Global Information Assurance Certification Paper

SANS
Global Information Assurance Certification (GIAC)
Program

GCUX
Certified Unix Security Administrator
Version 1.7

Installing And Securing Solaris 8

By
Timothy E. Raborn
September 2001

Table of Contents

Tables

Figures

Preface

The purpose of this practical assignment is to fulfill requirements for the Global Information Assurance Certification for Unix Security Administrators (GCUX), version 1.7.

Document Conventions

Typeface or Symbol	Meaning	Example
AaBbCc123	Normal text	The quick brown fox jumped over the lazy dog.
AaBbCc123	Book or document titles, terms, or words to be emphasized	Refer to Step 1.5 of *Securing Solaris Step-by-Step Version 2.0* by Hal Pomeranz for information on installing patches. This command *must* be run as root.
`AaBbCc123`	The names of commands, files and directories; on-screen computer output	Edit the `/etc/issue` file Run `make all` to compile the package `hostname% file not found`
`AaBbCc123`	What is typed, contrasted with on-screen computer output	`hostname%` **`su`** `password:`
`AaBbCc123`	Command-line placeholder: replace with a real name or value	To edit a file, type `vi` *`filename`*

Table 1 - Document Conventions

Purpose

The purpose of this document is to provide a detailed checklist of the steps a System Administrator (SA) must take to secure a Solaris™ 8 server for use as a public Internet server. The most likely use for such a server would be for web or ftp services. As an example, the final step of this process will demonstrate how to install and configure the Apache web server.

The steps demonstrated in this document were tested for accuracy using a SPARCserver 20 with 192MB of RAM, four 2.1GB and two 1.0GB SCSI disk drives, and one SCSI CD-ROM drive. However, the steps were written without regard to hardware specifics.

Before attaching any computer system or network device to a network, the organization's Information Technology (IT) security policy should be consulted.

Preliminary

The installation of a new server should be coordinated with the network administrator (NA). The NA will assign an Internet Protocol (IP) address and give other information needed during the installation such as the default router (or gateway) and the domain name service (DNS) servers. Often times, a network will have two or more DNS servers for redundancy, and two or more should be used when possible.

The NA may also need the Media Access Control (MAC) address of the network interface card (NIC). Every NIC manufactured has a unique MAC addressed assigned to it at the factory, and this address is "hard-coded" into the hardware. It uniquely identifies each node of a network. Some networks use "port locking" whereby a specific MAC address is assigned to a specific port on a hub or switch. Usually, this port is actually connected to a wall jack in the computer room. If another computer system (or even the same computer system with a different NIC) tries to use that port, it will lock and no longer allow packets to be transmitted. This is used as an added layer of security to prevent unauthorized computers from being physically connected to the network. If this is the case, the NA (or firewall administrator) must be notified as to which port is being utilized so the networking hardware can be programmed appropriately. Sun servers display their MAC address in the hardware banner when the system is first booted. Alternately, type `reset` at the EPROM `ok` prompt and the hardware banner will be redisplayed after a few seconds.

The NA will also configure the access control list (ACL) in the organization's firewall to allow proper access from the outside world.

Because it is never a good idea to connect a newly installed and unpatched operating system (OS) to a public network, there are precautions that must be taken to prevent the system from being compromised before the installation is completed. One method is to completely disconnect the system from the network and use tapes, CD-ROM's or floppy disks to transfer compiled software from a trusted development system to the new server. A more convenient method would be to connect the system to a small private/isolated network. A trusted development system with a full OS installation (X, software development tools, etc.) can be used on this network for building and compiling any needed software. The binaries can then be easily copied across the network and installed as needed.

If resources allow, a Solaris™ 8 workstation located on a private network, could be dedicated to building various software packages to be installed on the servers. If the software is always kept current, this will greatly decrease the installation time since all that will be necessary is to copy the software to the new system. Similarly, all of the binaries can be burned to a CD-ROM each time new versions are downloaded and re-compiled, and these new binaries can be copied to the new server when needed.

Physical Security

The physical security of a computer system is as important as the configuration of the OS and the software. How this is accomplished depends greatly on the size of the organization and the resources and space available. Typically, an Internet server should be located in a room with limited access. This can be as simple as a lock and key or more conveniently with a cipher lock or keycards. Keycards have the advantage of being able to provide an audit trail of the personnel entering and exiting the room since each card is coded for each individual. Only individuals who absolutely need access should be permitted into this area. If possible, the computer room should have no windows (this is especially true for exterior windows) and no exterior doors. Often times a computer room will have windows or glass doors which open to an interior hallway. If this is the case, the monitor and keyboard should be located to provide privacy from individuals who may peer in.

If the computer room uses a drop ceiling with ceiling tiles, make sure the walls continue up to the roof to prevent someone from climbing over the wall from an outside room. In some instances, a metal "cage" is located above the wall to provide protection, yet still allow proper ventilation. Ceiling tiles can also be secured to its metal framing with clips and screws, which prevent them from being removed without being broken (which would leave evidence of a breach).

Likewise, if a raised floor is used, it should be secured so that an individual would be unable to climb under the wall and into the computer room. This is usually accomplished with metal bars or a "cage" around the perimeter of the room. One should also make note of any network or power cables that are within an arm's reach of this cage from the exterior. All cables should be kept away from this area to prevent someone from physically tapping into the network, or causing a denial of service by severing a network cable or a power cord.

The server and keyboard should be housed in a locked cabinet (a mouse will not be necessary since X will not be installed). The system should be connected to an uninterruptible power supply (UPS). If resources allow, a UPS capable of providing power for at least two hours (or longer) is helpful during extended power outages.

Consideration must also be taken to ensure that there are no water lines running over or under the systems in this area. Thought must also be give to the use of overhead air conditioning units in the event that the condensation drain becomes blocked and overflows.

A fire retardation system should be used in the computer room, but water-based sprinkler systems should be avoided at all costs. Halon has been banned by the Environmental Protection Agency, therefore, a halon substitute, such as Inergen, should be considered.

Depending on the need, installation of a surveillance camera should be considered.

Operating System Installation

Preliminary Preparations

1. Gather network information
   Ethernet (or MAC) address: ____:____:____:____:____:____
   IP Address: _____._____._____._____
   Net Mask: _____._____._____._____
   Default Router (Gateway): _____._____._____._____
   Primary DNS Server: _____._____._____._____
   Secondary DNS Server: _____._____._____._____
2. Ensure that all hardware is properly assembled and that all cables are securely fastened.
3. Double-check the network connection to see if it is disabled or is properly connected to the private/isolated network.
4. Power the system up. If the system has previously been used for some other purpose, it may attempt to boot to the OS after the hardware and memory check. If it does, press `<STOP><A>` to stop the system and return to the EPROM `ok` prompt.

Boot from the OS Disk

1. Insert the CD-ROM labeled *Solaris™ 8 Software 1 of 2* into the CD-ROM drive.
2. At the `ok` prompt, enter `boot cdrom`.

```
ok boot cdrom
Boot device: /iommu/sbus/espdma@f,400000/esp@f,800000/sd@6,0:d File and args:
SunOS Release 5.8 Version Generic 32-bit
Copyright 1983-2000 Sun Microsystems, Inc. All rights reserved.
```

3. Select the language, typically English.
4. Select the Locale, typically English. After making the Locale selection, the X server will start, after which a console window will appear with the following message:

```
The system is coming up.  Please wait.
```

5. When the initial window appears, press `Continue` until the Network Connectivity window appears.
6. Select `Yes` when asked if the system will be networked.
7. It is not recommended that an Internet server use DHCP, therefore select `No` when asked if DHCP will be used. As mentioned earlier, a static IP address should be obtained from the NA.
8. Select the primary network interface. This will usually be the one with the lowest number, typically 0.
9. Enter the host name that will be used for this computer. This should be a name agreed

upon with the NA. It must be unique and must be added to the DNS server(s). Contact the DNS SA if necessary.

10. Enter the IP address assigned by the NA.
11. Answer `No` to enabling Ipv6.
12. Verify that all the information in the above steps is correct, then press `Continue`.
13. For this particular configuration, Kerberos Security will not be used, therefore answer `No` when asked to configure Kerberos.
14. For name service, select `None`.
15. The next question will ask if the system is part of a subnet. Normally networks are divided into subnets using routers and gateways. If unsure, ask the NA. Normally the answer will be `Yes`. If `No`, the process will skip ahead to the Time Zone settings.
16. If `Yes` is selected for Subnet, a *netmask* will be entered. For most class B networks, this will be 255.255.0.0. If unsure about the netmask, consult the NA.
17. To set the time zone, select `Geographic region`, then select `Set…`. Select the region from the list on the left, then the time zone from the list on the right.
18. Make sure the date and time are correct on the next window, and then press `Continue`.
19. Ensure that the subnet and time zone information is correct, then press `Continue`.
20. The installation will now ask if this is an `Upgrade` or an `Initial` installation. Select `Initial` (the default), and then `Continue`.
21. Press `Continue` again to advance to the next window.
22. The next window allows the selection of the geographic region for which support should be installed. This step may be skipped.

Software Group Selection

The OS cluster to be installed is now selected. Because this system is going to be an Internet bastion host, the proper selection is `Core System Support`. Other packages desired, but not included in the core cluster can be installed by selecting `Customize…`.

Optional Software				
Software Desired	Required Packages			
NTP				
	SUNWnptr	SUNWntpu		
Perl				
	SUNWlibm	SUNWlibms		
Oracle				
	SUNWarc	SUNWsprot	SUNWbtool	SUNWtoo
	SUNWlibCf			
Berkeley compatibility tools				
	SUNWscpu			
On-line manual pages				
	SUNWlibC	SUNWdoc	SUNWman	
X clients				
	SUNWxwrtl	SUNWxwfnt	SUNWxwice	SUNWtltk
	SUNWxilrl	SUNWxildh	SUNWxilow	SUNWxwplt
	SUNWmfrun			
Terminfo Database				
	SUNWter			
System Accounting				
	SUNWaccr	SUNWaccu		

Table 2 – Optional software and required packages

NTP, Perl, manual pages, the terminfo database, and system accounting are very desirable. NTP will be covered later in this document, as will system accounting. Quite often, an SA will write scripts (including `cron` job scripts) in Perl.

After selecting any optional packages, press `Continue`.

Disk Partition Layout

There are no set rules for the layout of disk partitions. Much will depend on the size of the hard drives and exactly what the server's main function will be. Because only the core elements of the OS will be installed, the `/usr` partition will contain less than 50MB. The `/var` partition should be large enough to manage all of the extra logging information. The `/usr/local` partition will be used to hold all third party software and information such as web pages or ftp archives.

Typical Disk Partition sizes as recommended by Lee Brotzman and Hal Pomerantz in the *Linux/Solaris Practicum*from SANS Baltimiore:[1]:

/	256 MB
/var	2048 MB
(swap)	2048 MB
/usr	1024 MB
/usr/local	Largest remaining partition

Table 3 – Typical disk partition sizes.

These partition sizes should only be used as a guideline.

1. At this point in the installation, a list of all of the hard drives on the system is displayed. The boot disk will be shown in the `Selected Disks` list. Add all of the `Available Disks` to the `Selected Disks` list by pressing >>. Press `Continue`.
2. Do **not** preserve any of the disks data by pressing `Continue`. (This document is assuming that a completely new load of the OS is being performed. If a hard drive with data has been installed, then preserve the data on that disk, if desired, but no data on the system partitions such as `/`, `/usr`, or `/var` should be preserved).
3. Do not use the `Auto Layout` feature for defining partitions. Press `Manual Layout`, and then when presented with a summary list of the layout, press `Customize`….
4. Use this window to layout the partitions in the manner that best is best suited for the required applications, then press `OK`, then `Continue`.
5. When asked to mount a remote file system, press `Continue` to skip this step.

A profile window showing the disk layout will be displayed. Review this list very carefully, and when satisfied the partitions have been defined properly and appropriately, press `Begin Installation` to start the installation process.

Before the install process begins, a prompt to `Auto Reboot` or perform a `Manual Reboot` will be displayed. A `Manual Reboot` is recommended. This step can be quite lengthy, and the console should be watched during the reboot for possible errors. If `Auto Reboot` is selected, it is quite likely any error messages will be missed (the alternative is to check the `/var/adm/messages` file for this information).

At this time, the install process will partition and format the disk partitions, then copy the software packages selected from the CD-ROM to the hard drive. Depending on the speed of the hardware, this could take a long time.

Configuring The System

Setting the root password

When the system has been rebooted, a login prompt will be displayed. When logging in as root, it is immediately obvious that there is no root password. Therefore, the first step should be to set the root password using the `passwd` command.

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

Asterisks are only shown as placeholders. No actual characters are displayed.

The password should follow industry standard practices regarding strong passwords. No easy-to-guess common names or words. Combinations of upper and lower case letters as well as numbers and special characters should be used. Solaris™ 8 will only recognize and use the first eight characters of any password string.

It is normally recommended that passwords *not* be written down. It is, however, acceptable to maintain a written password list for all important servers on a network. This list should be kept in a locked safe or filing cabinet that is only accessible by individuals with a need-to-know.

A good IT Security Plan will define an organization's password policy. This policy should be consulted and followed when creating or setting passwords.

Network Configuration

The following network configuration is recommended by Hal Pomeranz in the *Solaris Security Step by Step Version 2.0* guide available from the SANS Institute[2].

1. Create an `/etc/defaultrouter` file containing the IP address of the gateway (supplied by the NA). Various scripts in the startup directories (`/etc/rcX.d`) will check for this file and if it exists, will use the contents to set up proper routing.
2. Because the server should not act as a router, IP Forwarding is disabled by creating the `/etc/notrouter` file.

   ```
   # touch /etc/notrouter
   ```

3. Create the `/etc/resolv.conf` file. The local resolver routines will use either a local name resolution database (`/etc/hosts`, for example), or a remote domain server to resolve Internet names into addresses. Defining the `/etc/resolv.conf` will indicate to the resolver to use a remote domain server to resolve addresses. The file uses keyword-value pairs in the form of:

```
keyword value
```

Valid keywords include:

domain: specifies the default local domain to search
nameserver: defines the DNS server(s) to query for information. The IP address in dot-notation should be used. Up to three nameserver values are allowed.
search: a space or tab delimited list of domains in which to also search for host names. The value is limited to six domains and no more than 256 characters. Using this option is slow and will generate excessive network traffic if the domains listed are not local.
sortlist: will sort responses based on a set of rules
options: specifies optional behaviors. Possible parameters include: debug, ndots, retry, and retrans.

For more information on search, sortlist, and options, refer to the manual pages online:

```
# man resolv.conf
```

A typical `/etc/resolv.conf` file would look like this:

```
# cat /etc/resolv.conf
domain mycompany.com
nameserver 192.1.0.10
nameserver 192.1.0.20
nameserver 192.1.0.30
```

4. While it may be convenient to use DNS for hostname lookups, it is much safer to put a small list of trusted hosts in the local `/etc/inet/hosts` file. This protects the system from various DNS spoofing attacks (e.g. DNS cache poisoning). When administering a system in this manner, the SA needs to work closely with the NA and the DNS server SA in the event the IP address of a specific server changes.

   To ensure that the system looks at the local database before making a request to the DNS server, change the `hosts:` line of the `/etc/nsswitch.conf` file to be as follows:

   ```
   hosts:  files dns
   ```

5. Now reboot the system.

   ```
   # reboot
   ```

Installing Optional Software

See the section titled *Software Group Selection* for more information on the installation of optional software and the packages required (see Table 2). That section will indicate how to select additional packages while installing the OS. However, should a desired package be overlooked at that point of the installation, it is still possible to add it in later. As root, mount the Solaris™ 8 installation CD into the CD-ROM drive and issue the following commands:

```
# mount -r -F hsfs /dev/dsk/c0t2d0s0 /mnt
# cd /mnt/Solaris_2.8/Product
# pkgadd -d . pgk1 pkg2 pkg3 pkg4 …
```

The device shown is typically the CD-ROM drive, but could be different.

Applying Patches

Because vulnerabilities and other miscellaneous bugs will be discovered after the OS has been released, Sun continues to publish patches that are freely available to download and apply to the system.

It is a good idea to install any additional/optional packages from the OS *before* applying this patch cluster. Some of the patches may be for some of the optional software. If the patches are installed first, and the optional software second, the possibility exists for unpatched software to reside on the system! See Table 2 for more information on possible software of interest, and the previous section titled *Installing Optional Software* for instructions on how to add the software to the system.

Download ftp://sunsolve.sun.com/pub/patches/8_Recommended.zip to retrieve the latest cluster available. This file is very large (the August 10, 2001 release is 37.9MB), and therefore will require a high-capacity media type, CD-ROM or tape, for transfer to the new system versus floppy disks. For further information, `8_Recommanded.README` is also available from this same ftp directory and contains detailed information about the patches included and instructions for applying the cluster. Download the cluster archive to a working directory, and then change the run-level of the system to single-user mode.

```
# init S
```

When prompted, enter the root password to enter system maintenance mode.

Now move to the local directory and unzip the cluster:

```
# cd working_dir
# unzip -q 8_Recommended.zip
```

Using the `-q` option with the `unzip` command will suppress the voluminous output and will actually speed up the unzipping process. After the cluster has been unzipped, run the install script as follows:

```
# ./install_cluster -q -nosave
[…]
Installing 109279-14
Installing 109320-03
Installing 109326-06
Installing 109783-01
Installation of 109783-01 failed.  Return code is 2
Installing 109898-03
Installing 110075-01
Installing 110452-01
Installation of 110452-01 failed.  Return code is 8
Installing 110700-01
[…]
# reboot
```

The `-q` option suppresses the message warning the user to check for minimum disk space, and the `-nosave` option will suppress saving the base objects being patched, which prevents uninstalling a patch at a later date.

Various error codes will be displayed during the install process. `Return code 2` indicates that the patch was already installed from the OS, and `Return code 8` indicates that the patch applies to a software package that is not installed. This process is very lengthy. In fact, it will take longer than the actual install of the OS itself. It is very important to reboot when the process has completed so the patches will take effect.

The Sun patch directory will also contain a file named `Solaris8.PatchReport`, which is typically updated weekly, containing information about *all* patches, not just those included in the recommended patch cluster. This report shows *security* patches not included in the patch cluster. Sun Spport customers can use the `patchdiag` utility, which will notify system administrators of new patches.[3]

Removing Unnecessary Services

By default, many unnecessary services are started at boot time. There are two options: removing the files, or renaming them. Renaming may be preferable in the event that a service is turned off, but is later needed. The files in the `/etc/rc`*X*`.d` directory are actually symbolic links to files in the `/etc/init.d` directory. Links beginning with '`S`' will invoke the scripts with the "`start`" argument and those beginning with '`K`' will use the "`kill`" argument.

A good rule of thumb is to prepend `.NO` to the link name, which will prevent the file from being started by the OS at boot time.[4]

Files to rename in /etc/rc2.d[5]:

Prevents the automatic reconfiguration of Solaris'™ network parameters.

```
S30sysid.net
S71sysid.sys
S72autoinstall
```

NFS is a huge security hole and should not be used.

```
S73nfs.client
S74autofs
S73cachefs.daemon
S93cacheos.finish
```

RPC-based services are generally considered unsecure due to its limited authentication. If absolutely needed, Wietse Venema's version of `rpcbind` should be installed. It is available from ftp://ftp.porcupine.org/pub/security/rpcbind_2.1.tar.gz.

```
S71rpc
```

Disable the name service cache daemon.

```
S76nscd
```

Disable the LDAP cache manager.

```
S71ldap.client
```

Do not start up the sendmail daemon. Sendmail is *not* necessary for sending out e-mail from this system.

```
S88sendmail
```

`expreserve` is used to recover `vi` buffers in the event of a system halt during editing. It has historically had vulnerability problems.

```
S80PRESERVE
```

Files to rename in /etc/rc3.d[6]:

There is one other NFS-related link to delete:

```
S15nfs.server
```

Files to rename in /etc/rcS.d[7]:

Only remove this script if not using hot-pluggable devices:

```
S50devfsadm
```

Configuring Sendmail

As mentioned in the previous section, sendmail is not necessary for sending mail out to a mail/relay host. Therefore, it is disabled by renaming the `S88sendmail` file. Normally, the flow of mail out of a system is immediate, but due to the destination host being temporarily unavailable, the message could remain in a queue on the local system until the local queue is flushed. Add the following line to root's `crontab` file[8]:

```
7 * * * * /usr/lib/sendmail -q
```

This will flush the sendmail queue at seven (7) minutes after every hour. For an example of how to configure a minimal `sendmail.cf` file, see *Appendix B: Sendmail Configuration*.

Modifying Default Boot Services

Some services must be disabled by installing modified forms of the standard Solaris™ boot scripts. Modifying the original scripts has the undesirable side effect of possibly being overwritten by future patches or upgrades. New scripts should be written and placed in the `/etc/init.d` directory and the appropriate links created in the `/etc/rc`*X*`.d` directory.[9]

Save the following script from *Solaris Security Step by Step Version 2*.0 by Hal Pomeranz as `/etc/init.d/newinetsvc`[10]:

```
#!/sbin/sh

/usr/sbin/ifconfig -au netmask + broadcast +

if [ -f /usr/sbin/in.named -a -f
/etc/named.conf ] ; then
        /usr/sbin/in.named
        echo "starting internet domain name
server."
fi
```

Figure 1 – Replacement `/etc/init.d/newinetsvc` script

Now replace the link to `/etc/init.d/inetsvc` in `/etc/rc2.d` with a link to the new script:

```
# mv /etc/rc2.d/S72inetsvc /etc/rc2.d/.NOS72inetsvc
# ln -s /etc/init.d/newinetsvc /etc/rc2.d/S72newinetsvc
# chmod 744 /etc/init.d/newinetsvc
# chown root:root /etc/init.d/newinetsvc
```

This will disable DHCP, multicast routing and `inetd`.

Now make a copy of `/etc/init.d/devfsadm`:

```
# cp /etc/init.d/devfsadm /etc/init.d/newdevfsadm
# chmod 744 /etc/init.d/newdevfsadm
# chown root:root /etc/init.d/newinetsvc
```

Comment out any calls to `devfsadmd` and `devfseventd`[11].

Normally, `syslogd` will listen on UDP port 514 for messages from other hosts. This is only necessary on a remote logging host and would not be desirable on an Internet server. Therefore, create a new copy of the `/etc/init.d/syslog` file:

```
# cp /etc/init.d/syslogd /etc/init.d/newsyslog
# chmod 744 /etc/init.d/newsyslog
# chown root:root /etc/init.d/newsyslog
```

Then edit `/etc/init.d/newsyslog` and add the `-t` option to the `syslogd` invocation as follows to shut off UDP port 514:

```
/usr/sbin/syslogd -t > /dev/msglog 2>&1 &
```

Now modify the links[12]:

```
# mv /etc/rc2.d/S74syslog /etc/rc2.d/.NOS74syslog
# ln -s /etc/init.d/newsyslog /etc/rc2.d/S74syslog
```

Setting Kernel Parameters/Network Configuration

The initial configuration of network interfaces on a system is handled by the `/etc/init.d/inetinit` script. The parameters for these interfaces are initially set to their default values when the `inetinit` script is run as `/etc/rc2.d/S69inet`. Any changes to these parameters must be executed *after* `S69inet` is run to prevent the desired values from being reset back to the defaults. In addition, the changes to the parameter values should be executed as soon as possible after `S69inet` is executed because there is a short period of time in which the system will be vulnerable when booting. Adding these values to the end of the `inetinit` script is not a good idea since future patches or upgrades could overwrite this file[13].

The new file containing the configuration changes will be `/etc/init.d/netconfig`. Create this script as follows[14]:

```
1.  #!/bin/sh
2.  ndd -set /dev/tcp tcp_conn_req_max_q0 8192
3.  ndd -set /dev/tcp tcp_ip_abort_cinterval 60000
4.  ndd -set /dev/ip ip_respond_to_timestamp 0
5.  ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0
6.  ndd -set /dev/ip ip_respond_to_address_mask_broadcast 0
7.  ndd -set /dev/ip ip_ignore_redirect 1
8.  ndd -set /dev/ip ip_send_redirects 0
9.  ndd -set /dev/ip ip_forward_src_routed 0
10. ndd -set /dev/ip ip_forward_directed_broadcasts 0
11. ndd -set /dev/ip ip_forwarding 0
12. ndd -set /dev/ip ip_strict_dst_multihoming 1
13. ndd -set /dev/arp arp_cleanup_interval 60000
14. ndd -set /dev/ip ip_ire_arp_interval 60000
```

Figure 2 – The new `/etc/init.d/netconfig` script

In
Figure **2**, the line numbers used as a reference. An explanation of each line is as follows:

- SYN Floods – Attackers will sometimes perform a SYN Flood attack in which thousands of TCP connections are sent to a system with different (forged) source addresses, but the standard TCP 3-way handshake is not completed. Eventually the buffer for the pending connections will become "flooded" and the system will no longer be able to accept TCP connections.[15]
  2. `tcp_conn_req_max_q0 8192` – sets the limit for the number of half-open TCP connections the system will accept.
  3. `tcp_ip_abort_cinterval 60000` – Sets the time-out in milliseconds for how long the kernel will wait for a TCP connection to be completed. Increasing this value can help the system survive a *SYN Flood* attack.
- Smurfing - A *smurf* attack occurs when an attacker sends PING requests with faked source information to an Internet broadcast address, which transmits the PING to all the hosts on the network. When all of the hosts on the network reply to the "source" address, which is the victim, it gets flooded with bogus packets.[16] Setting all of the following parameters to 0 prevents a system from responding to particular ICMP messages.
  4. `ip_respond_to_timestamp 0`
  5. `ip_respond_to_timestamp_broadcast 0`
  6. `ip_response_to_address_mask_broadcast 0`
- Disable ICMP redirects – setting the following parameters will tell the system to ignore ICMP redirect packets and to not emit ICMP redirect packets.
  7. `ip_ignore_redirect 1`

8. `ip_send_redirects 0`

- Source routing – the following parameter setting will disable source routing on multi-homed systems. It has no effect on systems with only one NIC.
  9. `ip_forward_src_routed 0`
- Mapping
  10. `ip_forward_directed_broadcasts 0` – setting this to zero prevents a system from passing traffic to network broadcast address on an internal network.
- IP Forwarding – Turning off IP forwarding prevents a system from accepting and forwarding packets not destined for a local interface address. Failing to disable IP forwarding could possibly allow a hacker to bypass parts of network security.
  11. `ip_forwarding 0`
  12. `ip_strict_dst_multihoming 1`
- Adjusting ARP timeouts – The default Solaris™ value for holding cached ARP information is twenty minutes. Reducing this default value can help prevent ARP spoofing attacks, but can have a negative impact on network performance due to the increase in ARP traffic. The settings below will change the values from twenty minutes to 60 seconds.
  13. `arp_cleanup_interval 60000`
  14. `ip_ire_arp_interval 60000`

Now set the ownership and permission on the new script and create the new link:

```
# chown root:root /etc/init.d/netconfig
# chmod 744 /etc/init.d/netconfig
# ln –s /etc/init.d/netconfig /etc/rc2.d/S69netconfig
```

To further reduce resource consumption, add the following lines to the `/etc/system` file[17]:

```
set maxuprc = 128
set sys:coredumpsize = 0
```

The second line above will prevent any process from creating a `core` file. Since this system will be used as an Internet server and not a development system, there is no need for `core` files. `core` files contain a snapshot of the memory on the system when a program crashes. This file can have potentially sensitive information in it (e.g. possible password information) and could help an unauthorized user elevate his privileges.

Now reboot the system so that it will now use the updated kernel configuration.

Further Miscellaneous Configuration

Since no services will be started via `inetd`, `/etc/inetd.conf` and `/etc/inet/inetd.conf` can be removed and NFS-related configuration files can be deleted. Removing these files will help with system auditing. If any of the files reappear, this will be a

strong indication that the system has been compromised. Unused `crontab` files should also be removed.[18]

```
# rm /etc/inet/inetd.conf
# rm /etc/inetd.conf
# rm /etc/auto_*
# rm /etc/dfs/dfstab
# rm /var/spool/cron/crontabs/adm
# rm /var/spool/cron/crontabs/lp
```

For a hardened Internet server, only root should be allowed to execute at and cron jobs. Only users listed in `/etc/cron.d/cron.allow` and `/etc/cron.d/at.allow` can execute cron and at jobs respectively. To limit execute to root only, run the following commands[19]:

```
# echo root > /etc/cron.d/cron.allow
# echo root > /etc/cron.d/at.allow
# chmod 400 /etc/cron.d/*.allow
# chown root:root /etc/cron.d/*.allow
```

`sys` will need to be added to `/etc/cron.d/cron.allow` if system accounting is enabled (see section titled *System Accounting*).

Logging and Auditing

By default, `syslog` ignores messages logged to `LOG_AUTH`. `LOG_AUTH` is the authorization facility, which is made up of `login`, `su`, and `getty`. The priorities available (in descending order of severity) are *emerg*, *alert*, *crit*, *err*, *warning*, *notice*, *info*, *debug*, and *none*. `syslog` will log messages of the priority level selected plus all higher priority levels[20].

To enable logging of LOG_AUTH messages, add *at least* one of the following lines to the end of the `/etc/syslog.conf` file as follows[21]:

```
auth.info  /var/log/authlog
auth.info  @192.1.0.99
auth.info  /dev/term/a
```

The white space between the two columns *must* be a tab.

This will log messages of priority *info* and higher to LOG_AUTH. The first line will log messages to the file `/var/log/authlog`. The second line will log messages to a specific log host. If using this option be sure to use the IP address of the server rather than the hostname to ensure that the remote logging process does not fall victim to a DNS spoofing attack. The third line is an example of logging to a `tty` port. This could be a line printer or a personal computer (PC) connected to the `tty` port and configured to capture the logging information. Any combination

of the three examples can be used.[22]

To enable `syslog` logging, the initial log files must be manually created[23]:

```
# touch /var/log/authlog
# chmod 600 /var/log/authlog
# chown root /var/log/authlog
```

To enable logging of failed login data, the `loginlog` log file must be manually created[24]:

```
# touch /var/adm/loginlog
# chmod 600 /var/adm/loginlog
# chown root /var/amd/loginlog
```

Rotating Log Files

To keep log file sizes at a reasonable level, it is a good idea to rotate the log files on a regular basis.

Create the following file called `logrotate` (from Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*[25]) and place the file in an appropriate location on the file system (e.g., `/usr/local/cronjobs` or `/usr/local/bin`):

```
#!/bin/ksh
```

Figure 3 – `logrotate` script used for rotating log files.

Now add the following jobs to the `root`'s `crontab` using "`crontab -e`":

```
5 2 * * 0 /usr/local/cronjobs/logrotate /var/log/authlog 600 4
7 2 * * 0 /usr/local/cronjobs/logrotate /var/log/loginlog 600 4
```

This will rotate the logs once a week, every Sunday morning at 2:05 a.m. and 2:07 a.m. The log files will have permissions of 600, and 4 revisions will be kept of each. This means that only four weeks of log files will be maintained at any one time on the system. The log files should be included as part of the regular backup configuration.

The `/var/cron/log` file is a log file for `cron` jobs. By default the file `/etc/default/cron` should contain the line "`CRONLOG=YES`". This will log all `cron` activity to the `cron` log file. This log should be checked regularly for suspicious activity.

System Accounting

System accounting is used to control a set of tools used to build accounting systems. When

activated, processing accounting records are logged by the system kernel and connect time accounting is handled by the programs that write data to `/var/adm/wtmpx`.[26]

For system accounting to be enabled, the `SUNWaccr` and `SUNWaccu` packages must be installed. If they were not installed originally, see the section titled *Installing Optional Software* for information on how to add them to the system. In addition, the appropriate lines in `/etc/init.d/perf` must be uncommented (instructions are in the file by default). Also the edit the `sys` crontab by typing "`crontab -e sys`" and uncomment the appropriate lines.

The default `sys` crontab archives data every twenty minutes between 8:00 a.m. and 5:00 p.m., and a daily report is generated at 6:05 p.m. on Monday through Friday for data between 8:00 a.m. and 6:01 p.m. for 1200-second (20-minute) intervals. An alternative is to run the archive data every twenty minutes seven days a week and to generate a report every day of the week for data between 12:00 a.m. and 11:59 p.m. at twenty-minute intervals.[27]

For the alternative-logging scheme, leave the original crontab entries commented out and add the following two lines to the `sys` crontab:

```
0,20,40 * * * * /usr/lib/sa/sa1
45 23 * * * /usr/lib/sa/sa2 -s 0:00 -e 23:59 -i 1200 -A
```

Once these steps are completed, initialize the logs:

```
# /etc/init.d/perf start
```

The `acctcom` command will display the contents of the `/var/adm/pacct` file (the default file location for the accounting files).

Consult the `acct` manual pages for more information on system accounting.

Caveat! Process accounting can cause noticeable performance degradation. In addition, 40 bytes of data are logged per process, which could be significant on a busy web or file server. As helpful as it can be, it may not be feasible on all Internet servers.

Tightening User Access Controls

Remove unnecessary users from the `/etc/password` file[28]:

```
# passmgmt -d uucp
# passmgmt -d nuucp
# passmgmt -d lp
# passmgmt -d smtp
# passmgmt -d listen
# passmgmt -d nobody4
```

For non-interactive users in `/etc/passwd`, the shell should be set to `/dev/null`[29]:

```
# passmgmt -m -s /dev/null adm
# passmgmt -m -s /dev/null daemon
# passmgmt -m -s /dev/null bin
# passmgmt -m -s /dev/null nobody
# passmgmt -m -s /dev/null noaccess
```

If system accounting has been enabled, the `sys` account requires a valid shell. If not, then the shell for `sys` should also be set to `/dev/null`.

For more information on using `passmgmt`, consult the manual page.

Remove `.rhosts` support from the `/etc/pam.conf` file by either editing the file and removing all lines containing the string `rhosts_auth`, or by running the following commands[30]:

```
# grep -v rhosts_auth /etc/pam.conf > /etc/pam.new
# mv /etc/pam.net /etc/pam.conf
# chmod 644 /etc/pam.conf
# chown root:sys /etc/pam.conf
```

Edit `/etc/default/login` and change the line

```
CONSOLE=/dev/console
```

to

```
CONSOLE=
```

The default setting will disallow remote logins by `root`, but still allow logins from the console. The latter will still prohibit remote logins by `root`, but will also forbid logins directly to the console. A user must first login to their usual account, then run `su` to have root access.[31]

Displaying Warning Banners

The local security policy should be consulted on the use of login warning banners. A warning banner is a message displayed anytime a user, authorized or unauthorized, connects to the system (usually, but no limited to a telnet, ftp, or secure shell connection). Failure to display a banner could potentially be used by the defense in a criminal prosecution. A banner should *never* have a "welcome" message or appear as a friendly greeting.

The following is an example of a banner approved for use on Department of Defense computer systems. It was taken from the web site of the U.S. Department of Energy Computer Incident

Advisory Center (CIAC) [32]:

```
NOTICE TO USERS


This is a Federal computer system and is the property of the
United States Government. It is for authorized use only.
Users
(authorized or unauthorized) have no explicit or implicit
expectation of privacy.

Any or all uses of this system and all files on this system
may be intercepted, monitored, recorded, copied, audited,
inspected, and disclosed to authorized site, Department of
Energy, and law enforcement personnel, as well as authorized
officials of other agencies, both domestic and foreign.  By
using this system, the user consents to such interception,
monitoring, recording, copying, auditing, inspection, and
disclosure at the discretion of authorized site or Department
of Energy personnel.

Unauthorized or improper use of this system may result in
administrative disciplinary action and civil and criminal
penalties. By continuing to use this system you indicate your
awareness of and consent to these terms and conditions of
use.
LOG OFF IMMEDIATELY if you do not agree to the conditions
stated in this warning.
```

Figure 4 – Warning Banner approved for use by the Department of Defense.

Create `/etc/issue` and `/etc/motd` in accordance with the security policy on banner information.

On a side note, an Internet server should not be running telnet or non-anonymous ftp due to clear-text passwords. However, it is a good practice to configure these services for banners in the event the services are turned on at some point in the future.

Modifying /etc/default

Consult the password policy and set the appropriate values in the `/etc/default/passwd` file.

```
MAXWEEKS=10
MINWEEKS=
WARNWEEKS=1
```

In the above example, passwords will expire after 10 weeks, there is no minimum time a user must keep a password, a warning will be displayed beginning one week prior to the password expiring.

There may be a temptation to set `PASSLENGTH=10` (or even some higher number like 12) to force a minimum password length of 10 characters, which would increase the difficulty of a brute force attack. However, Solaris™ silently uses only the first eight characters of a password, so this would technically accomplish nothing.

It is possible to disable the `[STOP][A]` key sequence on the keyboard. Whether this is desired depends on the physical security of the system. If the system is properly housed in a locked cabinet in an access controlled computer room, it is not necessary and in fact would be undesirable. If the system were to hang for some reason (yeah, I know, this is not Windows 98), it would be possible to halt the system and synchronize the file systems from the PROM level. However, if the computer is in a less secure area the key sequence can be prevented by uncommenting the following line in the `/etc/default/kbd` file:

```
KEYBOARD_ABORT=disable
```

In the `/etc/default/login` file, set `RETRIES` to the number of failed login attempts that will be allowed before the login program exits. `SYSLOG_FAILED_LOGINS` is used to determine how many failed login attempts will be allowed before a failed login message is logged. It is highly recommended that this be set to `0` to log *all* failed login attempts. SLEEPTIME can be set to control the number of seconds the login command should wait before printing the "login incorrect" message to the console when a bad password is provided (valid values are from 0 to 5 seconds).

Setting the EPROM

The `EPROM` command can be used to view or change the values of parameters in the EEPROM. Only a handful of parameters will be discussed here. Consult the manual pages for a description of all available parameters.

The `security-modes` parameter can be see to `none`, `command`, or `full`. The default setting is `none`. When set to full, the system cannot reboot without a human operator. The best setting for

this parameter is `command`. This will require the EPROM password anytime an EPROM command is issued with the exception of `boot`.[33] When this parameter is set to `command` or `full` the user is prompted to enter the password. It is important to note that there is no way to recover this password should it ever be forgotten. The only way to recover from this situation is to order a new EPROM from Sun, which can take a significant amount of time. This password should be recorded and locked away in two or three very secure places (e.g. two or three locked file cabinets at the office and possibly at the SA's home). Leaving this parameter set to `none` leaves the system open to a denial of service attack should an attacker gain root access. The attacker could set this parameter himself and odds are that he or she will not be sharing it with the SA.

It is possible to monitor failed EPROM logins by monitoring the `security-#badlogins` parameter.

```
# eeprom security-#badlogins
security-#badlogins=0
```

The `oem-banner` parameter can be set to display a warning banner when the system is booted (see also the section on *Displaying Warning* Banners).

```
# eeprom oem-banner="Authorized users only.  All access is logged."
# eeprom oel-banner\?=true
```

This will replace the standard Sun banner, which will also mask the MAC address, host ID, and memory information.[34]

This particular setting may be optional depending again on the physical security of the machine. After all, if it is locked in a cabinet in a computer room, the SA will probably be the only person to ever see it.

Setting File System Security

To help protect the system binaries in `/usr` from Trojan horses (usually via rootkits), the file system can be mounted read-only by changing the configuration of the `/etc/vfstab` file. Other file systems should be mounted `nosuid` to prevent set-UID programs from executing there. The root file system cannot be mounted `nosuid` since `nosuid` also implies `nodev`.[35] The logging option is also added to writeable partitions. This helps prevent file system inconsistencies, which can slow or abort the boot process. The logging option makes the file system transactional. File modification requests are written to a hidden file before the actual file is modified. If the system is rebooted in an ungraceful fashion, these transactions are replayed on the file system, which eliminates the need to run `fsck`. This also prevents someone with physical access to the system from repeatedly crashing it until it needs to perform a file system consistency check. When this happens, the boot process is stopped when the system asks the SA to run `fsck`, at which point, someone would have physical access to the system with a root

shell.[36]

Here is how the new entries in the `vfstab` file might look:

```
/dev/dsk/c0t3d0s0    /dev/rdsk/c0t3d0s0   /           ufs   1   no   remount,logging
/dev/dsk/c0t1d0s7    /dev/rdsk/c0t1d0s7   /usr        ufs   1   no   ro
/dev/dsk/c0t3d0s3    /dev/rdsk/c0t3d0s3   /var        ufs   1   no   nosuid,logging
/dev/dsk/c0t5d0s7    /dev/rdsk/c0t5d0s7   /usr/local  ufs   2   yes  nosuid,logging
```

Third-Party Software Configuration

TCP Wrappers

TCP Wrappers was written by Wietse Venema. It acts as a "wrapper" around any TCP service (typically those found in the `/etc/inetd.conf` file). It has its own access control language that can be used to monitor and control incoming TCP connections. One of the best "features" of the tool is that it does not require any changes to existing software.

The source for TCP Wrappers can be obtained by accessing the URL ftp://ftp.porcupine.org/pub/security/index.html. Download the archive to the development system. When building this code for use on a Solaris™ 8 system, the ipv6 version of the software must be used.

After downloading the software, unzip and untar the archive:

```
# gunzip tcp_wrappers_7.6-ipv6.1.tar.gz
# tar xf tcp_wrappers_7.6-ipv6.1.tar
# cd tcp_wrappers_7.6-ipv6.1
```

Before continuing, be sure to read the `README` file packaged with the archive. It gives a detailed explanation of how the software works and explains (in more detail than given here) how to build and configure the software. Generally, the "Advanced configuration" option should be followed. In the case presented in this document, we are not running `inetd`, but integrating TCP Wrappers with Secure Shell will be covered in the next section.

Edit the `Makefile` and locate the setting for `REAL_DAEMON_DIR`. Set this to `/usr/sbin` (the default for Solaris™). This is where the "real" daemons live for telnet (`in.telnetd`), ftp (`in.ftpd`), etc. Also uncomment the line `STYLE=-DPROCESS_OPTIONS`. This will turn on the language extensions which will extend the capabilities of the access control language to display warning banners when a connection is made to the system (see section titled *Displaying Warning* Banners). The last change to make is to modify the logging facility used. By default, the `LOG_MAIL` facility is used, which means that all of the logs from TCP Wrappers will be mix in with the `sendmail` logs. Change the value for the `FACILITY` variable to be `LOG_AUTH`. In doing this, the logs for TCP Wrappers will now go in the same log file as the failed `su` and failed login attempts.

Now issue the make command as follows:

```
# make sunos5
gcc -O -DFACILITY=LOG_AUTH -DHOSTS_ACCESS -DPARANOID -DNETGROUP
-DGETPEERNAME_BUG -DBROKEN_FGETS -DLIBC_CALLS_STRTOK
-DSOLARIS_24_GETHOSTBYNAME_BUG   -DDAEMON_UMASK=022
-DREAL_DAEMON_DIR=\"/usr/sbin\" -DPROCESS_OPTIONS
-DSEVERITY=LOG_INFO -DRFC931_TIMEOUT=10
-DHOSTS_DENY=\"/etc/hosts.deny\"
-DHOSTS_ALLOW=\"/etc/hosts.allow\"  -DTLI     -DALWAYS_HOSTNAME
-o tcpd tcpd.o libwrap.a -lsocket -lnsl
[…]
```

The parameter `sunos5` says that we are using Sun OS version 5 (Solaris 8 = Sun OS version 5.8).

After a successful build, the binaries `tcpd`, `tcpdchk`, `tcpdmatch`, `try-from`, and `safe_finger` should be transferred to the new server.

An explanation of how to set up the `/etc/inetd.conf` file for a more typical Solaris™ installation will still be briefly shown. There were five binaries created from this build. They are `safe_finger`, `tcpd`, `tcpdchk`, `tcpdmatch`, and `try-from`. For more information on these utilities (and the `tcpd` service), see the `README` file packaged with the TCP Wrappers tar archive. On a typical install, `/etc/inetd.conf` would be updated to use `tcpd`, and `tcpd` would be copied to `/usr/sbin`. An example of how the `/etc/inetd.conf` would be changed for ftp and telnet are as follows:

Original file:

```
ftp     stream  tcp6    nowait  root    /usr/sbin/in.ftpd       in.ftpd
telnet  stream  tcp6    nowait  root    /usr/sbin/in.telnetd    in.telnetd
```

Changed file:

Figure 5 - Configuration of the `/etc/inetd.conf` file for a typical TCP Wrappers installation.

The two files needed from this build are `tcpd.h` and `libwrap.a`. These will be used to build SSH in the next section.

At the heart of TCP Wrappers is the access control language used grant or deny access. This is managed through the `/etc/hosts.allow` and /etc/`hosts.deny` files. Access will be granted when a rule is matched in the `hosts.allow` file. Otherwise, access is denied when a rule matches in `/etc/hosts.deny`. Otherwise, access is granted.

Since we are using the language extensions, the syntax of these files is a follows:

```
daemon_list : client_list option : option …
```

The `daemon_list` is a list of one or more daemon processes or a wildcard (`ALL` or `NONE`). The `client_list` is a list of host names, address, patterns or wild cards. An `option` is a keyword or value. Options are processed in the order they appear. The options available are quite lengthy and complex. Consult the `hosts_access.5` and `hosts_options.5` manual pages that are bundled with the software.

Usually, the best configuration is to set up a "deny-based" set of rules. This can be achieved as follows:

```
/etc/hosts.allow:

      sshd, ssh-x11 :   hostname1, \
                        hostname2, \
                        hostname3 : \
         banners /usr/sbin/banners/ : \
         spawn (/usr/bin/mailx -s  \
               "[TCP/Wrappers] ALLOW Access from %u@%h using %d." \
               sysadmin) : \
         ALLOW

/etc/hosts.deny:

      ALL : ALL : \
         spawn (/usr/bin/mailx -s  \
               "[TCP/Wrappers] DENY Access from %u@%h using %d." \
               sysadmin) : \
         DENY
```

Any host not matching a rule in `/etc/hosts.allow` will always match a rule in `/etc/hosts.deny` and will therefore be denied access. This particular `/etc/hosts.deny` file uses the `ALL` wildcard to match all daemons and all hosts.

The example shown for `/etc/hosts.allow` shows host names being used. In fact entire domains can be listed (`.mycompany.com`), an IP address can be used, a Class-B or Class-C IP address range (`192.1.` or `192.1.0.`) or the keyword `LOCAL`, which will match any host *not* containing a dot in the hostname.

The banners option is used in the `/etc/hosts.allow` file. After the banners option is a directory name. In this directory, a file with the same name as the daemon process will be displayed on the screen.

The spawn option in this case will send an e-mail to `sysadmin` (a mail alias defined in `/etc/aliases`) with a subject line containing information about the connection. It is usually

good idea to send denied connections to the e-mail of the SA. E-mail is usually checked more often than system logs, and if numerous e-mail notifications arrive in a short period of time, it could be an indication of an attack in progress. In the above example, an e-mail is also sent on successful connections. This is a matter of personal choice. A busy system with numerous users logging in during the day would more than likely generate too much e-mail traffic. On the other hand, a system being used only for serving web pages will usually be accessed only when the web master is updating the content of the data. In this case, it would be a good idea to use the e-mail example shown. If an increase in traffic is observed or if it is discovered that the web master is logging in every night between 2:00 a.m. and 4:00 a.m., it could of a tip off that an attacker has somehow successfully accessed the system.

To check the `/etc/hosts.allow` and `/etc/hosts.deny` files, use the `tcpdmatch` and `tcpdchk` utility programs.

```
# /usr/local/sbin/tcpdchk
warning: /etc/hosts.allow, line 8: sshd: no such process name in /etc/inet/inetd.conf
warning: /etc/hosts.allow, line 8: sshd2: no such process name in /etc/inet/inetd.conf
warning: /etc/hosts.allow, line 8: ssh-x11: no such process name in /etc/inet/inetd.conf
```

These warnings are OK in this example because the system demonstrated in this document does not use `inetd`.

```
# /usr/sbin/tcpdmatch sshd hostname2
warning: sshd: no such process name in /etc/inet/inetd.conf
warning: hostname2: hostname alias
warning: (official name: hostname2.mycompany.com)
client:   hostname hostname2.mycompany.com
client:   address  192.1.0.23
server:   process  sshd
matched:  /etc/hosts.allow line 8
option:   banners /usr/local/sbin/banners


+------------------------------------------------------------------------------+
|                                                                              |
|                            W A R N I N G ! !                                 |
|                                                                              |
|- - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -|
|                                                                              |
|  This computer is the propery of MyCompany, Inc.  It is for use by           |
|  authorized users only.  By accessing and using the computer system, you     |
|  are consenting to system monitoring, including the monitoring of            |
|  keystrokes.  Unauthorized or improper use of this system may result in      |
|  administrative disciplinary action and civil and criminal penalties.        |
|  By continuing to use this system, you indicat your awareness of and         |
|  consent to these terms and conditions of use.                               |
|                                                                              |
|  LOG OFF IMMEDIATELY if you do not agree to the conditions stated in this    |
|  warning!                                                                    |
|                                                                              |
+------------------------------------------------------------------------------+

You are unknown@hostname2.mycompany.com


option:   spawn (/usr/ucb/Mail -s "[www] ALLOW Access from unknown@hostname2.mycompany.com
using sshd." sysadmin)
option:   ALLOW
```

```
access:   granted
```

The example above indicates that the system named `hostname2` will be granted access. Notice also that the warning banner is displayed indicating that the banner configuration has been set up properly.

The example `hosts.allow` and `hosts.deny` files shown above are examples for use with secure shell, which is covered in the next section.

Secure Shell (SSH)

An Internet server should never run any sort of clear-text authentication protocols for user access. Therefore SSH can be installed to replace telnet, rsh, rlogin, and non-anonymous ftp. One freely available distribution is OpenSSH, which can be obtained from http://www.openssh.org. As of this writing, the latest version available to compile on Solaris systems is 2.9p2. Download and unpack this archive on the development system:

```
# gunzip openssh-2.9p2.tar.gz
# tar xf openssh-2.9p2.tar
# cd openssh-2.9p2
```

The `INSTALL` file included in the `tar` archive indicates that OpenSSH requires the Zlib data compression library and the OpenSSL encryption library. Zlib is available from http://www.freesoftware.com/pub/infozip/zlib/ and OpenSSL is available from http://www.openssl.org.

[**Important note**: As of this writing, the freesoftware.com domain has been experiencing problems, and therefore, the above link may not work. An alternative download link is ftp://ftp.uu.net/graphics/png/src/zlib-1.1.3.tar.gz]

The latest version of Zlib is 1.1.3. Download and unpack this archive:

```
# gunzip zlib-1.1.3.tar.gz
# tar xf zlib-1.1.3.tar
# cd zlib-1.1.3
```

Zlib uses a configuration script to query the system automatically for the setup. To compile and install the Zlib library, issue the following commands:

```
# ./configure
# make test
# make install
```

After issuing "`make test`" the last line of output should say, "`*** zlib test OK ***`". The install process will actually install the library on the development system. This is OK

because the OpenSSL library will be compiled in the next step and will need to access these files.

The latest version of OpenSSL is 0.9.6b. OpenSSL requires Perl. If Perl is not currently installed, see the section entitled *Installing Optional Software* for more information on Perl. Download and unpack OpenSSL:

```
# gunzip openssl-0.9.6b.tar.gz
# tar xf openssl-0.9.6b.tar
# cd openssl-0.9.6b
```

Just like the Zlib library, OpenSSH uses a configuration script. It can be configured, compiled, and installed with the following steps:

```
# ./config
# make
# make test
# make install
```

By default the OpenSSL library will be installed in `/usr/local/ssl`. Depending on the hardware being used, OpenSSL can take a very long time to compile.

Now that Zlib and OpenSSL are out of the way, OpenSSH can be compiled and installed. To continue from above, change back to the OpenSSH working directory and build the code.

The `INSTALL` file from the OpenSSH package states that OpenSSH will only compile using the GNU version of `make` which can be obtained from ftp://ftp.gnu.org/gnu/make. OpenSSH also uses the `configure` script to create the necessary Makefiles to build the software. By default OpenSSH will be installed in the `/usr/local` directory structure. Consult the `INSTALL` file if an alternative location is desired. The `--without-rsh` and `--disable-suid-ssh` options should be passed to the `configure` script to prevent the `ssh` client from using `rsh` if a secure shell service is not running on the remote machine and to disable installing `ssh` as suid. Depending on how the environment is set up for the compile process, it may be necessary to use the `--with-ssl-dir=`*`PATH`* option. However, if the OpenSSL software has been installed on the system and the `LD_LIBRARY_PATH` environment variable includes a pointer to the directory where it was installed, then this option should not be necessary. The `--with-tcp-wrappers` option is used because OpenSSH does not compile with TCP Wrappers support unless specifically told to do so when configure is run:

```
# ./configure --with-tcp-wrappers --without-rsh --disable-suid-ssh
creating cache ./config.cache
checking fo gcc… gcc
checking whether the C compiler (gcc) works… yes
[…]
OpenSSH has been configured with the following options:
                 User binaries: /usr/local/bin
               System binaries: /usr/local/sbin
           Configuration files: /usr/local/etc
               Askpass program: /usr/local/libexec/ssh-askpass
```

```
                Manual pages: /usr/local/man/manX
                    PID file: /var/run
      sshd default user PATH: /usr/bin:/bin:/usr/sbin:/sbin:/usr/local/bin
   Random number collection: Builtin (timeout 200)
             Manpage format: man
                PAM support: no
          KerberosIV support: no
                AFS support: no
               S/KEY support: no
        TCP Wrappers support: yes  <----------
        MD5 password support: no
IP address in $DISPLAY hack: no
   Use IPv4 by default hack: no
    Translate v4 in v6 hack: no

            Host: sparc-sun-solaris2.8
        Compiler: gcc
  Compiler flags: -g -O2 -Wall
Preprocessor flags: -I/usr/local/ssl/include  -I/usr/local/include
     Linker flags: -R/usr/local/ssl/lib -L/usr/local/ssl/lib  -L/usr/local/lib -
R/usr/local/lib
       Libraries: -lwrap -lz -lsocket -lnsl  -lgen -lcrypto
```

Check the messages displayed at the end of the configuration process to ensure that the option for TCP Wrappers has indeed been activated. If all looks correct, then compile the code:

```
# make
```

Now that the code has been compiled, it needs to be installed on the actual server. The install process for OpenSSH actually strips the debug information out of the executables using the strip command. Removing the debug information from the executables will make them smaller and faster. `strip` the files before moving them to the production server:

```
# strip sshd
# strip ssh-keygen
```

The files should be placed in the appropriate directories once moved to the production server. Even though the locations compiled into the code can usually be overwritten by passing the appropriate parameters from the command line, placing the files in the correct location will make the process much easier. For this example, `sshd` should be copied to the `/usr/local/sbin` directory and `ssh-keygen` should be copied to `/usr/local/bin`.

The `ssh_prng_commands` file should also be moved to the server and put in the directory compiled into the code (`/usr/local/etc` by default). If desired, the OpenSSH client programs (like `ssh`) can also be copied to the server.

OpenSSH must be properly configured when the installation is complete. Create a file named `/usr/local/etc/ssh_config` (the directory could be different based on the configuration options selected before compiling the code. The location of this file can also be passed to the `sshd` process on startup). The following is a sample `ssh_config` file that should be suitable for most systems. It was taken from *Solaris Security Step by Step Version 2.0* by Hal Pomeranz. A detailed explanation of each of the parameters can be found in the manual pages

for `sshd`.

```
# Standard SSHD port
Port 22
# Listen on all local addresses
ListenAddress 0.0.0.0
# Default to Protocol v2, then fall back to v1
Protocol 2,1
# Log messages from sshd to the AUTH facility
SyslogFacility AUTH
# Define the level of information to logged by sshd
LogLevel INFO
# The pid file contains the process ID of sshd
PidFile /usr/local/etc/sshd.pid
# Defines the files holding the private host keys
# sshd will refuse to use a file that is group/world-accessible.
HostDSAKey /usr/local/etc/ssh_host_dsa_key
HostKey /usr/local/etc/ssh_host_key
# The interval in seconds between ephemeral server key regenerations
KeyRegenerationInterval 900
# Number of Bits in the ephemeral server key.
ServerKeyBits 1024
# Time in seconds the server will wait for a successful login
LoginGraceTime 180
# Specifies whether X11 forwarding is enabled
X11Forwarding yes
# specifies that sshd should check file modes and ownership
# before accepting a login.
StrictModes yes
# Specifies whether the system should send keepalive messages
# to the host is still reachable.
KeepAlive no
# Specifies NOT to use login for interactive login sessions
UseLogin no
# Specifies that sshd should NOT check for new e-mail on
# interactive logins
CheckMail no
# Specifies that sshd should NOT print the message of the day
# for interactive logins
PrintMotd no
# Specifies that Password Authentication is allowed
PasswordAuthentication yes
# Specifies that empty passwords are not allowed.
PermitEmptyPasswords no
# Do not permit Root Logins remotely
PermitRootLogin no
# Do not use Rhosts authentication methods
IgnoreRhosts yes
RhostsAuthentication no
RhostsRSAAuthentication no
# tells sshd to ignore the user's $HOME/.ssh/known_hosts during
# RhostsRSAAuthentication or HostbasedAuthentications.
IgnoreUserKnownHosts yes
# Specifies that RSA Authentication is allowed
RSAAuthentication yes
# Specifies that DSA Authentication is allowed
DSAAuthentication yes
```

Figure 6 - Example /usr/local/etc/sshd_configfile [37].

One configuration parameter not shown here is `Banner`. If the banner option is not used with TCP

Wrappers, this would be an acceptable alternative. The file name assigned to this parameter will be displayed before login.

```
Banner /usr/local/etc/sshbanner.txt
```

A similar configuration file is used for the `ssh` client program. A global version can be placed in `/usr/local/etc/ssh_config`, and individual users can create their own configuration file in `$HOME/.ssh/config`. For a complete list of the parameters for this file, see the manual page for `ssh`.

Make sure that the permissions for the `sshd_config` and `ssh_config` file are set appropriately:

```
# chmod /usr/local/etc/sshd_config 600
# chmod /usr/local/etc/ssh_config 644
```

Now generate the server key files for the system:

```
# /usr/local/bin/ssh-keygen -t rsa1 -f /usr/local/etc/ssh_host_key -N ""
Generating public/private rsa1 key pair.
Your identification has been saved in /usr/local/etc/ssh_host_key.
Your public key has been saved in /usr/local/etc/ssh_host_key.pub.
The key fingerprint is:
9a:f6:7b:2a:eb:8e:15:ae:a4:b7:2d:67:ba:3a:a1:85 root@www
# /usr/local/bin/ssh-keygen -t dsa -f /usr/local/etc/ssh_host_dsa_key -N ""
Generating public/private dsa key pair.
Your identification has been saved in /usr/local/etc/ssh_host_dsa_key.
Your public key has been saved in /usr/local/etc/ssh_host_dsa_key.pub.
The key fingerprint is:
d1:52:a2:12:f3:cd:b8:18:19:7b:38:b4:7d:c0:2e:3f root@www
# /usr/local/bin/ssh-keygen -t rsa -f /usr/local/etc/ssh_host_rsa_key -N ""
Generating public/private rsa key pair.
Your identification has been saved in /usr/local/etc/ssh_host_rsa_key.
Your public key has been saved in /usr/local/etc/ssh_host_rsa_key.pub.
The key fingerprint is:
8d:12:b2:e7:a0:f0:b2:af:e1:37:dc:ab:b4:f5:b1:f2 root@www
```

Now create the script to start up the `sshd` process when the system is booted. This script was taken from *Solaris Security Step by Step Version 2*.0 by Hal Pomeranz, Hal.

```
#!/sbin/sh

case "$1" in
'start')
        if [ -x /usr/local/sbin/sshd -a -f
/usr/local/etc/sshd_config ]; then
                /usr/local/sbin/sshd -f
/usr/local/etc/sshd_config
        fi
        ;;
'stop')
        kill `cat /usr/local/etc/sshd.pid`
        ;;
*)
        echo "Usage: $0 { start | stop }"
        ;;
esac
exit 0
```

Figure 7 - Example /etc/init.d/sshd startup script[38]

Save this file as `/etc/init.d/sshd`. Now set the proper permissions and create a link to this script in `/etc/rc2.d`[39].

```
# chmod 744 /etc/init.d/sshd
# cd /etc/rc2.d
# ln –s ../init.d/sshd S75sshd
```

Start the secure shell daemon by either rebooting the system or issuing the following command:

```
# /etc/init.d/sshd start
```

Network Time Protocol (NTP)

Network Time Protocol is a time synchronization protocol that can be used to synchronize the date and time on all of the computer system located on a network. This can be useful for many reasons. Software developers sharing resources on NFS file system will be able to keep their source code properly synchronized, which helps with software projects using Makefiles. NTP is also of great use to SA's for keeping all of their primary servers synchronized. Should an

attacker successfully penetrate the network and any commandeer any of these systems, the time stamps in the log files will all be synchronized, which can help the SA track the path used to get through the network. Even more than the conveniences are the legal issues with using NTP. Should an attacker be charged with a crime for compromising a system or group of systems, the log files from multiple systems could be used as evidence in the case. Should these log files not be properly synchronized, it could potentially be detrimental to the prosecutor's case.

NTP comes standard with Solaris™ 8 but is not installed as part of the Core package. See the section entitled *Installing Optional Software* information on adding the NTP package if it has not already been installed.

The NTP protocol uses a hierarchical design. A *stratum* is an NTP classification, which is used to estimate the "distance" between a specified time server and a primary reference clock. The valid strata range is one to fifteen. The lower the value, the closer to the primary reference clock. Sixteen is used to describe an unreachable server. Zero represents a directly connected reference source.[40] A server's stratum is one plus the *lowest* stratum value of any server with which it is actively synchronizing.[41]

A typical approach would be for the network to have three NTP servers that each sync from different time servers (three each) and they should peer with each other. Local servers then peer with these machines and each other. Desktop clients should use `ntpdate` to synchronize with the local servers, but should never peer with them.[42]

The `/etc/rc2.d/S74xntpd` script checks for the existence of the `/etc/inet/ntp.conf` file (which does not exist by default). If `/etc/inet/ntp.conf` exists, the `/etc/rc2.d/S74xntpd` script will parse `/etc/inet/ntp.conf` and will set the system up as either a server or client based on the parameters defined in the file. There are numerous possible configurations for this file, but for the purposes of this document, an example for what an Internet server may use will be shown. The following is an `ntp.conf` file for a client machine[43].

```
driftfile /etc/ntp.drift

server 192.1.0.25
server 192.1.0.26
server 192.1.0.27

restrict default ignore
restrict 127.0.0.1 nomodify
```

The `driftfile` parameter is used to store information used by NTP to determine the drift tendencies of the system. By using this file, the system does not have to start recalculating the drift information every time the system is rebooted.

The three servers defined would generally be NTP servers internal to the local network.

Because an Internet server is often located in the Demilitarized Zone (DMZ) area of the network, a very restrictive security policy is implemented.

For more information on configurations for the `ntp.conf` file, consult the web page at http://nim.cit.cornell.edu/usr/share/man/info/en_US/a_doc_lib/files/aixfiles/ntp.conf.htm.

fix-modes

`fix-modes` is a tool written by Casper Dik, that modifies the ownership and permissions on various system files to be more secure. The source code can be retrieved from ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz. It will have to be built on the development system and then transferred to the new system.

```
# gunzip fix-modes.tar.gz
# tar xf fix-modes.tar
# make CC=gcc
gcc    -o secure-modes secure-modes.c -lintl
gcc    -o pmodes pmodes.c
# cd ..
# tar cf fix-modes.tar fix-modes
```

Now transfer the tar file to the new system and execute the following steps:

```
# tar xf fix-modes.tar
# cd fix-modes
# sh ./fix-modes
```

This script will remove world read and write permissions to all files listed in `/var/sadm/install/contents` with the exception of those listed in `exceptions.h` (`exceptions.h` is packaged with the fix-modes archive). When the program has completed execution, the `/var/sadm/install/contents.mods` file will contain a history of what files were changed.

Logcheck

Logcheck is an automated log file analyzer written by Craig Rowland. The program is a shell script run via `cron`. It analyzes the system's log files and if suspicious behavior is discovered, the root account is sent an e-mail. E-mail will only be sent if the log files are suspect.

The latest available version is 1.1.1 and it can be downloaded from a link on the web page at http://www.psionic.com/abacus/logcheck. Installation and configuration is quite easy. Move the archive to a working directory and execute the following commands:

```
# gunzip logcheck-1.1.1.tar.gz
# tar xf logcheck-1.1.1.tar
# cd logcheck-1.1.1
```

If the development system uses GNU C, edit the Makefile and change the `CC` definition so that it uses the correct compiler. Now run `make` to install the program.

```
# make sun
make install SYSTYPE=sun
Making sun
gcc -O -o ./src/logtail ./src/logtail.c
./src/logtail.c: In function `main':
./src/logtail.c:51: warning: return type of `main' is not `int'
Creating temp directory /usr/local/etc/tmp
Setting temp directory permissions
chmod 700 /usr/local/etc/tmp
Copying files
cp ./systems/sun/logcheck.hacking /usr/local/etc
cp ./systems/sun/logcheck.violations /usr/local/etc
cp ./systems/sun/logcheck.violations.ignore /usr/local/etc
cp ./systems/sun/logcheck.ignore /usr/local/etc
cp ./systems/sun/logcheck.sh /usr/local/etc
cp ./src/logtail /usr/local/bin
Setting permissions
chmod 700 /usr/local/etc/logcheck.sh
chmod 700 /usr/local/bin/logtail
chmod 600 /usr/local/etc/logcheck.violations.ignore
chmod 600 /usr/local/etc/logcheck.violations
chmod 600 /usr/local/etc/logcheck.hacking
chmod 600 /usr/local/etc/logcheck.ignore
Done. Don't forget to set your crontab.
```

This warning can be ignored. The return type is set to void.

All of the files are installed in the `/usr/local/etc` directory.

Since the `LOG_AUTH` authorization facility was turned on (see section entitled *Logging and Auditing*), the `logcheck.sh` file must be modified. Look in this file for the section labeled "`LOG FILE CONFIGURATION SECTION`," then for the sub-section entitled "`SunOS, Sun Solaris 2.5`." There will be two `$LOGTAIL` executions listed as follows:

```
# SunOS, Sun Solaris 2.5
$LOGTAIL /var/log/syslog > $TMPDIR/check.$$
$LOGTAIL /var/adm/messages >> $TMPDIR/check.$$
```

Add an entry for the authlog as indicated here:

```
# SunOS, Sun Solaris 2.5
$LOGTAIL /var/log/syslog > $TMPDIR/check.$$
$LOGTAIL /var/log/authlog >> $TMPDIR/check.$$
$LOGTAIL /var/adm/messages >> $TMPDIR/check.$$
```

Be very careful adding this entry and ensure that the append redirection (double >>) is entered properly.

The `INSTALL` file packaged in the archive recommends that the program be run hourly. Add the

following entry to the `root`'s crontab:

```
# logcheck
0 * * * * /bin/sh /usr/local/etc/logcheck.s
```

By default, the program will report *everything* in your log files. The program will "grep" the log files using entries defined in configuration files also located in `/usr/local/etc`. The following is an explanation of each file[44]:

- `logcheck.hacking` – contains pattern strings indicating a certifiable hack. If any matches are made from this file, the header "ACTIVE SYSTEM ATTACK" will be displayed in the e-mail report.
- `logcheck.violations` – contains pattern strings indicating events normally viewed as negative. Matches made from this file have the header "Security Violations" in the e-mail report.
- `logcheck.violations.ignore` – contains pattern strings that are reverse searched (using "`grep -v`") against the `logcheck.violations` file. Matches made here are *not* reported.
- `logcheck.ignore` – contains pattern strings that are used as a catch-all for data that will *not* be reported.

The e-mail report is sorted with the following priority[45]:

1. Active System Attacks.
2. Security Violations.
3. Unusual System Events.

Knowing what pattern strings should be used in each file may take some tweaking over a period of time. Err on the side of caution. It is better to report too much information than to risk missing a possible intrusion attempt by reporting too little.

The program uses a pointer to remember where it left off, which will prevent duplicated data in each e-mail report. It tracks the size and inode of the log file to enable it to tell when a log file has been rotated. If the inode of the log changes, or the file size is smaller than the last run, `logtail` will reset the counter and parse the entire file[46].

System Backups

Backups are an essential part of any disaster recovery plan. If the system has a hardware failure (e.g. bad hard drive or "fried" motherboard), or is damaged by fire, flood, or other natural disaster, the only way to retrieve the data on the system is through the backup archives.

A backup has another important feature in that it can be used to compare a clean operating

system against one that may be tainted if it is suspected that someone has successfully compromised the system.

The physical security of the backup medium is as important as the computer system itself, if not more. After all, tapes can be easily hidden in a pocket or briefcase. They should be stored in a safe or locked file cabinet at all times. For increased security, the storage cabinet should be located in an area with limited access. Multiple backup copies should be kept at all times. One copy can be kept close by for easy access, and the other should be kept in a secure off-site storage facility.

An initial backup of the system can be achieved using the `ufsdump` command. `ufsdump` should only be run when the file system is inactive (either unmounted or when in single-user mode). Boot the system to single-user mode:

```
# reboot -- -s
```

In the above example, the `--` parameter separates the `reboot` options from the `boot` options, and `-s` indicates to boot to single-user mode.

After the system has rebooted, run a file system consistency check and then mount all of the file systems:

```
# fsck
# mount -a
```

Now back up all of the file systems to some form of media, usually tape[47].

```
# mt /dev/rmt/0 rewind
# ufsdump 0f /dev/rmt/0n / /var /usr /usr/local /opt
# mt /dev/rmt/0 rewoffl
```

Now repeat this process on a fresh set of media for the second backup set.

A regular backup schedule should be implemented and rigidly followed. How often backups are run on the system will depend greatly on how often the data on the system changes. Consideration must be given to how often the log files are rotated and purged. Make sure backups are run often enough to preserve all log files. It is common for backups to be run on a daily basis. On a web server with static data, running a backup less often may be reasonable.

Potential Services

At this point, the server has been set up and is now ready to be deployed. It can safely be connected to the network. So far, the server is only running one service: Secure shell. SSH was

installed on the system to help with remote administration or to access or upload new data to the system in a secure manner. The NA should design the network so that SSH access can only be achieved from the private intranet, and even then, TCP Wrappers should limit which hosts from the internal network have access. Typical a system in a properly designed DMZ will not have access back to the internal network. If off-site access is desired by the webmaster or the SA, a secure dial-up server should be implemented on a separate system, of course, rather than opening a hole through the DMZ. Configuring a secure dial-up server is beyond the scope of this document, but implementation can be successfully and securely achieved through proper planning.

A server configured in this manner would quite likely be used as an anonymous FTP server or as an HTTP server. Therefore, installing the Apache web server will be illustrated.

Apache Web Server

Apache is an open-source web server maintained and distributed by the Apache Software Foundation (ASF), a non-profit corporation. Information on the Apache web server and the ASF can be found at the official web page at www.apache.org.

Apache is bundled with the Solaris™ 8 distribution, but is not installed with the Core System Support demonstrated in this document. It would be possible to add the Apache server to the installation demonstrated in the section on *Software Group Selection*, but downloading the latest version is more preferable because this will ensure the installation of the latest available version. More recent releases of the server are more likely to have been patched for any discovered vulnerabilities.

An in-depth discussion of every possible configuration setting for Apache is beyond the scope of this document (and would probably be longer than this document). Only basic settings will be discussed in this section. For more details on configuring an Apache web server, consult the documentation at http://www.apache.org/docs and for security tips, see http://httpd.apache.org/docs/misc/security_tips.html.

As of this writing, the latest version of the Apache web server is 1.3.20 and it can be downloaded from http://httpd.apache.org/dist/httpd/apache_1.3.20.tar.gz. A version 2.0 beta is currently available to download, but doing so is highly discouraged. Beta releases of software usually have not been properly tested and could contain vulnerabilities. It is much safer from a security standpoint to employ a more stable release.

After downloading the archive to the development system and moving it to a working directory, extract the contents:

```
# gunzip apache_1.3.20.tar.gz
# tar xf apache_1.3.20.tar
# cd apache_1.3.20
```

The `INSTALL` file will contain instructions for how to configure and compile the server. Pay close attention to the *Requirements* section for building the software; particularly the information regarding necessary disk space and use of an ANSI-C compiler. Perl 5 is optional, but is recommended.

It is possible to add other modules into Apache during the configuration process, including code for adding SSL support. For more information on these modules, see the web page for Related Projects at http://www.apache.org/related_projects.html. `README.configure` explains how to add these modules in during configuration.

Prior to working with the configuration script, the `<workingdir>/src/Configuration` file should be checked for any settings to modify.

- Towards the beginning of this file are settings related to the `Makefile` options (`CC=`, etc).
- Next is the rules configuration section. Read each section of this file very carefully to ensure that each parameter is set with a value appropriate for the system being configured. Since this document describes securing a Solaris™ system, the two Irix parameters (IRIXNIS and IRIXN32) can be ignored. If third-party modules are being installed, the `PARANOID` parameter should be set to `yes`. Modules can control the `configure` script in the same environment, and setting `PARANOID` to `yes` will cause `configure` to echo the code as the module is executed.
- In the modules configuration, be aware that the order matters! Modules listed first are executed last.

Apache uses a configuration script to prepare the necessary `Makefile` files for the installation. The `README.configure` file packaged with the archive will explain in detail all possible configuration options. One of the more important options will be the installation directory and web page directory. To determine the default information issue the `--show-layout` option. This will display the default information, then exit without actually configuring the `Makefile` files:

```
# ./configure --show-layout
Configuring for Apache, Version 1.3.20
 + using installation path layout: Apache (config.layout)

Installation paths:
              prefix: /usr/local/apache
         exec_prefix: /usr/local/apache
             bindir: /usr/local/apache/bin
            sbindir: /usr/local/apache/bin
         libexecdir: /usr/local/apache/libexec
             mandir: /usr/local/apache/man
         sysconfdir: /usr/local/apache/conf
            datadir: /usr/local/apache
           iconsdir: /usr/local/apache/icons
         htdocsdir: /usr/local/apache/htdocs
```

Apache is the default layout

```
                cgidir: /usr/local/apache/cgi-bin
            includedir: /usr/local/apache/include
          localstatedir: /usr/local/apache
             runtimedir: /usr/local/apache/logs
             logfiledir: /usr/local/apache/logs
          proxycachedir: /usr/local/apache/proxy

Compilation paths:
             HTTPD_ROOT: /usr/local/apache
      SHARED_CORE_DIR: /usr/local/apache/libexec
       DEFAULT_PIDLOG: logs/httpd.pid
   DEFAULT_SCOREBOARD: logs/httpd.scoreboard
     DEFAULT_LOCKFILE: logs/httpd.lock
      DEFAULT_XFERLOG: logs/access_log
     DEFAULT_ERRORLOG: logs/error_log
    TYPES_CONFIG_FILE: conf/mime.types
   SERVER_CONFIG_FILE: conf/httpd.conf
   ACCESS_CONFIG_FILE: conf/access.conf
 RESOURCE_CONFIG_FILE: conf/srm.conf
```

As indicated by the arrow in the above configuration test, the default "Apache" configuration is used. To select a particular file layout, use the `--with-layout=[F:]ID` option. The `config.layout` file contains information on all possible layout options. Similar to selecting disk partitions, the exact configuration will vary from system to system based on available file space or merely personal preference. Be sure to check the `README.configure` for more information on modifying any of these settings. For a typical build, the default settings may very well be suitable.

This example will demonstrate installing the web server with the default settings shown above:

```
# ./configure
Configuring for Apache, Version 1.3.20
 + Warning: Configuring Apache with default settings.   <-- warning
 + This is probably not what you really want.
 + Please read the README.configure and INSTALL files
 + first or at least run './configure --help' for
 + a compact summary of available options.
 + using installation path layout: Apache (config.layout)
Creating Makefile
Creating Configuration.apaci in src
Creating Makefile in src
 + configured for Solaris 280 platform
 + setting C pre-processor to gcc -E
 + checking for system header files
 + adding selected modules
 + checking sizeof various data types
 + doing sanity check on compiler and options
Creating Makefile in src/support
Creating Makefile in src/os/unix
Creating Makefile in src/ap
```

```
Creating Makefile in src/main
Creating Makefile in src/lib/expat-lite
Creating Makefile in src/modules/standard
#
```

In the above example, a warning is displayed about the use of the default settings for the `configure` script. This example uses the default settings; therefore, the warning can be ignored.

Now run `make` to compile the server code:

```
# make
===> src
===> src/os/unix
gcc -c  -I../../os/unix -I../../include   -DSOLARIS2=280 -DUSE_EXPAT -
I../../lib/expat-lite -DNO_DL_NEED
ED `../../apaci` os.c
gcc -c  -I../../os/unix -I../../include   -DSOLARIS2=280 -DUSE_EXPAT -
I../../lib/expat-lite -DNO_DL_NEED
ED `../../apaci` os-inline.c
rm -f libos.a
ar cr libos.a os.o os-inline.o
ranlib libos.a
<=== src/os/unix
===> src/ap
gcc -c  -I../os/unix -I../include   -DSOLARIS2=280 -DUSE_EXPAT -I../lib/expat-
lite -DNO_DL_NEEDED `../ap
aci` ap_cpystrn.c
[…]
gcc  -DSOLARIS2=280 -DUSE_EXPAT -I../lib/expat-lite -DNO_DL_NEEDED `../apaci` -o
ab   -L../os/unix -L../
ap ab.o -lap -los  -lsocket -lnsl
sed <apxs.pl >apxs \
    -e 's%@TARGET@%httpd%g' \
    -e 's%@CC@%gcc%g' \
    -e 's%@CFLAGS@% -DSOLARIS2=280 -DUSE_EXPAT -I../lib/expat-lite -DNO_DL_NEEDED
`../apaci`%g' \
    -e 's%@CFLAGS_SHLIB@%%g' \
    -e 's%@LD_SHLIB@%%g' \
    -e 's%@LDFLAGS_MOD_SHLIB@%%g' \
    -e 's%@LIBS_SHLIB@%%g' && chmod a+x apxs
<=== src/support
<=== src
```

Even though the build process is being run on the development system, go ahead and run the install process:

```
# make install
===> [mktree: Creating Apache installation tree]
./src/helpers/mkdir.sh /usr/local/apache/bin
mkdir /usr/local/apache
mkdir /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/bin
./src/helpers/mkdir.sh /usr/local/apache/libexec
mkdir /usr/local/apache/libexec
[…]
<=== [config]
+--------------------------------------------------------+
| You now have successfully built and installed the      |
| Apache 1.3 HTTP server. To verify that Apache actually |
| works correctly you now should first check the         |
| (initially created or preserved) configuration files   |
|                                                        |
|   /usr/local/apache/conf/httpd.conf
|                                                        |
| and then you should be able to immediately fire up     |
| Apache the first time by running:                      |
|                                                        |
|   /usr/local/apache/bin/apachectl start
|                                                        |
| Thanks for using Apache.        The Apache Group       |
|                                 http://www.apache.org/ |
+--------------------------------------------------------+
#
```

The default configuration will put all of the installed files under the `/usr/local/apache` directory tree. This makes the job of moving the installed files to the production server much easier. On the development system, run the following commands:

```
# cd /usr/local
# tar cf apache.tar apache
```

If desired, the `/usr/local/apache` directory can now be removed from the development system.

Now move the file `apache.tar` to the `/usr/local` directory of the production server and issue the following commands:

```
# cd /usr/local
# tar xf apache.tar
```

Now all that is necessary is to edit the configuration files and start the server.

Edit the file `/usr/local/apache/conf/httpd.conf`. Search for all references to the

hostname of the development system. The Apache install process will place these entries there automatically. Rename the hostname to the name of the production server. Uncomment the line containing the `ServerName` parameter and ensure that the value of the hostname given is a valid hostname (it should be the name the NA assigned to this host).

To start the web server, issue the following command:

```
# /usr/local/apache/bin/apachectl start
/usr/local/apache/bin/apachectl start: httpd started
```

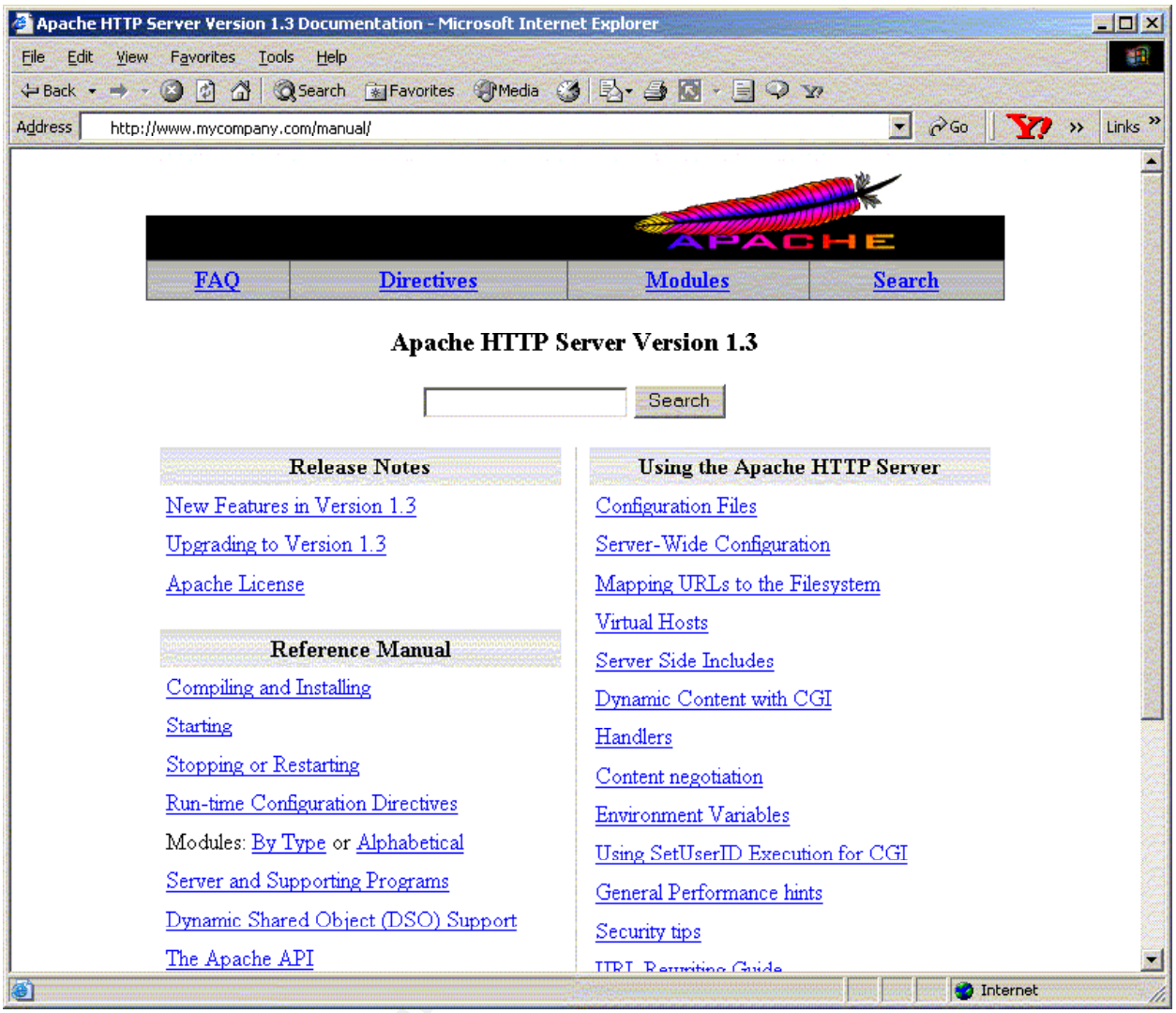
To ensure that the server has started and is running, check the process status table:

```
# ps -ef | grep httpd
  nobody  5237  5235  0 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
  nobody  5239  5235  0 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
    root  5235     1  1 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
    root  5242  5095  0 12:29:10 pts/2    0:00 grep httpd
  nobody  5240  5235  0 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
  nobody  5236  5235  0 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
  nobody  5238  5235  0 12:29:07 ?        0:00 /usr/local/apache/bin/httpd
```

By default the `httpd.conf` file assigns the parameter `StartServers` as `5` and the `User` parameter is set to `nobody`. As can be seen above, there are 5 `httpd` processes running owned by the user `nobody`.

The default location for the web page (html) data is in the `/usr/local/apache/htdocs` directory as defined by the `DocumentRoot` parameter in the `httpd.conf` file.

Now try to access the server from another system on the network using a web browser. A good starting place would be the on-line manual pages:



After the service has been tested and appears to be running OK satisfactorily, create a startup script to start the server up when the system boots. Save the script shown in Figure 8 as `/etc/init.d/apache`.

```
#!/bin/sh

APACHE_PATH=/usr/local/apache/bin

case $1 in
'start')
        if [ -f ${APACHE_PATH}/apachectl ]; then
                ${APACHE_PATH}/apachectl start
        fi
        ;;
'stop')
        if [ -f ${APACHE_PATH}/apachectl ]; then
                ${APACHE_PATH}/apachectl stop
        fi
        ;;
esac
```

Figure 8 - Apache server startup script

After saving the script, ensure the file permissions are correct and create a link in the `/etc/rc2.d` directory:

```
# chown root:root /etc/init.d/apache
# chmod 744 /etc/init.d/apache
# ln –s /etc/init.d/apache /etc/rc2.d/S95apache
```

As mentioned above, there is a manual page containing security tips that can be found at `http://<hostname>.mycomany.com/manual/misc/security_tips.html`. This file contains easy to follow information on general security measures that will help make the server more secure.

A few comments regarding security[48]:

- Pay particular attention to the settings for the file and directory permissions. The httpd binary, configuration files, and the log files can all be protected, but access to the `htdocs` directory can still be modified to allow modification by a webmaster (normal system user). The htdocs directory hierarchy does not contain any executables and is therefore safe.
- It is recommended that script aliased CGI be used over non-script aliased CGI. The SA is then in full control of all CGI content and can examine each program or script placed in the CGI All CGI programs or scripts should be carefully examine for any possible security holes. If the files are binary executables, ensure that the software developer has a strong understanding of buffer overflows and how to prevent them.

- Add the following to the `httpd.conf` file:

```
<Directory />
     Order Deny,Allow
     Deny from all
</Directory>
```

  This will block access to the entire file system and will ensure that clients (web browsers) are unable to "walk" through the entire file sytem. Now, the appropriate Directory directives can be added as needed:

```
<Directory /home/*/public_html>
     Order Deny,Allow
     Allow from all
</Directory>
<Directory /usr/local/apache/htdocs>
     Order Deny,Allow
     Allow from all
</Directory>
```

- When setting up the `UserDir` directive, be very careful not to open a hole in the system. Setting "`UserDir ./`" would map `/~root` to `/`, which would allow clients access to the entire file system. To prevent this, set the following `UserDir` directive in the `httpd.conf` file:

```
UserDir disabled root
```

The information given here should provide a good start to configuring and securing the Apache web server.

Conclusion

This document has presented the basic steps necessary to secure a Sun Solaris™ 8 system for use as an Internet server. There are other tools and techniques that can be added to the steps present here. Each layer of security added to a system is one more barrier to stopping an unauthorized intruder. Other techniques to consider are intrusion detection systems, or `chroot`-ing services like ftp or http.

After the server has been implemented, daily diligence is required to ensure that it is functioning properly and that no suspicious activity is occurring on the system. Log files must be checked often. If logs show that the webmaster is updating the web site at odd hours of the night, he or she should be contacted immediately to determine if in fact it was legitimate.

If any inappropriate activity is observed, the IT Security Plan should indicate a series of steps to follow. This will usually involve contacting members of an Incident Response Team.

Keep abreast of the latest vulnerabilities with the software being run on the system. Numerous e-mail lists distribute the latest vulnerability warnings for the more popular operating systems and third-party software packages. Very often the messages will specify a patch or work-around for a discovered vulnerability. See *Appendix D: Mailing Lists* for more information on mailing lists.

Appendix A: Acronyms

Definitions for most of these terms can be found by searching the dictionary at http://www.webopedia.com.

ACL – Access Control List
ARP – Address Resolution Protocol
ASF – Apache Software Foundation
CERT® – Carnegie Mellon University's Computer Emergency Response Team
CIAC – U.S. Department of Energy's Computer Incident Advisory Center
CGI – Common Gateway Interface
DHCP – Dynamic Host Configuration Protocol
DMZ – Demilitarized Zone
DNS – Domain Name Service
EPROM – Erasable Programmable Read-Only Memory
FTP – File Transfer Protocol
GCUX – Global Information Assurance Certification for Unix Security Administrators
GIAC – Global Information Assurance Certification
ICMP – Internet Control Message Protocol
IP – Internet Protocol
IT – Information Technology
LDAP – Lightweight Directory Access Protocol
MAC – Media Access Control
NA – Network Administrator
NFS – Network File System
NIC – Network Interface Card
NTP – Network Time Protocol
OS – Operating System
PC – Personal Computer
RPC – Remote Procedure Call
SA – System Administrator
SANS – System Administration, Networking, and Security
SSH – Secure Shell
SSL – Secure Socket Layer
TCP – Transmission Control Protocol
UPD – User Datagram Protocol
UPS – Uninterruptible Power Supple

Appendix B: Sendmail Configuration

The following is an example of a minimal `sendmail.cf` configuration file. This is a suitable file for systems where local mail delivery is not required. Make sure the `mailhost` is set properly for the `DR` parameter. One solution is to define a host named `mailhost` in `/etc/hosts` and use `mailhost` here as listed. This example was taken from *Solaris Security: Step-by-Step* by Hal Pomeranz[49].

```
# Minimal client sendmail.cf

### Defined macros
# The name of the mail hub
# PUT APPROPRIATE HOSTNAME FOR LOCAL SITE HERE!!!
Drmailhost

# Define version
V8

# Whom errors should appear to be from
DnMailer-Daemon

# Formatting of the UNIX from line
DlFrom $g $d

# Separators
Do.:%@!^=/[]

# Form of the sender's address
Dq<$g>

# Spool directory
OQ/usr/spool/mqueue

### Mailer Delivery Agents
# Mailer to forward mail to the hub machine
Mhub,   P=[IPC], S=0, R=0, F=mDFMuCX, A=IPC $h
# Sendmail requires these, but are not used
Mlocal, P=/dev/null, F=rlsDFMmnuP, S=0, R=0, A=/dev/null
Mprog,  P=/dev/null, F=lsDFMeuP, S=0, R=0, A=/dev/null

### Rule sets - WHITESPACE BETWEEN COLUMNS MUST BE TABS!!!

S0
R@$           +$#error $:            Missing user name
R$+           $#hub $@$R $:$1        forward to hub

S3
R$*<>$*       $n                     handle <> error address
R$*<$*>$*     $2                     basic RFC822 parsing
```

Appendix C: Useful URLs

The following is a list of useful URLs that can be helpful in securing a server or in keeping abreast of the latest security news or vulnerability notices.

Latest Recommended Patches from Sun – ftp://sunsolve.sun.com/pub/patches
Solaris Security FAQ – http://www.itworld.com/Comp/2377/security-faq/
Sunsolve – http://sunsolve.sun.com
Sun Security Bulletins – http://sunsolve.sun.com/security

Tripwire Intrusion Detection Software – ftp://coast.cs.purdue.edu/pub/tools/unix/ids/tripwire/
Wietse's tools and papers – ftp://ftp.porcupine.org/pub/security/index.html (including TCP Wrappers)
Logcheck – http://www.psionic.com/abacus/logcheck

Apache Software Foundation – http://www.apache.org
Apache suEXEC – http://www.apache.org/docs-2.0/suexec.html
CGIWrap – http://cgiwrap.unixtools.org/

Zlib Data Compression Library – http://www.freesoftware.com/pub/infozip/zlib (NOTE: As of this writing freesoftware.com is off-line. An alternative site is ftp://ftp.uu.net/graphics/png/src/zlib-1.1.3.tar.gz)
The OpenSSL Project – http://www.openssl.org/
OpenSSH – http://www.openssh.org/
Secure Shell FAQ – http://www.employees.org/~satch/ssh/faq/

Nessus Security Scanner – http://www.nessus.org/

Sendmail Consortium – http://www.sendmail.org

Security Focus – http://www.securityfocus.com
CERT Coordination Center – http://www.cert.org/
U.S. Department of Energy Computer Incident Advisory Center (CIAC) – http://www.ciac.org/ciac/
Internet Security Systems, Inc.: X-Force – http://xforce.iss.net/
AntiOnline.com – http://www.antionline.com/
Attrition Security – http://www.attrition.org/security/
The SANS Institute – http://www.sans.org (SANS also runs http://www.incidents.org).

Appendix D: Mailing Lists

Mailing lists are a good way of keeping up with the latest patch releases, security news, or to keep in touch with other SA's around the world. Be sure to follow proper netiquitte and refer to the rules of the mailing list before deciding to post a message.

To subscribe to Sun's Customer Warning System (CWS), send an e-mail to security-alert@sun.com with the command `subcribe` in the *subject line* (not the message body).

Carnegie Mellon University runs the Computer Emergency Response Team (CERT®), a federally funded research and development center. To subscribe to the CERT advisory mailing list, send an e-mail to majordomo@cert.org with the command subscribe cert-advisory in the message *body* (not in the subject line). These bulletins will also cover all major platforms.

SecurityFocus.com runs several mailing lists, which can be found by accessing their web site at http://www.securityfocus.com. Of particular interest to Sun SA's is their FOCUS-SUN mailing list. Unlike the Sun, CIAC, and CERT mailing lists, this one provides a forum for SA's to discuss security and system administration issues relating to the Sun platform. To subscribe to this list, you must fill out a form on their web site.

SecurityFocus.com also runs a popular mailing list called *Bugtraq*. Bugtraq is a moderated mailing list that discusses *in detail* information related to vulnerabilities, how they work, how to prevent them, and how to exploit them. To subscribe to this list, you must fill out a form on their web site.

The SANS Institute offers three different newsletters. Security Alert Consensus (weekly), SANS NewsBits (weekly), and the SANS Windows Security Newsletter (monthly). For information on subscribing, visit the following URL: http://server2.sans.org/sansnews.

Internet Security Systems' X-Force group hosts a web page listing a large variety of mailng lists. For more information on these, visit http://xforce.iss.net/maillists/otherlists.php.

References

[1] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 115.
[2] Pomeranz, Hal ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 2.
[3] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institude, Baltimore, Maryland, May 2001, pg 122.
[4] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 6.
[5] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 6.
[6] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 6.
[7] Sammons, Eric L., *Installing and Securing Solaris 8*, http://www.sans.org/y2k/practical/Eric_Sammons_GCUX.zip, 2001, pg. 17.
[8] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 20.
[9] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 8.
[10] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 32.
[11] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 8-9.
[12] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 9.
[13] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 127.
[14] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 10.
[15] Bob Toxen, *Real World Linux Security: Instrusion Prevention, Detection, and Recovery*, Prentice Hall PRT, 2001.
[16] INT Media Group, http://www.webopedia.com/TERM/s/smurf.html, 2001.
[17] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 11.
[18] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 12, and Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 136.
[19] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 166.
[20] Sun Microsystems, *syslog.conf man pages*, 1997.
[21] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 168.
[22] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 168.
[23] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 169.
[24] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 169.
[25] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 36.
[26] Sun Microsystems, *acct man pages*, 1997.
[27] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 176.
[28] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 16-17.
[29] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 17.
[30] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 17.
[31] Campione, Jeff, *Solaris 8 Installation Checklist*, http://www.sans.org/y2k/practical/Jeff_Campione_GCUX.htm, 2000.
[32] U.S. Department of Energy Computer Incident Advisory Center (CIAC), *Creating Login Banners*, http://www.ciac.org/ciac/bulletins/j-043.shtml.
[33] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 188.
[34] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 189.

[35] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 12-13.
[36] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 146.
[37] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, Appendix C, pg. 34
[38] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, Appendix D, pg. 35
[39] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 26.
[40] GPSClock.com, http://www.gpsclock.com/ntp/stratum.html, 1999.
[41] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 70.
[42] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 75.
[43] Brotzman, Lee and Pomerantz, Hal *Linux/Solaris Practicum*, SANS Institute, Baltimore, Maryland, May 2001, pg. 90.
[44] Rowland, Craig, `INSTALL` file packaged with Logcheck Version 1.1.1, http://www.psionic.com/tools/logcheck-1.1.1.tar.gz.
[45] Rowland, Craig, `INSTALL` file packaged with Logcheck Version 1.1.1, http://www.psionic.com/tools/logcheck-1.1.1.tar.gz.
[46] Rowland, Craig, `INSTALL` file packaged with Logcheck Version 1.1.1, http://www.psionic.com/tools/logcheck-1.1.1.tar.gz.
[47] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, pg. 27.
[48] Apache Software Foundation., *Security Tips for Server Configuration*, http://httpd.apache.org/docs/misc/security_tips.html, 2001.
[49] Pomeranz, Hal, ed., *Solaris Security Step by Step Version 2.0*, The SANS Institute, 2001, Appendix B, pg. 33.