



Global Information Assurance Certification Paper

Copyright SANS Institute
Author Retains Full Rights

This paper is taken from the GIAC directory of certified professionals. Reposting is not permitted without express written permission.

Installation and Configuration of a Solaris 8 Security Server



Practical

GCUX Version 1.8

Robert C. Donaldson

June 2002

Index

Index.....	2
Description of the system.....	3
General Description	3
Final role of the system.....	3
Risk analysis of the system.....	5
Step by Step Procedures.....	6
Initial O/S Installation.....	6
Package additions and removal.....	8
Patch Installation and Removal.....	11
Post Installation Configuration.....	12
Password/Login Policies.....	12
Hardening O/S.....	14
Installation of security software.....	29
Testing and Verification	43
Ongoing maintenance.....	58
Summary/Conclusions	59
References.....	60

Description of the system

General Description

Hardware	-	Sun Ultra 10
Operating System	-	SunOS 5.8 (sun4u)
CPU	-	300 Mhz
Ram	-	640 MB
Hard disks	-	(2) Seagate Medalist PCI 4 GB
Hub	-	Arsence Palm-9E 8 Port Ethernet
Laptop	-	Dell Latitude (Windows 2000 Server)

The hub is used to connect the Sun Server and the Laptop. Sftp will be used to transfer files from the Laptop to the Sun Server. Laptop will be used to connect to the Internet via modem to download Sun packages, patches, and applications. Sun server is in locked location with no access other than system admin.

Final role of the system

This server will provide three primary functions:

1. First function will be as a test platform used to experience and investigate the ramifications of installing the Solaris 8 operating system using the SANS Institute Step-by-Step hardening practices and installing many "Security" applications such as Nessus, Ethereal, Snort, Dsniff, Etc. Will use the "core" installation, but will add packages to add the "CDE" environment so that the system can function in a multi-window GUI environment. In this test environment, the server can function as a "Model Office" type of server to incorporate various security techniques and experience first hand the tradeoffs between tighter security and user friendliness. This will be a single purpose system with no other applications installed other than the "security" apps, and no other users will be added to the system.
2. Second function will be as a Security Assessment Server using the Security software to assess any of its own security vulnerabilities. I call it a Security Server because it will be a server for Nessus as well as a client. Many of the other Security applications will be run as a client so a case could be made that this Sun box is primarily a workstation, but since it is functioning as a true server at least in the case of Nessus, I will continue to call it a server.
3. Third function will be as a potential Security Assessment Server attached to some other local LAN by either physically moving the server to the new location, or providing the dial up capability through a modem to allow access to a LAN.

Here is a list of the software that will be added that is not included in the O/S but will be added to enable the system to accomplish the above goals:

1. aide-0.7 ## Freeware replacement for tripwire, monitors files for any file changes such as permissions, checksums, etc.
2. dsniff-2.3 ## Checks for passwords in packets.
3. ethereal-0.9.0 ## Is a GUI network protocol analyzer.
4. fixmodes-2.8 ## Tightens Solaris file security by changing file permissions, ownerships, groups.
5. gcc-2.9.5 ## Required to compile applications that do not have pre-compiled packages, or to have more control over parameters to the compiler. Can remove if server is connected to a different LAN.
6. ip_fil.3.4.27 ## Filters packets by protocol, port, source and destination
7. logcheck-1.1.1 ## Check specified log files on re-occurring basis
8. openSSH-3.1 ## Secure replacement for ftp and telnet.
9. openSSL-0.9.6c ## Needed for openSSH.
10. snort-1.8.1 ## Allows for network traffic analysis.
11. sudo-1.6.3 ## Used to control and log access to root.
12. nessus-1.2.0 ## Security Scanner and security auditing tool.
13. nmap-2.54 ## Port scanner, also used by Nessus.
14. pgp-6.5.8 ## Encrypts data such as Nessus reports.
15. tcp_wrappers-7.6 ## Used to help make “remote” services more secure via “hosts.allow”.

Risk analysis of the system

As long as Sun server remains on the private LAN, disconnected from any network access and physically secured in a locked room, there are no real security concerns other than someone breaking into the house and stealing the computer.

It is more interesting to consider the security concerns if the server were to be connected to a LAN that has access to the Internet and users on that LAN or others with access that could try to exploit any weaknesses of the server. Under that scenario, there would be two primary security concerns:

1. The first and most serious risk would be for someone to gain access to any security assessment reports. Since these reports would potentially contain sensitive information for the vulnerabilities of the client's LAN, it could be very harmful for their business for that information to be in the wrong hands. The primary means to deal with this risk is to use PGP to encrypt the Security Reports and to immediately store the Reports on offline media such as a tape or even a floppy disk.
2. The second risk and equally as serious is for anyone to gain unauthorized access to the server either via root access or the admin user. Since the admin user has access to all commands via "sudo" both the super-user and the admin user has access to all commands. This gives the user the potential to take over the system. However, as long as unauthorized access is detected, the system can simply be rebuilt with minimal serious repercussions. Again, the only real sensitive data on the system would be the Security Assessment Reports. If the unauthorized access is undetected, that could prove to be very serious, because the unauthorized user could modify system files, which could result in the Security Assessment providing invalid data. Also, if an unauthorized user gained access to the system, they could use the Security Assessment tools on that LAN to find vulnerabilities on the LAN and to issue attacks on those vulnerabilities. The primary means of avoiding access to the server is the general concept of "hardening" the operating system.

The system has to be secure as the Security Reports are valuable to the client. If the client is a bank, for example, and millions of dollars could be lost if their servers were compromised, then the server needs to be most secure.

Generally speaking, it is not known who the new clients might be. Therefore, it would not be known how important their Security Assessment data would be. It would best to make the Sun Server as secure as possible to accommodate any potential clients security needs regarding the importance of their data and their server. In terms of the client's server, it is important to note that not only is their data potentially very important, but also the client may have very important requirements for server uptime. A hacker could possibly use the Security Assessment report to generate a denial of service attack that could result in serious consequences to the client.

3. Because this server will at times be running with the CDE functionality enabled, and also running as an X server, this increases the vulnerability of the server while it is on the network. Server will also be running, at times, `nessusd`, which is by default port 1241. `nessusd` will not be started automatically with a startup script, but only run manually when a security assessment is ready to be run. This will help minimize the vulnerability of the `nessusd` being always available for someone to try to login to it. Even though the Nessus client requires an id and password to connect to the Nessus server, it is still important to minimize the availability of this service. In order to mitigate these risks, IP Filter will be installed. This allows system to pass or block specific ports for all address or by specific addresses. Appendix A – Common Vulnerable Ports (SANS Institute resources, p 26) lists X Windows - 6000/tcp through 6255/tcp as some of the ports commonly probed and attacked. During the configuration of IP Filter, specific rules will be set to disable these ports.

Step by Step Procedures

Note: Will use the following typeface convention to help distinguish the type of text. Commands entered into the computer will be shown as normal text in bold with a preceding # to designate the root prompt, for example:

cat /etc/passwd

Configuration file input or output and scripts will also be normal type in bold. For example:

ndd -set /dev/ip ip_forwarding 0

Comments will be preceded by two #'s, for example:
This is a comment

Initial O/S Installation

1. Make sure system is not connected to a network.
2. Insert "Solaris 8 Installation" CD.
3. Enter "**Stop A**" to go to the boot prompt.
ok boot cdrom
4. Answer "**y**" for initial install.
5. Format disk for clean install.
6. Since this server will only be running "Security" software, 640 MB of swap space should be sufficient.
7. Allow swap to start at beginning of disk.
8. Select "**Networked**"; **No DHCP**;
9. host: **ss-802**

10. IP address - **192.168.10.100**; Netmask - **255.255.255.0**
11. Answer **no** for Ipv6.
12. Check “**none**” for Name Service.
13. Select geographic region and set time if not correct.
14. Follow recommended procedures and select an 8-digit password for root. Use some capital letters, some digits, and lower case characters. Make sure that it is not dictionary searchable.
15. Check no power management.
16. Direct connection to the Internet. (Won’t use a proxy, but will not connect directly to the Internet with this server, will use sftp to transfer files from server that has access to the Internet.
17. Select custom install.
18. No additional languages.
19. No additional software.
20. Check “with 32-bit support only” .
21. Check “Core Solaris Software Group”. This will install the minimal set of packages. We will then add and remove packages to try to include only the packages that are required for the functionality we desire.
22. No documentation, will add man pages package later.

23. Disk Layout:

- Use the selected “c0t0d0” as the disk for file system layout.
- Select “modify” to edit the disk layout.

Slice	File System	Size (MB)	Size (Cyls)
0	/	200	434
1	swap	640	1388
3	/usr	1000	2168
4	/var	1000	2168
5	/opt	1000	2168
6			
7	/export/home	240	521

- /usr needs plenty of space because many of the security applications will be installed in /usr/local/bin.
- /var need lots of space to accommodate log files.
- /opt for other installed applications

Package additions and removal

The “core” installation on this Sun platform installed 51 packages. This is compared to 297 packages that the “end user” installation installed on this server. Before removing unnecessary packages, packages needed for additional functionality should be added. Packages need to be added first, and test that the functions work, before removing packages that may somehow be needed. The overall functionality desired is the CDE environment. This is desired to allow for GUI access for Security Applications, and multi screen capability to allow for more efficient workspace. Note that with CDE functionality available, user still has option to use command line login. With command line login and the stopping of all CDE daemons, the server would be in a more secure mode.

Packages to be added were determined by first installing O/S choosing “end-user” software installation, then selectively deleting packages until it appeared no more packages could be deleted and still have “CDE” functionality. I also used (Pomeranz, 6.5, p 126) as a list of packages to be added to the “core” installation, to help determine which packages were really needed.

1. Use following procedure to add packages:

```
## Insert Solaris 8 Software 1 or 2 into CD.  
# mkdir /cdrom  
# mount -F hsfs -o ro /dev/dsk/c0t2d0s0 /cdrom  
# cd /cdrom/Solaris_8/Product  
# ls    ## shows the packages available  
# pkgadd -d <pkgname>  ## will add package
```

Packages to be added - Solaris 8 Software 1 of 2:

SUNWaudio	Audio applications (required for SUNWolrte)
SUNWtoo	Programming Tools
SUNWdoc	Documentation Tools
SUNWhmdu	SunSwift Sbus Adapter Headers
SUNWtltk	ToolTalk runtime
SUNWlibC	Sun Workshop Compilers Bundled libC
SUNWntpu	NTP, (Root)
SUNWntpu	NTP, (Usr)
SUNWctpls	Portable layout services for Complex Text Layout support
SUNWdtbas	CDE application basic runtime environment
SUNWdtdmn	CDE daemons
SUNWdtdte	Solaris Desktop Login Environment
SUNWdtgezt	Solaris Desktop Extensions Applications
SUNWdthe	CDE Help Runtime
SUNWicn	CDE icons
SUNWdtim	Solaris CDE Image Viewer
SUNWdtlog	System boot for Desktop Login
SUNWdtwm	CDE Desktop Windows Manager

SUNWfns	Federated Naming System
SUNWnamdt	Northern America CDE Support
SUNWocf	Open Card Framework (Required for CDE)
SUNWocfh	Open Card Framework header files
SUNWocfr	Configuration files for Open Card Framework
SUNWocfx	Open Card Framework
SUNWoldte	Open Look Desktop Environment
SUNWowbcp	OpenWindows binary compatibility
SUNWolrte	OPEN LOOK toolkits runtime environment
SUNWscpr	Source Compatibility, (Root)
SUNWxim	X Window System X Input Method Server Package
SUNWxwcf	X Window System common (not required) fonts
SUNWxwopt	nonessential MIT core clients and server extensions
SUNWxwpsr	Sun4u-platform specific X server auxiliary filter modules

Packages to be added - Solaris 8 Software 2 of 2:

SUNWter	Terminal information
SUNWman	On-Line Manual Pages
SUNWaccr	System Accounting, (Root)
SUNWaccu	System Accounting (Usr)
SUNWgzip	The GNU Zip (gzip) compression utility
SUNWspot	Solaris Bundled tools
SUNWhea	SunOS Header Files
SUNWarc	Archive Libraries
SUNWbtool	CCS tools bundled with SunOS
SUNWtnfc	TNF Core Components
SUNWtnfd	TNF Developer Components
SUNWxwrtl	X Window System & Graphics Runtime Library
SUNWxwfmt	X Window platform required fonts
SUNWxwice	ICE components
SUNWxilrl	XIL Runtime Environment
SUNWxildh	XIL Loadable Pipeline Libraries
SUNWxilow	XIL Desktop Loadable Pipeline Libraries
SUNWxwplt	X Window System platform software
SUNWmfrun	Motif RunTime Kit
SUNWlibm	Sun Workshop bundled libm
SUNWscpu	Source Compatibility, (Usr)

Packages added for OpenSSH X tunneling which will be needed when openSSH is installed (Noordergraaf, p 13).

SUNWxcu4 XCU4 Utilities

Here are the very minimal packages needed for Solaris 8 in 32 bit mode as listed in: (Noordergraaf, p 11):

SUNWcar Core Architecture, (Root)

SUNWcsd	Core Solaris Devices
SUNWcsl	Core Solaris, (Shared Libs)
SUNWcsr	Core Solaris, (Root)
SUNWcsu	Core Solaris, (Usr)
SUNWesu	Extended System Utilities
SUNWhmd	SunSwift Sbus Adapter Drivers
SUNWkvm	Core Architecture, (Kvm)
SUNWlibms	Sun WorkShop Bundled shared libm
SUNWloc	Sun Localization
SUNWnamos	Northern America OS Support
SUNWpd	PCI Drivers
SUNWswmt	Install and Patch Utilities

Will use the above information that list the minimal packages needed and info from the SANS Solaris Practicum documentation to remove “unnecessary” packages. Server will be rebooted frequently to make sure server will come up and CDE is still functioning.

1. # **pkginfo > pkgs.core**
2. ## inspect the file pkgs.core and list packages to be removed
3. # **pkgrm <pkgname>**
4. Here is a list of removed packages:

SUNWatfsr	AutoFS, (Root)
SUNWatfsu	AutoFS, (Usr)
SUNWudf	Universal Disk Format 1.50, (Usr)
SUNWudfr	Universal Disk Format 1.50, (Root)
SUNWusb	USB Device Drivers
SUNWadmr	System & Network Administration Root
SUNWaudd	Audio Drivers
SUNWauda	Audio Applications
SUNWdfb	Dumb Frame Buffer Device Drivers
SUNWerego	Solaris User Registration Installation ID file
SUNWftpr	FTP Server, (Root)
SUNWftpu	FTP Server, (Usr)
SUNWluxop	Sun Enterprise Network Array firmware and utilities
SUNWnamox	North American 64-bit OS Support
SUNWnistr	Network Information System, (Root)
SUNWnistru	Network Information System, (Usr)
SUNWpcelx	3COM EtherLink III PCMCIA Ethernet Driver
SUNWpcmci	PCMCIA Card Services, (Root)
SUNWpcmci	PCMCIA Card Services, (Usr)
SUNWpcmci	PCMCIA memory card driver
SUNWpcser	PCMCIA serial card driver
SUNWpsdpr	PCMCIA ATA card driver
SUNWqfed	Sun Quad FastEthernet Adapter Driver
SUNWses	SCSI Enclosure Services Device Driver
SUNWsndmr	Sendmail root

SUNWsndmu	Sendmail user
SUNWsolnm	Solaris Naming Enabler
SUNWtleu	Thai Locale Environment User Files

Patch Installation and Removal

1. Download from www.sunsolve.sun.com/pub-cgi/show.pl?target=patches/patch-access latest Solaris 8 recommended cluster security patches onto PC.
2. Download MD5 checksum.
3. Check MD5 checksum on Laptop before transferring to Sun server.
 ## Had a problem with the MD5 checksums not matching the downloaded checksum values from the source when MD5 was used on the Sun server, but was originally downloaded onto the PC and ftp was used to transfer to the Sun. This problem did not occur when MD5 was used directly on the PC. However this should not be a problem because the transfer of files from the PC to the Sun server is made directly through the hub.
4. Connect PC to hub and assign an IP to PC (I used 192.168.10.101).
5. Ftp the 8_Recommended.zip to Sun server into desired directory (/opt/admin/downloads was used).
 Note: Have to use ftp until sftp is installed.
6. # /usr/bin/unzip 8_Recommended.zip
7. # cd 8_Recommended
8. # ./install_cluster
9. Here are the patch install error codes obtained from:
<http://www.cisco.com/univercd/cc/td/doc/product/access/sc/rel9/relnote/sol8rn.htm>

Exit Codes:

- 0 No error
- 1 Usage error
- 2 Attempt to apply a patch that's already been applied
- 3 Effective UID is not root
- 4 Attempt to save original files failed
- 5 pkgadd failed
- 6 Patch is obsolete
- 7 Invalid package directory
- 8 Attempting to patch a package that is not installed
- 9 Cannot access /usr/sbin/pkgadd (client problem)
- 10 Package validation errors
- 11 Error adding patch to root template
- 12 Patch script terminated due to signal
- 13 Symbolic link included in patch
- 14 NOT USED
- 15 The prepatch script had a return code other than 0.
- 16 The postpatch script had a return code other than 0.

- 17 Mismatch of the -d option between a previous patch install and current one.
 - 18 Not enough space in the file systems that are targets of the patch.
 - 19 \$SOFTINFO/INST_RELEASE file not found
 - 20 A direct instance patch was required but not found
 - 21 The required patches have not been installed on the manager
 - 22 A progressive instance patch was required but not found
 - 23 A restricted patch is already applied to the package
 - 24 An incompatible patch is applied
 - 25 A required patch is not applied
 - 26 The user specified backout data can't be found
 - 27 The relative directory supplied can't be found
 - 28 A pkginfo file is corrupt or missing
 - 29 Bad patch ID format
 - 30 Dryrun failure(s)
 - 31 Path given for -C option is invalid
 - 32 Must be running Solaris 2.6 or greater
 - 33 Bad formatted patch file or patch file not found
 - 34 The appropriate kernel jumbo patch needs to be installed
10. Also check the /var/sadm/patch directory to see the log files for each patch installed and check for any errors
 11. Download patchk.pl.tar.gz and patchdiag.xref from:
<http://sunsolve.sun.com/pub-cgi/show.pl?target=patchk>
 Unzip and untar patchk.pl and place patchdiag.xref into the newly created patchcheck_1.1 directory. Run patchk.pl from this directory and inspect report for any necessary security patches that need to be installed that were not included in the 8_Recommended patch cluster.

Post Installation Configuration

Password/Login Policies

In this section, the password/login policies will be modified to adhere to Minimum Security Requirements for Multi-user Operating Systems (Kuntz, p 396-398)

1. "The system-supplied default minimum length shall be eight characters."
2. "The system shall enforce password aging on a per-userID or per-group basis ... The system-supplied default for those userIDs that might acquire privileges shall be 30 days."
3. "The system-supplied default shall be three times. (Failed login attempts)"
4. "The system shall generate an alarm when this threshold is exceeded."
5. "... advisory warning message to user regarding unauthorized use ... by default: NOTICE: This is a private computer system. Unauthorized access or use is prohibited and may lead to prosecution."

Procedures for above:

1. Edit the /etc/default/passwd: **PASSLENGTH=8**
 ## Sets minimum password length
2. Edit the /etc/default/passwd: **MAXWEEKS=4**
 ## Sets the maximum number of days the password is valid until it must be changed.
3. Edit the /etc/default/login **RETRIES=3**
 ## Sets the number of failed logins before login program exits.
4. Edit the /etc/default/login **SYSLOG_FAILED_LOGINS=3**
 ## Sets the number of failed login attempts before failed login attempt is logged via the syslog daemon.

Note: The above password rules apply only to users changing their passwords, such as the admin user on this system. These rules do not apply to root. For example, even though the /etc/default/passwd file sets the PASSLENGTH variable to 8, root could make its password shorter. Obviously this is not recommended, because longer passwords are more difficult to crack, but just be advised that root can violate these rules if it wants to violate them.

5. Edit /etc/issue, and /etc/motd replace with above notice in # 5.
6. Note that the line: **CONSOLE=/dev/console** ## is not commented out, this restricts root to only login from the console, no remote logins via telnet.
7. Add new user to be used for all logins, and also to be used with “sudo” once it is installed:

```
# useradd -u 100 -g 14 -m -d /export/home/admin -s /bin/ksh admin
```

```
-u sets uid to 100
-g sets gid to 14 “sysadmin”
-d set home directory to /export/home/admin
-m says create the home directory
-s set login shell to /bin/ksh
```

admin is username

```
# passwd admin           ## Give new user a password
```

Note: Banner for ftp and telnet typically give away information as to the OS type and version. They do not have to be changed because openSSH will replace those services. Ftp and telnet will be turned off once openSSH is operational.

Hardening O/S

The steps for the following section are taken directly from SANS Institute Track 6 Securing UNIX 6.5 Linux/Solaris Practicum section Solaris Security: Step-by-Step, pages 106-235.

Since this server will be primarily used on a private LAN in a lab environment, this has been taken into consideration when looking into following the suggestions recommended in the document referenced above. If the server is connected to a LAN with Internet access, adjustments may have to be made.

1. There is no need for DNS services, so the /etc/resolv.conf file will not be used. The /etc/resolv.conf file is used to specify the name of the DNS server to use and also can specify the domain name(s) to use for the search order.
2. The line in /etc/nsswitch.conf file will remain as: **hosts files**
All name resolution will come from the /etc/hosts file.
3. Also, no need for /etc/defaultrouter since all work will be on the LAN. If the server is moved temporarily to a site that requires DNS, server can be reconfigured with the new IP address, default route, and DNS server lookup through /etc/resolv.conf if required.
The /etc/nsswitch.conf file could then be changed so the hosts line reads:
hosts dns files
4. Prepend .NO to the following links in /etc/rc2.d so that they will execute during startup:

Note: In the directories where the startup scripts are located, for example, /etc/rc2.d, all files the begin with S (capital S) will be executed with the “start” parameter, so long as the name is changed to something other than that beginning with “S” it will not be considered as a startup script and will not be executed.

S30sysid.net Not needed unless sys-unconfig is used to reconfigure network information such as IP address, netmask.

For example, S30sysid.net was renamed .NOS30sysid.net

S71sysid.sys	Same as above for system info such as hostname.
S72autoinstall	Used with jumpstart, not needed now.
S76nsd	Provides name service caching, using /etc/hosts so no need.
S73nfs.client	Not using NFS to mount any remote systems.
S73Cachefs.daemon	Used if /usr is a chachefs, NA.
S93cachefs.finish	NA, as above.
S71ldap.client	Not going to be using an LDAP server.
S80PRESERVE	No need to move “editing” files into /usr/preserve dir.

/etc/rc3.d:

S15nfs.server	Not going to be nfs mounting files to share remotely so by disabling this daemon, it will also eliminate another potential vulnerability.
---------------	---

Note: Autofs package has been removed to eliminate any vulnerabilities that may be introduced with this package. Automount is not required for this system.

Also, CDE has a requirement to use rpcbind, so S71rpc was not changed.

5. Following commands were placed in the file /etc/init.d/inetinitA, made it same permissions, owner and group as ./inetinit, then linked /etc/rc2.d/S69initA to it as:

```
# ln -s /etc/init.d/inetinitA /etc/rc2.d/S69inetA
```

Note: Since scripts are executed in “ascii order”, S69inetA will execute after S69inet. This is when the ndd commands should be executed. These commands were placed in a separate file from S69inet so that the original system files remain intact. In case of system upgrade or patch installation, the original files may be modified, and the changes may be compromised if they were included in the original system file.

```
## Increase the following value from a default of 1024 to 8124 to allow  
## for a larger queue, which is basically the “length of the incomplete  
## connection queue” (Voeckler, p 10).
```

```
ndd -set /dev/ip tcp_conn_req_max_q0 8192
```

```
## Lower the cinterval value from default of 180000 (3 min) to 60000 (1 min) to decrease  
## SYN attack vulnerability. This is the amount of time that a connection is allowed to  
## stay in a half opened state. Attackers try to make many network connections and  
## not respond leaving the connections in half opened state on purpose, thereby trying to  
## “flood” the server with these half opened connections.
```

```
ndd -set /dev/ip tcp_ip_abort_cinterval 60000
```

```
## Turn off following to make system less vulnerable regarding ICMP messages.
```

```
ndd -set /dev/ip ip_respond_to_timestamp 0  
ndd -set /dev/ip ip_respond_to_timestamp_broadcast 0  
ndd -set /dev/ip ip_respond_to_address_mask_broadcast 0
```

```
## Lower amount of time arp values stay in arp cache, which can  
## make IP spoofing harder. The arp value is the unique “MAC” address (the NIC cards  
## hardware address as assigned by the NIC vendor), and its corresponding  
## IP address.
```

```
ndd -set /dev/arp arp_cleanup_interval 60000  
ndd -set /dev/ip ip_ire_arp_interval 60000
```


Following five commands relate to routing packets, and they
are not applicable on my private LAN. However, this could be
an issue if system was moved temporarily to another LAN with
gateways and routers that connected to other LANs and to the Internet.

```
nnd -set /dev/ip ip_ignore_redirect 1
nnd -set /dev/ip ip_send_redirects 0
nnd -set /dev/ip ip_forward_src_routed 0
nnd -set /dev/ip ip_forwarding 0
nnd -set /dev/ip ip_strict_dst_multihoming 1
```

Note: Here is a script used to write all of the “nnd -get “ values to a file.
Script was used before and after modifying the values using “nnd” so the
default values could be noted, and it could be confirmed that they were set
correctly.

```
#!/bin/ksh
## Prints out all names and values obtained by “nnd -get”
## for devices /dev/ip, /dev/tcp, /dev/udp, /dev/arp
## Note: need to use the sed command to eliminate the “?”,
## because the “?” is a valid device name that just lists all of the
## other device names. The other sed command gets rid of any
## occurrences of “(“ followed by anything, because the output of the
## “nnd -get /dev/ip?” produces a few instances of names such as:
## “ip_respond_to_address_mask_broadcast(read and write)” which when
## sent to awk to print the first field has the “(read” as part of the name.
```

```
OUTFILE="allVals.out"
cat /dev/null > $OUTFILE
for DEV in ip tcp udp arp
do
  for NAME in `nnd /dev/$DEV \? | awk '{ print $1 }' | sed s/\(.*// | sed s/\?//`
  do
    echo $NAME >> $OUTFILE
    nnd -get /dev/$DEV $NAME >> $OUTFILE
    echo >> $OUTFILE
  done
done
# mv /etc/rc2.d/S72inetsvc /etc/rc2.d/.NoS72inetsvc
## disable inetd once openSSH is installed.
```

7. Add a new version of /etc/init.d/syslog and change command that starts syslog to
add -t option so the syslogd will not listen for logging messages from other systems.
This removes vulnerability of some system purposefully trying to bring down the

server by overwhelming the syslog daemon or by creating huge syslog files.
Here are the steps:

- 1) # cp /etc/init.d/syslog /etc/init.d/newsyslog
- 2) # chmod 744 /etc/init.d/newsyslog
- 3) # chown root:root /etc/init.d/newsyslog
- 4) # edit /etc/init.d/newsyslog:
The line should now read:
/usr/sbin/syslogd -t >/dev/msglog 2>&1 &
- 5) mv /etc/rc2.d/S74syslog /etc/rc2.d/.NOS74syslog
- 6) ln -s /etc/init.d/newsyslog /etc/rc2.d/S74syslog

9. Moved /etc/rcS.d/S50devfsadm to /etc/rcS.d/.NOS50devfsadm to disable support for “hot_swappable” devices and dynamic pty allocation.

10. Removed the following lines in /etc/inittab:

Note: If modem or serial device is to be used at a later date, then these will have to be added back.

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

**co:234:respawn:/usr/lib/saf/ttymon -g -h console login: “ -T sun -d
/dev/console -m ldterm,ttcompat**

Note: Can still login using “CDE” login GUI, and can also login via “command line” by clicking on “Options” → “Command Line Login”.

These lines were removed from the /etc/inittab file so that these services are not started and therefore it takes away potential vulnerabilities to try to exploit.

11. Will use “aide” (which is a freeware version of “tripwire” to monitor files such as /etc/inet/inetd.conf. If a monitored file is modified, an entry is placed in the log file. All services will be turned off once openSSH is installed by moving the /etc/rc2.d/S72inetsvc startup script to ./NOS72inetsvc.
12. Removed /etc/dfs/dfstab file since the NFS functionality has been removed with the removal of the appropriate NFS packages. The dfstab file controls file sharing through NFS by specifying which files are to be shared.
13. Removed the files “adm”, and “lp” in /var/spool/cron/crontabs directory. These files contain the commands that the “cron” daemon will execute on behalf of that user. This makes it more difficult for a hacker to try to use “cron” to execute commands to gain root access or to explore the system. The “adm” and “lp” accounts have /dev/null as their login shell, to keep anyone from using those ids to get a shell, and this is just a way to further restrict any access using these accounts. The only files left in that directory are the “root” file so root will be allowed to run and edit cron jobs and “sys” because “sys” is needed to run “Sar” jobs. Sar is “System Accounting” and will be more thoroughly explained when it is configured. Also removed all files in the /var/spool/cron/atjobs directory. We will use “cron” to execute jobs “at a later time”, so will remove all users files to execute “at” jobs.

14. Here are the results of the `ps -ef` command after all of the above adjustments:

Command line login:

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	0	0	0	19:55:15	?	0:09	sched
root	1	0	0	19:55:15	?	0:00	/etc/init -
root	2	0	0	19:55:15	?	0:00	pageout
root	3	0	0	19:55:15	?	0:17	fsflush
root	149	1	0	19:55:38	?	0:00	/usr/dt/bin/dtlogin -daemon
root	664	149	0	20:46:56	console	0:00	-sh
root	140	1	0	19:55:36	?	0:00	/usr/lib/utmpd
root	107	1	0	19:55:33	?	0:00	/usr/sbin/rpcbind
root	132	1	0	19:55:35	?	0:00	/usr/sbin/cron
root	117	1	0	19:55:34	?	0:00	/usr/sbin/inetd -s
root	131	1	0	19:55:35	?	0:00	/usr/sbin/syslogd -t
root	683	664	0	22:00:18	console	0:00	ps -ef
root	226	117	0	19:56:29	?	0:00	rpc.ttdbserverd

So, other than those processes required for CDE, and the extra process that will be removed once OpenSSH is installed, there are only 9 processes, the rpc processes are required for CDE to run.

Here are the extra processes when logging in through CDE:

UID	PID	PPID	C	STIME	TTY	TIME	CMD
root	149	1	0	19:55:38	?	0:00	/usr/dt/bin/dtlogin -daemon
root	527	149	1	20:39:59	?	0:03	/usr/openwin/bin/Xsun :0 - nobanner -auth /var/dt/A:0 6IaOsa
root	585	546	0	20:40:12	pts/3	0:00	/usr/dt/bin/sdt_shell -c unset DT; DISPLAY=:0; /usr/dt/bin/dt
root	600	1	0	20:40:13	pts/3	0:00	/usr/dt/bin/ttsession
root	546	528	0	20:40:11	?	0:00	/bin/ksh /usr/dt/bin/Xsession
root	587	585	0	20:40:12	pts/3	0:00	-sh -c unset DT; DISPLAY=:0; /usr/dt/bin/dtsession_res -merge
root	592	1	0	20:40:12	?	0:00	/usr/dt/bin/dsdm
root	607	601	0	20:40:14	?	0:02	dtwm
root	528	149	0	20:39:59	?	0:00	/usr/dt/bin/dtlogin -daemon
root	601	587	0	20:40:13	pts/3	0:00	/usr/dt/bin/dtsession
root	608	601	0	20:40:16	??	0:01	/usr/dt/bin/dtterm -session dtq7aO7a

Eleven extra processes for CDE.

15. The Sendmail package is not installed on this server since it is not absolutely necessary. There are many vulnerabilities involving Sendmail, and it is listed as one of the “Top Vulnerabilities To UNIX Systems” (SANS Institute resources, p 20). If Sendmail is later required, could look into documentation on page 146 of the SANS Step-by-Step Documentation regarding configuring Sendmail to run as a cron job instead of as a daemon.
16. Edited /etc/vfstab to turn on **logging** for /, **nosuid** for /var, /opt/, and /export/home, and **ro** for /usr.

Logging for / makes the root filesystem more stable. This prevents the root filesystem from becoming corrupt when a hacker repeatedly crashes the system trying to corrupt the root filesystem so that a root shell can be obtained. A transaction log is used to maintain file system integrity. If the system crashes, the transaction log is “replayed” to keep the filesystem from becoming corrupt.

Nosuid for /var, /opt, and /export/home means the OS will ignore suid bit for all files in those filesystems. Suid files are often targets of hackers trying to gain root access. By removing suid functionality on these three filesystem, much of the suid vulnerabilities have been mitigated.

Ro for /usr prevents file system modification for all files in /usr. This is where many of the configuration files are, so eliminating the possibility of their modification as long as /usr stays mounted “ro” makes the system more secure.

Because of these attempts to try to restrict access to these filesystems, we need to make sure that the system is not rebooted with modified mount options. “Aide” will be able to produce a report of any files modified such as /etc/vfstab to change the mount options, so it is important to regularly monitor the “aide” log file.

Can also monitor “uptime” to see when system was last rebooted; check /var/adm/messages for reboots, issue command “last” to see logins; check /var/log/authlog for logins; check files in the /var/audit directory for latest file to check for reboots, and check /var/adm/sulog.

A hacker may be able to place a “rootkit” on the system, but the more redundancy there is in being able to monitor logins, reboots, and authorizations, the harder it is for them to try to insert their version of all of these programs to try to conceal their modification of the system.

Here is the /etc/vfstab as modified:

##device ##to mount ##	device to fsck	mount point	FS type	fsck pass	mount at boot	mount options
###/dev/dsk/c1d0s2	/dev/rdisk/c1d0s2	/usr	ufs	1	yes	-
fd	-	/dev/fd	fd	-	no	-
/proc	-	/proc	proc	-	no	-
/dev/dsk/c0t0d0s1	-	-	swap	-	no	-
/dev/dsk/c0t0d0s0	/dev/rdisk/c0t0d0s0	/	ufs	1	no	logging
/dev/dsk/c0t0d0s3	/dev/rdisk/c0t0d0s3	/usr	ufs	1	no	ro
/dev/dsk/c0t0d0s4	/dev/rdisk/c0t0d0s4	/var	ufs	1	no	nosuid
/dev/dsk/c0t0d0s7	/dev/rdisk/c0t0d0s7	/export/home	ufs	2	yes	nosuid
/dev/dsk/c0t0d0s5	/dev/rdisk/c0t0d0s5	/opt	ufs	2	yes	nosuid
swap	-	/tmp	tmpfs	-	yes	-

Note:

Since /usr will be “ro”, will link /usr/local/var to /opt/local/var, that way Nessus can write its log and other files to /opt/local/var which is “nosuid” but “rw”.

```
# mkdir /opt/local/var
# cp -r /usr/local/var/* /opt/local/var
# rm -fr /usr/local/var
# ln -s /opt/local/var /usr/local/var
```

18. Install tcp_wrappers

- 1) Unzip and use pkgadd to install as previously described
 - 2) modify inetd.conf
 - 3) Here are the lines in the /etc/inetd.conf file, (shows the in.ftpd and in.telnetd changes)
 - 4) **##ftp stream tcp6 nowait root /usr/sbin/in.ftpdin.ftpd**
 - 5) **ftp stream tcp nowait root /usr/local/bin/tcpd in.ftpd**
 - 6) **##telnet stream tcp6 nowait root /usr/sbin/in.telnetd in.telnetd**
 - 7) **telnet stream tcp nowait root /usr/local/bin/tcpd in.telnetd**
- Note: Make sure to change “tcp6” to “tcp”, otherwise you may see an entry in /var/log/syslog regarding “refused connection from 0.0.0.0”.
- 8) Here are the hosts.allow and hosts.deny files for access control:

hosts.allow: in.ftpd, in.telnetd, nessusd, sshd:
192.168.10.101, localhost

IP address of my PC for access through a hub to the Sun server
on the private LAN

hosts.deny: ALL: ALL:

- 9) Also, “PuTTY” was used for ssh connection from my W2000 Laptop to the Sun server. This includes ssh connection for secure telnet type session and

sftp for secure ftp access. PuTTY is a free implementation of SSH for Win32 platforms. Download PuTTY from:

<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>

19. OpenSSH installation procedures:

- 1) Download the following packages and install them in order:
 - prngd-0.9.23-sol8-sparc-local
 - zlib-1.1.4-sol8-sparc-local
 - libgcc-3.0.3-sol8-sparc-local
 - openssl-0.9.6c-sol8-sparc-local
 - openssh-3.0.2p1-sol8-sparc-local
- 2) Note: Make sure to download zlib-1.1.4. Earlier versions are vulnerable to denial of service attacks as documented in Cert Advisory CA-2002-07.
- 3) Feed the prngd-seed file with some log file data:
cat /var/log/* > /usr/local/etc/prngd/prngd-seed
- 4) Start the prngd daemon
/usr/local/bin/prngd /var/spool/prngd/pool
- 5) Generate public and private keys (Only generate for Version 2, which is a bit more secure).

/usr/local/bin/ssh-keygen -t rsa -f /usr/local/etc/ssh_host_rsa_key -N ""

/usr/local/bin/ssh-keygen -t dsa -f /usr/local/etc/ssh_host_dsa_key -N ""

- 6) Start the sshd
/usr/local/sbin/sshd

- 7) PuTTY was used for “ssh” and “sftp” to confirm it works

Here is the message that is displayed for 1st login to a server:

psftp> open 192.168.10.100

login as: **admin**

The server's host key is not cached in the registry. You have no guarantee that the server is the computer you think it is.

The server's key fingerprint is:

ssh-rsa 1024 44:a4:38:a3:25:02:1e:5a:d6:f8:42:bf:e1:ae:62:e0

If you trust this host, enter “y” to add the key to

PuTTY's cache and carry on connecting.

If you want to carry on connecting just once, without adding the key to the cache, enter “n”.

If you do not trust this host, press Return to abandon the connection.

Store key in cache? (y/n) **y**

You can compare the fingerprint above with the fingerprint that was created when openssh was first configured on the server.

- 8) Here are the steps to create start up scripts for prngd and sshd:
- First I created a “TEMPLATE” file in /etc/init.d
- Note: This TEMPLATE is a modified version of the scripts used in (Christensen, p 5) for the prngd startup script.

```
#!/sbin/sh
case "$1" in
'start')
if [ /usr/bin/pgrep -f xxx ]; then
    echo "$0: xxx is already running"
    exit 0
fi
if [ -x xxx ]; then
    xxx &
fi
;;
'stop')
if [ /usr/bin/pgrep xxx ]; then
    /usr/bin/pkill -f xxx
fi
;;
*)
echo "Usage: $0 { start | stop }"
exit 1
;;
esac
exit 0
```

- Then the following command was issued in vi in the TEMPLATE file:

```
:%s/xxx/usr/local/bin/prngd/g
```

```
## this will change all occurrences of xxx to
/usr/local/bin/prngd
```

- :w prngd**
- Do the same for sshd
- # chown root:root /etc/init.d/prngd**
- # chown root:root /etc/init.d/sshd**
- # chmod 744 /etc/init.d/prngd**
- # chmod 744 /etc/init.d/sshd**
- This is a quick way to create the files and still can reuse the “TEMPLATE”
- Make links in /etc/rc2.d to the new files:
ln -s /etc/init.d/prngd /etc/rc2.d/S90prngd
ln -s /etc/init.d/sshd /etc/rc2.d/S95sshd

- ix. Gave `prngd` a lower “S” number so it will execute before `sshd`: `S90prng`, `S95sshd`
- x. Reboot and check to see the `prngd` and `sshd` processes are running correctly
- xi. Test the “stop” option to see that the script can also kill the process
- xii. Modified following lines in the `/usr/local/etc/sshd_config` file to issue the warning message for `ssh` login, and also to deny root login from `ssh`:

PermitRootLogin no
Banner /etc/issue

- xiii. Restart `sshd` and test that the “banner” message shows up when accessing `ssh`, and that root is not permitted to login via `ssh` or `sftp`.

20. Deleted unwanted users (`uucp`, `nuucp`, `smtp`, `listen`, `nobody4`).

```
# userdel uucp
# userdel nuucp
# userdel smtp
# userdel listen
# userdel nobody4
```

Note: These users have been deleted in keeping with the theme that “less is more”, and that all services and functions not necessary are removed to eliminate any potential for exploitation by persons trying to gain access to the system.

21. Set shell to “`/dev/null`” for others (`daemon`, `bin`, `lp`, `nobody`, `noaccess`). Again, we want to lower the risk that these accounts can be used to gain access, but since these accounts are used by the system, by setting their login shell to `/dev/null`, it eliminates the possibility of someone logging in using these ids.

Here is the /etc/passwd file with the modifications:

```
# cat /etc/passwd
root:x:0:1:Super-User:/:/sbin/sh
daemon:x:1:1:/:/dev/null
bin:x:2:2:/:usr/bin:/dev/null
sys:x:3:3:/:/sbin/sh
adm:x:4:4:Admin:/var/adm:/dev/null
lp:x:71:8:Line Printer Admin:/usr/spool/lp:/dev/null
nobody:x:60001:60001:Nobody:/:/dev/null
noaccess:x:60002:60002:No Access User:/:/dev/null
admin:x:100:14:/:export/home/admin:/bin/ksh
```

22. Verified root and all other “system” users are in /etc/ftpusers file, chmod to 600, and changed group to root.
All ids in the /etc/ftpusers file are NOT allowed to use ftp. We want to make sure Only our “admin” account can ftp , that makes it harder to ftp into privileged filesystems such as “/”.
23. Modified /etc/pam.conf by removing rhosts_auth entries so that the system ignores .rhosts style authentication for rlogin, rsh rcp etc. services.
This is important because if an /.rhosts entry exists on say serverA and has, and has serverB’s IP address in it, then if a user has root access to serverB and can get to serverA via for example “rsh”, then they do not need to know the password for serverA.
24. Created empty files for /.rhosts, /.shosts, /.netrc, /etc/hosts.equiv and made their permissions 000. Makes it more difficult to add entries to .rhosts file.
25. Added “**auth.info** /var/log/authlog” to /etc/syslog file. Authorizations will be logged to the file /var/log/authlog such as login attempts, su’s, etc.
26. Confirmed that “ssh” successful and unsuccessful logins are being logged to /var/log/authlog.
27. Added /var/adm/loginlog file to monitor login attempts. More redundancy we have in monitoring authorization the better. Just makes it harder for someone to try to cover their tracks if they get in.
28. Confirmed that repeated login failures are logging to /var/adm/loginlog
29. Here is the procedure used to rotate the log files:
 - 1) Removed the directory /var/adm/log ## nothing in it.
 - 2) Modified script so that instead of doing the rotating on one file, it is done in a loop on all “log” files in the list as set in the script, and it also compresses the file so it takes up less space.

3) Here is the modified script:

```
#!/bin/sh
cd /var/adm
for LOG in aculog lastlog loginlog sulog messages
do
    test -f $LOG.2.gz && mv $LOG.2.gz $LOG.3.gz
    test -f $LOG.1.gz && mv $LOG.1.gz $LOG.2.gz
    test -f $LOG.0.gz && mv $LOG.0.gz $LOG.1.gz
    mv $LOG $LOG.0
    /usr/bin/gzip $LOG.0
    cp /dev/null $LOG
    chmod 600 $LOG
done
cd /var/cron
for LOG in log
do
    test -f $LOG.2.gz && mv $LOG.2.gz $LOG.3.gz
    test -f $LOG.1.gz && mv $LOG.1.gz $LOG.2.gz
    test -f $LOG.0.gz && mv $LOG.0.gz $LOG.1.gz
    mv $LOG $LOG.0
    /usr/bin/gzip $LOG.0
    cp /dev/null $LOG
    chmod 600 $LOG
done
LOGDIR=/var/log
cd $LOGDIR
for LOG in authlog sysidconfig.log syslog
do
    if test -s $LOG
    then
        test -f $LOG.6.gz && mv $LOG.6.gz $LOG.7.gz
        test -f $LOG.5.gz && mv $LOG.5.gz $LOG.6.gz
        test -f $LOG.4.gz && mv $LOG.4.gz $LOG.5.gz
        test -f $LOG.3.gz && mv $LOG.3.gz $LOG.4.gz
        test -f $LOG.2.gz && mv $LOG.2.gz $LOG.3.gz
        test -f $LOG.1.gz && mv $LOG.1.gz $LOG.2.gz
        test -f $LOG.0.gz && mv $LOG.0.gz $LOG.1.gz
        mv $LOG $LOG.0
        /usr/bin/gzip $LOG.0
        cp /dev/null $LOG
        chmod 600 $LOG
        sleep 40
    fi
done
##
kill -HUP `cat /etc/syslog.pid`
```

30. Cleaned out crontab entries for root by removing following lines:

```
15 3 * * 0 /usr/lib/fs/nfs/nfsfind
## NFS is not being used on this server
30 3 * * * [-x /usr/lib/gss/gsscred_clean ] && /usr/lib/gss/gsscred_clean
## This script was not installed.
```

31. Enable system accounting:

- 1) Added following lines to sys crontab file
“/var/spool/cron/crontabs/sys” to begin collecting Sar data:

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

32. Enabled kernel auditing by running /etc/security/bsmconv; then rebooted.

- 1) Can now see the /usr/sbin/auditd running.
- 2) Edited the /etc/security/audit_control file:

```
dir:/var/audit
flags:lo,ad,nt
minfree:20
naflags:lo,ad,nt
```

Note: lo flag - logins, logouts
ad flag - administrative actions, mount etc.
nt flag - network events, connect, etc.
minfree - sets at 20% when system will warn if disk reaches that threshold for free disk space of filesystem containing current audit file
naflags - monitor even if action can not be bound to a specific user

Above from man page on audit_control.

Decided against using flags -all, as it tended to fill up the audit file too quickly.

- 3) Rebooted, made unsuccessful logins.
- 4) # /usr/sbin/praudit -l /var/audit/*'hostname' ## to view audit entries, could see reboot and logins.
- 5) Use the following command to see the praudit output in a more easily readable form:

```
# praudit -s <auditfile> | egrep “^header|^return” | more
```

This just prints out the “header” and “return” lines and you can see a little easier what event and what return values are there.

Note: Stop-A has been disabled when auditing was enabled. This is done by adding the following line to the /etc/system file:

set abort enable = 0

If you want to run audit, but not disable “Stop-A”, then you can remove this entry or comment it out (use * for comment character) and reboot.

33. telnet and ftp are not being run on this server. Set the /etc/default/telnetd and /etc/default/ftpd files to “” anyway. This way if either service is added, no banner message will be displayed when accessing server using telnet or ftp and giving away information regarding the OS type and version.

Edit /etc/default/inetinit: **TCP_STRONG_ISS=2**

(2) above sets RFC 1948 sequence number generation, unique-per-connection as documented in comments in the script.

/etc/default/cron **CRONLOG=YES** was already set by default.

34. /etc/default/su PATH and SUPATH have been left as default, same as the /etc/default/login.
35. Leave the /etc/default/init CMASK value at 022. This is basically a one-user system, and there is not much of a vulnerability of users being able to read files. There should be no users without root access on this system so /etc/profile and /etc/skel won't be used. The /etc/profile file can be used to set up default shell variable values, and /etc/skel is a directory that contains files that are used to create new users .login, .cshrc, and .profile files depending on the shell they will be using.
36. As mentioned previously, /etc/default/login file sets /dev/console so root cannot remotely login to the system. However, all remote login services have been disabled except for ssh and sftp. These services also prohibit root login through the /usr/local/etc/sshd_config **PermitRootLogin no**
37. Will leave eeprom **security-mode=none** because system is in locked environment; would consider changing to full, if system were to be moved temporarily to another location to do a security assessment.
38. Set eeprom **oem-banner="Authorized users only. All access may be logged and reported."**

Note: Hides system network info that could be useful for break in attempt.

39. Modified /etc/system:

```
## Some protection for buffer overrun.  
set noexec_user_stack = 1  
set noexec_user_stack_log = 1  
## No core dumps. It is possible to use "strings" command to look for passwords  
and ids in coredumps.  
set sys:coredumpsize = 0  
## NFS use privileged ports for NFS client requests.  
set nfssrv:nfs_portmon = 1
```

Note: Since this is a single user system, won't worry about limiting user processes, or file descriptors.

This concludes the step-by-step procedures following the SANS Institute Track 6 Securing UNIX 6.5 Linux/Solaris Practicum section Solaris Security: Step-by-Step, pages 106-200.

Before finishing the final recommended procedures on pages 200-233 such as running fix-modes, testing and verification, log file checking, backups, automated tools, and monitoring for security (tripwire, watcher); the security software needs to be loaded.

This way the fix-modes, log files, aide, and all will also work on the Security software. After that is accomplished, will finish off with the last part of the Linux/Solaris Solaris Security: Step-by-Step, pages 200-235.

Installation of security software

1. Sudo allows users to execute specified commands as root and logs their use. Use the following procedures for installation of sudo:

- 1) Download from <http://www.sunfreeware.com>:

- `sudo-1.6.3p5-sol8-sparc-local.gz`

- 2) Check MD5 checksum on Laptop before transferring to Sun Server.
- 3) Use sftp to transfer files to Sun server.
- 4) Unzip and use pkgadd to install.
- 5) # **visudo** ## command to make changes to sudo.

User Alias **ADMINS = admin**

Defaults **syslog=auth** ## sudo events log to /var/log/authlog.

```
Defaults:ADMINS    !lecture    ## no lecture message for bad PW.
```

```
ADMIN ALL=(ALL) ALL    ## admin get to issue all commands
```

Note: This is a one-user system. If other users were added, they would be added to the list of ADMINS (only “root” users on this system) so that they would then get the ability to use “sudo”.

All remote logins must be made using the “admin” account. Use sudo to preface any command that requires root authority. This way all privileged commands are logged in the /var/log/authlog file via syslogd. For example, as admin:

```
$ sudo cat /etc/shadow
```

This allow you cat run the “cat” command as root, and an entry is placed in the /var/log/authlog file.

Can use: `sudo su -` to become super-user, but this will defeat the logging. By policy, local logins on the console should also be made using the non-root account. This is good administrative policy, and if another non-root user were added to the system, the mechanisms would be in place to track “privileged commands”. Also, since `/usr` is mounted “ro”, visudo will not work. This is because the “sudo” configuration file is by default located in `/usr/local/etc/sudoers`. If any changes need to be made to the sudoers file, system will have to be rebooted with `/usr` “rw” to make the changes. Just edit the `/etc/vfstab` file, replace “ro” with “-” in the line that controls the `/usr` mount, and reboot. When finished changing “sudoers” file, change `/etc/vfstab` file back and reboot so `/usr` is again “read only”. It may have been possible to move the `/usr/local/etc/sudoers` file to the `/opt` filesystem, but the sudoers files should not need to be changed anyway, unless a new user were added to the system, so this is a good security measure.

2. Nmap is a port scanner that can be used to determine which services a system is offering. Use the following procedures for installation of Nmap:
- 1) Download from <http://www.sunfreeware.com>:
 - nmap-2.54BETA28-sol8-sparc-local.gz
 - libpcap-0.6.2-sol8-sparc-local.gz
 - 2) Check MD5 checksum on Laptop before transferring to Sun Server.
 - 3) Use sftp to transfer files to Sun server.
 - 4) Unzip and use pkgadd to install.
3. IP Filter allows filtering of incoming and outgoing IP packets based on protocol, port, source and destination address. Use the following procedures for installation of IP Filter:

- 1) Download from:
<ftp://coombs.anu.edu.au/pub/net/ip-filter/ip-fil3.4.27.tar.gz>
- 2) Use sftp to transfer files to Sun server.
- 3) Unzip and untar
- 4) `# cd ./ip_fil3.4.27`
- 5) `# /usr/ccs/bin/make solaris ## Don't use Gnu Make`
- 6) `# cd ./SunOS5`
- 7) `# /usr/ccs/bin/make package ## makes and installs package`
- 8) `## Edited the /etc/rc2.d/S65ipfboot startup script to log to a file rather than to syslog to keep the IP Filter log in a separate file:`

Was: `ipmon -Ds ## Starts as a daemon and logs to /var/log/syslog`
Now: `ipmon -D /var/log/ipflog ## Starts as a daemon but logs to ipflog`

- 9) As part of the installation a script is included called "mkfilters" located by default in the /opt/ipf/bin directory. Issued the following command to start the initial filter configuration file:

```
# /opt/ipf/bin/mkfilters > /etc/opt/ipf/ipf.conf ## This is where ipf expects
to find the configuration file by default and by default the ipf.conf file is
initially empty.
```

Here is what was put into the ipf.conf file:

```
# The following routes should be configured, if not already:
# route add 192.168.10.100 localhost 0
#
block in log quick from any to any with ipopts
block in log quick proto tcp from any to any with short
pass out on hme0 all head 150
block out from 127.0.0.0/8 to any group 150
block out from any to 192.168.10.100/32 group 150
pass in on hme0 all head 100
block in from 127.0.0.0/8 to any group 100
block in from 192.168.10.100/32 to any group 100
```

Here is a brief summary of the way IPF handles the “rules” file:

- reads file from top to bottom.
- the last rule that matches the packet takes precedence.
- if “quick” is used then rule is applied immediately and no further examination of the rules for that packet is done.
- basic action is to “pass” or “block” a packet.
- Can also block all packets with “IP Options” set, which is considered a security threat and also can filter on “fragments” and “shorts”. See:
<http://coombs.anu.edu.au/~avalon/examples.html#ipopts>
for an explanation.

See the following URL for an in depth discussion of the IP Filter rules:

<http://www.obfuscation.org/ipf/ipf-howto.html>

Based on above recommendation from “mkfilters”, place the command:

route add 192.168.10.100 localhost 0

at the beginning of the /etc/init.d/inetinitA. This keeps the system file /etc/init.d/inetinit unmodified and protects against any added patches or updates overwriting these modifications. /etc/init.d/inetinitA has all of the “ndd” settings for system tuning and those commands were placed there for the same reason.

- 10) We can use the /etc/opt/ipf/ipf.conf file to control which packets are allowed into or out of the Sun server on its interface “hme0”.

11) Use the following command to have ipf “reread” its configuration file:

```
# /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf
```

12) Use the following command to see which “rules” are currently valid:

```
# /sbin/ipfstat -io ## Shows which rules are currently in effect for  
inbound and outbound packets.
```

13) Here is an example of a rule to block the X Windows ports 6000 to 6255 outbound from any address to any address:

```
block out log proto tcp from any to any port 5999 >< 6256
```

4. Snort is a real time intrusion detection system capable of performing traffic analysis and packet logging. Use the following procedures for installation of Snort:

1) Download from <http://www.sunfreeware.com>:

- snort-1.8.1-sol8-sparc-local.gz

2) Check MD5 checksum before transferring to Sun Server.

3) Use sftp to transfer files to Sun server.

4) Unzip and use pkgadd to install.

5. Dsniff is a software sniffer that looks for passwords in network packets. Use the following procedures for installation of Dsniff:

1) Download from <http://www.sunfreeware.com>:

- dsniff-2.3-sol8-sparc-local.gz
- glib-1.2.10-sol8-sparc-local.gz
- libnids-1.16-sol8-sparc-local.gz
- libnids-1.16-sol8-sparc-local.gz

2) Check MD5 checksum on Laptop before transferring to Sun Server.

3) Use sftp to transfer files to Sun server.

4) Unzip and use pkgadd to install.

6. Nessus is a security scanner made up of a client and server. It is used to test for vulnerabilities in a system, and to optionally attack those vulnerabilities. Use the following procedures for installation of Nessus:

- 1) Download the following from: <http://www.sunfreeware.com/>:
 - gdk+-1.2.10-sol8-sparc-local.gz
 - glib-1.2.10.-sol8-sparc-local..gz
- 2) Download following from <ftp://nessus.stanford.edu/nessus-1.2.0/src>:
 - libnasl-1.2.0.tar.gz
 - nessus-core-1.2.0.tar.gz
 - nessus-plugins-1.2.0.tar.gz
 - nessus-libraries-1.2.0.tar.gz
 - MD5 ## checksums
- 3) Check MD5 checksum on Laptop before transferring to Sun Server.
- 4) Use sftp to ftp files to Sun Server.
- 5) Unzip and untar files into appropriate directory.
- 6) Use pkgadd to install gdk+-1.2.10-sol8-sparc-local.gz.
- 7) Use pkgadd to install glib-1.2.10-sol8-sparc-local.gz.
- 8) **# cd nessus-libraries**
./configure
make
make install
- 9) **# cd ../libnasl** ## follow steps same as nessus-libraries.
- 10) same as above for nessus-core and nessus-plugins.
- 11) Made sure /usr/local/lib was in the LD_LIBRARY_PATH (Is in /.profile).
- 12) **# cd ../nessus-core**
- 13) **# ./nessus-adduser**
Login: <nessus_id> ## This is separate from regular UNIX account, and will be used for client login to Nessus server.
- 14) Authentication (pass/cert) [pass]: **pass** ## Use password authentication.
- 15) Login password: *********
- 16) **^D** ## Empty rules set for now.
- 17) ## Prints out your selections and asks for confirmation.
- 18) Inspect /usr/local/etc/nessus/nessusd.conf, made following changes:

Lines below with “##” are commented out, line below replaces it.

```
## logfile = /usr/local/var/nessus/logs/nessusd.messages ## /usr is “ro”  
logfile = /var/log/nessusd.messages  
## dumpfile = /usr/local/var/nessus/logs/nessusd.dump ## /usr is “ro”  
dumpfile = /var/log/nessusd.dump  
## checks_reads_timeout = 15  
check_reads_timeout = 5 ## gives a little better performance
```

19) # **nessus-mkcert**

Generates the following interaction:

Script started on Tue 23 Apr 2002 11:33:13 AM PDT

nessus-mkcert

/usr/local/var/nessus/CA created

/usr/local/com/nessus/CA created

Creation of the Nessus SSL Certificate

This script will now ask you the relevant information to create the SSL certificate of Nessus. Note that this information will **NOT** be sent to anybody (everything stays local), but anyone with the ability to connect to your Nessus daemon will be able to retrieve this information.

CA certificate life time in days [1460]:

Server certificate life time in days [365]:

Your country (two letter code) [FR]: **US**

Your state or province name [none]: **CA**

Your location (e.g. town) [Paris]: **Sacramento**

Your organization [Nessus Users United]:

Creation of the Nessus SSL Certificate

Congratulations. Your server certificate was properly created.

/usr/local/etc/nessus/nessusd.conf updated The following files were created:

. Certification authority:

Certificate = /usr/local/com/nessus/CA/cacert.pem

Private key = /usr/local/var/nessus/CA/cakey.pem

. Nessus Server :

Certificate = /usr/local/com/nessus/CA/servercert.pem

Private key = /usr/local/var/nessus/CA/serverkey.pem

Press [ENTER] to exit

script done on Tue 23 Apr 2002 11:37:33 AM PDT

Here are the additions added to the /usr/local/etc/nessus/nessusd.conf file:

Added by nessus-mkcert

##

cert_file=/usr/local/com/nessus/CA/servercert.pem

key_file=/usr/local/var/nessus/CA/serverkey.pem

ca_file=/usr/local/com/nessus/CA/cacert.pem

If you decide to protect your private key with a password,

uncomment and change next line

pem_password=password

If you want to force the use of a client certificate, uncomment next line

force_pubkey_auth = yes

Since /usr will be mounted “read only”, files in the /usr/local/var/nessus will have to move to a filesystem that allow writing. Here is the procedure to move these files to /opt:

If /usr/ is currently mounted “ro”, then replace “ro” with “-” on the line
that contains the /usr/ mount options and reboot:

/dev/dsk/c0t0d0s3 /dev/rdsk/c0t0d0s3 / ufs 1 no ro

Change to:

/dev/dsk/c0t0d0s3 /dev/rdsk/c0t0d0s3 / ufs 1 no -

```
# cd /usr/local/var/nessus
# tar cvf nessus.tar *
# mkdir /opt/local
# mkdir /opt/local/var
# mkdir /opt/local/var/nessus
# chmod 700 /opt/local/var/nessus
# mv /usr/local/var/nessus /opt/local/var/nessus
# cd /opt/local/var/nessus
# tar xvf nessus.tar
# mv /usr/local/var/nessus /usr/local/var/nessus.old ## for backup
# ln -s /opt/local/var/ /usr/local/var/
```

Once it is working, can remove nessus.old

rm -fr /usr/local/var/nessus.old

Once finished with modifying /usr, can put back the “ro” for
/usr/ and reboot system so /usr is back to “read only”.

nessud -D ## Start the Nessus server in Daemon mode

Will not use a startup script to start nessud. This way nessud will
only be running when we want to run a report. Once report is finished,
can stop the nessud:

```
# ps -ef | grep nessud
# kill <pid of the nessud>
```

Here is the command to start the nessus client from a
command line session (make sure nessud is running):

nessus -q 192.168.10.100 1241 <id> <pw> <targetFile> <resultsFile>

1241 is the default port Nessus uses.

targetFile is a file that lists target hosts (192.168.10.100) in this case.

resultsFile is a file to place the Nessus report.

7. Ethereal is a GUI network protocol analyzer. Use the following procedures for installation of Ethereal:

1) Download from <http://www.sunfreeware.com>:

- ethereal-0.9.0-sol8-sparc-local.gz

2) Check MD5 checksum before transferring to Sun Server.

3) Use sftp to transfer files to Sun server.

4) Unzip and use pkgadd to install.

8. This version of PGP is a command line version that allows a user to encrypt/decrypt data. Use the following procedures for installation of PGP:

1) Download from:

<http://www.pgpi.org/products/pgp/versions/freeware/unix/6.5.8>

- PGPCmdln_6.5.6.SolPkg_FW.tar.gz

2) Use sftp to transfer files to Sun server.

3) Unzip and use pkgadd to install.

4) First step is to generate private/public key pair:

Set the PGPPATH variable to tell PGP where to place its files.

pgp -kg # starts key generation

Chose RSA for public-key algorithm

Chose 1024 bits for key size

Entered an id and pass phrase

Entered some random keystrokes for the key
generation process to use.

Can now see the following files:

pubring.pkr and secring.skr

5) To encrypt a file type: **pgp -e <filename> <id>**

This will create a new file called <filename>.pgp which is the
encrypted version of the original file

6) To decrypt the file: **pgp <filename.pgp>**

System will prompt for your passphrase, then decrypt the file and

- name it <filename> without the pgp extension.
- 7) Use the following procedures to prepare a floppy for use:
 - # **fdformat** ## formats the floppy
 - # **newfs /dev/rdiskette** ## puts ufs filesystem on floppy
 - # **mount /dev/diskette /floppy**
 - 8) Encrypt Nessus report as an example “nessusNBE.rpt:
 - # **pgp -e nessusNBE.rpt** <id>
 - 9) Move report to floppy: **mv nessusNBE.rpt.pgp /floppy**
 - 10) Remove floppy disk and place in secure location
 - 11) If reports are large, may want to compress reports before encrypting them.
 - 12) Can also move reports to tape.
9. Logcheck checks specified logs for keywords and alerts on a regular basis via a cron job. Use the following procedures for installation of Logcheck:
- 1) Download from <http://www.sunfreeware.com>:
 - logcheck-1.1.1-sol8-sparc-local.gz
 - 2) Check MD5 checksum before transferring to Sun Server.
 - 3) Use sftp to transfer files to Sun server.
 - 4) Logcheck will want to place some files in /usr/local/var.
 - If /usr is currently mounted “ro” have to change the /etc/vfstab file, take out the “ro” and replace with “-“ for /usr and reboot. Then, put /usr back as “ro” when finished.
 - 5) Unzip and use pkgadd to install.
 - 6) Moved all logcheck files from /usr/local/etc to /opt/local/etc
 - # **mv /usr/local/etc/logcheck* /opt/local/etc**
 - # **mkdir /opt/local/etc/tmp**
 - # **chmod 700 /opt/local/etc/tmp**
 - # **rmdir /usr/local/etc/tmp**
 - 7) Edited logcheck.sh and changed /usr/local to /opt/local for the ./tmp directory and all of the logcheck config files:

TMPDIR=/opt/local/etc/tmp

HACKING_FILE=/opt/local/etc/logcheck.hacking

VIOLATIONS_FILE=/opt/local/etc/logcheck.violations

VIOLATIONS_IGNORE_FILE=/opt/local/etc/logcheck.violations.ignore

IGNORE_FILE=/opt/local/etc/logcheck.ignore

- 8) Check that syslogd is running ## Remember it is running with the -t

option
Check /etc/syslog.conf ## Ok, it is logging to /var/log/messages and
 /var/log/authlog

- 9) Inspected logcheck.hacking, logcheck.violations, logcheck.ignore and logcheck.violations.ignore and left defaults for now. These files contain the keywords that the program uses as it inspects the log files to determine if an alert should be generated.
- 10) Added authlog and loginlog to list of log files logcheck monitors:

```
$LOGTAIL /var/log/authlog > $TMPDIR/check.$$
```

```
$LOGTAIL /var/adm/loginlog >> $TMPDIR/check.$$
```

The logtail program just checks the log files from where it last left off.

“\$\$” is the pid of the current shell process, used just to get an unique filename for this session.

- 11) Since we are not running mail on this system, will modify the logcheck.sh script to write the info to a file instead of using mail. Mail would be preferable, but for now the Sendmail package is not installed so we will rely on inspecting the log files manually. Here are the changes I made to the script (commented out the line and put replacement below):

```
if [ "$ATTACK" -eq 1 ]; then
##      cat $TMPDIR/checkreport.$$ | $MAIL -s "$HOSTNAME $DATE
ACTIVE SYSTEM ATTACK!" $SYSADMIN
      cat $TMPDIR/checkreport.$$ > /var/log/sysattack.$DATE.$$
elif [ "$FOUND" -eq 1 ]; then
##      cat $TMPDIR/checkreport.$$ | $MAIL -s "$HOSTNAME $DATE
system check" $SYSADMIN
      cat $TMPDIR/checkreport.$$ > /var/log/syscheck.$DATE.$$
fi
```

Note that the part that sent mail was commented out and instead will copy the file to /var/log and call it either sysattach.<date>.pid or syscheck.<date>.pid

Also had to change the way the date command was used because the “/” is not allowed in a file name:

Was: DATE='date +%m/%d/%y:%H:%M'

Now: DATE='date +%m-%d-%y:%H:%M'

- 12) Run logcheck.sh once manually to see if it can read log files and report any alerts. Made unsuccessful attempt to “su”, when logcheck.sh was run could see the file syscheck.05-01-02:12.54.19575 with reference to failed su attempts.

- 13) Made logcheck files owned by root so it could write into /var/log
- 14) Added logcheck.sh to crontab to run every half hour:

0,30 * * * * /opt/local/etc/logcheck.sh > /dev/null 2>&1

10. Fix-modes is a program that checks each file in the /var/sadm/contents file and changes their permissions ownership, etc. to be more secure. Followed following procedure to install and run Fix-modes program:

- 1) Download fix-modes program from:
<http://www.science.uva.nl/pub/solaris/>
- 2) Used sftp to transfer file to Sun server.
- 3) uncompressed FixModes.tar.Z.
- 4) **# tar xvf FixModes.tar**
- 5) **# cd FixModes**
- 6) **# ./fix-modes**
- 7) Inspected /var/sadm/install/contents.mod file to see which files were changed.
- 8) rebooted and tested system confirmed “ssh” and “sftp” works, inspected log files, everything seems normal.

11. Aide stands for (Advanced Intrusion Detection Environment). It checks the integrity of files and is a free program similar to tripwire. Use the following procedures for installation of Aide:

- 1) Download from <http://www.sunfreeware.com:>
 - aide-0.7-sol8-sparc-local.gz
- 2) Check MD5 checksum before transferring to Sun Server.
- 3) Use sftp to transfer files to Sun server.
- 4) Unzip and use pkgadd to install.
- 5) Aide documentation suggests placing Aide binary on read only media, since /usr is mounted ro, /usr/local/bin should work.
- 6) Moved /usr/local/aide to /opt/local/aide
- ## Follow previously described procedure for making /usr/ “writable” if it is currently mounted “ro”.

```
# mkdir /opt/local/aide
# chown root:root /opt/local/aide
# cp /usr/local/aide/aide.conf /opt/local/aide
```

- 7) Edited /opt/local/aide/aide.conf file: **@@define TOPDIR /opt/local/aide**
- 8) Files to be added to Aide’s database are controlled by the aide.conf file.

- 9) aide.conf also controls what parameters the program will check:

```
##
##p:  permissions
##i:  inode
##n:  number of links
##u:  user
##g:  group
##s:  size
##b:  block count
##m:  mtime
##a:  atime
##c:  ctime
##S:  check for growing size
##md5: md5 checksum
##sha1: sha1 checksum
##rmd160: rmd160 checksum
##tiger: tiger checksum
##R:  p+i+n+u+g+s+m+c+md5
##L:  p+i+n+u+g
##E:  Empty group
##>:  Growing logfile p+u+g+i+n+S
```

- 10) For example, to check all files in /etc/ for permissions, inode, user, and group you would place the following in /etc:

```
/etc p+I+u+g
```

If you want to ignore a subdirectory can use the “not” operator:

```
!/var/log/*    ## ignore all file in /var/log
```

Here are the changes made to aide.conf to select files to be monitored:

```
/ R
!/proc.*
/usr R
/var R
/opt R
/export/home R
!/opt/local/var.*
!/tmp
!/tmp.*
!/proc
!/proc.*
!/dev/fd
!/dev/fd.*
!/var/log.*
!/var/adm.*
```

!/var/audit/*

“R” is defined in aide.conf as:

p+i+n+u+g+s+m+c+md5

Note: removed /var/tmp and linked it to /tmp

ln -s /tmp /var/tmp

That way when system reboots /var/tmp is cleaned out as well as /tmp and Aide can monitor all of /var except for /var/log, /var/adm, and /var/audit.

- 11) Once the system is set up there should not be much changing of any files other than the log files and files for Sar, and audit. It is best to include most files and then examine the report to see how large it is and what directories can be ignored.
- 12) Can use: **aide -verbose=255** to debug any problems with the aide.conf file
- 13) Once config file is created with all the files selected and their parameters the database needs to be initialized with the following command:

aide -c /opt/local/aide/aide.conf—init

This created a 3.9 Mb db file that includes all the information for each file including md5 checksums, permissions, etc.

Note: database, config file, aide binary, and even the manual pages should also be on the media as suggested by the Aide manual. Actually, the manual suggests that the config file be placed offline, perhaps on a floppy, so that any attacker can not read it or alter it. This configuration file is very important, because it contains information as to where a cracker could place their rootkit and avoid Aide detection.

- 14) Integrity of files is checked by the following command:

aide -c /opt/local/aide/aide.conf—check

Aide reads the database and compares it to the actual condition of the files on the disk. Any discrepancies between the database and actual condition of the files is noted in the report. It takes about 10-15 minutes to run the aide -check command.

Here is the output from the aide -check command which was run just after the initial data base was created:

error mmap'ing /etc/mnttab

AIDE found differences between database and filesystem!!

Start timestamp: 2002-04-30 19:19:00

Summary:

Total number of files=31005,added files=2,removed files=0,changed files=5

Added files:

added:/usr/local/aide/aide.db.new

added:/opt/admin/aide/x

Changed files:

changed:/usr/local/aide

changed:/opt/admin/aide

changed:/opt/admin/aide/aide.out

changed:/devices/pseudo/pts@0:6

changed:/.sh_history

Detailed information about changes:

File: /usr/local/aide

Mtime: old = 2002-04-30 19:01:21, new = 2002-04-30 19:02:56

Ctime: old = 2002-04-30 19:01:21, new = 2002-04-30 19:02:56

File: /opt/admin/aide

Mtime: old = 2002-04-30 18:40:31, new = 2002-04-30 19:19:00

Ctime: old = 2002-04-30 18:40:31, new = 2002-04-30 19:19:00

File: /opt/admin/aide/aide.out

Mtime: old = 2002-04-30 19:02:03, new = 2002-04-30 19:19:00

Ctime: old = 2002-04-30 19:02:03, new = 2002-04-30 19:19:00

File: /devices/pseudo/pts@0:6

Mtime: old = 2002-04-30 19:02:03, new = 2002-04-30 19:18:30

File: /.sh_history

Size: old = 54684 , new = 54900

Mtime: old = 2002-04-30 19:02:03, new = 2002-04-30 19:19:00

Ctime: old = 2002-04-30 19:02:03, new = 2002-04-30 19:19:00

MD5: old = c6qIObqeNFfV5ZeQQLkrNA== , new =

cYG6FejrBs6thHARS1Y2Gg==

End timestamp: 2002-04-30 19:30:34

Note: I added /opt/admin/x just to see if aide would catch it.

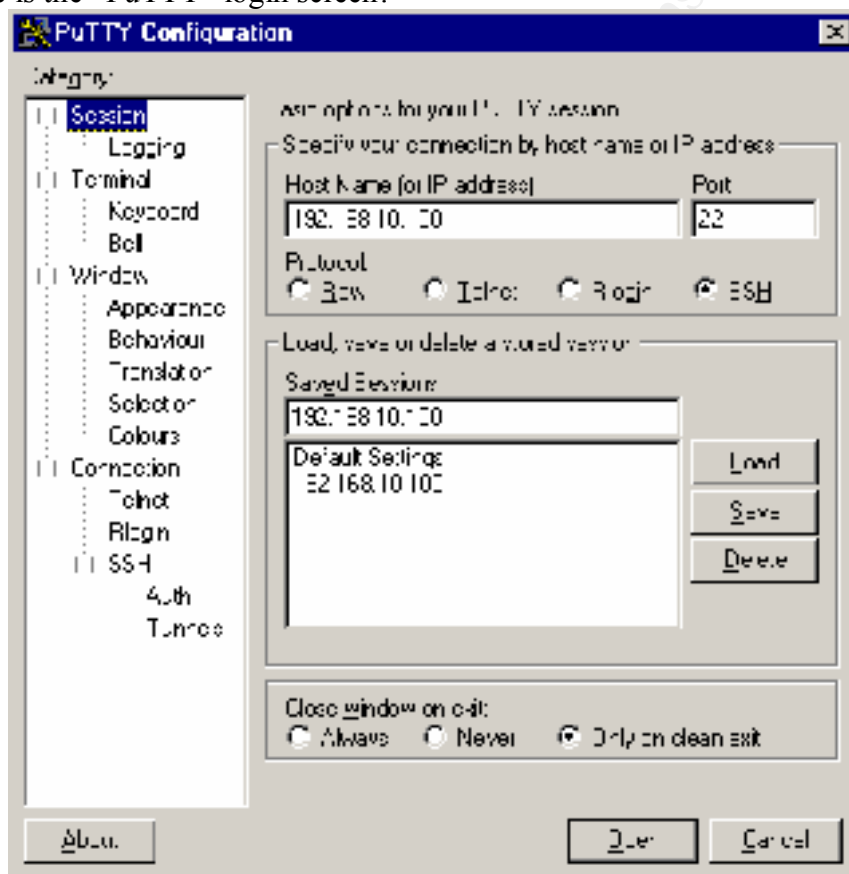
Could add /.sh_history to the ignore list in aide.conf file since that file will change every time computer is used by super-user.

- 15) Report is now manageable enough to be able to quickly notice any unauthorized changes. Will wait a day or so and see how any cron jobs and additional usage affects the report, but the computer would normally be idle until a Security Assessment is required, so no major changes to files should occur under normal circumstances.

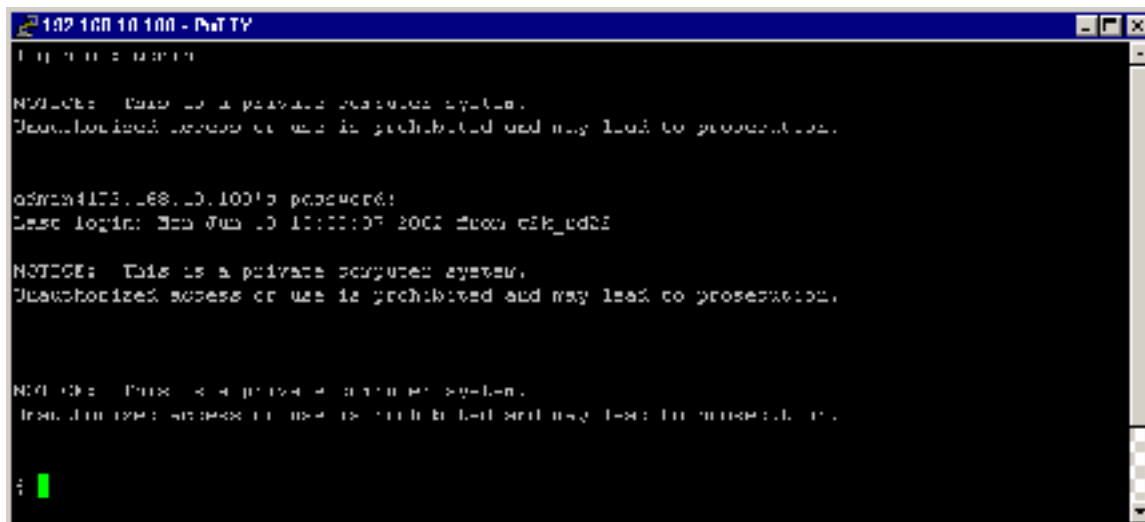
Testing and Verification

Most of the testing and verification was done along the way, but now that the server is complete and /usr is mounted “ro”, will do the final testing and verification. This is the completion from SANS Institute Track 6 Securing UNIX 6.5 Linux/Solaris Practicum section Solaris Security: Step-by-Step, pages 203-235.

1. Here is the “PuTTY” login screen:



2. Here is the login session through “PuTTY” using ssh, port 22:



```
192.168.10.100 - PuTTY
login as: root

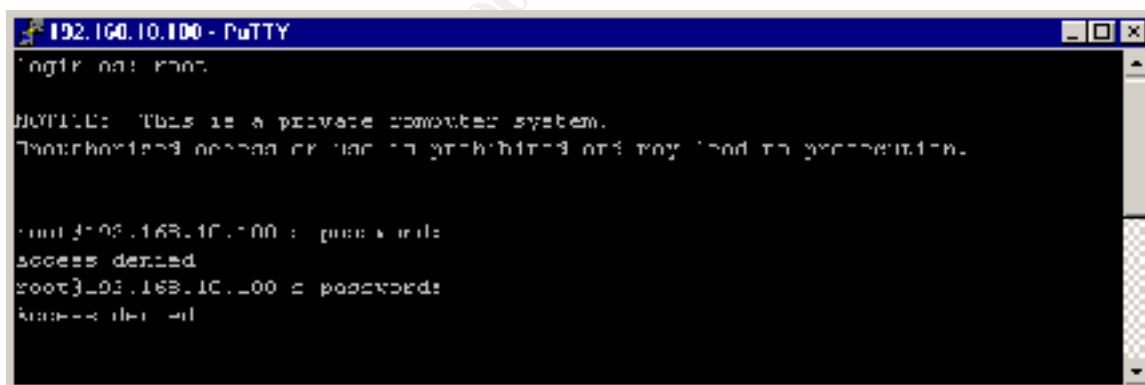
NOTICE: This is a private computer system.
Unauthorized access or use is prohibited and may lead to prosecution.

admin@192.168.10.100's password:
Last login: Sun Jun 13 11:01:07 2002 from ctk.pds

NOTICE: This is a private computer system.
Unauthorized access or use is prohibited and may lead to prosecution.

root@192.168.10.100's password:
Last login: Sun Jun 13 11:01:07 2002 from ctk.pds
```

3. Here is the login screen trying to login as root:



```
192.168.10.100 - PuTTY
login as: root

NOTICE: This is a private computer system.
Unauthorized access or use is prohibited and may lead to prosecution.

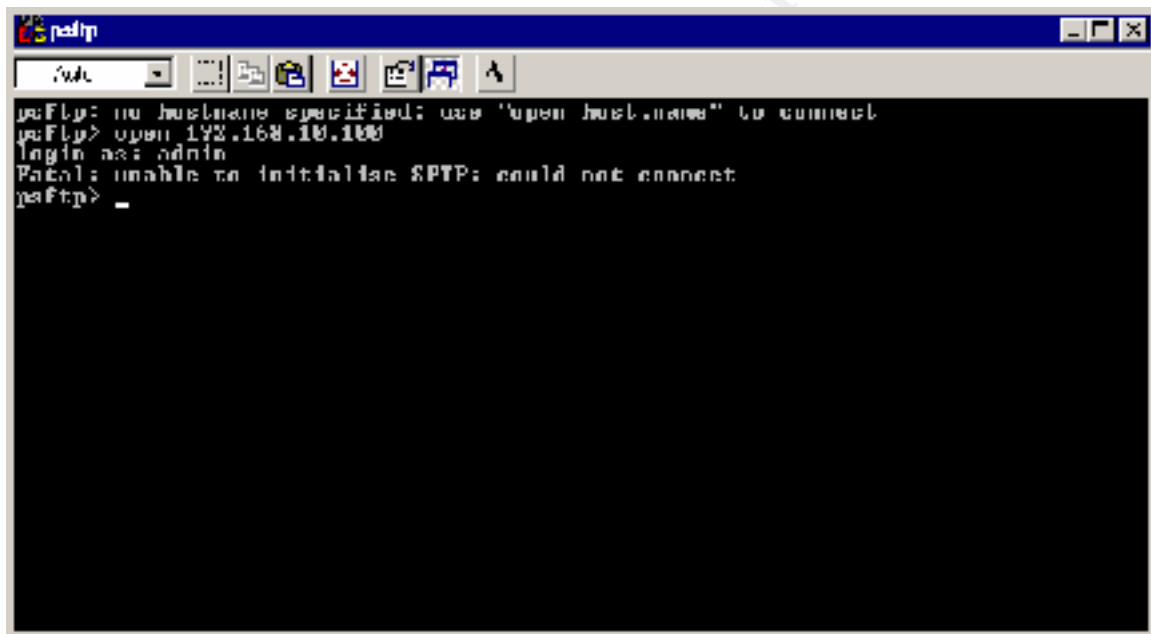
root@192.168.10.100's password:
access denied

root@192.168.10.100's password:
access denied
```



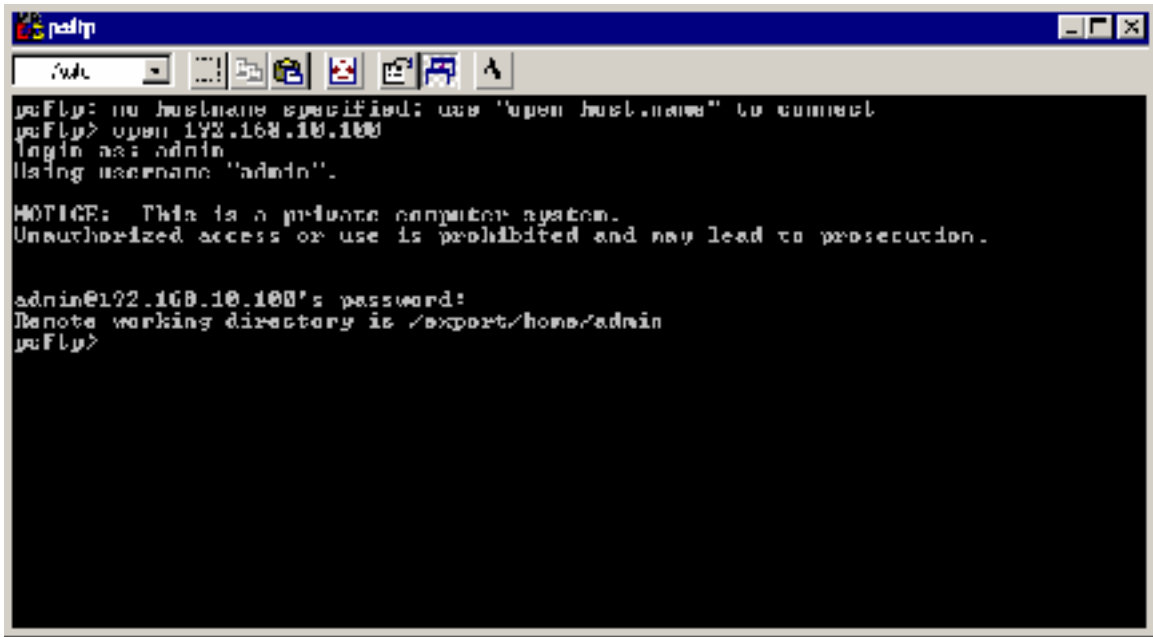
Note: The reasons why root can't login:

- 1) The /usr/local/etc/sshd_config file has
"PermitRootLogins=no"
 - 2) The /etc/default/login has CONSOLE=/dev/console, so no
remote logins allowed for root.
4. The IP address of the PC was changed to 192.168.10.105, the hosts.allow only has 192.168.10.101 in it so when a login is tried using ssh then screen just disappears. Here is the output trying to use sftp:



```
psftp> no hostname specified: use "open hostname" to connect
psftp> open 192.168.10.105
login as: admin
Warning: unable to initialize SFTP: could not connect
psftp> _
```

5. So the IP address was changed back to 192.168.10.101 and rebooted it worked again. Here is the output for the sftp session:



```
msf5: no hostname specified: use "open.hostname" to connect
msf5> open 192.168.10.100
login as: admin
Using username "admin".

NOTICE: This is a private computer system.
Unauthorized access or use is prohibited and may lead to prosecution.

admin@192.168.10.100's password:
Remote working directory is /export/home/admin
msf5>
```

6. Here is the output of the “mount” command showing the filesystems set as “nosuid”:

```
/ on /dev/dsk/c0t0d0s0
read/write/setuid/intr/largefiles/logging/onerror=panic/dev=2200000 on
Wed May 8 14:32:22 2002

/usr on /dev/dsk/c0t0d0s3 read
only/setuid/intr/largefiles/onerror=panic/dev=2200003 on Wed May 8
14:32:18 2002

/proc on /proc read/write/setuid/dev=3380000 on Wed May 8 14:32:20
2002

/dev/fd on fd read/write/setuid/dev=3440000 on Wed May 8 14:32:22
2002

/etc/mnttab on mnttab read/write/setuid/dev=34c0000 on Wed May 8
14:32:23 2002

/var on /dev/dsk/c0t0d0s4
read/write/nosuid/intr/largefiles/onerror=panic/dev=2200004 on Wed
May 8 14:32:23 2002
```


10. Here are the steps to check IP Filter:

- 1) Check to see if port 6000 is filtered out by using Exceed to try to connect from PC. First will see if we can get an "Exceed connection"
- 2) Make sure in.telnetd is not commented out in /etc/inetd.conf:

```
telnet stream tcp nowait root /usr/sbin/in.telnetd in.telnetd
```

- 3) # /etc/rc2.d/.NOS72inetd start ## Temporarily start up the inetd
- 4) Enter following rule in /etc/opt/ipf/ipf.conf file:

```
pass in log on hme0 all  
pass out log on hme0 all
```

Note: This will allow everything in and out, but log into /var/log/ipflog

- 5) Re-Read the ipf.conf file:

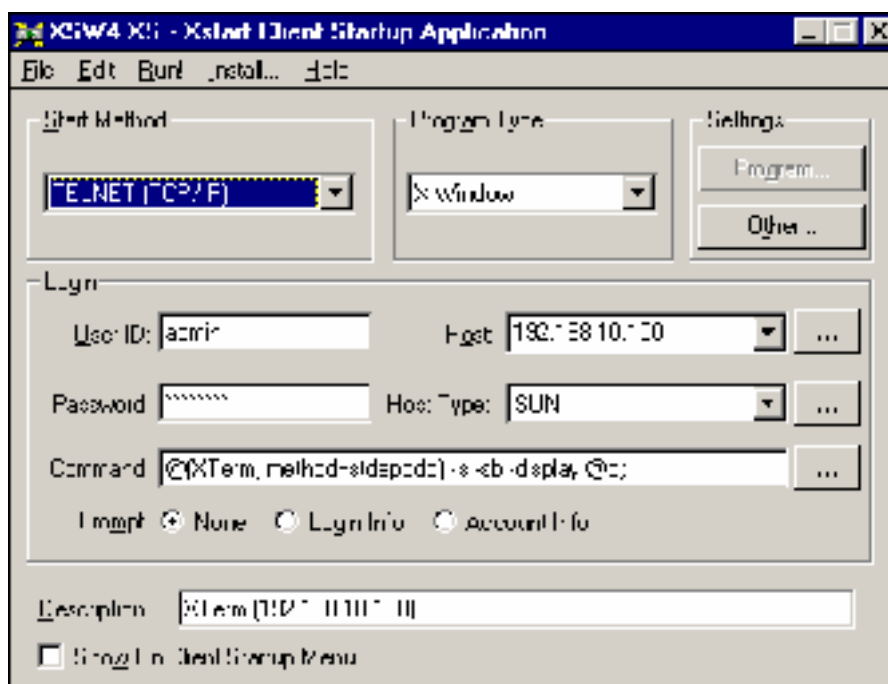
```
# /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf
```

- 6) Check to see which rules are in effect:

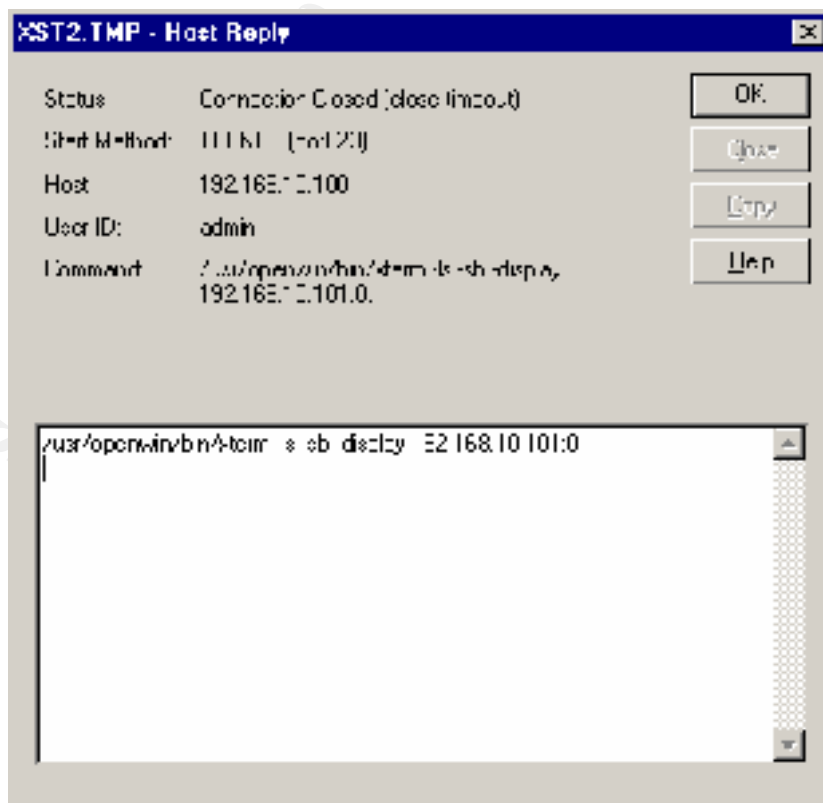
```
# /sbin/ipfstat -io
```

```
pass out log on hme0 from any to any  
pass in log on hme0 from any to any
```

7) Here is the Exceed setup screen:



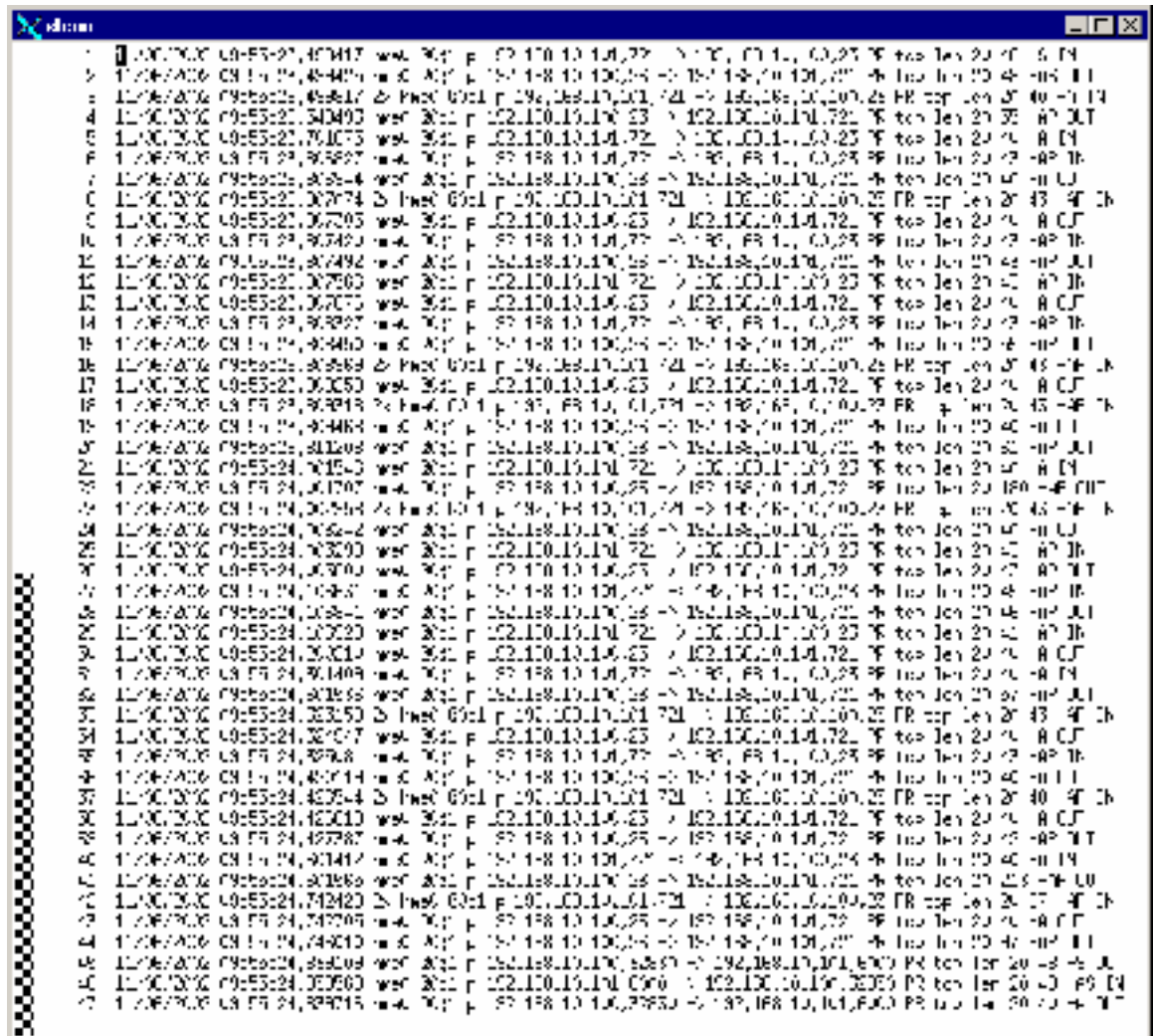
8) Here is the host reply screen showing "Telnet" as start method:



9) Here is the actual login screen:



10) Here is the log file /var/log/ipflog showing the first few lines:



```
1 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
2 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
3 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
4 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
5 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
6 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
7 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
8 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
9 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
10 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
11 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
12 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
13 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
14 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
15 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
16 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
17 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
18 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
19 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
20 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
21 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
22 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
23 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
24 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
25 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
26 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
27 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
28 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
29 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
30 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
31 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
32 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
33 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
34 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
35 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
36 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
37 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
38 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
39 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
40 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
41 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
42 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
43 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
44 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
45 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
46 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
47 192.168.1.100:5527->192.168.1.101:23 TCP ESTABLISHED
```

- 11) We can see that port 23 was used for telnetting into the server and then on line 45 we can see the port 6000 beginning to be used for “Xwindows” via xterm.
- 12) Next we block port 6000 to see it that will stop the ability to open an “Xwindows” session.

Note: You will have to do this from a console login, or using “PuTTY” and ssh, because once you block using the following rule, your “Exceed” session will hang.

- 13) Add the following line to /etc/opt/ipf/ipf.conf:

block out log proto tcp from any to any port 5999 <> 6256

14) Re-Read the ipf.conf file:

```
# /sbin/ipf -Fa -f /etc/opt/ipf/ipf.conf
```

15) Empty out the /var/log/ipflog so it will be easier to see the new messages:

```
# cat /dev/null > /var/log/ipflog
```

16) Check to see which rules are in effect:

```
# /sbin/ipfstat -io
```

```
pass out on hme0 from any to any
block out log proto tcp from any to any port 5999 >= 6256
pass in on hme0 from any to any
```

Note: Will only log the blocked packets.

17) Try to start a telnet session using “Exceed”

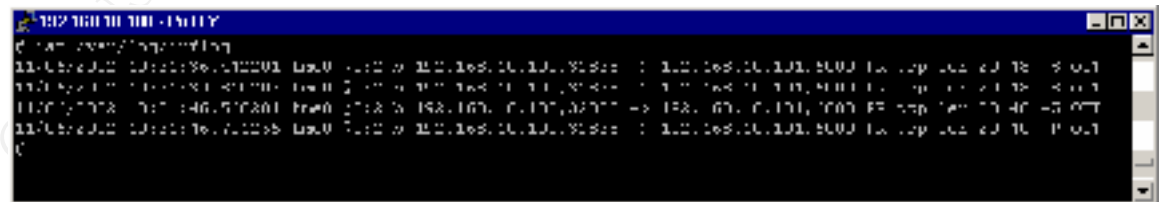
Hangs on:



18) Finally comes back with:



19) Here is the output from the /var/log/ipflog file:



20) This shows that IP Filter can effectively block port 6000.

21) Once testing is completed, make sure to put /etc/opt/ipf/ipf.conf back to how it should be, and also with /etc/initt.d.conf we want to comment out the telnet line.

Now that all applications have been installed and the system is relatively stable, we can do a Security Assessment of the system. First let's do a port scan using nmap:

nmap -P0 -v localhost

```
Starting nmap V. 2.54BETA28 ( www.insecure.org/nmap/ )
No tcp,udp, or ICMP scantype specified, assuming vanilla tcp connect() scan. Use
-sP if you really don't want to portscan (and just want to see what hosts are up).
Host localhost (127.0.0.1) appears to be up ... good.
Initiating Connect() Scan against localhost (127.0.0.1)
Adding open port 1241/tcp
Adding open port 22/tcp
Adding open port 111/tcp
Adding open port 6000/tcp
Adding open port -32765/tcp
Adding open port -32764/tcp
The Connect() Scan took 1 second to scan 1548 ports.
Interesting ports on localhost (127.0.0.1):
(The 1542 ports scanned but not shown below are in state: closed)
Port      State      Service
22/tcp    open      ssh
111/tcp   open      sunrpc
1241/tcp  open      msg
6000/tcp  open      X11
32771/tcp open      sometimes-rpc5
32772/tcp open      sometimes-rpc7
```

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

Note: Port 1241 is for the Nessus server
Port 22 is used by ssh
RPC ports are used by CDE

Next we will do a Security Assessment using Nessus.

1. Makes sure nessusd is running (**nessusd -D**) ## I don't have it as a start up script,
rather start it when needed
2. Run Nessus to start the client:
nessus &
3. Login using your Nessus id and pw
4. Chose "Display and remember the server certificate, do not care about the CA"
5. Target selection: **localhost**
6. All plugins except "Denial of Service" is selected

7. Here is the report from Nessus:

```
timestamps|||scan_start|Tue May 7 12:02:45 2002|
timestamps||localhost|host_start|Tue May 7 12:02:45 2002|
results|localhost|localhost|general/tcp|10336|Security Note|Nmap only scanned 0
TCP ports out of 65535.Nmap did not do a UDP scan, I guess.
results|localhost|localhost|general/udp|10287|Security Note|For your information,
here is the traceroute to 127.0.0.1 : \n127.0.0.1
results|localhost|localhost|unknown (177/udp)|10891|Security Warning|\n
The plugin sends a XDMCP QUERY request to see if the remote\nhost is running
XDM (or similar display manager) with XDMCP\nprotocol enabled.\n\nThis
protocol was used to provide X display connections for old \nX terminals.
XDMCP is completely insecure, since the traffic and\npasswords are not
encrypted. \n\nRisk factor : Medium\nSolution : Disable XDMCP
timestamps||localhost|host_end|Tue May 7 12:08:36 2002|
timestamps||scan_end|Tue May 7 12:08:36 2002|
```

8. Here is the procedure for disabling XDMCP as suggested by Nessus and taken from <http://home.attbi.com/~sabernet/papers/Solaris.html>:

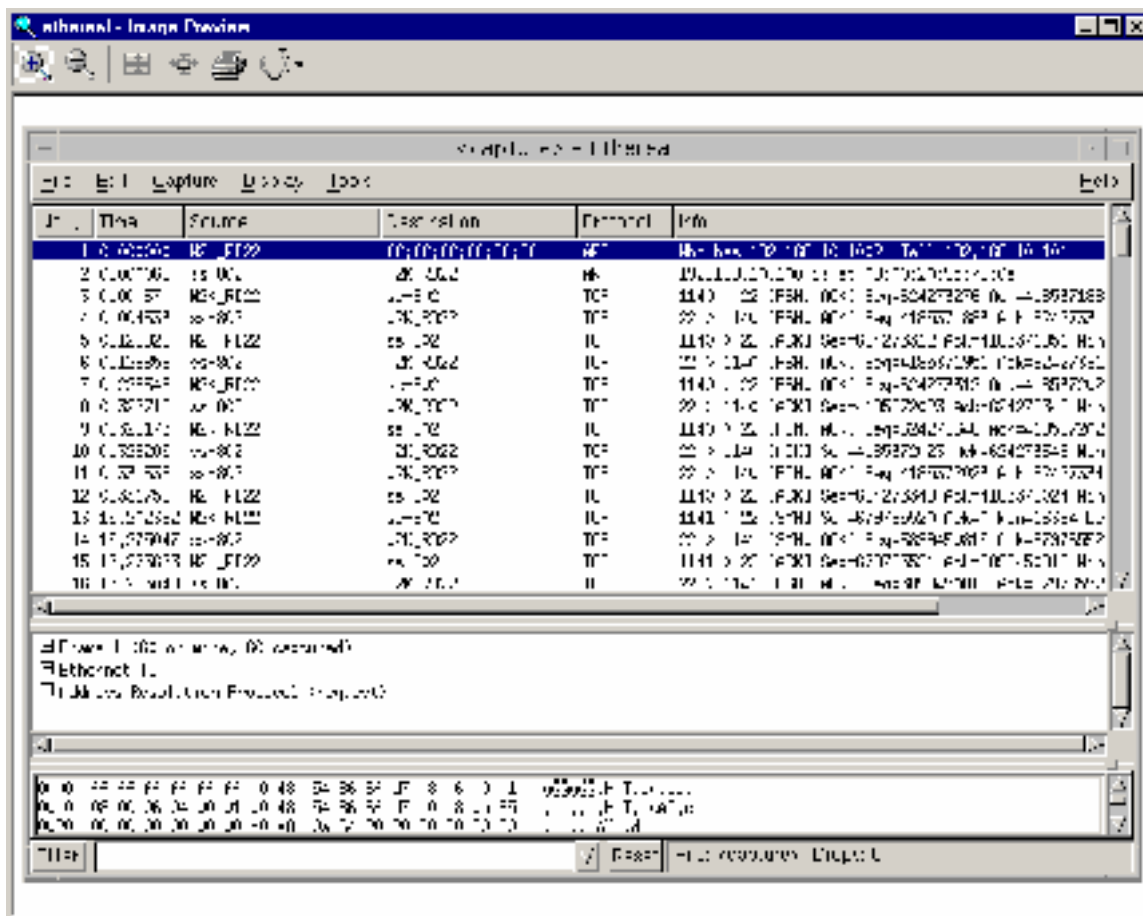
Place following information in the file /etc/dt/config/Xaccess

```
##
## Xaccess - disable all XDMCP connections
##
!*
```

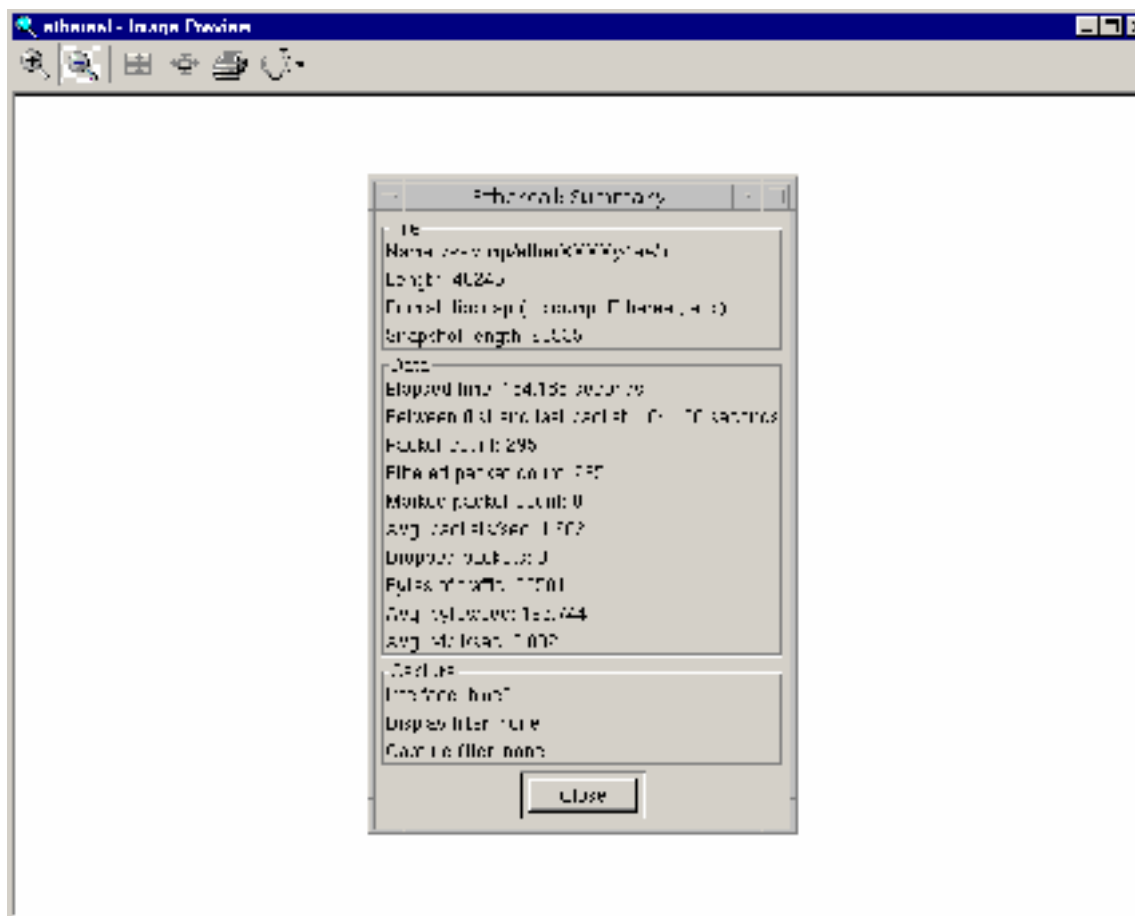
Set the permissions on /etc/dt/config/Xaccess to 444:
chmod 444 /etc/dt/config/Xaccess

Here is a example session using ethereal for packet capture:

1. # **ethereal**
2. ## Under capture enter "**hme0**" for interface
3. # On the PC, I started an ssh session using PuTTY.
4. Here is the full screen after capture in ethereal:



5. The Ethereal full screen is divided into three sections. The top section lists each packet captured, the middle section shows summary info for packet type, and the bottom section shows a hex dump of the packet selected in the top section.
6. Here is the summary from ethereal:



Last thing to look at is to show how the id and passwords are shown in clear text when using telnet as an example:

1. First I turned on temporarily the telnet service by starting .NOS72inetsvc:
/etc/rc2.d/.NOS72inetsvc
2. Edit the /etc/hosts.allow and add "in.telnetd"
3. Temporarily change "admin's" password to "tmp_Pw"
3. Start "dsniff"
4. telnet to Sun server

6. Here is the dsniff output:

sudo dsniff

listening on hme0

05/09/02 12:18:09 tcp W2K_RD22.1143 -> ss-802.23 (telnet)
admin

```
tmp_Pw
exit
```

Note that “dsniff” does not show any output when using tcp_wrappers in.telnetd and also no output when using ssh to access server via port 22. This confirms that the id and password are transmitted in clear text form with “telnet”, but not with tcp_wrappers and ssh.

6. Here is the output of the file “snort” created when it was set to capture data:

```
# pwd
```

```
/var/log/snort/192.168.10.101
```

```
# ls -l
```

```
total 2
```

```
-rw----- 1 root  other    280 May  9 13:02 TCP:1150-23
```

```
# cat TCP:1150-23
```

```
[**] TELNET access [**]
```

```
05/09-13:02:54.598948 192.168.10.100:23 -> 192.168.10.101:1150
```

```
TCP TTL:64 TOS:0x0 ID:41915 IpLen:20 DgmLen:55 DF
```

```
***AP*** Seq: 0x8D164A4F Ack: 0x6B3DE14A Win: 0x60F4 TcpLen: 20
```

Note that snort created a file /var/log/snort/192.168.10.101/TCP:1150-23.

In that file is reference to the telnet session on my PC.

Snort uses the /usr/local/doc/snort/rules/snort.conf file to set up which rules to follow. It can generate alerts based on the type of traffic received. These alerts could be sent to syslog, and logcheck could monitor them to be included in its “grand” alert scheme.

Ongoing maintenance

1. The logcheck program check the log files every half hour and places “syscheck” or “sysattack” file in the /var/log directory. The security administrator has to check these files every day to assess the status of the system. It might be a good idea to add the Sendmail package and have logcheck mail its alerts, but still the admin would have to regularly check their mail for the alerts.
2. Admin should regularly check filesystem status: “**df -k**” to make sure log files are not filling up any filesystem.
3. Should also check to make sure the /usr/lib/newsyslog script is properly rotating the log files.
4. Backups need to be performed regularly to insure minimal disruption to service in the event of any catastrophe.
5. Admin must check on a weekly basis the Cert advisories:
<http://www.cert.org/advisories/>
6. Admin must also check on a weekly basis Sun’s patch advisory page:
<http://sunsolve.sun.com/pub-cgi/show.pl?target=home>
7. This system has “Aide” running, which is a freeware program similar to “tripwire”. Admin must regularly (at least weekly) run “**aide -check**” and inspect the output file (/opt/admin/aide/aide.out). This program will show any file changes and Admin can make a determination as to the legitimacy of the changes.
8. There are a number of “Security” programs on this Sun server. Admin can periodically use these tools to assess its vulnerabilities. Nessus is particularly good at preparing a report that can be compared with previous reports to see if any changes have been made.

Summary/Conclusions

The original goal of this project was to build a Sun “Security” Server using the SANS Institute’s O/S hardening techniques in their Solaris Security: Step-by-Step documentation and experience first hand the problems and tradeoffs that would be encountered during the process. Significant effort was involved in exploring the packages included in the O/S installation and which packages to include and remove. I think with the ability to run the CDE environment, and to also be able to login from the command line and turn off the extra “CDE” required processes, you can have the best of both worlds, that is a very secure server with just command line functionality and minimal services, or the “CDE” environment with more functionality and a bit more exposure. With Command Line Login you can still run nessusd, and can run nessus client in “batch” mode (see Nessus configuration section for an example). Also, in Command Line Mode, you can still connect from the PC using “PuTTY” and ssh.

IP Filter gives the ability to filter source and destination packets by port and by packet type. Basic actions are to block or pass. Packets can even be filtered by “shorts” or “fragments”. There is no “implicit deny” so all packets are allowed unless a match is found in the rules file. Rules file is read from top to bottom and last rule read that matches conditions of the rules is applied unless a previous match had “quick” specified. Mfilter uses ifconfig then generates some recommended rules to start with.

IP Filter can be used to protect server from outside network attacks, but when ports need to be opened, for example, to do a Nessus report, then ports can be opened for the target host only while the report is being run, then closed down again after Nessus is finished. See the IP Filter section for examples of rules files and port shutdown.

Using /usr as a “ro” filesystem with so many programs in the /usr/local directory structure was also enlightening. Some subdirectories had to be moved to “/opt” because they used them to write “log” files, “pid” files, “CA” files, etc. As an unexpected bonus, some configuration files such as /usr/local/etc/sudoers cannot be changed unless the server is rebooted with /usr “rw”. This provides additional security since “sudo” has the power to grant root access. The files and procedures to move them is included in the sections for configuring the security software, for example, the logcheck files in the logcheck configuration section.

Many “Security Applications” were installed on the server. Only a minimal amount of their overall functionality was used, but the initial installation, configuration, and the testing of the network between the PC and the Sun server proved quite informative. Overall, it appears that the initial objective was accomplished, and I consider this project to have been very meaningful. The end result is a document which should prove useful to anyone wishing to install and configure a server such as described in this practicum.

References

“CERT/CC Advisories”. URL:
<http://www.cert.org/advisories/>

“Double Free Bug in zlib Compression Library. URL:
<http://www.cert.org/advisories/CA-2002-07.html>

Christensen, Steven. “Installing OpenSSH Packages.” URL:
<http://www.sunfreeware.com/openssh.html>

“Installing and configuring TCP Wrappers on Solaris 7 and Solaris 8.”
Solaris Resources at Kempston. URL:
<http://www.kempston.net/solaris/tcpwrappers.html>

Kurtz, Ronald L., Vines, Russell Dean. The CISSP Prep Guide: Mastering the Ten
Domains of Computer Security: John Wiley & Sons, Inc. 2001.

Lehti, Rami. “The Aide Manual”. URL:
<http://www.cs.tut.fi/~rammer/aide/manual.html>

Noordergraaf, Alex. “Solaris™ Operating Environment Minimization for Security: A
Simple Reproducible and Secure Application Installation Methodology.” Sun
BluePrints™ OnLine. November 2000. URL:
<http://www.sun.com/solutions/blueprints/1100/minimize-updt1.pdf>

Pomeranz, Hal. SANS Institute Track 6 Securing Unix.

SANS Institute resources. “The Twenty Most Critical Internet Security Vulnerabilities.”
URL:
<http://www.sans.org/top20.htm>

“Solaris Security Guide.” URL:
<http://home.attbi.com/~sabernet/papers/Solaris.html>

“Tcp wrappers (tcpd) thinks all hosts are 0.0.0.0 in Solaris 8 or in some versions of AIX.”
URL: <http://www.faqs.org/faqs/computer-security/most-common-qs/section-16.html>

Voeckler, Jens. “Solaris™ 2.X – Tuning Your TCP/IP Stack and More”. URL:
http://athena.vvsu.ru/docs/tcpip/solaris_tcpip/tune.html