet, ftp, and SNMP. These services present risks to the system, and the services are not necessary for our environment.

2.1.2 Physical security

The *logger* server should be placed in a physically secure computer room. All servers (including *logger*) that are attached to the production environment should be physically located in a secure computer room. This ensures adequate physical security, which is a vital part of OS security. Ideally, a computer room has strong physical access control mechanisms in place such as active badge readers. Only authorized personnel are allowed in and out of the computer room. All physical access is logged.

2.1.3 UPS power

Because *logger* is the central logging server for the site, we want to ensure it stays online as much as possible. To safeguard against power outages, *logger* should be placed on an uninterruptible power supply (UPS). It should also have a generator backup to supply power in the case of extended power outages. A UPS alone will only buy a limited amount of time.

2.1.4 Software Config errors

We need to be very aware of the possibility of typos and human error when configuring *logger*. To this end, the author has created a perl script called SecurityCheck.pl that can systematically and automatically run to verify the security settings in this document. We will check most steps as they are configured and then check the overall configuration after all steps are complete. It is possible that one step could undo the configuration of a previous step.

2.1.5 Old Passwords

One of the weakest links in any UNIX machine is the password authentication mechanism. We will harden this area with a combination of configuration settings along with an additional perl script written by the author called passwd_age.pl. This script will be configured to run daily and send e-mails to the 'users' for 14 days prior to the password expiring. It will automatically disable the password for accounts that reach the expiration date.

2.1.6 Minimized number of users

Another "best practice" is to minimize the number of users on the local system. The only real "users" in the traditional sense of the UNIX operating system will be system administrators. Estimates are 2-4 UNIX system administrators. Other users should not have access to the system logs for security reasons. Any access to other production servers that requires elevated privileges needs to be logged and traceable back to an individual user while being protected from modification from the user. The primary threats to the system are expected to come from the end-users (SAs) themselves.

2.1.7 Firewalls, IDS, and backups

Logger will be placed on a corporate network behind an Internet-facing firewall. Network intrusion detection devices are expected to be in place to monitor the network for attacks. The setup also assumes there is a corporate Veritas Netbackup server though which backups will be performed. Additionally, there are corporate mail servers and timeservers on the internal corporate network.

2.1.8 Denial of Service to the syslog service

Syslog-ng is an alternative syslog service that could be used if we were truly worried about denial of service attacks. We will accept the risk of denial of service attacks on the syslog service because *logger* is inside our corporate firewall, and the fact that we get hourly *logcheck* e-mails. We will be monitoring the hourly *logcheck.sh* e-mails. If they jump in size, we will notice quickly and react to the flood of syslog messages. Additionally, our network has IDS sensors to look for these types of attacks. For our environment, the Sun Solaris syslog service will suffice. The syslog-ng utility is mentioned as an alternative if it would work better in your environment.

2.1.9 Availability

It is recommended to have a secondary *logger* system on the network that would be configured identical to the first one. Other UNIX machines and network devices can be configured to send syslog messages to this additional source as well as the original *logger*. This would provide redundancy in the case of hardware failure, as well as a single source to capture logs when patch cycles are happening. When *logger* is in single-user mode for patch cycles, it will not be accepting syslog messages.

2.1.10 OS & Product Hardening

We will harden as much of the Operating System and add-on packages as possible with the configuration files and utilities available. This includes implementing sudo, TCP wrappers, configuring OpenSSH, as well as restricting file/directory permissions.

2.1.11 Patching

Maintaining current patches on the machine is critical. Refer to the "Ongoing Maintenance" section for more info on patching. Basically, the author recommends a monthly patching cycle for the system, which includes bringing the system down to single user mode and applying the latest "Recommended" patches from Sun. Additionally, when a new CERT or Sun Security advisory arrives in e-mail (from the e-mail lists recommended later), the patch cycle will be advanced.

3.0 STEP-BY-STEP GUIDE

3.1 Assumptions:

- Power is available
- A dumb terminal is available (like a vt100, or serial cable attached to a PC computer running a program like Teraterm). The serial port runs at 9600,N,8,1. Be sure your terminal is set to these settings.
- The installer has some knowledge of how to use the 'vi' text editor.

3.1.1 Follow your organization's procedures before connecting a device on your network. A Security Plan that is approved by your accrediting official may be required.

3.2 Install the Base Operating System:

Install the base operating system from CD. When prompted to do so, select the option to install the core distribution. We will be locking down the insecure protocols throughout the remainder of this document. The reason to install the core distribution is minimize the amount of vulnerabilities. Some less-than-secure options like telnet/ftp will still be installed with the core. We will be disabling these in later steps.
It is recommended for this server to use the following partition table assuming a single 18 gig drive for the OS. If a larger drive is used, the primary partition to grow is the /var filesystem. This is where the "data" for *logger* will be stored.

100Meg	/	
1.7 Gig	/usr	
1.0 Gig	swap	
1.0 Gig	/opt	
2.0 Gig	/tmp	
2.0 Gig	/export/home	#For users home directories (SA's)
9.5 Gig	/var	#This is where the logs will go. (Remaining space)

This partition scheme will protect the filesystems from each other in case one of them fills up 100% (usually the /var or /export/home partitions fill up.) The root filesystem should be extremely static and should not need more than 100 Megs. The /var filesystem is where the syslog logs will be stored for the other network devices (i.e. other UNIX machines and/or syslog-enabled network devices.) It would be best to install the machine in an offline state from a security perspective. It will be necessary to pre-fetch 3rd party packages from the Internet. The administrator will need to pre-fetch the required packages (listed throughout this document and summarized in Appendix H) and burn them to a CD in iso9660 format.

The user can verify the md5 hashes from the source (when available) web sites on the machine that downloads the software. It will be up to the user to use the correct utilities on their "download" machine, as it could be a Microsoft Windows or Macintosh machine. The basic idea is to verify that what was downloaded from the Internet is what we expected to get. Most web sites offer an expected md5 hash to compare the downloaded file to verify authenticity.

Logger should be installed while it is disconnected from the network.

Turn on the machine. Hit Stop-A (or send a break signal).
Insert the Solaris 9 (9/02 is what we have available and will use) Software 1 disc. Don't use the "Solaris 9 Installation" disc as this runs a web service interface to install the machine. It would also put the swap space at the beginning of the hard disk.

At the "OK" prompt, type
boot cdrom

Choose:
0 - English
0 - English (C - 7-bit ASCII)
1 - ANSI Standard CRT
Networked – Yes
DHCP – No
Hostname: *logger*
IP: *<enter your IP address here>*
Part of a subnet: YES
Netmask: *255.255.255.0 (or yours here)*
Enable Ipv6: No
Default Route: Specify One
Router IP Address: *<enter your default route here>*
Configure Security Policy: No
Name service: *None*
Continents and Oceans: *Americas*
Countries and Regions: *United States*
Time zones: *Central Time <or yours here>*
Date and time: *<enter the correct date and time>*
… <wait a few minutes> ...
Solaris Interactive Installation: *Select Initial install with F4 (Esc-4)*
Solaris Interactive Installation: *Standard Install with F2 (Esc 2)*
Select Geographic Regions: *North America - U.S.A. (UTF-8)*
Select 64 Bit: *yes*
Select Software: *Core System Support 64-bit ..... 705.00 MB*
Select Disks: *c0t0d0 (17356 MB) boot disk 17356 MB*
Preserve Data?: *no*
Do you want to use auto-layout to automatically layout file systems?: *Yes*
Automatically Layout File Systems: *Select /, /opt, /usr, /var, /swap*
File System and Disk Layout: *Customize with F4 (Esc-4)*
Setup the filesystems so they look like this:

Slice	Mount Point	Size (MB)
0	/	104
1	/var	9352
2	overlap	17356
3	swap	1025
4	/tmp	2048
5	/opt	1025
6	/usr	1752
7	/export/home	2048

Select F2 (Esc 2) to continue
File System and Disk Layout: *Select F2 (Esc 2) to continue*
Mount Remote File Systems? *Select F2 (Esc 2) to continue*
Profile: *Select F2 (Esc 2) to continue*
Reboot After Installation? *Auto Reboot*
… <This takes about 10-15 minutes> …
logger console login: *root*
passwd
Enter a complex root password twice.

Once the base OS is installed, continue with the next section to add further security to the operating system.

Remove some of the core utilities from the OS:

```
pkgrm SUNWrcmdr     #Remote Network Server Commands (Root)
pkgrm SUNWlldap     #LDAP services
```

Install additional software from the Solaris9 Software 1 CD.

```
mkdir /mnt/cdrom
mount -o ro -F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
cd /mnt/cdrom/Solaris_9/Product
pkgadd -d . SUNWscpu        (needed for NTP)
pkgadd -d . SUNWntpr        (Network Time Protocol)
pkgadd -d . SUNWntpu        (Network Time Protocol)
pkgadd -d . SUNWtcpd        (TCP Wrappers)
cd /
umount /mnt/cdrom
```

Now load Software 2 CD.

```
mount –o ro –F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
cd /mnt/cdrom/Solaris_9/Product
pkgadd –d . SUNWgzip        (Compression utilities)
```

Now verify that these packages exist with this command:

```
pkginfo SUNWscpu SUNWntpr SUNWntpu SUNWtcpd SUNWgzip
```

We should see one line per package with a brief description. If you do not see one line per package, retry the step for the package that is missing.

Package for SUNWscpu will add the 'whoami' command into /usr/ucb for use in a later section on modifying the default prompt.
Package SUNWntp* packages will add the xntpd daemon for time services.

We are done with the Solaris 9 Software 2 CD. Let's unmount it and remove it from the drive.

```
cd /
umount /mnt/cdrom
```

Eject the cd and remove it.

Install the latest Solaris 9 recommended patch cluster. This will be done with compact disc media. From another machine that is connected to the Internet, download the latest recommended patch cluster from http://sunsolve.sun.com for Solaris 9. The file will be named 9_Recommended.zip. Burn this file onto a CD. Mount the CD on *logger* with the following commands:

mount –o ro –F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
cd /mnt/cdrom

Copy the recommended patch cluster to the /tmp directory and unzip with

cp 9_Recommended.zip /tmp/9_Recommended.zip
cd /tmp
unzip –q 9_Recommended.zip

If you want to see each file in the zip file as it is uncompressed, leave off the "-q" flag on the previous command.

Switch to single user mode and install the cluster.

init S
<give the root password when prompted>
cd /tmp/9_Recommended
./install_cluster

The install of the patch cluster will take approximately 20-30 minutes with a core OS installation. There will be several patches (about 21) that fail because we only have the core cluster packages installed. These are normal and expected error messages.

When the patch cluster is finished installing, reboot with:

init 6

Log in as root when the machine is finished rebooting.

3.3 Banners/Disclaimers

Banners are necessary as legal warnings that violators can be prosecuted in a court of law. Prosecution of violators is more difficult without the legal warning banners in place. Even though we do not expect to be using the telnet services, having legal warning banners in place offers some legal protection in the rare cases when a SA may turn on telnet at a later date.

3.3.1 Telnet Banner:

Create the */etc/issue* file to display the message found in **Appendix A** to remote users before they login.

vi /etc/issue

3.3.2 File Transfer Protocol (FTP) Banner:

Create the FTP banner. For Solaris, the file is /etc/ftpd/banner.msg. Modify the BANNER variable to contain the short version of the contents of **Appendix A**, so that individuals who ftp to the machine are presented with the warning. Again, we do not expect to be using the ftp services, but having legal warning banners in place offers some legal protection in the rare cases when a SA may turn on ftp at a later date.

vi /etc/ftpd/banner.msg

3.4 Accounts

3.4.1 Shadow Passwords

Solaris does this by default. Shadow passwords are important because the encrypted versions of every user's password are stored in this file. When the passwords are "shadowed", the only ID that can read them is the root ID. When passwords are not shadowed, the encrypted string for each user's password is stored in clear text in a world readable file /etc/passwd. Any user on the system could take this information and run a program to crack the passwords.

3.4.2 Restrict Global Privileges

The definition "chown giveaway" is that a user can "give" his/her file away to another user. At this point, the owner of the file is some other user on the system. This is not necessarily a bad thing generally, but it could be abused to make the SA think some other user has created or modified files on the system. In an effort to avoid confusion and better ensure security of the files, the "chown giveaway" feature should be turned off for operating systems that default to this option. For Solaris, this feature is turned off by default. No action is required to turn it off. It is mentioned here for education only.

3.4.3 **Create accounts for the system administrators** (users of the system).

3.4.3.1 As a recommend policy, all passwords should:

- be 8 characters long. (by default Solaris only requires 6)
- consist of 1 uppercase letter.
- consist of 1 lowercase letter.
- consist of 1 numeric character, 1 special character and as many other characters to bring the length to 8 characters.
- No dictionary words. (Solaris does not natively support this option. This policy would be a written company policy. Note: It is possible to install a 3rd party product or PAM module to enforce this rule. For logger, we will rely on written company policy alone.)

3.4.3.2 Set password rules as closely as possible to match that of the policy from the previous section. Below are listed the settings for the operating systems included in this document. Some of the written policy rules cannot be enforced with OS built-in parameters. For this machine, we will rely on the users following the written policy for the rules that cannot be enforced by the OS. Alternatively, a PAM module like *npasswd* that could be plugged into /etc/pam.conf, or a commercial third-party product like PowerPassword (from http://www.symark.com) would provide a complete, enforceable set of password rules.

```
TERM=vt100
export TERM
vi /etc/default/passwd
MAXWEEKS=13
MINWEEKS=1
PASSLENGTH=8
WARNWEEKS=2
```

3.4.3.3 Create System Administrator accounts

Create the actual accounts on the system for the System Administrators. The following is an example of one account creation.

```
/usr/sbin/useradd -g 14 -c "Travis Hildebrand"-m  -d /export/home/travis -s /bin/ksh travis
passwd travis
  <Enter a complex password for this user (and verify by entering again)>
```

The "-g" flag sets the group to "sysadmin."
The "-c" flag is for the GCOS (comment) field.
The "-d" flag sets the user's home directory.
The "-m" flag creates the user's home directory.
The "-s" flag sets the user's shell.

We can check to see if the ID was created by looking at the /etc/passwd and /etc/shadow files.

```
grep travis /etc/passwd /etc/shadow
```

We should see something like:

```
/etc/passwd:travis:x:100:14:Travis Hildebrand:/export/home/travis:/bin/ksh
/etc/shadow:travis:IsTobQJHkM0Lw:12284:7:91:14:::
```

Repeat this process for the other System Administrators.

3.4.4 Modify /etc/profile

Set the default prompt to display "username@hostname>" as the user's shell prompt (PS1 variable). Set the default PATH variable to a minimal path to avoid Trojan programs that could be placed in paths with additional "write" permissions by other users. Set the default "umask" variable so when new files and directories are created, they do not have world read/write/execute permissions. The umask variable is also set in another system configuration file below. It is understood that individual users can modify/override these system defaults in their own .profile.

vi /etc/profile

Set the following variables:

```
PS1=`/usr/ucb/whoami`'@'`hostname`' > '
PATH=/usr/bin:/sbin:/usr/sbin
export PS1 PATH
umask 027
```

Manually run the /etc/profile for the login session we already have:

```
. /etc/profile
```

Our prompt should have changed. The other variables did also. We can check by running the commands:

```
umask
echo $PATH
```

3.4.5 Disable unnecessary accounts

As a minimum, ensure that these accounts are locked or do not have a password associated with it for login. This action will ensure that the user cannot login to any of these accounts with a username and password.

Delete the following accounts: uucp, nuucp.

```
userdel uucp
userdel nuucp
```

Disable the following accounts: daemon, bin, sys, adm, lp, listen, smmsp, nobody4, nobody, noaccess, as well as any other unnecessary accounts

```
for i in daemon bin sys adm lp smmsp listen nobody noaccess nobody4
do
        passwd –l $i
done
```

The "-l" flag on the *passwd* command locks the account.

The end result should be verifiable by looking at the /etc/shadow file. The 2nd entry of all the locked accounts should have a *LK* in them.

cat /etc/shadow

root:7pi5aXlb8yH2g:12284::::::
*daemon:*LK*:12284::::::*
*bin:*LK*:12284::::::*
*sys:*LK*:12284::::::*
*adm:*LK*:12284::::::*
*lp:*LK*:12284::::::*
*smmsp:*LK*:12284::::::*
*listen:*LK*:12284::::::*
*nobody:*LK*:12284::::::*
*noaccess:*LK*:12284::::::*
*nobody4:*LK*:12284::::::*
travis:IsTobQJHkM0Lw:12284:7:91:14:::

Allow direct root account logon only at the system console, which should be physically located in a secure space like a computer room. This ensures more accountability to an individual user.

vi /etc/default/login

Ensure there is no "#" sign at the beginning of the line:

CONSOLE=/dev/console

3.4.6 Restrict the use of the su command to the admin group and root account only

The idea here is to not allow non-SA users to become another user.
This step is useful if there are ever going to be other users on the system besides system administrators.
If the sysadmin group does not exist, create the sysadmin group in /etc/group file. (GID=14) Otherwise ensure /etc/group file contains "sysadmin::14:" Now, change the group ownership of the /usr/bin/su command.

chgrp sysadmin /usr/bin/su
chmod o-rwx /usr/bin/su

Caution: /usr/bin/su has the SetUID bit turned on, su will no longer work if this bit is turned off.

The above permissions and ownership is set to assist with enforcing the following suggested policy passage: (System Administrator Security Procedures):

> Accountability and Traceability for All Privileged System Commands: All commands issued by computer system operators must be traceable to specific individuals via the use of comprehensive logs. For systems that provide the capability to turn off, bypass, modify, or delete such logs, system administrators must never do so without first obtaining the concurrence of the <*insert approving official title*>. As an example (but not limited to this example), UNIX system administrators must not use 'sudo' commands to avoid logging root-level commands.

3.4.7 Password Age Script:

This script is used to "age" passwords according to rules set in the script itself. Passwords should be changed every 90 days to provide better security and minimize the chances of a hacker gaining unauthorized access to the system. E-mail warnings should be sent for the last 14 days of a password's life to remind the user that his/her password is about to expire. Since the OS does not include this custom script natively, the author wrote script to provide this functionality. The key here is to warn the user with daily e-mails near the password expiration date. At the 90-day mark, the password is disabled.

Install a password age script in /usr/local/scripts and make a root crontab entry so it runs daily:

EDITOR=/bin/vi;export EDITOR #sets the default editor to vi
crontab -e root

Insert the following line:
1 5 * * * /usr/local/scripts/passwd_age.pl > /dev/null 2>&1

Ensure that every ID in the /etc/passwd file has a corresponding entry in the /etc/aliases file so the warning e-mails will be delivered to the correct person.
Note: The passwd_age.pl script is included here in a*comment* format in this word document so as to not dramatically increase the size of this document. To get the full text of this script, you will have to place the cursor over this sentence and "edit" the comment.

mount –o ro –F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
mkdir –p /usr/local/scripts
cp /mnt/cdrom/passwd_age.pl /usr/local/scripts/passwd_age.pl
chmod 700 /usr/local/scripts/passwd_age.pl

Modify the script and replace the 6 variables at the top with information for your site. Feel free to make other customizations to the script to meet your needs.

vi /usr/local/scripts/passwd_age.pl

Now run the script manually to ensure there aren't errors.

/usr/local/scripts/passwd_age.pl
ls –l /var/log/passwd_age.log

The passwd_age.log file should now exist if the script ran OK. If the "ls" command shows no output, the log did not get created.

c*rontab –l root*
This will show the root crontab is installed. Look for the line we added.

3.5 Directory/File Permissions

3.5.1 Set UMASK & TIMEOUT variables

The UMASK variable is used to set the default permissions for new files and directories.
A UMASK of 027 will result in **file** permissions that are read-write for the "user" and read only for the primary "group". There are NO "world" permissions.
A UMASK of 027 will result in **directory** permissions that are read-write-execute for the "user", read-execute for the group, and NO "world" permissions.
The main idea here is to not give excessive file/directory permissions to other users of the system.

Use secure settings in /etc/default/login:

vi /etc/default/login

Set the following variables (and ensure there is no "#" before them):

UMASK=027 (Keep in mind that individual users can set their own umask in their .profile.)
TIMEOUT=60 (This will limit the number of seconds for a user to login to 60 seconds to help avoid a Denial of Service attack.)

Remove '.' from PATH environment variable. (This is an open door for Trojan Horses), and change the 'umask' environment variable to a setting of '027' in the same configuration file:

vi /etc/skel/local.profile (remove the ":" and the ".")
vi /etc/skel/local.cshrc (remove the " " and the ".")

3.5.2 Deny user access to 'crontab' and 'at'

Limit the users that can and cannot run "cron" and "at" jobs. By placing IDs in the cron.deny and at.deny files, these IDs will not be allowed to run "cron" or "at" jobs.
Place the IDs of the specific users on the system that should be allowed to run "cron" and "at" jobs in the at.allow and cron.allow files. These files should contain the root ID, plus the IDs of the System Administrators that need to execute "cron" and "at" jobs on the system.

vi /etc/cron.d/cron.allow
vi /etc/cron.d/cron.deny
vi /etc/cron.d/at.allow
vi /etc/cron.d/at.deny

By default the following accounts should appear on the deny files:

daemon
bin
smtp
listen
nobody
noaccess

Place the userid's that are "allowed" to run cron and at in the *.allow files.

Log cron activity by ensuring that 'CRONLOG=yes' setting is enabled in the /etc/default/cron file.

To test this step, setup a temporary cronjob like:

crontab –e root

Insert the following line:

** * * * /bin/date >>/tmp/date.log*

This command will write the current time and date to the /tmp/date.log file every minute on the minute. Check the end of the /var/cron/log file by viewing the log file to see if the 'date' command ran as expected. Look for error messages.

tail /var/cron/log

Remove the temporary cronjob.

3.5.3 Set /etc file permissions to deny group and others to write on /etc

This step ensures that members of the "group" that owns the /etc directory as well as the "world" group cannot write to any files in the /etc directory. This directory mainly contains configuration files for the operating system and should be protected from users with group and world writable access to them.

find /etc –type f –perm –o+w -exec chmod o-w {} \;
find /etc –type f –perm –g+w -exec chmod g-w {} \;

If these commands are successful, they will not output anything to the terminal. If there are errors, there was a typo in the command.

A list of valid group writable files can be found in Appendix B.

3.5.4 Verify need for all files with the setuid and setgid permission bits

Setuid and setgid programs execute as another userid to control aspects of the system. It is a good idea to verify these programs. To search for files with these permission bits set, use:
find / -type f \(-perm -u+s –o -perm -g+s \) –ls

A list of typical valid setuid and setgid files can be found in Appendix C.
Other setuid and setgid programs would need to be examined for authenticity by the system admin.

3.6 Logging

3.6.1 Time Services

Time services are very important for logs. It is very important that the time be synchronized within an environment. It would be best if the time is set correctly, as well, but it is more important to have the time be consistent across all machines in the environment. Set time services up by ensuring that the /etc/inet/ntp.conf file reads:

vi /etc/inet/ntp.conf

Set the following values in this file:

server <YOURNTPserver1>.<YOURDOMAIN>
server <YOURNTPserver2>.<YOURDOMAIN>
driftfile /etc/inet/ntp.drift
logconfig =all

The xntp process will run as a daemon after the next system reboot. Do not reboot at this point.

3.6.2 Log all syslog messages locally

Modify the /etc/syslog.conf file so it basically reads as follows (*** Ensure that the space in the uncommented line has TABS (not spaces!)):

vi /etc/syslog.conf

Change the file to the following:

```
# For minimal console messages, such as "SU":
auth.err                /dev/console
# display emergencies on all terminals (uses WALL)
*.emerg                 *

kern.debug              /var/log/kernlog
user.debug              /var/log/userlog
mail.debug              /var/log/maillog
daemon.debug            /var/log/daemonlog
auth.debug              /var/log/authlog
lpr.debug               /var/log/lprlog
news,uucp.debug         /var/log/newslog
cron.debug              /var/log/cronlog

## other "local" messages not yet used
local0.debug            /var/log/local0.log
local1.debug            /var/log/local1.log
local2.debug            /var/log/sudo.log
local3.debug            /var/log/tcpwrap.log
local4.debug            /var/log/ssh.log
local5.debug            /var/log/local5.log
```

local6.debug /var/log/local6.log
local7.debug /var/log/local7.log

Ensure that the above listed files exist with the following command:

touch /var/log/kernlog /var/log/userlog /var/log/maillog /var/log/daemonlog /var/log/authlog /var/log/lprlog /var/log/newslog /var/log/cronlog /var/log/local0.log /var/log/local1.log /var/log/sudo.log /var/log/tcpwrap.log /var/log/ssh.log /var/log/local5.log /var/log/local6.log /var/log/local7.log

The touch command creates empty files if the files didn't already exist.
To verify the files exist run the command:

ls –l /var/log

Now, restart the syslog daemon manually:

/etc/init.d/syslog stop
/etc/init.d/syslog start

Look for any typo errors when syslog starts. If there are no errors, the only message you will see is "syslog service starting." To test if the syslog facility is working, become root with the following command (yes, we are already root, but this will create a log entry.)

/usr/bin/su –

Look at the /var/log/authlog file. There should be an entry of the form:

cat /var/log/authlog

You should see something like:
Aug 14 12:44:15 logger su: [ID 366847 auth.info] 'su root' succeeded for root on /dev/console

Exit the "su" shell we entered two commands ago.

exit

This step allows the local machine to store syslog messages based on the category of messages into separate files for ease of parsing. This step is the key to creating a worthwhile *logger* machine. However, this step can also be used on all other UNIX servers to break up the syslog messages locally in the same manner that *logger* does.

3.6.3 Log all syslog messages to a remote logging machine.

This step would be used on all other hosts besides *logger* to send syslog messages to the syslog server *logger*.

Modify the /etc/hosts file and insert a line that reads:
X.X.X.X logger.<YOUR-DOMAIN> logger loghost

where X.X.X.X is the IP address of *logger*. This will ensure that the IP address of the syslog server is defined to the system independent of DNS.
Put the following lines in the /etc/syslog.conf file. (*** Ensure that the space in the uncommented line has TABS (not spaces!))
 # To send all debug + higher messages to loghost
**.debug @logger*

3.6.4 Log all SU attempts

Log all su (switch user) attempts to /var/log/syslog file, to the console, and to the /var/adm/messages file.

Ensure the following entries exist in the /etc/default/su file (and that they are not commented out with a "#" at the beginning of the line:
vi /etc/default/su

SULOG=/var/adm/sulog
SYSLOG=YES
CONSOLE=/dev/console

To verify this step, become your local userid from the current root shell with a '*su*' command. Check the /var/adm/sulog file to see if the attempt was actually logged.

su – travis
exit
cat /var/adm/sulog

The sulog should show that root became the user 'travis'.

3.6.5 Log third consecutive unsuccessful login attempt to /var/adm/loginlog

To detect potential brute force login attacks, we want to capture to a log when someone fails to log in more than 3 consecutive times.

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

The touch command created the log. The chown command sets root as the owner and sys as the group owner. The chmod command makes the file read-write by root only.

3.7 Services

Disable network services that are not needed (or not authorized per the approved Security Plan, if applicable). For Solaris, these services are defined in the

/etc/inet/inetd.conf file. Additionally, some services are started up from the /etc/rc3.d startup script directory. Solaris runs the startup scripts in /etc/rc2.d and /etc/rc3.d for run level 3, which is the normal run level, so we will concentrate on those startup scripts.

- Edit the /etc/inetd.conf file and comment out (by inserting a "#") all the unnecessary services. Reference http://online.securityfocus.com/infocus/1490 for some tips on what some of these services do. We will be disabling all of them except the 4 necessary for our backup solution.

  For Solaris, a good /etc/inetd.conf file will only have the following services running (which only includes backup client services):

  ```
  bpcd    stream tcp  nowait  root /usr/openv/netbackup/bin/bpcd bpcd
  vopied  stream tcp nowait  root    /usr/openv/bin/vopied vopied
  bpjava-msvc stream tcp nowait  root  \
    /usr/openv/netbackup/bin/bpjava-msvc bpjava-msvc -transient
  vnetd   stream  tcp     nowait root    /usr/openv/bin/vnetd    vnetd
  ```

  If backups were not necessary, this entire inetd.conf file could be commented out. Better yet, the inet services would not need to be started at all. Please note that the 4 entries listed above for the NetBackup client are added in a later step.

  vi /etc/inetd.conf (Comment out every line in the file)

To verify that all services are commented out, run this command:

grep -v "^#" /etc/inetd.conf

There should be no output if all lines begin with a "#" in /etc/inetd.conf.

Take special note that telnet and ftp services are not running, as ssh covers both of these functions in encrypted format.

The only services that need to be running at startup time from the */etc/rc2.d* directory are:

S01MOUNTFSYS	(mounts filesystems on startup)
S05RMTMPFILES	(removes temp files on startup)
S69inet	(TCP configuration for the machine)
S72inetsvc	(only needed for backup services)
S74syslog	(for logging locally and remotely)
S74xntpd	(time services)
S75cron	(cron jobs)
S75savecore	(in case of panics, cores are saved)
S77nddconfig	(sets our local network parameters)
S88sendmail	(for sending mail off the machine)

S88utmpd (utmp daemon)
S90sshd (secure shell / file transfer daemon)

And from the */etc/rc3.d* startup script directory are:
S76snmpdx (snmp for network monitoring)
S81volmgt (volume management for cdroms etc.)

Note: most of the above listed services will be configured later in this document.
Disable the rest by renaming the startup scripts to start with an underbar '_'

```
cd /etc/rc2.d

for i in S30sysid.net S71rpc S71ldap.client S72autoinstall
S99audit S93cacheos.finish S71sysid.sys
S73cachefs.daemon S89PRESERVE
do
  mv $i _$i
done

mv /etc/rc3.d/S16boot.server /etc/rc3.d/_S16boot.server
```

- Send inetd a SIGHUP (kill –1 $PID) to restart the daemon
  /bin/kill -HUP `/usr/bin/pgrep inetd`
- Check the configuration of the inetd services by attempting to ftp/telnet/rexec/rsh to *logger* from the local machine. None of these services should answer requests to gain access.

3.7.1 Implement inetd connection tracing

This tracing provides additional logging for inetd services. While we turned off most inetd services in the previous step, if a system admin turns them on later, we will have additional logging.
Modify /etc/init.d/inetsvc by adding –t option to the inetd command.

vi /etc/init.d/inetsvc

The new line in the */etc/init.d/inetsvc* file should look like:

/usr/sbin/inetd -s -t

3.7.2 Configure DNS (client)

Because this machine is to be a DNS client, and it is inside the firewall, create the resolv.conf file with trusted internal DNS servers and with the local domain set

and the domain to search. Then modify the nsswitch.conf file so the hosts: line reads "hosts: files dns".

vi /etc/resolv.conf
search <yourdomain>
nameserver <IP addr DNS1>
nameserver <IP addr DNS2>

vi /etc/nsswitch.conf (should have the following line:)
hosts: files dns

Lastly, lock down the file ownership and permissions.
chown root:root /etc/resolv.conf /etc/nsswitch.conf
chmod 644 /etc/resolv.conf /etc/nsswitch.conf

3.7.3 Automounter

Disable automounter. Because we aren't running NIS/NFS, we don't need automounter running. Remove any /etc/auto_* files and rename /etc/rc2.d/S74autofs to /etc/rc2.d/_S74autofs.

*mv /etc/rc2.d/*S74autofs /etc/rc2.d/_S74autofs
*rm /etc/auto**

Kill the automounter process.
/bin/kill `/usr/bin/pgrep automountd`

This step can be tested by looking for a automount process running.
ps –ef |grep automountd |grep –v grep

We should see no output if the automount processes are dead. In a later section, we will see if the startup scripts are working properly.
*ls –l /etc/auto**

There should be no automounter processes running, and there should be no files named /etc/auto*.

3.7.4 File Transfer Protocol (FTP):

Restrict ftp access to system accounts. To accomplish this, a ftp config file must be modified. Modify the /etc/ftpd/ftpusers file. Do not configure anonymous or ftp account IDs. This file defines which users are **NOT** allowed to use FTP. FTP is a **clear-text** protocol that should not be used if it is technically possible to use an encrypted protocol instead like (scp or sftp).

vi /etc/ftpd/ftpusers

Ensure the following IDs exist in this file (1 per line):
root, daemon, bin, sys, adm, lp, smmsp, listen, nobody, noaccess, and nobody4.
We can remove uucp and nuucp from this file as we have deleted those IDs earlier.

Earlier, we turned off ftp services in the /etc/inetd.conf file. This step is done as an additional safeguard for when ftp is turned on by a system administrator at a later date.

3.7.5 Mail

3.7.5.1 Mail setup for servers that are not mail relays or receive user email:

Solaris 9 ships with sendmail enabled in "-bd" (daemon) mode. Due to the historical problems with buffer overflows and root exploits in sendmail, we do not want to run sendmail in full daemon mode (one that answers to anybody that can reach our IP address). To avoid this, we need to modify the /etc/mail/sendmail.cf file and restart the sendmail daemons.

vi /etc/mail/sendmail.cf

Remove these three lines in /etc/mail/sendmail.cf

O DaemonPortOptions=Name=MTA-v4, Family=inet
O DaemonPortOptions=Name=MTA-v6, Family=inet6
O DaemonPortOptions=Port=587, Name=MSA, M=E

Add this line:

O DaemonPortOptions=Name=MTA-v4, Family=inet, Addr=127.0.0.1

and modify the line that starts with 'DS' to read:

DS<YOURMAILSERVER>.<YOURDOMAIN>

This set the SmartHost for sending mail. Lastly, stop and restart sendmail.

/etc/init.d/sendmail stop
/etc/init.d/sendmail start

3.7.5.2 Aliases

Ensure that the root and postmaster aliases go to real people (i.e. *root: First.Last@<yourdomain>*).

- *vi /etc/mail/aliases*
  Ensure there is a line that starts with "root:" and goes to a real person's

e-mail address like user@<yourdomain> by adding a line like:
root: user@yourdomain.com

- Ensure the permissions are set to 644
  chmod 644 /etc/mail/aliases
- Run the 'newaliases' command to update binary database file.
  (Alternatively, '/usr/lib/sendmail –bi' will accomplish the same thing when newaliases is phased out of new versions)
  /usr/sbin/newaliases

3.7.6 Network File System (NFS)

NFS is a **clear-text** protocol that should not be used if there is a technical alternative. For *logger,* NFS services are not needed and will be turned off.

3.7.6.1 Disable NFS server capabilities

*M*ove the startup script so it doesn't run on reboot.
mv /etc/rc3.d/S15nfs.server /etc/rc3.d/_S15nfs.server

3.7.6.2 Disable NFS client capabilities

mv /etc/rc2.d/S73nfs.client /etc/rc2.d/_S73nfs.client
(Kill any nfs processes that are currently running).
/etc/init.d/nfs.client stop
We can verify that there are no nfs processes running with this command:
ps –ef |grep nfs |grep –v grep
There should be no ouput if there are no nfs processes running.

3.7.7 Name Service Cache Daemon (NSCD)

Due to historical buffer overflow attacks, if the startup script is not already disabled completely, set the following variables to avoid buffer overflow attacks. If it exists, disable the startup script.

mv /etc/rc2.d/S76nscd /etc/rc2.d/_S76nscd

In the event that nscd is started manually, we will disable the below listed items in the configuration file.
vi /etc/nscd.conf

To do so for Solaris, add the following lines to /etc/nscd.conf:
enable-cache hosts no
enable-cache passwd no
enable-cache group no

3.7.8 SNMPD

SNMP is not running on *logger*, but in the event that it would be installed and started at a later date, we have changed the default community name to something other than 'public.' We change this because there are many tools that can exploit this vulnerability.

Edit SNMPD configuration file to reflect <pick-a-word> instead of public. This will make the community value much harder to guess.

mkdir –p /etc/snmp/conf
vi /etc/snmp/conf/snmpd.conf

Put these settings in there:

system-group-read-community *<SomethingHard>*
system-group-write-community *<SomethingElseHard>*

Change all instances of string 'public' to <pick-a-word>

3.7.9 UTMPX

Ensure that this 'last users' history file is not world or group writable.
Run '*ls -l /var/adm*' to verify permission of 644 for utmpx. If it is not set to 644, run this command:

chmod 644 /var/adm/utmpx

Repeat the 'ls –l /var/adm/utmpx' command to verify the change in permissions.

3.7.10 Networking

To help prevent SYN floods (tcp_conn_req_max_q0), ignore web server redirects (ip_ignore_redirect), turn off ICMPv4 redirects (ip_send_redirects), set IP route flush timeout to 10minutes, shorten the ARP cleanup interval to 6ms, turn off IP forwarding (ip_forward_directed_broadcasts, ip_forward_src_routed, ip_forward_src_routed, and ip_forwarding), and finally, prevent an attacker from passing packets across a machine with multiple interfaces that is not acting as a router (ip_strict_dst_multihoming to 1),
Create the following /etc/init.d/nddconfig file:

vi /etc/init.d/nddconfig

```
case "$1" in
'start')
        ndd –set /dev/tcp tcp_conn_req_max_q0 10240
        ndd –set /dev/ip ip_ignore_redirect 1
        ndd –set /dev/ip ip_send_redirects 0
        ndd –set /dev/arp arp_cleanup_interval 60
        ndd –set /dev/ip ip_forward_directed_broadcasts 0
        ndd –set /dev/ip ip_forward_src_routed 0
```

```
                ndd –set /dev/ip ip_forwarding 0
                ndd –set /dev/ip ip_strict_dst_multihoming 1

        ;;

'stop')
        ;;

*)
        echo "Usage: $0 { start | stop }"
        exit 1
        ;;
esac
exit 0
```

After creating the file, ensure the permissions are 744 and owned by root:sys, and create a link to /etc/rc2.d so the script will run when the machine boots.

```
chmod 744 /etc/init.d/nddconfig
chown root:sys /etc/init.d/nddconfig
ln –s /etc/init.d/nddconfig /etc/rc2.d/S77nddconfig
/etc/init.d/nddconfig start
```

Verify there were no errors when running the script. If there are no errors, there will be no output.

To ensure this machine is not used as a bridge if another network interface is configured, disable IP forwarding.

```
touch /etc/notrouter
```

3.7.11 Prevent TCP sequence prediction attacks

Change the /etc/default/inetinit file the so the "TCP_STRONG_ISS" line is set to '2'.

```
vi /etc/default/inetinit
```

Enabling this setting will make it more difficult for hackers to gain access to the machine by making the prediction of TCP sequence numbers harder to predict.

3.7.12 Prevent any code from executing in the stack

This helps make buffer overflow attacks more difficult. This is accomplished by adding the following lines to /etc/system:

```
vi /etc/system
        set  noexec_user_stack=1
        set  noexec_user_stack_log=1
```

These variables will take effect on the next reboot of the machine. Do not reboot at this time.

3.8 Additional Packages

3.8.1 Sudo

Download, install and configure sudo. Sudo is the one of the top utilities (in the author's opinion) to help trace privileged commands to a specific user ID. It offers a wide range of flexibility to configure who is allowed to run which commands.

If you haven't already, download sudo from http://www.sunfreeware.com/programlistsparc9.html, and place on a compact disc from another machine connected to the Internet.

If the CD isn't already mounted, mount it and copy the compressed file to /tmp:

```
mount –o ro –F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
cd /mnt/cdrom
cp sudo-1.6.7p5-sol9-sparc-local.gz /tmp
cd/tmp
gunzip sudo-1.6.7p5-sol9-sparc-local.gz
pkgadd –d ./sudo-1.6.7p5-sol9-sparc-local
/usr/local/sbin/visudo   #(to create '/usr/local/etc/sudoers' )
```

The sudoers file that is created here is the main control mechanism that describes who gets the authority to run specific commands.

The visudo command puts the administrator in the vi editor so he/she can modify the sudoers file. When the administrator exits this editor, the visudo command checks the validity and syntax of the sudoers configuration file.

Here is a sample sudoers file that will give full root authority (save the shells). Replace the stock sudoers file with the following one and modify it appropriately for your environment. Change the usernames and command aliases appropriately to your environment.

```
#User aliases are first.
#These are a list of users on the system that can all be grouped together.
#Unix Support Team
User_Alias UST=travis,stan,benj,stever

#LoggerAdmins used an example for other user groups.
User_Alias LoggerAdmins=hope,youse,gyes,rgood

#Command aliases group a list of command together in a group
#A whole group of command can be given to a user or a "user alias" at one time.
# Cmnd alias specification
Cmnd_Alias      SHELLS=/bin/sh,/bin/jsh,/sbin/sh,/bin/csh,/bin/tcsh,\
```

```
        /bin/bash,/bin/ksh,/usr/local/bin/ksh,\
        /usr/local/bin/tcsh,/usr/local/bin/bash,/bin/zsh,\
        /usr/local/bin/zsh,/usr/bin/zsh,/usr/bin/csh,\
        /usr/bin/tcsh,/usr/bin/sh,/usr/bin/jsh,\
        /usr/bin/bash,/usr/bin/rsh,/bin/rsh,/bin/remsh,\
        /usr/bin/remsh

Cmnd_Alias      SU=/usr/bin/su,/bin/su,/sbin/su,/usr/bin/newgrp,/bin/newgrp

Cmnd_Alias      PASSWD_CMDS=/usr/bin/passwd ?*,/bin/passwd ?*,!/usr/bin/passwd root,\
        !/usr/bin/passwd ?*su,!/usr/bin/passwd ?*root*,!/bin/passwd root,\
        !/bin/passwd ?*su,!/bin/passwd ?*root*

Cmnd_Alias LOG_CMDS=/etc/init.d/syslog stop, /etc/init.d/syslog start

# Host alias specification

# User privilege specifications
#Root has all access on all machines to run all commands
root            ALL=(ALL) ALL

#The Unix Support Team can run ALL commands except the shells as
#  any user on any machine
UST             ALL=(ALL) ALL,!SHELLS

#The sample LoggerAdmins group can run web commands.  We also explicitly list
# that they cannot run shells, or switch users, or change passwords in the event that
#someone later adds "ALL" privileges.
LOGGERADMINS    ALL=LOG_CMDS,!SHELLS,!SU,!PASSWD_CMDS

#user2 can run ALL commands except shells, switch user, or password commands
# as any user on any machine.
user2           ALL=ALL,!SHELLS,!SU,!PASSWD_CMDS
```

To test sudo, run the command "/usr/local/bin/*sudo –l"* to see if the output shows the allowed commands for the currently logged in user.

3.8.2 TCPWrappers

TCPWrappers offers fine-grained control over who can and cannot access services (that are TCPWrappers aware) on your server. The granularity is down to the host IP address level. This means that the SA can pre-define which hosts are allowed to connect with pre-defined protocols to this server via the /etc/hosts.allow and /etc/hosts.deny files. Solaris 9 comes with TCPWrappers pre-installed (but in a disabled state). We also added the utilities when we first installed the Operating System. To enable TCP wrappers, we must modify the /etc/default/inetd file.

vi /etc/default/inetd

Remove the "#" at the beginning of the line. Ensure there is a line that reads:

ENABLE_TCPWRAPPERS=YES

To implement the new /etc/inet/inetd.conf, send a restart signal to inetd.
/bin/kill -HUP `/usr/bin/pgrep inetd`

Next, we need to create the /etc/hosts.allow file and populate it with information about the machines that are allowed to connect to *logger*. Modify the data in the lines that start with "sshd:" to specify your hosts that should be allowed to ssh to *logger*. The author recommends leaving the telnet and ftp lines commented out as we previously disabled the services in the /etc/inetd.conf file.
vi /etc/hosts.allow

(insert machines Fully-Qualified-Domain-Name(FQDN) or IP addresses of the System Administrators and Applications Support Personnel that are allowed to connect via each protocol to this UNIX server.) Here is an example /etc/hosts.allow file:

```
# A "#" is a comment in this file.  These next lines are
# to be used in emergencies if sshd is dead for some reason.
# They would have to be un-commented after the clear-text
# protocol was manually started
#in.telnetd:  10.1.1.1
#in.ftpd:     10.1.1.1

#Sysadmins
#change these lines to reflect your actual hosts on your network
sshd: Sahost1.<mydomain.com>
sshd: Sahost2.<mydomain.com>

#Applications Support Personnel
#change these lines to reflect your actual hosts on your network
sshd: AspHost1<mydomain.com>
sshd: AspHost2<mydomain.com>
```

Next, we need to create the /etc/hosts.deny file and populate it with our default deny rule. All attempts will be rejected if the connection gets this far.

vi /etc/hosts.deny

There should be a single line that reads:

ALL:ALL

Run the tcpdchk script to check implementation.

/usr/sfw/sbin/tcpdchk

You should see a warning response about sshd not being a known process in inetd. Because sshd was compiled (and will be installed in a later step) with TCP wrapper support enabled, we can safely ignore this warning.

We will test TCP wrapper usage in a later section.

3.8.2.1 Install Gnu C libraries(libgcc)

We should have already download the latest libgcc version for Solaris 9 from http://www.sunfreeware.com/ and/or ftp://sunsite.unc.edu/pub/solaris/sparc. This should be placed on a CD and copied to logger via CD.

cp /mnt/cdrom/libgcc-3.3-sol9-sparc-local.gz /tmp
cd /tmp
gunzip libgcc-3.3-sol9-sparc-local.gz
pkgadd –d libgcc-3.3-sol9-sparc-local
rm libgcc-3.3-sol9-sparc-local

These libraries are needed for other programs like SSH (installed next).

3.9 Secure Shell (SSH)

Secure shell offers secure connections to UNIX hosts by encrypting the channels used to communicate with the UNIX host. It is a replacement tool for undesirable clear-text protocols like telnet, ftp, rlogin, rsh, rcp, etc. In fact, these clear-text protocols can and should be disabled after installing ssh. Clear-text protocols transmit sensitive information in such a way that the data can be eavesdropped or otherwise captured or hijacked. Secure shell also offers secure tunnels for other protocols like X session traffic.

3.9.1 Secure Shell (SSH) Server

Solaris 9 has a version of SSH available on the install media. The author recommends not using this version because the version from Sun is older than OpenSSH 3.6.1 at this time.

Implement OpenSSH by doing the following:

(Additional locations to acquire the below listed files are:
ftp://ftp.sunfreeware.com/pub/freeware/SOURCES or the faster mirror located at
ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9)

PATH=$PATH:/usr/local/bin:/usr/local/sbin
export PATH

We have previously acquired the following packages from ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9 and placed them on a CD.

openssh-3.6.1p1-sol9-sparc-local.gz
openssl-0.9.7b-sol9-sparc-local.gz
zlib-1.1.4-sol9-sparc-local.gz
logrotate-3.6.9-sol9-sparc-local.gz
popt-1.7-sol9-sparc-local.gz

For each of the above listed packages, copy the compressed file to /tmp
cp /mnt/cdrom/<package_name> /tmp

Uncompress all of the packages with one command:

cd /tmp
*gunzip *.gz*

If there are no errors unzipping, there will be no output from the previous command.

Install zlib

pkgadd -d zlib-1.1.4-sol9-sparc-local

Install openssl

pkgadd -d openssl-0.9.7b-sol9-sparc-local

Install openssh

pkgadd –d openssh-3.6.1p1-sol9-sparc-local

Configure sshd by modifying the /usr/local/etc/sshd_config file.
vi /usr/local/etc/sshd_config

Set the following variables and ensure they aren't commented out with a "#":

Protocol 2
PermitRootLogin no
X11Forwarding yes

Configuring the above settings disables the less-secure ssh version 1 protocol, disallows direct root logins via ssh, and finally allows X traffic to flow through encrypted channels instead of the usual clear-text protocol.

OpenSSH 3.6.1 has a privilege separation feature that helps defeat buffer overflow attacks to gain root access to the machine. We need to create a non-privileged id on the machine to enable this feature.

Create the privilege separation user ID and group ID including the directory creation, ownership, and permissions:
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
passwd –l sshd

Create ssh version 2 host keys:
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 ""

Create start/stop scripts. Because the package from sunfreeware.com does not include the startup/shutdown scripts, we will need to create our own.

Place the script in /etc/init.d and create a link to it in /etc/rc2.d
vi /etc/init.d/sshd

The file should look like this

```
#!/bin/sh
#script to start/stop sshd

pid=`/usr/bin/ps -e | /usr/bin/grep sshd | /usr/bin/sed -e 's/^ *//' -e 's/ .*//'`
case $1 in
'start')
        /usr/local/sbin/sshd
        ;;
'stop')
        if [ "${pid}" != "" ]
        then
                /usr/bin/kill ${pid}
        fi
        ;;
*)
        echo "usage: /etc/init.d/sshd {start|stop}"
        ;;
esac
```

The above startup script allows the ssh daemon to start when the machine is booted. We now need to set the permissions on the startup script and link it to the /etc/rc2.d directory so it is run at start time. And lastly, manually start the ssh daemon on the currently running machine.

chmod 744 /etc/init.d/sshd
chown root:sys /etc/init.d/sshd
ln –s /etc/init.d/sshd /etc/rc2.d/S90sshd
/etc/init.d/sshd start

Check to see if the ssh daemon is running:
ps –ef |grep ssh

We should see something like:
root 400 1 0 13:07:15 ? 0:00 /usr/local/sbin/sshd

Test to see if ssh is working:
ssh localhost

We should see a message like:

```
The authenticity of host 'localhost (127.0.0.1)' can't be established.
RSA key fingerprint is c0:8f:d3:4f:cb:94:49:aa:50:5c:ef:05:ba:f9:33:c0.
Are you sure you want to continue connecting (yes/no)? yes
```

Warning: Permanently added 'localhost' (RSA) to the list of known hosts.

Note that root ssh will not work as we configured the ssh daemon to not allow this activity. Try to ssh to localhost using a valid local ID.

ssh travis@localhost
exit #To get back to the root shell

We will test the privilege separation process in a later step.

3.9.2 Secure Shell (SSH) Clients

To use ssh from a windows machine, install a program called '*putty*' (available at ***http://www.chiark.greenend.org.uk/~sgtatham/putty/*****).** Putty is a free win32 telnet/ssh client. It also includes pscp and psftp, which allows secure file transfers between the Windows client and the UNIX server.

An alternative to putty is Teraterm. Teraterm and ttsh must be installed on the client machines. This software can be downloaded from:
http://hp.vector.co.jp/authors/VA002416/ttermp23.zip

3.10 Fix-Modes

The fix-modes software rectifies various permissions and ownership issues on files all through the Solaris Operating Environment. This program should be re-run every time packages or patches are added to the machine. It is a good idea to run this program from cron on a regular basis.
Download a pre-compiled version of fix-modes from ftp://ftp.CISecurity.org/pub/pkgs/Solaris/fix-modes.tar.Z.
Uncompress the software and install it in /usr/local/bin:

cp /mnt/cdrom/fix-modes.tar.Z /tmp
uncompress fix-modes.tar.Z
tar –xvf fix-modes.tar
cd fix-modes
cp secure-modes /usr/local/bin
cp fix-modes /usr/local/bin
/usr/local/bin/fix-modes

This process will complain about several files that do not exist. Therefore the fix-modes program cannot change the permissions. These messages can be safely ignored. Now we need to add a daily crontab entry so the file permissions are set appropriately and remain that way (at least once per day).

crontab –e root

Insert a line like:

*3 3 * * * /usr/local/bin/fix-modes -q >/dev/null 2>&1*

The above crontab entry will run the fix-modes program every night at 3:03 am.

3.11 Intrusion Detection: Tripwire

We will be installing tripwire indirectly based on the README file included in the source download. The purpose of installing Tripwire is for file integrity protection. Tripwire is a utility tool that compares a designated set of files and directories against information stored in a previously generated database. When tripwire is run against system files on a regular basis, any changes in critical system files will be visible, and appropriate damage control measures can be taken. We will have to create a package for Solaris 9 on another machine that has all the necessary header files. Use these instructions to compile the source code. The author has gone a step further and created a Solaris "package." This package can also be placed on the CD used to install the machine. The quick install is to acquire the package tripwire-1.3.1-sol9-sparc-local-TJH from http://www.jedi.net/travis/security and run:

cd /mnt/cdrom
pkgadd -d tripwire-1.3.1-sol9-sparc-local-TJH

Now skip to step 7 below if you already have the package listed above.

Obtaining Tripwire software:
Download the academic version from http://www.tripwiresecurity.com or more specifically, http://www.tripwire.com/downloads/tripwire_asr. This will download the Academic Source Release. Please check the license agreement included in the package. It may not be appropriate to use at your site. An alternative intrusion detection software that could be used is AIDE. Save the tripwire package to a CD or other removable media if this has not already been done.

2) Transfer the package to /tmp:

cd /tmp
cp /mnt/cdrom/Tripwire-1.3.1-1.tar.gz /tmp
gunzip Tripwire-1.3.1-1.tar.gz
tar –xvf Tripwire-1.3.1-1.tar
cd tw_ASR_1.3.1_src

3) Modify Makefile:

INSTALL= /usr/ucb/install
HOSTNAME= 'uname –n'

4) Modify /opt/local/tw_ASR_1.3.1_src./include/config.h file:
Confirm the following statement is included in the ./include/config.h file:

include '../configs/conf-svr4.h'

Confirm the definition of CONFIG_PATH and DATABASE_PATH:
#define CONFIG_PATH 'usr/local/bin/tw'
#define DATABSE_PATH 'var/tripwire'
Confirm the definition of CONFIG_FILE and DATABASE_FILE:
#define CONFIG_FILE 'tw.config'
#define DATABSE_FILE tw.db_@

5) Customize *./config/tw.conf* based on your needs for file protection.
mkdir -p /usr/local/bin/tw
cp ./configs/tw.conf.sunos5 /usr/local/bin/tw/tw.config
A sample *tw.config* file is included in the note on the previous line.
vi /usr/local/bin/tw/tw.config and put the contents of the above listed note into the file.

6) Perform the following steps to configure and install the package:
make
mkdir –p /usr/man/man8
make test
make install

7) Create initialization database and set permissions so only root can read and write the database file:
cd /tmp
/usr/local/bin/tripwire –initialize
This will create a datafile file called *./databases/tw.db_<hostname>*
mv ./databases/tw.db_<hostname> /var/tripwire
chmod 600 /var/tripwire/tw.db_<hostname>

8) Make a copy of the tw.db_<hostname> to removable media. When *logger* is put on the network, we could *scp* the file to a remote host with a CDROM burner.

Here is an example of copying the initial database to a tape (if you have a local tape drive):
tar cvf /dev/rmt/0 ./database

9) Install a crontab to run the *tripwire* program in query mode at least once per day and log the output to your syslog server. If the package was used to install the program, this crontab entry will already exist. The specific command to put in root's crontab is as follows:

*4 4 * * * /usr/local/bin/tripwire -q |/bin/logger -p local5.notice -t TRIPWIRE*

Because we are logging the output of the *tripwire-q* command daily, the syslog logs will contain the changes to the files monitored by *tripwire*. In the next step, we install the *logcheck.sh* software that will mail us the logs hourly. Be sure to

check the first log message of the day for the tripwire logs. They will only appear once per day due to the settings of the above listed cronjob. This is an ongoing task that should be done daily.

3.12 Logcheck

Logcheck is a program that will analyze the syslog-style logs on any UNIX machine. It is designed to run from cron and works by eliminating the routine "boring" items from the logs. What remains are the "interesting" logs that are e-mailed to a person to be specified in the configuration file.

Start by acquiring the Logcheck package from http://www.sunfreeware.com/programlistsparc8.html#logcheck if you have not already pre-fetched the package. Place this package on a CD. This package can be installed using the pkgadd command as root.

```
cp /mnt/cdrom/logcheck-1.1.1-sol8-sparc-local.gz /tmp
cd /tmp
gunzip logcheck-1.1.1-sol8-sparc-local.gz
pkgadd –d  logcheck-1.1.1-sol8-sparc-local
```

If needed, the source code is available for download from http://sourceforge.net/projects/sentrytools. If you have to install from source code on another Solaris 9 machine, the basic installation/configuration steps are as follows:

```
gunzip logcheck-1.1.1.tar.gz
tar -xvf logcheck-1.1.1.tar
cd logcheck-1.1.1
make sun
make install
```

At this point, the logcheck programs have been installed in /usr/local/etc. This document will briefly touch the major settings that need to be modified for our specific system. More information is available in the README file from the source directory "logcheck-1.1.1".

Because we have previously setup the /etc/syslog.conf file appropriately, we need to configure *logcheck.sh* to point to where our specific logs are located. These are the files that Logcheck will parse hourly from a root cron job we will also be configuring in this section. Use your favorite text editor to modify the *logcheck.sh* script located in */usr/local/etc*. For example:

```
vi /usr/local/etc/logcheck.sh
```

Locate the section that starts with "$LOGTAIL " for the OS (Solaris). Replace the lines that are uncommented with the following lines that match our syslog.conf configuration:

```
$LOGTAIL /var/log/syslog > $TMPDIR/check.$$
$LOGTAIL /var/log/tcpwrap.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/sudo.log >> $TMPDIR/check.$$
$LOGTAIL /var/adm/messages >> $TMPDIR/check.$$
$LOGTAIL /var/log/kernlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/userlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/maillog >> $TMPDIR/check.$$
$LOGTAIL /var/log/daemonlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/authlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/lprlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/newslog >> $TMPDIR/check.$$
$LOGTAIL /var/log/cronlog >> $TMPDIR/check.$$
$LOGTAIL /var/log/local0.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/local1.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/local4.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/local5.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/local6.log >> $TMPDIR/check.$$
$LOGTAIL /var/log/local7.log >> $TMPDIR/check.$$
```

While in the same file, find the section entitled: CONFIGURATION SECTION. Change the name of the SYSADMIN variable to a real person. For example:

SYSADMIN=YourFirst.LastNAME@<DOMAIN.COM>

If an alias was configured for root (as required in an earlier section), the default ID of 'root' would actually be sent to a real person. It is now time to modify the logcheck.hacking, logcheck.violations, logcheck.violations.ignore, and logcheck.ignore files located in /usr/local/etc to your company's specific needs. These files have a good start for items that are "less-than-interesting", but will need to be tweaked to meet your specific needs. These files are explained in the README file in the source directory. Basically, they contain patterns of things to search for in the logs. The *.ignore files are things to weed-out. The *.hacking file is things to specifically look for that indicate potential hacking activity.

Lastly, we need to add a cronjob schedule so *logcheck.sh* will run on a regular schedule.

crontab –e root #add the following lines:

```
0 8,9,10,11,12,13,14,15,16,17 * * 1,2,3,4,5 /usr/local/etc/logcheck.sh
55 23 28,29,30,31 * * /usr/local/etc/logcheck.sh
```

The two lines above will run the logcheck.sh every hour on the weekdays between 8am and 5pm, which correlates to a historically typical workday. Obviously, this can be adjusted to your particular work schedule. The 2nd line is set to run at 11:55pm on the last days of every month. This is a one-line quick fix to accommodate a monthly log rotation (next section) that would occur on the 1st of every month at 12:00am. It does produce up to 3 extra e-mails in a given month,

but covers all but 5 minutes of logs for every month assuming log rotations occur as described above. The reason for the 5-minute time frame is that it does take some time to parse the logs with the logcheck.sh script. An assumption is made that it will take less than 5 minutes to complete its run.

Lastly, it is recommended that the logcheck.sh script be run manually to see if there are any errors that need to be corrected. If there are no errors and there are actual logs to be parsed, an e-mail will be generated to the ID specified in the logcheck.sh script. Again, check the README for more detailed instructions.

/usr/local/etc/logcheck.sh

The output from the logcheck.sh root crontab should be analyzed on an ongoing basis. Investigations of unusual events will be necessary to determine if there is malicious activity, or if the *.ignore files need to be modified for this new "normal" syslog activity.

3.13 Logrotate

Logrotate is a program designed to rotate logs. The binary package and required popt program can be acquired from ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9. The filenames are: logrotate-3.6.9-sol9-sparc-local.gz and popt-1.7-sol9-sparc-local.gz. Copy these files to /tmp from the CD and pkgadd the program.

cp /mnt/cdrom/logrotate-3.6.9-sol9-sparc-local.gz /tmp
cp /mnt/cdrom/popt-1.7-sol9-sparc-local.gz /tmp
cd /tmp
gunzip logrotate-3.6.9-sol9-sparc-local.gz
gunzip popt-1.7-sol9-sparc-local.gz
pkgadd –d logrotate-3.6.9-sol9-sparc-local
pkgadd –d popt-1.7-sol9-sparc-local

The source code is available from http://software.stanford.edu/pub/Solaris/freeware/SOURCES/. Since we did not install necessary headers and compilers on this production machine to compile code, we would have to compile the source code on a different Solaris 9 machine.

This simple program contains one basic configuration file and will require a crontab entry to start the process.

vi /usr/local/etc/logrotate.conf

For *logger*, the /usr/local/etc/logrotate.conf file will basically look like this:

```
# see man logrotate for details
# rotate log files weekly
monthly
create 640 root sys
# keep 12 months worth of backlogs
rotate 12
```

```
# send errors to somebody
# modify the next line to reflect your e-mail address
errors <YOUR USERID>@<DOMAIN.COM>
# create new (empty) log files after rotating old ones
create
# uncomment this if you want your log files compressed
compress
# Common Solaris system files and settings
/var/adm/wtmpx {
# multiple of 372 bytes (each record size for solaris)
    size=363k
    rotate 4
}

/var/adm/wtmp {
    size=363k
}

/var/adm/lastlog {
    size=300k
    rotate 1
}
/var/log/alertlog {
}
/var/log/authlog {
}
/var/log/cronlog {
}
/var/log/daemonlog {
}
/var/log/kernlog {
}
/var/log/local0.log {
}
/var/log/local1.log {
}
/var/log/local4.log {
}
/var/log/local5.log {
}
/var/log/local6.log {
}
/var/log/local7.log {
}
/var/log/lprlog {
}
/var/log/maillog {
}
/var/log/newslog {
}
/var/log/ntp.log {
}
/var/log/sudo.log {
}
/var/log/sysidconfig.log {
}
/var/log/syslog {
}
/var/log/tcpwrap.log {
}
/var/log/userlog {
    postrotate
        rm -f /var/log/*offset
        kill -HUP `cat /etc/syslog.pid`
    endscript
}
```

There is an entry for each syslog file on *logger*. At the end of this configuration file, we find a brief section that will run after the rotation is done. This step will re-initialize the *.offset files for the logcheck.sh script that runs hourly (from the

section above). This way, the next time the logcheck.sh script runs, it will know to start from the beginning of the log files. The postrotate script also restarts the syslog daemon so it will start logging to the newly created files. The final step is to create a root crontab entry to run at midnight on the first of each month.
crontab –e root

Add the following line:
*0 0 1 * * /usr/local/bin/logrotate /usr/local/etc/logrotate.config*

Logs will now be kept for a period of 1 year. Each month will have its own logs in a gzipped format. Additionally, the logs will be backed up to the backup server for longer term storage.

Unmount the cdrom and eject it.
cd /
umount /mnt/cdrom

3.14 Backups

As stated previously, our corporate backup strategy utilizes Veritas NetBackup. The backup server is a separate machine. For *logger*, we need to install the client so the backup server can pull backups. The original NetBackup 4.5 CD media is necessary for this step. Follow the install instructions for installing the client on *logger.*
Mount the CD-ROM, and run the "*install*" program.

mount -o ro -F hsfs /dev/dsk/c0t6d0s0 /mnt/cdrom
cd /mnt/cdrom
./install

Select option 2 to "Install NetBackup Client". Part of the install asks a question about the name or IP address of the master backup server. Enter this info for your master backup server. We can now unmount the cd and eject it.
cd /
umount /mnt/cdrom

The install script modifies the /etc/inetd.conf and the /etc/services file for all needed daemons. The following 4 lines are added to the /etc/inetd.conf file:

```
bpcd    stream  tcp     nowait  root    /usr/openv/netbackup/bin/bpcd bpcd
vnetd   stream  tcp     nowait  root    /usr/openv/bin/vnetd vnetd
vopied  stream  tcp     nowait  root    /usr/openv/bin/vopied vopied
bpjava-msvc stream tcp nowait root /usr/openv/netbackup/bin/bpjava-msvc \
                                    bpjava-msvc -transient
```

Modify the /etc/hosts.allow for these services so our backup server can access them:
vi /etc/hosts.allow

Add these lines:

bpcd:<hostname-of-backup-server>
vnetd:<hostname-of-backup-server>
vopied:<hostname-of-backup-server>
bpjava-msvc:<hostname-of-backup-server>

Lastly, have the backup administrator set a policy and a schedule for backing up this machine (on the master backup server.)
For *logger*, a good suggested backup schedule is to do full backups once per week on a Saturday for all filesystems on the machine. For the remainder of the days, an incremental backup is sufficient. Keep in mind that the majority of the "data" for this machine is contained in the /var partition (specifically /var/log). Depending on the syslog usage, these files could grow quite fast. Alternatively, if you have a small set of machines logging to *logger,* they won't grow fast at all. Logging approximately 70 UNIX machines for 1 year has resulted in about 2.5 gigs of logs (keep in mind that 11 months of those are in a compressed gzip format.)

3.15 EEPROM

Obtain one or more copies of the system's PROM, particularly if one or more software products are bound to the system's hostid.

```
eeprom >/eeprom.txt
chmod 600 /eeprom.txt
```

Set EEPROM security-mode to 'command' to further prevent physical access attacks.

```
eeprom security-mode=command
```

Note that if EEPROM security-mode is set to 'full', a password will need to be entered to boot the machine from disk. If the password is lost or forgotten, a new EEPROM chip will need to be installed on the hardware. Any software that is tied to the hostid on that EEPROM will need to be re-licensed.

3.16 Put *logger* on the network

At this point we are ready to plug the machine into the network. To ensure the machine comes up ok, we will shut the machine down completely. It is not necessary to power the machine off, simply bring it down to run level 0. Once it is down, plug in the network cable and turn the machine on.

```
init 0
```

When the machine gets to the 'ok' prompt, type:

```
boot
```

Watch the screen as it comes up for errors. Once the machine is up, login and look in some of the logs for any errors.

/usr/local/bin/*sudo cat /var/log/kernlog /var/log/daemonlog*
/usr/local/bin/*sudo cat /var/adm/messages*

3.17 Validate configuration

Refer to Section 5.0 to validate the configuration of *logger*.

4.0 ONGOING MAINTENANCE

We can now login with our individual account and use the sudo function to look at logs and configure other services as needed. This should be the normal way to log in to the machine from now on. Usage of the root ID should be limited to emergency repair situations only to provide adequate individual accountability.

4.1 SecurityCheck.pl – (At least Monthly +after patch application)

As described in section 5.1, the SecurityCheck.pl will be run from cron. This monthly cronjob will send output to the System Admin on the 1st of every month. The output should be seriously analyzed. If there are any "Errors" in the output, these items should be remediated. The script can also be run manually with a command like:

/usr/local/bin/sudo /usr/local/bin/SecurityCheck.pl

by a user with sufficient sudo authority to run the command. This script should also be run any time patches are applied to the system or a new package is installed to help ensure that previously configured items have not been undone!

4.2 Verify syslog is working from remote machines. (Daily)

To accomplish this task, we will need to configure some remote UNIX or network devices to send their syslog messages to *logger*. The section above entitled "Log all syslog messages to a remote logging machine" described the procedure to accomplish this task.

From the remote machine, run the command:

/bin/logger -p local5.notice -t test test123

On *logger,* there should be a new line at the end of the /var/log/local5.log file. This test can be repeated for each facility in the syslog.conf file on *logger*.

tail –f /var/log/local5.log

We should see a line like:
Aug 21 16:36:47 aacsec1 test: [ID 702911 local5.notice] test123

This test can be set to run from cron on a remote machine. View the e-mail for that hour's *logcheck.sh* output to verify the syslog test arrived on logger.

4.3 *logcheck.sh* output (Hourly)

The output from the logcheck.sh root crontab should be analyzed on an ongoing basis. Investigations of unusual events will be necessary to determine if there is malicious activity, or if the *.ignore files need to be modified for this new "normal" syslog activity. Additionally, it is a good idea to look at the original logs on *logger* to see that there is still activity being logged. We are now relying on the logcheck.sh program to filter logs, but haven't definitively determined that logs are still being populated. This ongoing task will be to examine the hourly e-mail messages from the logcheck.sh program. The *.ignore files will need to be modified to filter out more and more "boring" items in the syslogs as new services and programs are added to the machines being monitored by *logger*. Another reason to periodically look at the original logs is to determine that the *.ignore configuration files are still adequate. We don't want to filter out items that should be 'interesting.'

A sample of the hourly e-mail is located in Appendix I.

4.4 Detecting Denial of Service of the syslog facility (Hourly)

We are now getting hourly e-mails during work hours. We can quickly determine if there is an unusual amount of syslog information by looking solely at the size of the e-mails generated. You will quickly get a feel for what a normal hourly e-mail looks like. If there is an unusual spike in the size, some investigation will be quickly required to determine what is happening on your network.

It would also be a good idea to look at the filesystems on *logger* to monitor for available filesystem space. When the /var filesystem is bombarded with a denial-of-service, the filesystem will fill faster than normal. You will quickly get a feel for how quick your logs are growing on a daily/weekly basis. It would be a good idea to set up a cronjob to send e-mail on a daily basis with the size of the /var filesystem.

/usr/local/bin/sudo crontab –e

Add a line like:

```
5 23 * * * /bin/df –k |/bin/mailx root >/dev/null 2>&1
```

This will mail a *df* listing at 11:05 every evening to the root e-mail alias (which was configured to go to a real person earlier.) Pay close attention to the size of the /var filesystem. If the filesystem grows faster than normal for your environment, take a closer look at the logs to see which log is filling up the filesystem.

An alternative syslog service like syslog-ng (previously discussed) could be used instead of the Sun syslog service. It is also possible to install the *logger* machine behind another "security" firewall so the offending machine could be blocked at the firewall from sending syslog messages to *logger*. In our environment, we have accepted the risk of denial of service attacks because *logger* is already inside our corporate firewall.

4.5 Maintain Backups (Daily)

For *logger*, a good suggested backup schedule is to do full backups once per week on a Saturday for all filesystems on the machine. For the remainder of the days, an incremental backup is sufficient. Keep in mind that the majority of the "data" for this machine is contained in the /var partition (specifically /var/log). Depending on the syslog usage, these files could grow quite fast. Alternatively, if you have a small set of machines logging to *logger,* they won't grow fast at all. Logging approximately 70 UNIX machines for 1 year has resulted in about 2.5 gigs of logs (keep in mind that 11 months of those are in a compressed gzip format.)
Because logs can be used in legal arenas, they should be preserved longer than 12 months. The backup strategy should retain the tapes of the logs for a period of 7 years to meet most legal obligations.
It is important to verify the success of weekly and daily backups to maintain the integrity of the data. It would be a great idea to periodically (once per quarter) restore a tape and verify that the data is readable.

4.6 Tripwire (Daily)

We configured tripwire ASR in such a way that it is automatically run every day. The output of this is sent to the syslog facility. The *logcheck.sh* script will parse the syslogs and mail the 'interesting' items. All tripwire output is 'interesting' so it should not be filtered out with the /usr/local/etc/*.ignore logcheck filters. The tripwire output will ultimately be delivered to your e-mail server. This output should be analyzed daily to look for items that have changed on your system. If the changes were expected, the tripwire database can be updated to reflect the expected changes using the procedures previously discussed to create the database. If other files have changed on the machine beyond the expected ones, you should start an investigation into the cause of the file changes. Sometimes, cron jobs and at jobs affect other files on the machine. The worst case scenario is that someone has hacked into our *logger* machine and has started taking control of the machine.
We should check the output daily and periodically (monthly) check the machine manually by running "*tripwire –q"* while using the read-only copy of the tripwire database.

4.7 Security Vulnerability Scans (Quarterly)

It is a good idea to run a current vulnerability scan against the *logger* machine on a regular basis. This should be run at least once per quarter. The author recommends a better schedule of monthly. Vulnerability scanners are discussed in more depth in section 5. The results of the scan should be scrutinized. Any errors that show up should be remediated. New vulnerabilities are found every day and added to these vulnerability scanners.

4.8 Mailing Lists (Constantly)

Get the latest CERT advisories at http://www.cert.org/advisories/ as well as Solaris Security Bulletin reports and check for any last-minute security holes that must be remedied.

Subscribe to two or more security incident mailing lists and read all of the advisories that arrive. A good starting point is http://www.sans.org/sansnews and clicking on "subscriptions" for SANS lists and http://xforce.iss.net/xforce/maillists, which contains the ISSFORUM@iss.net mailing list. http://xforce.iss.net/xforce/maillists/otherlists.php contains links to other mailing lists like BugTraq, and the COAST Security Archive. Last, but not least check with the OS vendor for security patches. For Sun Solaris, look at the http://sunsolve.sun.com/security web page. Check with Security Services to see what they have on their security incident mailing.

This step is an ongoing step as new vulnerabilities are found and fixed every day. It is important to keep up-to-date on the latest security patches available.

4.9 Ping Monitoring (Constantly)

It is a good idea to use a simple tool from another machine to periodically ping *logger* to see if it is still on the network. A good fault management system could be used as well. The basic idea is that we want to be notified if *logger* falls off the network. A quick script can be written to watch for a failed ping and let you know via e-mail to your phone/pager/e-mail system the status of *logger*.
A combination of scripts that look like this could be run from a remote machine on the same network.

```
#!/bin/ksh
echo "waitfor _NO_ ping $1"
while (( 1 ))
do
  ping -c3 $1
  if [ $? -eq 1 ]
  then
    /bin/mail -s "$1 is down..."  YOURPHONENUMBER@YOURCARRIER.COM, \
                                  travis@YOURCOMPANY.COM \</dev/null
    exit 0
  fi
  sleep 10
done
```

4.10 Patches (Monthly)

Monthly, install all required patches and all security patches that are currently available. Reference http://sunsolve.sun.com/pub-cgi/show.pl?target=patches/patch-license&nav=pub-p. There are two basic tools to accomplish this goal. One is a fairly new product called PatchPro (aka Patch Manager) available from Sun at https://sunsolve.sun.com/patchpro/patchpro.html. This program will be a mature viable option soon. As of this writing, there are a few bugs that Sun is working out (namely, aborted downloads if a patch is not currently available on the Sun's site). The author recommends using the patchdiag tool (aka patchcheck) also available at Sun's website http://sunsolve.sun.com/pub-cgi/show.pl?target=patchk. After downloading the patchcheck_1.2.tar.Z program from another machine, use ssh to secure copy the file to *logger*. Place this file in /usr/local and uncompress and untar the file.
cd /usr/local
uncompress patchcheck_1.2.tar.Z
tar –xvf patchcheck_1.2.tar

This will create three files in /usr/local called: COPYRIGHT, userguide, and patchck.pl. The above instructions are a one-time install. The remainder of this section should be done monthly to keep up-to-date on patches.

- Retrieve the patchdiag.xref file from http://sunsolve.sun.com/pub-cgi/patchDownload.pl?target=patchdiag.xref&method=H
  And use ssh to securely copy this file to *logger*. Next copy the patchdiag.xref file to /usr/local/patchcheck_1.2/
- Run the patchck.pl tool to see which patches are needed.
  cd /usr/local/patchcheck_1.2
  /usr/bin/perl patchk.pl –b –l

  You will be asked if you want to launch a browser, select no. The patchck.pl program will save the html file to /tmp. Copy the html output file to another machine with a web browser and open the file said web browser.
- Create patch cluster, and download the cluster from sunsolve.sun.com
  This is done by clicking on the "Create PatchSuite" link in the html file that was just created and now being viewed with a web browser.
  Note: A valid SunSolve ID is needed. This requires a Sun Support contract.
  Once the patch cluster is downloaded, use ssh to copy the cluster back to *logger,* and extract the tar file, and unzip the zip files in /tmp.
- Review all patch README files to ensure machine stability.
- In single user mode, install the patches (following the instructions in the README files).

NOTE: This step is an ongoing step as new vulnerabilities are found and fixed every day. It is important to keep up-to-date on the latest security patches available. While it may

not be necessary to fully patch every day, regular patch cycles are necessary. The author recommends doing a patch cycle once per month. This patch cycle will be accelerated when a CERT (or other) advisory becomes known to the system administrator to accommodate the immediate threat.

4.11 Miscellaneous

We have previously configured other programs like:

- *fix-modes* program to run from cron on a daily basis
- *logrotate* program runs on a monthly basis
- *passwd_age.pl* runs on a daily basis

It is recommended to periodically (monthly) run the fix-modes program manually to verify it is still functioning.
It is recommended to check the /var/log/passwd_age.log to look for errors. E-mails will be sent to the user and the admin previously configured in the script as passwords approach expiration.

5.0 CONFIGURATION CHECK

Throughout this document, we have checked many of the items to see if they produce the expected output. Some examples were to verify ftp settings for the *root* ID. We verified that services like telnet that were turned off in inetd.conf were actually disabled. We verified that the dns configuration is working with the "*nslookup"* command. Additionally, we verified that automounter will not run, and that the sudo configuration returns expected results for a given user. It is recommended to quickly go back through each of those "check" steps again to verify they are still acting as expected. It is possible that some of the configuration changes made later affected changes made earlier. Typos and missed steps are also possible while manually configuring this machine. The next logical step is to create an automated script to check the configuration of the machine.

5.1 SecurityCheck.pl

To verify the configuration settings of this document, it is a good idea to write a script to automate the settings of the machine versus that of this Security Baseline. [Due to the size of the perl script (approximately 41k), it is not included in this document. To receive a copy of the script that verifies the majority of the settings of this secure configuration guide, visit http://www.jedi.net/travis/security.] This script is quite comprehensive and certainly checks most of the main points of the step-by-step guide. It can easily be modified to look for more settings as desired. Once you have a script to verify the settings, run a monthly cron job to verify the settings. The results of the script will be mailed to the system administrators for

review. This script is a huge time saver and is less prone to human error in manually checking configuration settings (albeit, the script was written by a human).

Copy this script to /usr/local/bin. If it is on the CD we created to install the machine, run:

/usr/local/bin/sudo cp /mnt/cdrom/SecurityCheck.pl /usr/local/bin

Add the SecurityCheck.pl script to root's crontab.

/usr/local/bin/sudo crontab –e root

Add this line (This is all one line!):

```
1 1 1 * * /usr/local/bin/SecurityCheck.pl -v 2>&1
>/var/log/SecurityCheck.`hostname`.`/bin/date +\%Y-\%m-\%d`.log &&
/bin/chmod 600 /var/log/SecurityCheck.`hostname`.`/bin/date +\%Y-\%m-
\%d`.log ; /bin/mailx -s 'SecurityCheck.pl -v on `hostname`' root
</var/log/SecurityCheck.`hostname`.`/bin/date +\%Y-\%m-\%d`.log
```

Ensure the permissions are 700 and the owner is root and run the script manually to verify all items have been completed in this procedure.

/usr/local/bin/sudo chmod 700 /usr/local/bin/SecurityCheck.pl
/usr/local/bin/sudo /usr/local/bin/SecurityCheck.pl |more

See Appendix E for a sample output from the SecurityCheck.pl script.
See Appendix B for a list of acceptable group/world writable permissions.

5.2 Security Vulnerability Scans

Perform a vulnerability scan of the system. The author recommends that a scanner like ISS's *Internet Scanner* or *Nessus* (free) with current module updates be run against the newly installed machine. A *Nessus* scan result for a Solaris version of *logger* can be found in Appendix D. Only the ports expected answered to the scan. There were 0 security holes found, and 5 warnings, which are all normal and expected. The scan even determined that the backup ports are protected with TCP wrappers.

5.3 Netstat

Run *netstat -an* to see that only the expected configured ports are listening. We should see that ssh (port 22), ntp (port 123), syslog (port 514), sendmail (port 25) are running, and four ports (13724, 13782, 13783, and 13722) for NetBackup. The NetBackup ports will only answer to the backup server because the services are protected with TCP wrappers. The inetd services for the backups are restricted

to only answer to our backup server based on the configuration we implemented in the /etc/hosts.allow file.

See Appendix F for a full output of logger.

5.4 Check Services

We checked many services for functionality as we configured them. Run through as many of those checks again as time permits. The results should be the same as before for all tests. We can test a few here that we were unable to check before as *logger* was not on the network when we originally configured it.

5.4.1 DNS

To test dns, execute the command:
/usr/sbin/nslookup logger

We expect to see some output like:

```
Server:  ns1.XXX.XX.com
Address:  10.101.122.17

Name:    logger.XXX.XX.com
Address:  10.101. 22.219
```

The expected results are that an IP address for machine *logger* will be returned. You will also notice that the IP address of the DNS server used will be listed first. This is the machine that the information comes from (instead of the /etc/hosts file.) If DNS was not working properly, we would need to double check the settings in the /etc/resolv.conf /etc/nsswitch.conf files to make sure we modified those files correctly. If those look good, see if we can ping the dns server for our site. If that doesn't work, we have a networking problem that will need to be fixed in one or more of the following files: /etc/hosts, /etc/defaultrouter, /etc/netmasks.

5.4.2 FTP

For testing purposes, the /etc/inetd.conf file can be modified to allow ftp. Restart inetd (or start it if it isn't running at all.)
Set the root password to something for a temporary test while logged onto the console. From a remote machine, attempt to use the root ID and password to ftp into the server. Expected results are that the root ID will not be allowed, yet a personal userid (defined on the *logger* system) will be allowed to ftp. Turn off the ftp services when finished testing this step. From the local console, change the root password back to your complex password. Be sure to change it because the root password just went over the network in clear-text.

Other inetd services:
Check the configuration of the inetd services by attempting to ftp/telnet/rexec/rsh to *logger* from a remote machine. None of these services should answer requests to gain access. We can check the TCP wrapper logs to see the refused connections. We will also get an e-mail with the logs at the top of the next hour because the *logcheck.sh* program should be working. The TCP wrapper logs will show up in /var/log/daemonlog.

5.4.3 Automount

Since we just rebooted the machine at the end of step 3, we can now test to see if the automounter daemon is running. Execute the following commands:

ps –ef |grep automountd
*ls –l /etc/auto**

There should be no automounter processes running, and there should be no files named /etc/auto*.

5.4.4 Sendmail:

Now we can test to ensure that our mailhost will accept mail from logger.
/bin/mailx –v –s "Test message" root </dev/null

You should see some verbose messages between the local *logger* machine and your mail server. The e-mail should arrive in your e-mail box. (This assumes that the root ID is aliased to your personal e-mail address as required in an earlier section.)

If the mail was not successful, redo the sendmail portion of the step-by-step guide. Verify that you have network connectivity to the sendmail server by trying to telnet to the mailserver on port 25.
telnet <yourmailserver> 25

If this works, your e-mail should go through to the mailserver. If not, there are other networking issues that will need to be resolved. Some other things to check are to see if DNS will resolve your mailserver hostname. Use the *nslookup* command to see if it resolves in DNS.

5.5 TCP Wrappers

Test to see if entries are getting posted to the syslog file.
/usr/local/bin/sudo tail –f /var/log/daemonlog

Temporarily turn telnet back on in /etc/inetd and send a HUP signal to the inetd daemon. Telnet to localhost. The syslog should display a new entry for the telnet session you just created. Now modify the /etc/hosts.allow so it reads:
in.telnetd: ALL

Try to telnet to localhost again. The logs for the last two tests will look like this in the /var/log/daemonlog file:

Aug 20 17:22:53 logger inetd[610]: [ID 808958 daemon.warning] refused connect from localhost (access denied)
Aug 20 17:23:08 logger inetd[613]: [ID 927837 daemon.info] connect from localhost

5.6 logcheck.sh

Check to see if the hourly e-mail is coming to your configured e-mail address. Configure a machine to send their syslogs to our newly created *logger*. We can see from the output in Appendix I, that another machine *aacsec1* is successfully logging syslog messages to *logger*. These logs are "interesting" enough to not be parsed out with the logcheck.sh script. They are emphasized with a **bold font** to separate them from the local *logger* syslog messages.

A sample of the hourly e-mail is located in Appendix I.

It is interesting to note that the "interesting" logs found that a process on aacsec1 is having some problems with the 'srs' program and also that the '/' filesystem is full!

5.7 Tripwire

Check to see if the tripwire database is working. We have previously initialized the tripwire database with a *tripwire –init* command and placed the database in /var/tripwire. Now, we can modify a couple files to see the differences.

/usr/local/bin/sudo touch /etc/passwd1
/usr/local/bin/sudo cp /etc/shadow /etc/shadow1
/usr/local/bin/sudo /usr/local/bin/tripwire –q
/usr/local/bin/sudo rm /etc/passwd1
/usr/local/bin/sudo rm /etc/shadow1

The output looks like this:

```
added:   -rw-r----- root          0 Aug 22 11:17:59 2003 /etc/passwd1
added:   -r-------- root        452 Aug 22 11:18:05 2003 /etc/shadow1
```

After a patch cycle, we will need to re-initialize the database and store a copy on read-only media. Remember that we will be generating this output everyday at 4:04 am based on our root cronjob. We should get an e-mail the next time the *logcheck.sh* script runs as this output is sent to the syslog facility via the cron job.

5.8 SSH

While logged into the console, try to ssh to logger from another machine on the network.

ssh travis@logger

It should allow you to connect using password authentication. This test will also show that the privilege separation is working. Before the login is complete (i.e. – while there is a "password:" prompt), run the following command on logger (from another window/session:

ps –ef |grep sshd

You should see a process that is owned by the "sshd" user:

```
root   448   400  1 13:31:29 ?        0:00 /usr/local/sbin/sshd
sshd   449   448  7 13:31:29 ?        0:01 /usr/local/sbin/sshd
```

You should also see one that is owned by root.

After the login is successful, there will be one owned by root and one owned by the user that logged in:

```
  root   448   400  1 13:31:29 ?        0:00 /usr/local/sbin/sshd
travis   452   448  0 13:31:34 ?        0:00 /usr/local/sbin/sshd
```

References

1. Gregory, Peter H. Solaris Security Sun Microsystems Press, 2000.
2. Andrason, Jackie. "HP-UX 11.00 Installation Checklist" SANS Institute: http://www.giac.org/practical/Jackie_Andrason_GCUX.doc
3. Ellis, Theodore. "HP-UX 11.0 Installation and Security Verification Checklist for Lawson. Application Server" SANS Institute: http://www.giac.org/practical/Theodore_Ellis_GCUX.doc
4. Schmidt, Della. "HP-UX 11.0 Installation Checklist" SANS Institute: http://www.giac.org/practical/Della_Schmidt_GCUX.doc
5. The Sendmail Consortium. "Sendmail 8.12.9" http://www.sendmail.org
6. Tripwire Academic Source Release 1.3.1 for UNIX. User Manual. Tripwire Security Systems, Inc., 1999.
7. Track 6 – Common Issues and Vulnerabilities in UNIX Security, 6.1 Acheson, Green, and Pomeranz; The SANS Institute, 2002.
8. Track 6 – UNIX Security Tools, 6.2; Hal Pomeranz; The SANS Institute, 2002.
9. Track 6 – Topics in UNIX Security, 6.3; Hal Pomeranz; The SANS Institute, 2002.
10. Track 6 – Running UNIX Applications Securely, 6.4; Hal Pomeranz; The SANS Institute, 2002.
11. Track 6 – UNIX Practicum, 6.5; Hal Pomeranz; The SANS Institute, 2002.

Appendix A - Banners

These are the official legal warning banners from Security. Please use the long version where technically possible. The only place the short version should be used is in the /etc/default/ftpd file.

Short Version:

This system is for official use by authorized users only with no expectation of privacy. It may include records protected by law. All use constitutes consent to authorized monitoring. Misuse may result in criminal or other penalties.

Preferred Long Version:

This system is intended for official and authorized use only by authorized users with no reasonable expectation of privacy. The system may include records protected by various Federal statutes including the Privacy Act (5 U.S.C. § 552a) and 38 U.S.C. §§ 5701 and 7332. Access to data is on a need-to-know basis only. All use of this system constitutes user understanding of unconditional consent to review and action including (but not limited to) monitoring, recording, copying, auditing, inspecting, investigating, restricting access, blocking, tracking, disclosing to authorized personnel, or any other authorized actions by law enforcement personnel. Unauthorized access to or misuse of this system is strictly prohibited may result in criminal, civil, or administrative penalties.

In addition, if the server holds or allows Privacy Act data to pass through, it must also contain the following text:

PRIVACY ACT WARNING
THE UNAUTHORIZED DISCLOSURE OF INFORMATION FROM THIS SYSTEM COULD RESULT IN A VIOLATION OF AN INDIVIDUAL'S RIGHT TO PRIVACY. SECURITY MEASURES REQUIRE THAT THE INFORMATION CONTAINED HEREIN BE USED ONLY BY AUTHORIZED PERSONS IN THE CONDUCT OF OFFICIAL BUSINESS. UNAUTHORIZED DISCLOSURE OF PERSONAL INFORMATION, TO ANY PERSON OR AGENCY NOT ENTITLED TO RECEIVE IT, MAY RESULT IN A FINE NOT MORE THAN $5,000.

THE PRIVACY ACT OF 1974, 5 U.S.C. 552A, PROHIBITS UNAUTHORIZED RELEASE OF PERSONAL DATA CONTAINED HEREIN.

In addition, if the server holds or allows Federal Tax Information (FTI) to pass through, it must also contain the following text:

CONFIDENTIAL FEDERAL TAX INFORMATION ENCLOSED. ACCESS IS RESTRICTED PER IRC 6103 (P) (4) (C). MISUSE OR UNAUTHORIZED ACCESS TO THIS INFORMATION IS PROHIBITED AND SUBJECT TO PROSECUTION AND PENALTIES.

Appendix B – Group write permissions

Group Write Permissions acceptable:

/etc/dumpdates
/etc/lp
/etc/vx/*
/etc/ntp.drift
/etc/*JASS*

Appendix C – suid files

Typical list of suid files for Solaris:

/usr/bin/sparcv7/newtask
/usr/bin/sparcv7/uptime
/usr/bin/sparcv7/w
/usr/bin/at
/usr/bin/atq
/usr/bin/atrm
/usr/bin/crontab
/usr/bin/eject
/usr/bin/fdformat
/usr/bin/login
/usr/bin/mail
/usr/bin/mailx
/usr/bin/netstat
/usr/bin/newgrp
/usr/bin/passwd
/usr/bin/pfexec
/usr/bin/su
/usr/bin/tip
/usr/bin/write
/usr/bin/sparcv9/newtask
/usr/bin/sparcv9/uptime
/usr/bin/sparcv9/w
/usr/bin/rcp
/usr/bin/rdist
/usr/bin/rlogin
/usr/bin/rsh
/usr/bin/mailq
/usr/bin/chkey
/usr/lib/fs/ufs/quota
/usr/lib/fs/ufs/ufsdump
/usr/lib/fs/ufs/ufsrestore
/usr/lib/pt_chmod
/usr/lib/utmp_update
/usr/lib/sendmail
/usr/platform/sun4u/sbin/eeprom
/usr/platform/sun4u/sbin/prtdiag
/usr/sbin/sparcv7/prtconf
/usr/sbin/sparcv7/swap
/usr/sbin/sparcv7/sysdef
/usr/sbin/sparcv7/whodo
/usr/sbin/allocate
/usr/sbin/sacadm
/usr/sbin/traceroute
/usr/sbin/wall
/usr/sbin/deallocate
/usr/sbin/list_devices
/usr/sbin/sparcv9/prtconf
/usr/sbin/sparcv9/swap
/usr/sbin/sparcv9/sysdef
/usr/sbin/sparcv9/whodo
/usr/sbin/ping
/usr/SUNWale/bin/mailx
/usr/ucb/sparcv7/ps

Appendix D – Nessus output

This is a text version of the *Nessus* vulnerability scan output:

Nessus Scan Report

This report gives details on hosts that were tested and issues that were found. Please follow the recommended steps and procedures to eradicate these threats.

Scan Details
Hosts which where alive and responding during test 1
Number of security holes found 0
Number of security warnings found 5

Host List
Host(s) Possible Issue
10.224.227.XXX Security warning(s) found

[return to top]

Analysis of Host
Address of Host Port/Service Issue regarding Port
10.224.XXX.XXX ssh (22/tcp) Security warning(s) found
10.224.XXX.XXX VeritasNetbackup (13722/tcp) Security notes found
10.224.XXX.XXX unknown (13724/tcp) Security notes found
10.224.XXX.XXX bpcd (13782/tcp) Security notes found
10.224.XXX.XXX vopied (13783/tcp) Security notes found
10.224.XXX.XXX general/tcp Security warning(s) found
10.224.XXX.XXX general/udp Security notes found
10.224.XXX.XXX ntp (123/udp) Security warning(s) found
10.224.XXX.XXX general/icmp Security warning(s) found

Security Issues and Fixes: 10.224.XXX.XXX
Type Port Issue and Fix
Warning ssh (22/tcp)
You are running OpenSSH-portable 3.6.1p1 or older.
If PAM support is enabled, an attacker may use a flaw in this version
to determine the existence or a given login name by comparing the times
the remote sshd daemon takes to refuse a bad password for a non-existant
login compared to the time it takes to refuse a bad password for a
valid login.
An attacker may use this flaw to set up a brute force attack against
the remote host.
*** Nessus did not check whether the remote SSH daemon is actually
*** using PAM or not, so this might be a false positive
Solution : Upgrade to OpenSSH-portable 3.6.1p2 or newer
Risk Factor : Low
CVE : CAN-2003-0190
BID : 7482
Nessus ID : 11574
Informational ssh (22/tcp) An ssh server is running on this port
Nessus ID : 10330
Informational ssh (22/tcp) Remote SSH version : SSH-2.0-OpenSSH_3.6.1p1
Nessus ID : 10267
Informational ssh (22/tcp) The remote SSH daemon supports the following versions of the
SSH protocol :
. 1.99
. 2.0
Nessus ID : 10881
Informational VeritasNetbackup (13722/tcp) The service closed the connection after 0 seconds without sending
any data
It might be protected by some TCP wrapper
Nessus ID : 10330
Informational unknown (13724/tcp) The service closed the connection after 0 seconds without sending any data
It might be protected by some TCP wrapper
Nessus ID : 10330
Informational bpcd (13782/tcp) The service closed the connection after 0 seconds without sending any data
It might be protected by some TCP wrapper
Nessus ID : 10330
Informational vopied (13783/tcp) The service closed the connection after 0 seconds without sending any data
It might be protected by some TCP wrapper

Nessus ID : 10330
Warning general/tcp
The remote host does not discard TCP SYN packets which
have the FIN flag set.
Depending on the kind of firewall you are using, an
attacker may use this flaw to bypass its rules.
See also : http://archives.neohapsis.com/archives/bugtraq/2002-10/0266.html
http://www.kb.cert.org/vuls/id/464113
Solution : Contact your vendor for a patch
Risk factor : Medium
BID : 7487
Nessus ID : 11618
Informational general/tcp Nmap found that this host is running Solaris 9 with TCP_STRONG_ISS set to 2
Nessus ID : 10336
Informational general/tcp Remote OS guess : Solaris 9 with TCP_STRONG_ISS set to 2
CVE : CAN-1999-0454
Nessus ID : 11268
Informational general/udp For your information, here is the traceroute to 10.224.XXX.XXX :
10.224.XXX.XXX
Nessus ID : 10287
Warning ntp (123/udp)
An NTP server is running on the remote host. Make sure that
you are running the latest version of your NTP server,
has some versions have been found out to be vulnerable to
buffer overflows.
*** Nessus reports this vulnerability using only
*** information that was gathered. Use caution
*** when testing without safe checks enabled.
If you happen to be vulnerable : upgrade
Solution : Upgrade
Risk factor : High
CVE : CVE-2001-0414
BID : 2540
Nessus ID : 10647
Informational ntp (123/udp)
It is possible to determine a lot of information about the remote host
by querying the NTP variables - these include OS descriptor, and
time settings.
Theoretically one could work out the NTP peer relationships and track back
network settings from this.
Quickfix: Set NTP to restrict default access to ignore all info packets:
restrict default ignore
Risk factor : Low
Nessus ID : 10884
Warning general/icmp
The remote host answers to an ICMP timestamp request. This allows an attacker
to know the date which is set on your machine.
This may help him to defeat all your time based authentication protocols.
Solution : filter out the ICMP timestamp requests (13), and the outgoing ICMP
timestamp replies (14).
Risk factor : Low
CVE : CAN-1999-0524
Nessus ID : 10114
Warning general/icmp
The remote host answered to an ICMP_MASKREQ
query and sent us its netmask (255.255.255.0)
An attacker can use this information to
understand how your network is set up
and how the routing is done. This may
help him to bypass your filters.
Solution : reconfigure the remote host so
that it does not answer to those requests.
Set up filters that deny ICMP packets of
type 17.
Risk factor : Low
CVE : CAN-1999-0524
Nessus ID : 10113

This file was generated by Nessus, the open-sourced security scanner.

Appendix E – SecurityCheck.pl output

```
HOSTNAME: logger             Security Check v 1.07
OS: SunOS 5.9
Date: 08.20.2003
   OK     :/etc/issue banner
   OK     :/etc/default/ftpd banner
   OK     :/etc/default/passwd PASSLENGTH=8
   OK     :/etc/profile PATH=/usr/bin:/sbin:/usr/sbin
   OK     :/etc/profile umask 027 or 077
   OK     :/etc/profile PS1=hostname #
   OK     :/etc/default/login CONSOLE=/dev/console  (aka root login)
   OK     :/etc/default/login TIMEOUT=60
   OK     :/etc/default/login umask 027 or 077
   OK     :/usr/bin/su exists
   OK     :/usr/bin/su mode is 4550 (should be 4550)
   OK     :/bin/su exists
   OK     :/bin/su mode is 4550 (should be 4550)
   OK     :/usr/bin/su gid=sysadmin or suadmin?
   OK     :/etc/skel/local.profile "." in PATH
   OK     :/etc/skel/local.profile umask=027 or 077
   OK     :/etc/skel/local.cshrc "." in PATH
   OK     :/etc/skel/local.cshrc umask 027 or 077
   OK     :/etc/cron.d/cron.deny user list
   OK     :/etc/default/cron "CRONLOG=YES"
   OK     :/etc/* group & world write permissions
          :Did you verify? : find / -type f \( -perm -u+s -o -perm -g+s \) -ls
   OK     :/etc/default/su SULOG=/var/adm/sulog
   OK     :/etc/default/su SYSLOG=YES
   OK     :/etc/default/su CONSOLE=/dev/console
   OK     :/var/adm/loginlog exists
   OK     :/var/adm/loginlog mode is 0600 (should be 0600)
   OK     :/var/adm/loginlog owner = root
   OK     :/var/adm/loginlog group = sys
          :       Active Services from /etc/inetd.conf
          :            bpcd stream      tcp     nowait  root   /usr/openv/netbackup/ \
                                                               bin/bpcd bpcd
          :           vnetd stream      tcp     nowait  root   /usr/openv/bin/vnetd vnetd
          :          vopied stream      tcp     nowait  root   /usr/openv/bin/vopied vopied
          :     bpjava-msvc stream      tcp     nowait  root   /usr/openv/netbackup/  \
                                                          bin/bpjava-msvc bpjava-msvc -transient
   OK     :inetd -t option in /etc/init.d/inetsvc
          :Did you log active services in the server log book?
   OK     :/etc/auto_* files
   OK     :/etc/rc2.d/S74autofs
   OK     :/etc/ftpd/ftpusers user list
   OK     :/etc/mail/sendmail.cf "D[S|R]mail*"
   OK     :/etc/mail/sendmail.cf "OI" line is not active
          :       Active Aliases
          : postmaster: root
          : MAILER-DAEMON: postmaster
          : bin:               root
          : daemon:            root
          : system:            root
          : toor:              root
          : uucp:              root
          : manager:    root
          : dumper:            root
          : operator:   root
          : decode:            root
          : nobody: /dev/null
          : root: <SOMEUSER>@<SOMEDOMAIN.COM>
          : travis: <SOMEUSER>@<SOMEDOMAIN.COM>
   OK     :/etc/aliases exists
   OK     :/etc/aliases mode is 0644 (should be 0644)
   OK     :/etc/rc3.d/S15nfs.server
   OK     :/etc/rc3.d/S73nfs.client
   OK     :NFS services (nfsd) should not be running.
   OK     :/etc/nscd.conf "enable-cache hosts no" option
   OK     :/etc/nscd.conf "enable-cache passwd no" option
```

```
OK      :/etc/nscd.conf "enable-cache group no" option
OK      :/etc/snmp/conf/snmpd.conf
OK      :sudo installed (via package [SFWsudo|SMCsudo].)
OK      :/var/log/sudo.log exists
OK      :/usr/local/bin/sudo exists
OK      :/etc/syslog.conf "local2.debug /var/log/sudo.log" option
OK      :/usr/sfw/sbin/tcpd exists
OK      :/usr/sfw/sbin/tcpdchk exists
OK      :/usr/sfw/sbin/tcpdmatch exists
OK      :/etc/syslog.conf "local3.debug /var/log/tcpwrap.log" option
OK      :/var/log/tcpwrap.log exists
OK      :/etc/init.d/nddconfig "ndd -set /dev/tcp tcp_conn_req_max_q0 10240" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_ignore_redirect 1" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_send_redirects 0" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/arp arp_cleanup_interval 60" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_forward_directed_broadcasts 0" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_forward_src_routed 0" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_forwarding 0" option
OK      :/etc/init.d/nddconfig "ndd -set /dev/ip ip_strict_dst_multihoming 1" option
OK      :/etc/notrouter exists
        :Did you verify? : http://www.cert.org/advisories
        :Did you verify? : http://sunsolve.sun.com/security
OK      :/etc/default/inetinit "TCP_STRONG_ISS=2" option
OK      :/etc/system "set noexec_user_stack=1" option
OK      :/etc/system "set noexec_user_stack_log=1" option
        :Did you verify patches? : http://sunsolve.sun.com/ \
                pub-cgi/show.pl?target=patches/patch-license&nav=pub-p
        :Run patchdiag and apply applicable patches.
        :Newest patch/package was installed on August 19, 2003
OK      :Perl 5 installed.
OK      :/usr/local/ssl exists
OK      :/usr/local/etc/sshd_config exists
OK      :/etc/init.d/sshd exists
OK      :/usr/local/etc/sshd_config "X11Forwarding yes" option
        :Unable to verify TripWire install.

SysAdmin:_____________________________      Date:_________________
Manager:______________________________      Date:_________________
```

Appendix F – netstat –an output

```
root@logger >netstat -an

UDP: IPv4
   Local Address         Remote Address     State
-------------------- -------------------- -------
      *.514                                Idle
      *.123                                Idle
127.0.0.1.123                              Idle
10.224.XXX.XXX.123                         Idle
      *.*                                  Unbound
      *.*                                  Unbound
      *.*                                  Unbound

TCP: IPv4
   Local Address        Remote Address    Swind Send-Q Rwind Recv-Q  State
-------------------- -------------------- ----- ------ ----- ------ -------
      *.*                  *.*                0      0 49152      0 IDLE
      *.13782              *.*                0      0 49152      0 LISTEN
      *.13724              *.*                0      0 49152      0 LISTEN
      *.13783              *.*                0      0 49152      0 LISTEN
      *.13722              *.*                0      0 49152      0 LISTEN
      *.22                 *.*                0      0 49152      0 LISTEN
      *.22                 *.*                0      0 49152      0 LISTEN
127.0.0.1.25               *.*                0      0 49152      0 LISTEN
      *.*                  *.*                0      0 49152      0 IDLE

TCP: IPv6
   Local Address  Remote Address     Swind Send-Q Rwind Recv-Q   State      If
---------------- ----------------- ----- ------ ----- ------ ----------- -----
      *.*           *.*                0      0 49152      0 IDLE
      *.22          *.*                0      0 49152      0 LISTEN
```

Appendix G – Process listing

```
root@logger >ps -ef
     UID   PID  PPID  C    STIME TTY      TIME CMD
    root     0     0  0 10:24:17 ?        0:00 sched
    root     1     0  0 10:24:17 ?        0:00 /etc/init -
    root     2     0  0 10:24:17 ?        0:00 pageout
    root     3     0  0 10:24:17 ?        0:00 fsflush
    root   174     1  0 10:24:34 ?        0:00 /usr/lib/saf/sac -t 300
    root   177   174  0 10:24:34 ?        0:00 /usr/lib/saf/ttymon
    root    48     1  0 10:24:24 ?        0:00 /usr/lib/sysevent/syseventd
    root    56     1  0 10:24:26 ?        0:00 /usr/lib/picl/picld
    root   117     1  0 10:24:30 ?        0:00 /usr/sbin/inetd -s -t
    root   128     1  0 10:24:30 ?        0:00 /usr/sbin/syslogd
    root   134     1  0 10:24:31 ?        0:00 /usr/sbin/cron
    root   194     1  0 10:24:55 ?        0:00 /usr/lib/inet/xntpd
    root   175     1  0 10:24:34 console  0:00 -sh
    root   162     1  0 10:24:32 ?        0:00 /usr/lib/utmpd
    root   199   175  0 10:25:32 console  0:00 ps -ef
    root   167     1  0 10:24:33 ?        0:00 /usr/local/sbin/sshd
   smmsp   188     1  0 10:24:54 ?        0:00 /usr/lib/sendmail -Ac -q15m
    root   189     1  0 10:24:54 ?        0:00 /usr/lib/sendmail -bd -q15m
```

Appendix H – Pre-fetch package list

File/Package	From
9_Recommended.zip	http://sunsolve.sun.com
SecurityCheck.pl	http://www.jedi.net/travis/security
fix-modes.tar.Z	ftp://ftp.CISecurity.org/pub/pkgs/Solaris/fix-modes.tar.Z
libgcc-3.3-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
logcheck-1.1.1-sol8-sparc-local.gz	http://www.sunfreeware.com/programlistsparc8.html#logcheck
logrotate-3.6.9-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
openssh-3.6.1p1-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
openssl-0.9.7b-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
passwd_age.pl	The note file in this doc or http://www.jedi.net/travis/security
popt-1.7-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
sudo-1.6.7p5-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9
tripwire-1.3.1-sol9-sparc-local-TJH	http://www.jedi.net/travis/security
zlib-1.1.4-sol9-sparc-local.gz	ftp://sunfreeware.secsup.org/pub/solaris/freeware/sparc/9

Appendix I – logcheck.sh sample e-mail

```
Security Violations
=-=-=-=-=-=-=-=-=-=
Aug 21 12:45:30 logger xntpd[181]: [ID 988144 daemon.debug]
signal_no_reset: signal 18 had flags 20000
Aug 21 12:54:38 aacsec1.XXX.XXX.COM EventProvider[23450]: [ID 395154
daemon.error] srs_register call failed
Aug 21 12:54:38 aacsec1.XXX.XXX.COM trend_pvr[23451]: [ID 395154
daemon.error] srs_register call failed
Aug 21 12:51:51 logger su: [ID 366847 auth.notice] 'su root' succeeded
for travis on /dev/console

Unusual System Events
=-=-=-=-=-=-=-=-=-=-=
Aug 21 12:51:03 logger /usr/local/bin/sudo: [ID 702911 local2.alert]
travis : user NOT in sudoers ; TTY=console ; PWD=/export/home/travis ;
USER=root ; COMMAND=/usr/bin/cat /var/log/kernlog
Aug 21 12:53:45 logger /usr/local/bin/sudo: [ID 702911 local2.notice]
travis : TTY=console ; PWD=/export/home/travis ; USER=root ;
COMMAND=/usr/bin/cat /var/log/kernlog
Aug 21 12:14:46 logger last message repeated 7 times
Aug 21 12:15:54 logger hme: [ID 786680 kern.notice] SUNW,hme0 : No
response from Ethernet network : Link down -- cable problem?
Aug 21 12:16:18 logger last message repeated 2 times
Aug 21 12:28:17 logger hme: [ID 786680 kern.notice] SUNW,hme0 : No
response from Ethernet network : Link down -- cable problem?
Aug 21 12:28:50 logger last message repeated 3 times
Aug 21 12:29:37 logger hme: [ID 786680 kern.notice] SUNW,hme0 : No
response from Ethernet network : Link down -- cable problem?
Aug 21 12:29:48 logger last message repeated 1 time
……
Aug 21 12:44:55 logger hme: [ID 517527 kern.info] SUNW,hme0 : Internal
Transceiver Selected.
Aug 21 12:44:55 logger hme: [ID 517527 kern.info] SUNW,hme0 :   100
Mbps Full-Duplex Link Up
Aug 21 12:45:00 logger pseudo: [ID 129642 kern.info] pseudo-device:
devinfo0
Aug 21 12:45:00 logger genunix: [ID 936769 kern.info] devinfo0 is
/pseudo/devinfo@0
Aug 21 12:47:00 aacsec1.XXX.XXX.COM ufs: [ID 845546 kern.notice]
NOTICE: alloc: /: file system full
Aug 21 12:51:41 aacsec1.XXX.XXX.COM last message repeated 9 times
Aug 21 12:51:50 aacsec1.XXX.XXX.COM ufs: [ID 845546 kern.notice]
NOTICE: alloc: /: file system full
Aug 21 12:58:00 aacsec1.XXX.XXX.COM last message repeated 12 times
Aug 21 12:59:00 aacsec1.XXX.XXX.COM ufs: [ID 845546 kern.notice]
NOTICE: alloc: /: file system full
 …
```