Global Information Assurance Certification Paper

SANS Global Information Assurance Certification Program

GCUX Certified UNIX Security Administrator Version 1.8

The Design and Configuration of a Secure SMB File Server

Shigeki Misawa
January 2002

Introduction

This document describes the design and configuration of a SMB (aka Samba[1]) server to serve selected UNIX file systems within a data center to desktop computers running Microsoft Windows that are outside of the data center. The SMB server is slightly non-standard for two reasons. First, it serves file systems that are NFS mounted from several file servers. Second, it is a dual ethernet interface system that SMB serves file systems, residing inside a data center firewall, to desktop clients residing outside of the data center firewall.

The SMB server is one component in a large data center that is in the process of tightening its security. With many systems in need of securing, instead of configuring and hardening the SMB server "by hand", an attempt was made to automate as much of the work as possible (given the resources at hand) so that the work that is done could be used to help secure other systems. The system chosen to accomplish this was Sun Microsystems' JumpStart™ technology[2] complemented with Sun Professional Service's JASS (Jumpstart Architecture Security Scripts) toolkit[3].

System Configuration Overview

The hardware used to create the SMB server consisted of the following:

- The Jumpstart server :
  - Pentium II based system with 128 MB of RAM
  - 20GB hard disk
  - Video monitor
  - Video Card
  - Ethernet Card
  - Keyboard
  - Mouse
  - CD-ROM drive
  - All components must be on the Solaris™ 8 Hardware Compatibility List[4].

- The SMB Server :
  - Dual CPU capable Pentium III system with one 800 MHz CPU
  - Dual SCSI buses on board
  - Two 9GB SCSI disks
  - 100 BaseTx Ethernet cards
  - One CD-ROM drive
  - On-board Cirrus Logic based video adapter
  - Keyboard
  - Mouse
  - All components must be on the Solaris™ 8 Hardware Compatibility List[5].

The software used to create the SMB server was as follows:

- The Jumpstart Server :
  - Solaris 8 for Intel 10/01
  - JASS[6] Version 0.3.3 – Solaris OS hardening Toolkit.
  - Sun C or GNU C compiler.

- The SMB Server :
  - Solaris 8 for Intel 10/01
  - Samba[7] Version 2.2.2 – Unix based SMB server
  - OpenSsh[8] Version 3.0.2p1 – Ssh server
  - ANDIrand[9] Version 0.7 - /dev/random implementation
  - OpenSSL[10] Version 0.9.6c – SSL/TSL library implementation
  - Tripwire[11] Version 1.2 – Software integrity checker
  - TCP Wrappers[12] Version 7.6-ipv6.1 – TCP Wrappers with IPv6 support
  - Stunnel[13] Version 3.22 – Universal SSL/TSL wrapper
  - IP-Filter[14] Version 3.4.20 – Host based firewall
  - Patchdiag[15] Version 1.0.4 – Solaris OS installed patch diagnostic tool
  - Patchcheck[16] Version 1.1 – Solaris OS installed patch diagnostic tool
  - Log rotation script – Log file rotation script[17]

The SMB system was configured to work with pre-existing services/servers at the data center. These included :

- Three NIS/DNS/NTP servers
- A secure system administration server
- A serial port console switch server
- A syslog/tripwire server
- Print services from a handful of corporate and data center print servers

The NIS servers provided the home directory information required by the SMB server. The internal DNS servers provided name service and the internal NTP servers provided synchronized time service. The secure system administration server allowed secured network access to the system for system administration purposes. The serial port console switch server provided secure access to the system in the event of network failure. The syslog/tripwire server provided a secure location where system messages could be archived and where nightly tripwire system integrity checks could be initiated. Finally, SMB based print services were provided by the SMB system to printers located at various points around the corporate subnet by accessing the print services of these network based printers.

To protect the SMB server from direct network assaults, a host based firewall was used to filter out inappropriate network traffic. To protect the SMB account passwords from

being sniffed, encrypted passwords where used and a local SMB password file was used. This latter step helped in decoupling the data center Unix passwords from the SMB passwords. Finally, an encrypted password management scheme using SWAT (provided with Samba) and Stunnel, an SSL wrapper, was used[18].

Risk Assessment

To understand the design of the server and to make a risk assessment, it is necessary to describe the context in which the server resides. The data center is a departmental data center located within a corporate site. The data center consists of a set of file servers that serve file systems (via NFS) to a large farm of computational nodes running the Linux operating system (OS), also contained in the data center. The Linux nodes are both batch and interactive. Interactive access to the facility is given to on-site personnel as well as affiliated off-site personnel from collaborating institutions from around the world. The off-site personnel constitute a significant fraction of the active interactive users of the facility.

In addition to providing centralized data processing resources, the facility is responsible for providing some centralized resources for on-site users that are outside of the data center. One resource that is mandated by the community is access to home directories that reside at the data center from desktop Windows based computers that reside outside of the data center. Since this is a mandated service, the wisdom in providing such a service is not an issue.

Security and Network Context

The method by which the SMB file service is provided is dependent on the security and network environment that currently exists at the facility and will hopefully evolve as the environment evolves. Figure 1 shows the network topology at the data center. The data center consists of five separate subnets, all behind the data center "firewall". OpenSubnet-1 is currently relatively open to access from the corporate subnets and the rest of the Internet. However, the firewall rules for this subnet will eventually evolve to a "default deny" set of rules.This subnet is for systems that require extensive connectivity to the outside world (e.g., time servers, DNS servers, AFS servers, etc). ClosedSubnet-1 through 4 are closed to access from outside the data center. These subnets contain the NFS file servers and the farm of Linux based computation nodes. For performance and availability reasons, the data center firewall is not implemented as a single box. Instead, the firewall consists of a hardware firewall in parallel with a set of dual interfaced systems that act as Ssh proxies and FTP proxies.

The Corporate subnets, the subnets that reside behind the corporate firewall, are where the Windows based desktop systems reside. In addition to these Windows based systems, the corporate subnet is home to a substantial number of Linux desktop systems and a

constantly changing population of laptops running, various operation systems, that are owned by visiting staff.
The Corporate firewall is currently in a “default allow” configuration with blocking rules that are fairly loose.

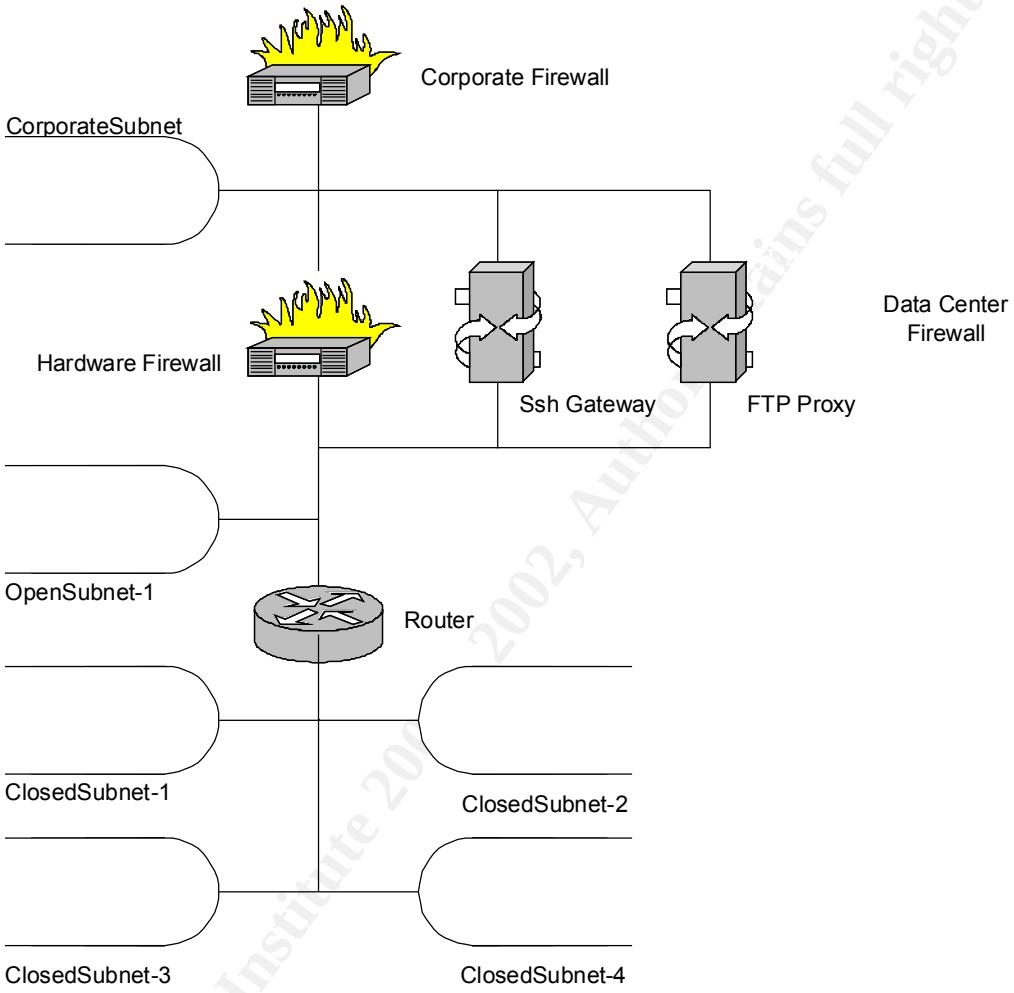

Figure 1 Data Center Network Topology

Given the security and network context, the next step is determining what we are attempting to protect, i.e., what are the crown jewels we are attempting to protect, and from what threats. The primary asset that is being protected is the integrity and operation of the data center, that is the NFS server, the Linux farm, and the data that is processed at the farm. Note that the integrity and operation of the SMB server is not perceived to be of primary importance, except if it compromises the integrity and operation of the data center.

With this network topology, population of computing systems, and user community, the perceived threats are of three types. Direct assaults to the facility from the internet, direct

assaults from compromised systems in the corporate subnet, malicious authorized and unauthorized interactive users on the system.

SMB Server Design

With the existing network topology, the choice was made to use a dedicated SMB server instead of installing SMB servers on the NFS servers. This would isolate the NFS servers from the SMB clients and simplify things from a user perspective by providing a single server from which they could gain access to the file systems. Also, securing one SMB server is considerably easier than securing multiple SMB servers. This is particularly true given the high data rates on the NFS servers. With a dedicated SMB server, it was possible to add a host based firewall to protect the system, something that could not be done on the NFS servers.

Of the three possible locations for the SMB server; inside the firewall; outside the firewall; by-passing the firewall (like the Ssh gateways and FTP proxies), the choice was made to by-pass the firewall. Although placing the server inside the firewall would have allowed the firewall to protect the server, availability and performance of the SMB service is limited by that of the hardware firewall. Historical availability and performance issues with the firewall made this choice unacceptable. Putting the SMB server in the corporate subnet suffers from the same performance and availability problems. In addition, it would require passing NFS service through the firewall, which has cause problems with the firewall before (this traffic has since been shut off) and also has known security problems. It would also require figuring out how to duplicate the DNS, NTP, and NIS services that are available inside the firewall.

With the server straddling the firewall, it is possible to consider using the DNS, NTP, and NIS services provided by the data center. In addition, it decouples the performance and availability of the service from that of the firewall. The downside is the system is not protected by the firewall and it also becomes a potential route for unauthorized access into the facility.

Infrastructure Support

Since the SMB server sits within an existing data center, the SMB server can conceivably use existing services that are provided by other systems at the data center. The services that are available for use by the SMB server are as follows:

- NIS/DNS/NTP servers
- Secure system administration server
- Serial Port console switch server
- Syslog/Tripwire server

User information, required for SMB home directory service, can be provided by the NIS server. Although, a relatively unsafe service, current practices at the facility prevent a more secure distribution of user information from being implemented at this time. For this reason, the NIS server is used to provide user information. By using this service, rpcbind and ypbind become a potential conduit for intrusion into the SMB server and the SMB server becomes an easier conduit for gaining access to other systems within the data center. At this time, it was decided that the combination of a host based firewall and proper hardware firewall rules mitigate most of these issues. (Note also that the use of automount and NFS client software, required for the operation of the server, require that rpcbind be run, so the increased risk is really only through the need to run ypbind.)

Using the data center NTP server allows accurate and synchronized time service without having to expose the SMB server to external time servers. An alternate method would have been to synchronize time via periodic cron jobs, but the risk of using NTP when combined with the host based firewall and the hardware firewall reduce the exposure sufficiently to warrant its use. For the same reasons, the internal DNS servers were used for name resolution.

(It should be noted that NIS, DNS, and NTP are all served by the same set of servers making them a choice conduit for data center intrusions. As a result, the effort that has been made to automate the SMB server configuration will be used to tighten these servers. If more equipment becomes available, it is likely that the NIS/NTP services will be de-coupled from DNS service.)

Since the SMB server is a single service server, the only interactive access that is required is administrative. To increase the security of the system, administrative access is allowed only from the secure system administrator server. This reduces the possibility of security problems resulting from bugs in the Ssh daemon. With the availability of a secure serial port console switch server, it is also possible to access the SMB server via the serial port.

The final server that is available is a combined Syslog/Tripwire server. This server provides a secure location where syslog output can be recorded and where tripwire system integrity checks can be launched and where the tripwire database can be stored. Given the way the Syslog/Tripwire server interacts with the system, it provides an additional means by which the SMB server could be compromised. However, in the bigger picture, if the Syslog/Tripwire server were compromised, there would be more serious problems.

Securing the server

With the environment of the SMB server defined, details on how to secure the SMB server can be discussed in more detail. First, to minimize potential points of compromise, a minimal installation of the operating system is made. Note that one might argue that this is not very critical since it is not an interactive system. However, the less software

installed, the fewer services need to be disabled. The second step in securing the server is hardening the configuration of the operating system. This is done via the JASS[19] toolkit in a process detailed later in this document. The third, and most important step in securing the system is configuring the host based firewall.

With the dual ethernet interfaces and the network topology, the configuration of the host based firewall is relatively simple. The only network traffic allowed through the external interface is SMB traffic, outgoing mail to the corporate mail server, and outbound connections to network printers. To protect the system further, only traffic to and from the corporate subnets are allowed, no Internet access is allowed. On the internal network interface, only traffic to and from the internal subnets is allowed. This configuration assumes that all systems in the internal data center subnets are secure. A more secure configuration would be to isolate access to only required systems within the internal subnets. However, the decision was made to keep the firewall rules relatively simple since other, more valuable targets are accessible from a compromised internal system. In this configuration, the security of the SMB server is heavily dependent on the security of the internal systems on which it relies, the proper configuration of the hardware firewall, and the security of the SMB protocol and Samba server implementation.

Finally, since the system is located at the data center, the system is physically secure. The data center is guarded by a card key system that allows only authorized users into the facility. As with most data centers, all electrical power to the facility is conditioned and there is a UPS to provide backup power.

Some final thoughts on risk assessment

When viewed in isolation, several tradeoffs have been made that simplify maintenance and configuration and correspondingly reduce security. However, when viewed in the context of the entire data center, these additional security risks are considered to be minor when compared to the other risks that are at the facility. (The most notable are the interactive users that are accessing the Linux farm. As well as the management of the authorization database that contains the list of authorized users.) It is more productive to expend resources on reducing these other risks than to spend them reducing the risks on the SMB server.

Configuring the system.

Sanitized Network Configuration

In order to discuss the actual configuration of the system, a sanitized network configuration is used. The necessary network parameters are as follows :

- **NIS00-IP**, **NIS01-IP**, **NIS02-IP** – NIS server IP addresses.
- **NIS00-Name**, **NIS01-Name**, **NIS02-Name** – NIS server host names.

- **DNS00-IP**, **DNS01-IP**, **DNS02-IP** – DNS server IP addresses.
- **DNS00-Name**, **DNS01-Name**, **DNS02-Name** – DNS server host names.
- **NTP00-IP**, **NTP01-IP**, **NTP02-IP** – NTP server IP addresses.
- **NTP00-Name**, **NTP01-Name**, **NTP02-Name** – NTP server host names.
- **SYSLOG-IP**, **SYSLOG-Name** – IP address and host name of syslog server
- **TRIP-IP**, **TRIP-Name** – IP address and host name of tripwire server.
- **SYSADM-IP**, **SYSADM-Name** – IP address and host name of system administration server.
- **SUBNETIn00-IP**, **SUBNETIn01-IP**, **SUBNETIn02-IP**, **SUBNETIn03-IP**, **SUBNETIn04-IP** – Addresses of the internal subnets. Note that the inside interface of the SMB server is in **SUBNETIn00-IP**.
- **DNIn** – Domain name used by the internal subnets.
- **MASKIn00**, **MASKIn01**, **MASKIn02**, **MASKIn03**, **MASKIn04** – Subnet masks for the internal subnets.
- **SUBNETOut-IP** – Address of the external subnet, that is the subnet that is outside the data center firewall and is directly connected to it. Note that the outside interface of the SMB server is in **SUBNETOut-IP.**
- **DNOut** – Domain name used by the external subnets.
- **MASKOut** – Subnet mask for the external subnet.
- **SUBNETSite-IP** – Address of the corporate or site subnet.
- **DNSite** – Domain name used by the corporate network.
- **MASKSite** – Subnet mask for the corporate or site subnet.
- **GATEWAYIn-IP** – IP address of the router in SUBNETIn00-IP.
- **GATEWAYOut-IP** – IP address of the router in SUBNETOut-IP.
- **SMBIn-IP**, **SMBOut-IP** – SMB server inside and outside IP addresses.
- **SMBIn-Name**, **SMBOut-Name** – SMB server inside and outside IP addresses.
- **ETHERNET-MAC** – SMB server inside interface ethernet MAC address.
- **HOSTNAME** – Unqualified hostname. (**SMBIn-Name** and **SMBOut-Name** are of the form **HOSTNAME.DNIn** and **HOSTNAME.DNOut**, respectively.)
- **NISDOMAIN** – NIS domainname.
- **JS-IP** – IP address of the Jumpstart server.
- **INSTALL-IP** – IP address used during the install process.

Configuring a Jumpstart Installation Server

As was stated previously, the SMB server is one of many systems at the data center that requires a secure configuration of the operating system, many of which will be identically configured (e.g., the Ssh gateways). In addition, the primary recovery procedure for a failed system disk is a complete reinstallation of the operating system from fresh media. As a result, a highly automated, repeatable, and quick installation and configuration system for the OS and third party software was required. The system chosen for this task was Sun's Jumpstart™[20] technology complemented with Sun's JASS [21](Jumpstart

Architecture Security Scripts) toolkit. The Jumpstart technology provides a means of automating the installation process while the JASS toolkit provides the implementation of a method of customizing and hardening the installed operating system.

The hardware required for the Jumpstart server was outlined earlier in this document as was the software required. In addition to the software and hardware, also required is an isolated subnet. The isolated subnet is required to protect the install client and the jumpstart server from network based attacks.

The creation of a Jumpstart server is not necessarily trivial from the description given in the Solaris™ 8 Advanced Installation Guide[22]. For this reason, the creation of the simplest possible Jumpstart server is outlined below along with the procedure for installing the JASS™ toolkit (Much of this is a consolidation of the information in the Solaris 8 Advanced Installation Guide[23]) :

1. Complete install of Solaris 8 on the system, using the "Entire Distribution" software group. (Since the system will be on an isolated subnet, the installation of unneeded packages is not a concern here.)
2. Install the latest recommended patches[24].
3. Install the Sun C compiler or GNU[25] C compiler[26] on the system.
4. Obtain the JASS toolkit tar file from Sun[27]. The version used here is 0.3.3.
5. Obtain the FixModes tar file from Sun[28].
6. Untar the FixModes tar file and recompile it for Solaris x86 (The tar file contains source and SPARC binaries).
7. Rebuild the FixModes tar file with the new Solaris x86 binaries.
8. Move the JASS toolkit tar file and FixModes tar file onto the system.
9. Log in as root on the system.
10. Change directories to /
11. Untar the JASS toolkit tar file. This will create the directory jass-0.3.3.
12. Move the jass-0.3.3 directory to /jumpstart
13. Verify that the /jumpstart directory has permissions set to 0755
14. Change directories to /jumpstart/Drivers.
15. Copy the file user.init.Sample to user.init.
16. Replace 192.168.11.33 in the definition of JASS_PACKAGE_MOUNT and JASS_PATCH_MOUNT in user.init with the IP address of the Jumpstart server (**JS-IP**).
17. Copy the FixModes.tar file into /jumpstart/Packages.
18. In the directory /jumpstart/OS, create a directory called Solaris_8_2001-10
19. In the directory /jumpstart/OS/Solaris_8_2001-10, create two subdirectories, x86 and sparc.
20. Insert the Solaris 8 Disk 1 for x86 into the CD-ROM drive.
21. Volume manager will mount the disk as /cdrom/sol_8_1001_ia/s0 and /cdrom/sol_8_1001_ia/s2.
22. Change directories to /cdrom/sol_8_1001_ia/s2/Solaris_8/Tools.

23. Copy the contents of the CD-ROM with the command: "./setup_install_server /jumpstart/OS/Solaris_8_2001-10/x86".
24. After the copy of the first installation CD completes, change directories to / and eject the cdrom with "cd /;eject cdrom".
25. Insert the Solaris 8 Disk 2 for x86 into the CD-ROM drive.
26. Volume manager will mount the disk as /cdrom/sol_8_1001_ia_2.
27. Change directories to /cdrom/sol_8_1001_ia_2/Solaris_8/Tools
28. Copy the contents of the CD-ROM with the command: "./add_to_install_server /jumpstart/OS/Solaris_8_2001-10/x86"
29. Change directories to / and eject the cdrom with the command : "cd /;eject cdrom".
30. If creating a jumpstart server that supports SPARC, insert the Solaris 8 Disk 1 for SPARC into the CD-ROM drive. Otherwise skip to step 40.
31. Volume manager will mount the disk as /cdrom/sol_8_1001_sparc/
32. Change directories to /cdrom/sol_8_1001_sparc/Solaris_8/Tools.
33. Copy the contents of the CD-ROM with the following command: "./setup_install_server /jumpstart/OS/Solaris_8_2001-10/sparc"
34. Change directories to / and eject the cdrom with the command : "cd /;eject cdrom".
35. Insert the Solaris 8 Disk 2 for SPARC into the CD-ROM drive.
36. Volume manager will mount the disk as /cdrom/sol_8_1001_sparc_2/
37. Change directories to /cdrom/sol_8_1001_sparc_2/Solaris_8/Tools.
38. Copy the contents of the CD-ROM with the following command: "./add_to_install_server /jumpstart/OS/Solaris_8_2001-10/sparc"
39. Change directories to / and eject the cdrom with the command : "cd /;eject cdrom".
40. Obtain the recommended patch cluster for x86 and SPARC from SUN[29].
41. Move the patch clusters to the install server and unzip in the cluster files in the directory /jumpstart/Patches/. Two directories will be created in the /jumpstart/Patches directory, 8_Recommended and 8_x86_Recommended, corresponding to the SPARC and x86 patch clusters respectively.
42. Add the following to the /etc/dfs/dfstab file: "share –F nfs –o ro,anon=0 /jumpstart".
43. Share the /jumpstart directory with the command "shareall"
44. Verify that the necessary NFS and NFS related daemons are running by running the command "rpcinfo –p". Mountd, nlockmgr, nfs and the status services should be available.
45. Configure the Jumpstart server as a "Profile" server by adding the following line to the beginning of the /etc/bootparams file : "* install_config=**JS-IP**:/jumpstart", where "**JS-IP**" is the ip address of the Jumpstart server.

At this point, a "trivial" Jumpstart server has been created, what needs to be done now is configuring the server to accept Jumpstart clients and to customize the server for local needs.

Configuring for the SMB server install client

The Sysidcfg file.

In this section the Jumpstart server is configured to accept the SMB server installation client. The first step is to specify identification information[30] for the SMB server, this is accomplished with a "sysidcfg" file. In the file system namespace created by the JASS toolkit, a sysidcfg file is placed in the /jumpstart/Sysidcfg/Solaris_8 directory tree. For this server, the sysidcfg file was placed in the directory /jumpstart/Sysidcfg/Solaris_8/smb/. The name of the file is, by requirement, sysidcfg. The contents of the sysidcfg file are in Tables 1 and 2

Table 1 Sysidcfg contents

```
name_service=DNS {
     domain_name=DNIn00
     name_server=DNS00-IP,DNS01-IP,DNS02-IP
     search=DNIn,DNOut,DNSite
}
network_interface=iprb0 {
     hostname=HOSTNAME
     default_route=GATEWAYOut
     ip_address=INSTALL-IP
     netmask=MASKOut
     protocol_ipv6=no
}
security_policy=NONE
system_locale=C
timezone=US/Eastern
timeserver=localhost
```

Table 2 Sysidcfg (cont'd) Hardware dependent part

```
keyboard=HOSTNAME:msus101 {
     dev=/dev/vt00
     kbdfile=msus101.kbd
     kdbChksum=0x96185083
}
display=HOSTNAME:SUNWi810 {
     ddxHandler=ddxSUNWi810.so.1
     ddxInitFunc=SUNWi810Init
}
pointer=HOSTNAME:ps23b {
     csize=0
```

```
        buttons=3
        strmod=vuid3ps2
        ddxInitFunc=ddxSUNWmouseProc
        ddxHandler=ddxSUNWx86mouse.so.1
        dev=/dev/kdmouse
        ptrfile=ps23b.ptr
        PtrChksum=0x45cafdaa
    }
    monitor=HOSTNAME:i810-4 {
        device=SUNWi810
        res=1024x768
        defdepth=8
        size="17-inch (43cm)"
        board=intel/i810-4.xqa
        monitor=mfreq/mfreq56.vda
        dpix=75
        dpiy=75
        DisplayChksum=0xfe437a4b
        hz=70
        dcm=Adapter
    }
```

The definitions of the names in bold face were given previously in the Sanitized Network Configuration section. Table 1 contains the sysidcfg information that is independent of the hardware in use, with the exception of the iprb0 label in the network_interface section. (The iprb0 corresponds to the Solaris designation of the first Intel 10/100 Ethernet interface on the system. This is the interface that is on the SMB server motherboard.) The name_service section configures the system for DNS name resolution and creates the /etc/resolv.conf file with the parameters specified between the parethensis. The network_interface section configures the first ethernet interface on the system. Here the hostname, default router, ip address, netmask and IPv6 support (none) options are specified. In this case we use an ip address that is for installation use only. It will be changed after the OS has been installed and the system is rebooted.

Looking further at the entries in Table 1, the (security_policy=NONE) entry specifies Kerberos authentication is not being used. (system_locale=C) specifies that the default location or language is "C". (timezone=US/Eastern) specifies that the time zone is US Eastern standard time. Finally, (timeserver=localhost) specifies that the local clock is correct.

Now a few caveats. Note that the Jumpstart server needs to be in the same IP subnet for this to work. Thus, if the permanent IP addresses are used, then the Jumpstart server and the installation server subnet needs to be configured to look like the subnet where the SMB server will reside. For this reason we use a temporary IP address (**INSTALL-IP**) that is changed later. Next, in this particular configuration, only one interface (iprb0) is configured. The sysidcfg file can specify more than one ethernet port; however, only one

is configured at this time due to limitations in later install scripts that are specific to the data center.

Table 2 contains the Solaris x86 hardware specific information. This is where the keyboard, display, pointer, and monitor used during the system install are specified. As would be expected, these entries are extremely system dependent and are for x86 installs only. These entries were obtained from an identical system already running Solaris 8 x86 using the kdmconfig command with the "-d" option. Without these entries, the install process will query for the parameters at install time.

The /etc/ethers, /etc/hosts and /etc/bootparams files.

At this point there is sufficient information to configure /etc/ethers, /etc/hosts and /etc/bootparams files on the Jumpstart server for the SMB client. The /etc/ethers file on the Jumpstart install server is used to specify the mapping between ethernet MAC address and IP address. When the SMB server (before the OS is installed) is initially booted using the Solaris 8 Installation CD disk 1, it sends a RARP request to determine its IP address[31]. The rarpd server (/usr/sbin/in.rarpd) on the Jumpstart server responds to the RARP request with the IP of the SMB server.

The mapping between ethernet MAC address and IP address is specified in the /etc/ethers file and /etc/hosts. The /etc/ethers file contains the mapping between MAC address and hostname. The format of the file is line oriented, one host/MAC address pair per line, with the MAC address specified first, e.g. 00:60:08:bf:cc:cc, followed by a space and the host name.

```
ETHERNET-MAC HOSTNAME
```

The /etc/hosts file on the Jumpstart server contains the mapping between hostname and IP address. Again, the format is line oriented, one host/IP address pair per line with the IP address first, e.g., 192.168.1.1, followed by a white spaces and the hostname.

```
INSTALL-IP HOSTNAME
```

Once the SMB server has obtained its IP address, it needs to find its sysidcfg file and the OS boot and install images. This information is obtained from the /usr/sbin/rpc.bootparamd daemon that is running on the Jumpstart server. The bootparamd server finds this information in /etc/bootparams. For the SMB client, the bootparams entry is

```
HOSTNAME \ root=JS_IP:/jumpstart/OS/Solaris_8_200110/x86/Solaris_8/Tools/Boot \
install=JS_IP:/jumpstart/OS/Solaris_8_2001-10/x86 \
boottype=:in \
sysid_config=JS_IP:/jumpstart/Sysidcfg/Solaris_8/smb \ rootopts=:rsize=8192
```

Because of the format limitations of this document, \ denotes a line break needed for formatting purpose only, in the bootparams file, the entry for a host is all on one line.

All three of the above files and the /tftpboot directory are managed with the add_install_client command, found in /jumptart/OS/Solaris_8_2001-10/x86/Solaris_8/Tools/. The complete command line to add the SMB install client is:

```
./add_install_client  \
     -i INSTALL-IP \
     -e ETHERNET-MAC \
     -s JS_IP:/jumpstart/OS/Solaris_8_2001-10/x86 \
     -p JS_IP:/jumpstart/OS/Sysidcfg/Solaris_8/smb \
     HOSTNAME i86pc
```

Note that after using this command, the /etc/bootparams file needs to be edited to replace any reference to the hostname of the Jumpstart server with the Jumpstart servers ip address.

Initial configuration of the OS image.

With most of the mechanics of the Jumpstart server in place, the next step is to set the configuration of the OS image that will be installed with a "profile" file[32] on the Jumpstart server. The profile file, in this case named smb_prof, is placed in the directory /jumpstart/Profile/. The contents of the profile file for the SMB server is given in Tables 3 and 4.

Table 3 Profile file contents

```
     install_type initial_install
     system_type standalone
     fdisk c0t0d0 0x0A delete
     fdisk c0t0d0 solaris delete
     fdisk c0t0d0 dosprimary delete
     fdisk c0t0d0 solaris maxfree
     partitioning explicit
     filesys c0t0d0s1 2048 swap
     filesys c0t0d0s0 free /
```

Table 4 Profile contents (cont'd)

```
     cluster SUNWCreq
     cluster SUNWCadm add
     cluster SUNWCfwcmp add
     cluster SUNWCfwutil add
```

```
cluster SUNWCfwshl add
cluster SUNWClp add
cluster SUNWCnca add
cluster SUNWCntp add
cluster SUNWCacc add
package SUNWami add
package SUNWamir add
package SUNWast add
package SUNWmfrun add
package SUNWgssdh add
package SUNWgssc add
package SUNWgss add
package SUNWj2rt add
package SUNWrsg add
package SUNWdoc add
package SUNWman add
package SUNWmp add
package SUNWctpls add
package SUNWsutl add
package SUNWlibC add
package SUNWlibCf add
package SUNWtltk add
package SUNWfns add
package SUNWfnsx5 add
package SUNWgssk add
package SUNWrsgk add
package SUNWtoo add
package SUNWsprot add
locale en_US
```

Table 3 contains the information in the profile file that describe the install type and the configuration of the root disk. The `install_type` entry specifies that the installation will be a fresh install of the OS, not an upgrade of an existing image. The `system_type` entry specifies that the SMB server will be a standalone system, not a diskless client or a dataless client. The remaining entries specify the layout of the root disk.

The first three `fdisk` entries attempt to delete all existing partitions on the system. The fourth `fdisk` entry creates a single Solaris partition on the hard disk, using all the free space on the disk. The partitioning explicit entry specifies that the Solaris partition will be explicitly partitioned. The `filesys` entries partition the Solaris partition into a 2GB swap partition and a root (/) partition taking up the rest of the free disk space in the Solaris partition.

Note that a single partition plus a dedicated swap partition removes some protection that might be gained by using separate partitions for subdirectories like /usr and /var. Given the size of the disks that are available and the fact that a denial of service from disk resource exhaustion is considered to be acceptable, a single partition is used.

Table 4, containing the rest of the profile file, specifies the OS software that will be installed on the system. The cluster and package entries specify the OS packages that will be installed on the system. The first cluster entry, SUNWCreq, is the installation "metacluster" that corresponds to the "Core Solaris Software Group", the minimum installation that is supported by the installation software. (Contents of the SUNWCreq metacluster can be found in the .clustertoc file in /jumpstart/OS/Solaris_8_2001-10/x86/Solaris_8/Product directory).

The SUNWCadm is the System and Network Administration cluster that contains administrative tools, in particular the "showrev" command necessary for the patchdiag and patchcheck tools that examine the patches installed on the system. The SUNWCfwcmp, SUNWCfwutil, and SUNWfwshl packages contain third party compression tools, other utilities and shells that are of use. For a truly minimal system, these packages can be deleted. In this case, they are installed to provide the base environment that is expected by other system administrators at the facility.

The SUNWClp and SUNWmp packages contain the software required for printer services (needed to support SMB based printing). The SUNWCnca is the Network Cache and Accelerator software, in particular nscd. Although this is considered to be a security risk according to the SANS Solaris Security Step by Step Guide[33], it is installed anyway, albeit with some configuration changes, to avoid any problems that might occur if ncsd were not installed. The SUNWCntp package contains the Network Time Protocol software that is used to synchronize the system clock to the time servers at the data center. The SUNWCacc package contains the system accounting software that is used to monitor and log processes that are run on the system.

The SUNWami, SUNWamir, SUNWgssdh, SUNWgssc, SUNWgss SUNWrsg, SUNWfns, SUNWfnsx5, SUNWgssk, and SUNWrsgk are packages that contain software that may be of use if changes are made to the authentication system on the SMB server (for example GSS or Kerberos enable authentication systems). The SUNWast package contains the ASET (Automated Security Enhancement Tool) software that can be used to harden the OS configuration and check the integrity of installed software. (Although installed, the ASET tool is currently not utilized due to time contraints in the rollout of the server.)

The SUNWj2rt package is the Java runtime environment, which is a dependency of some of the other packages that are installed. SUNWmfrun is the Motif runtime, that is also another dependency. The SUNWdoc and SUNWman packages contain the tools and files for on line man pages. SUNWtltk is the Tooltalk runtime that is required by other installed packages. SUNWctpls is another package (Portable layout services for Complex Text Layout support) that is required by other installed packages. SUNWlibC and

SUNWlibCf are the Sun Workshop C and Cfront libraries libraries that may be required compiled applications (e.g. SUNWlibC is required by SUNWfns).

The SUNWsutl, SUNWtoo and SUNWsprot packages contain additional software most notably, some statically linked utilities, diagnostic tools (truss) and other tools (make) that may be of use. These three packages are not really required but are added anyway. One package that was not installed, that probably should have been installed is SUNWter which contains terminal information.

Finally, the `locale` entry in the profile file specify that the default locale is en_US.

Local configuration and JASS hardening.

With the installation of the OS image specified, the next step is making local changes to the system configuration and hardening the system. As with any process that attempts to fully automate and configure a system, the extent to which the complete procedure can be automated is an example of the law of diminishing returns. Due to time constraints on the rollout of the SMB server, the number of automated configuration changes were kept to a minimum. Therefore, configuration changes made "by hand" were still required. The framework for making the automated local configuration changes and hardening the system is the JASS framework. As a result, it is best to discuss the basic JASS components that were used to harden the systems, and then discuss the changes that were made to these components to make the local configuration changes. Note that most if not all of the information about the JASS toolkit components is found in the Toolkits internals document[34].

The JASS toolkit works through the "finish" script mechanism that is provided by the Jumpstart installation technology. These finish scripts run after the operating system has been installed on the SMB server. The toolkit uses the "driver" finish script to run a set of other scripts that make changes to the installed system to "harden" it. Several example driver scripts are found in the /jumpstart/Drivers directory, corresponding to different system configurations. The basis of the SMB server configuration are in the desktop-config.driver, desktop-hardening.driver and desktop-secure.driver found in the /jumpstart/Drivers directory. The desktop-secure.driver is the main driver script, which in turn runs the desktop-config.driver and desktop-hardening.driver. For the SMB server, the driver scripts were smb-secure.driver, smb-config.driver, smb-hardening.driver and smb-package.driver. In what follows, the smb-*.driver files are discussed. However, the discussion is limited to the differences between the smb-*.driver files and the desktop-*.driver files, the effects of the smb-*.driver files that are relevant to the configuration of the SMB server, and the end results of the locally developed scripts that are run by the smb-*driver files (This is instead of showing the source code of these locally developed scripts.)

The smb-secure.driver script.

The difference between the smb-secure.driver and the desktop-secure.driver is very minimal. First, the smb-secure.driver calls the smb-config.driver, smb-hardening.driver and smb-package.driver (in that order) instead of the desktop-*.drivers. Second, the smb-secure.driver disables telnet, ftp, and dtspc in inetd.conf, unlike desktop-secure.driver which keeps them enabled. The remaining service in the resulting inetd.conf is rstatd, which is used to monitor the performance of the system. This is yet another security/maintenance-monitoring tradeoff. For a more secure system, rstatd can also be turned off.

The smb-config.driver script.

The smb-config.driver script attempts to add additional configuration information. It copies, using the JASS supplied mechanism, the JASS standard .cshrc and .profile files for root , which among other things issues the umask 022 command to set the default file creation mode bits on files created by root. They also set the core file size for root to 0MB. New /etc/nsswitch.conf, /etc/netmasks, and /etc/auto_master files are copied over to the system using the JASS supplied mechanism. The contents of these files are given below. The root password is set using the JASS supplied set-root-password.fin script. The sendmail.cf file is edited to specify the site specific mailhost using a locally created script. An /etc/inet/ntp.conf file is created (see below) using a locally created script. The /etc/hosts file is edited to add the IP addresses of the NIS servers and an /etc/defaultdomain file is created using a locally created script. Also, the recommended patches are installed using a locally created script that runs faster than the JASS supplied script. (The JASS supplied script can also be used, the two scripts function identically.)

Configuration of /etc/nsswitch.conf

The following contains the most important entries in the /etc/nsswitch.conf file.

```
hosts: files dns
passwd: files nis
group: files nis
netgroup: files nis
```

This reflects the fact that hostnames will be looked up using DNS, whereas the password, group and netgroup entries are looked up first in files and then in the NIS database. The other entries in /etc/nsswitch.conf are set to files. This reflects the fact that only the passwd, group, and netgroup maps are propagated by the local NIS configuration.

Configure /etc/inet/netmasks

The /etc/inet/netmasks file was changed to reflect the netmasks of the two subnets in which the SMB server resides.

```
SUBNETIn00-IP MASKIn00
SUBNETOut-IP MASKOut
```

Configure /etc/auto_master

The /etc/auto_master is set the following to enable automounting of the home directories:

```
/-   auto_direct    -nosuid
```

The corresponding /etc/auto_direct file containing the NFS file systems and their mountpoints is created "by hand". A sample entry of the containing the options used on the SMB server is shown below.

```
/fs1        -rw,nosuid,hard,intr nfsserver.name:/fs1
```

Changes to sendmail.cf

```
DSmailhost.DNSite
```

Configuration of /etc/inet/ntp.conf

```
driftfile /etc/ntp.drift
server NTP00-IP
server NTP01-IP
server NTP02-IP
restrict default notrust nomodify
restrict NTP00-IP nomodify
restrict NTP01-IP nomodify
restrict NTP02-IP nomodify
restrict 127.0.0.1
restrict SMBIn-IP
restrict SMBOut-IP
```

The contents of /etc/inet/ntp.conf specifies the drift file, the NTP servers and access controls for the ntp daemon. The first restrict directive tells ntpd to disallow synchronization with all hosts and disallow runtime reconfiguration requests from all hosts. The next three restrict directive tells ntpd to allow time synchronization with the three NTP servers. The final three directives allow the SMB server to accept runtime reconfiguration requests from itself.[35]

Configuration of /etc/hosts

The /etc/hosts file is edited to add the NIS server IP addresses.

```
127.0.0.1 localhost
SMBIn-IP SMBIn-Name HOSTNAME loghost
SMBOut-IP SMBOut-NAME
#
# NIS Servers
#
NIS00-IP NIS00-Name
NIS01-IP NIS01-Name
NIS02-IP NIS02-Name
```

Note that the final state of the /etc/hosts file is shown, which includes some hand editing, in particular, the setting of the SMBIn and SMBOut entries. After the JASS scripts run, the SMBIn and SMBout entries do not exist. Instead, the mapping of **HOSTNAME** to INSTALL-IP exists.

Configuration of /etc/defaultdomain

```
NISDOMAIN
```

The smb-hardening.driver script.

The smb-hardening.driver script runs the bulk of the JASS hardening scripts. It also runs a handful of locally created scripts. Of the JASS provided hardening scripts, a few were modified to add additional functionality. The following discusses the scripts run by the smb-hardening.driver script and the effects they have on the system. Note, all the scripts described here are supplied by the JASS toolkit unless explicitly stated otherwise. Also the description are taken from the descriptions in the JASS internals document[36].

The following are files that are copied from the /jumpstart/Files directory to the SMB server.

```
/etc/init.d/nddconfig
/etc/init.d/set-tmp-permissions
/etc/issue
/etc/motd
/etc/notrouter
/etc/rc2.d/S00set-tmp-permissions
/etc/rc2.d/S07set-tmp-permissions
/etc/rc2.d/S70nddconfig
/etc/syslog.conf
```

The /etc/issue and /etc/motd files are set to the site recommended text messages, similar to the messages recommended by SANS[37].

The /etc/notrouter file is created which disables routing by the system. This is particularly important on this system since it has to interfaces and can potentially route packets from the corporate subnet into the more secure data center subnets.

The nddconfig file sets network settings to values that are recommended in the Solaris blueprint article Solaris Operating Environment Network Settings for Security[38]. The settings cover those recommended in step 2.3.1 of the Solaris Security Step by Step (SSSS) guide[39]. The S70nddconfig file is actually a symlink to the nddconfig script. The JASS file copy function copies the symlink into the SMB server.

The set-tmp-permissions files are scripts that check the permissions of the /tmp and /var files at various points during the boot sequence to verify that they are correct.

The syslog.conf file that is copied over to the SMB server is shown below.

Contents of /etc/syslog.conf are as follows:

```
*.err; kernel.notice;auth.notic    /dev/sysmsg
auth.info /var/log/authlog
mail.debug     /var/log/syslog
*.alert   root
*.emerg   *

*.info    /var/adm/messages
*.info    @SYSLOG-IP
```

This syslog.conf file sends all info or more serious messages to the Syslog server and the local /var/adm messages file. It also sends auth.info messages to the authlog file as recommended by the SSSS guide[40].

The following are the scripts run by the smb-hardening.driver that explicitly disable services:

disable-ab2.fin – No effect
disable-apache.fin – No effect
disable-asppp.fin – No effect
disable-autoinst.fin
disable-core-generation.fin
disable-dhcpd.fin – No effect
disable-dmi.fin – No effect

```
disable-dtlogin.fin – No effect
disable-keyserv-uid-nobody.fin
disable-ipv6.fin – No effect
disable-ldap-client.fin
disable-mipagent.fin – No effect
disable-nfs-server.fin
disable-nscd-caching.fin
disable-preserve.fin
disable-power-mgmt.fin – No effect
disable-remote-root-login.fin
disable-rhosts.fin
disable-sendmail.fin
disable-slp.fin – No effect
disable-snmp.fin – No effect
disable-spc.fin
disable-syslogd-listen.fin
disable-system-accounts.fin
disable-uucp.fin – No effect
disable-vold.fin – No effect
disable-wbem.fin– No effect
```

The scripts with the phrase "No effect" means that the scripts had no effect on the configuration of the system. The reasons are varied, but most of the time the reason is the functionality was not installed when the OS was loaded.

The disable-autoinst.fin script prevents the startup of the system identification scripts, corresponding to step 2.1.2 of the (SSSS) guide[41].

The disable-core-generation.fin script forces all core files to be zero length, thus preventing core files from being used as a means of gathering information from the system. This is part of step 2.3.5 in the SSSS guide[42].

The disable-keyserv-uid-nobody.fin script starts up keyserv with the –d option, thus preventing user nobody from accessing secure RPC services. Since secure RPC is not used by the SMB server, the keyserv daemon should probably be disabled completely.

The disable-ldap-client.fin scripts disable the LDAP client cache daemon as recommended in step 2.1.6 of the SSSS guide[43].

The disable-nfs-server.fin script prevents the startup of NFS server, corresponding to part of step 2.1.3 in the SSSS guide[44].

The disable-nscd-caching.fin script disable nscd caching of passwd, host, and group information. It does not completely stop nscd services as suggested in step 2.1.5 in the SSSS guide[45].

The disable-preserve.fin script disables recovery of vi buffers as suggested in step 2.1.8 in the SSSS guide[46].

The disable-remote-root-login.fin script prevents root from logging in from places other than the console. However, this setting is NOT honored by Ssh so it is of limited use. There is no SSSS guide step that corresponds to this one.

The disable-rhosts.fin script disables .rhosts authentication in the pam.conf file. Since no BSD based remote services are available, this is of limited use. This is equivalent to step 2.8.4 in the SSSS guide[47].

The disable-sendmail.fin script places sendmail in send only mode. This is equivalent to running sendmail out of cron instead of in daemon mode as specified in step 2.1.7[48] and 2.10.3[49] in the SSSS guide. Note however that with this script sendail continues to run as a daemon. This may be a problem if an unknowing administrator were to "upgrade" sendmail to the stock distribution of sendmail from sendmail.org[50] since the configuration of "send only" mode may be specific to the Sun version of sendmail.

The disable-spc.fin script disables startup of the Solstice Print Client daemon.

The disable-syslogd-listen.fin script prevents syslogd from accepting syslog request from other systems, as suggested in step 2.2.8 of the SSSS guide[51].

The disable-system-accounts.fin script disables login access to several standard system accounts by giving them a shell that logs the login attempt and then logs the user out. This corresponds to step 2.8.2 of the SSSS guide[52].

The following are scripts in smb-hardening.driver that make configuration changes.

enable-ftp-syslog.fin
enable-inetd-syslog.fin
enable-priv-nfs-ports.fin
enable-process-accounting.fin
enable-rfc1948.fin
enable-stack-protection.fin
enable-accounting.fin
install-at-allow.fin
install-ftpusers.fin
install-loginlog.fin
install-newaliases.fin – No effect
install-sadmind-options.fin – No effect
install-security-mode.fin – No effect
install-shells.fin
install-sulog.fin
install-authlog.fin

```
remove-unneeded-accounts.fin
set-banner-ftpd.fin
set-banner-telnetd.fin
set-ftpd-umask.fin
set-login-retries.fin
set-power-restrictions.fin
set-rmmount-nosuid.fin – No effect
set-supath.fin
set-sys-suspend-restrictions.fin
set-system-umask.fin – No effect
set-tmpfs-limit.fin
set-user-umask.fin
update-at-deny.fin
update-cron-allow.fin
update-cron-deny.fin
update-cron-log-size.fin
update-inetd-conf.fin
install-fix-modes.fin
```

The enable-ftp-syslog.fin scripts enables logging of ftp requests via syslog. Since ftpd is disabled, this is of limited use.

The enable-inetd-syslog.fin script enables logging of connections to inetd based services via syslog. There is no equivalent step in the SSSS guide.

The enable-priv-nfs-ports.fin script requires NFS requests to originate from client port numbers less than 1024. This corresponds to step 2.3.6 in the SSSS guide[53].

The enable-process-accounting.fin script enables process accounting.

The enable-rfc1948.fin script enables strong TCP sequence number generation as specified by step 2.11.4 of the SSSS guide[54].

The enable-stack-protection.fin script enables the no executable stack system option as recommended in step 2.3.4 of the SSSS guide[55]. There is some issue as to whether or not this change has any effect on the x86 platform. The Sun Tunable Parameters reference manual suggests that this is valid only for specific SPARC based systems[56].

The enable-accounting.fin script is a locally written script that sets up the system accounting that is recommended in step 2.6.7 and 2.6.8 of the SSSS guide[57].

The install-ftpusers.fin creates the /etc/ftpusers file to disable ftp access to the system from selected system accounts. This is of limited use since ftpd is not enabled on the system.

The install-at-allow.fin script creates an empty at.allow file. It is used with the update-at-deny.fin script to disable access to the at command to selected users.

The install-loginlog.fin script creates the loginlog file to document failed logins as recommended by steps 2.6.3 and 2.6.5 of the SSSS guide[58]. It was locally modified to add an entry in root crontab file to rotate the log using the rotate log script provided in the SSSS guide[59].

The install-shells.fin script creates an /etc/shells file and installs the shells shipped with Solaris 8.

The install-sulog.fin script creates the sulog file to log executions of the "su" command.

The install-authlog.fin script creates the authlog file and a root crontab entry that rotates the log file as suggested by steps 2.6.2 and 2.6.5[60]. It was locally created. This is used in conjunction with previously made changes to the /etc/syslog.conf file.

The remove-unneeded-accounts.fin script removes the listen and nobody4 users from the /etc/password file. This is less aggressive than what is recommended by the SSSS guide[61]. Also, depending on the contents of the NIS maps, this may be ineffective.

The set-banner-ftpd.fin and set-banner-telnetd.fin scripts set the ftp and telnet banners as recommended in steps 2.9.3 and 2.9.2 in the SSSS guide[62].

The set-ftpd-umask.fin sets the umask used when files are ftp'ed. (Again, of limited use since ftp is disabled)

The set-login-retries.fin sets the maximum number of failed login attempts before a login failure is logged in loginlog.

The set-power-restrictions.fin script restricts access to the system power management functions.

The set-supath.fin script is a locally written scripts that sets SUPATH in /etc/default/login which is used to set root's path.

The set-sys-suspend-restrictions.fin script restricts access to system suspend and resume functions.

The set-tmpfs-limit.fin script configures the system to limit the use of swap space by the /tmp file system to 512MB.

The set-user-umask.fin sets the users umask via default .login, profile and other files.

The update-at-deny.fin, update-cron-allow.fin, and update-cron-deny.fin scripts disable access to the at and cron facilities to selected users as recommended by step 2.8.6 of the SSSS guide[63].

The update-cron-log-size.fin script limits the size of the cron log file. When the limit is reached the file will be rotated.

The update-inetd-conf.fin script disables services in inetd.conf. For the SMB server, only rstatd is left in the inetd.conf file.

The install-fix-modes.fin script installs Caspar Dik's fix-modes[64] executables onto the system. The script also executes the fix-modes binary on the system, as is recommended by the SSSS guide[65]. This alters the permission bits on selected files and directories to more secure values.

The smb-package.driver script

The smb-package.driver is a very simple, locally written, shell script that copies a list of files from the /jumpstart/Packages directory to /var/spool/pkg on the SMB server. This driver is used to copy the third software that will be installed on the SMB server, but will not actually install the software. The installation will be accomplished "by hand" after the compete Jumpstart installation process has finished. This is because of the limitations of the packaging of the third party software. All the software is packaged as Solaris pkg[66] format packages. However, since the packages were locally created, they have not been fully qualified for non-interactive installations.

The Rules file

This completes the discussion of the changes made during the Jumpstart process and almost all the components that need to be specified for a Jumpstart install. To complete the configuration of the Jumpstart server, a "rules" file[67] needs to be created so that the Jumpstart server can determine what Jumpstart installation process is used for a given install client. The rules file, located in /jumpstart and named rules, used here is very simple:

```
hostname HOSTNAME - Profiles/rsmb_prof Drivers/smb-secure.driver
```

This rules file simply says, for the install client **HOSTAME** use the profile found in /jumpstart/Profiles/rsmb_prof and the finish script /jumpstart/Drivers/smb-secure.driver. Before actually installing, the rules file needs to be checked. To do this, run the following command while in the /jumpstart directory :

```
/jumpstart/OS/Solaris_8_2001-10/x86/Solaris_8/Misc/jumpstart_sample/check
```

Installation of the SMB server.

To actually do an install, the SMB server needs to be connected to the subnet of the Jumpstart server and booted with the Solaris 8 Installation CD-ROM disk 1 of 2 in the SMB server CD-ROM drive. The system will come up in the device configuration assistant. Successive screens will show up that prompt the user to press F2 to continue. When the screen to choose boot devices is reached, the "Net" option should be chosen, at which point, the SMB server will contact the Jumpstart server and continue the boot process. During the boot process, the system will query for the type of installation, "Interactive" or "Jumpstart". The Jumpstart option should be chosen. From this point on, the process is completely "hands free". At this point, the Installation CD-ROM should be removed from the SMB server. Once the system has installed and rebooted, the process of installing and configuring the third party packages remains.

Third Party software installation

With the base OS loaded and basic hardening accomplished during the install process with the JASS scripts, the remaining task of installing the third party software must be done "by hand". Fortunately, all the software has been precompiled on a software "build" system that is dedicated to building applications. In addition, all the software has been packaged in "pkg" format, as a result, for the most part installation is as simple as pkgadd <pkgname>. The process of compiling the applications and converting them to pkg format is outside the scope of this document. Only the more salient points in building the application are discussed.

Installation of Patchdiag and Patchcheck

The installation of patchdiag and patchcheck was accomplished using the standard Solaris pkgadd command since both tools were converted to Solaris pkgs at our facility. However, they are easily installed from the distribution tar file by untarring them in the location of choice. There is no configuration except for downloading the latest patchdiag.xref file from SunSolve, which must be done whenever the tools are run.

Installation and Configuration of TCP-Wrappers.

The installation of TCP wrappers was accomplished with a standard Solaris pkgadd since it has been converted into a Solaris pkg at our facility. However, building and installing TCP wrappers for the SMB server without the pkg format is relatively simple[68]. Follow the instructions provided in the TCP wrappers distribution to make the binaries. (In this case "make REAL_DAEMON_DIR=/usr/sbin sunos5"). Copy the resulting binaries, include files, and libraries over to the SMB server. The critical component is the configuration of the /etc/hosts.allow and /etc/hosts.deny files on the SMB server.

TCP wrapper configuration consists of the /etc/hosts.allow and /etc/hosts.deny files. The content of the hosts.allow file is as follows:

```
ALL: localhost
ALL: SMBIn-Name
ALL: SMBOut-Name
sshd, sshdfwd-X11: TRIP-Name
sshd, sshdfwd-X11: SYSADM-Name
```

The hosts.deny file is:

```
ALL: ALL
```

Actually, in retrospect, TCP wrappers is not necessary if OpenSsh is linked with static TCP wrapper libraries. In this case the only thing that is necessary is the hosts.allow and hosts.deny files since OpenSsh depends on them. For this specific system, the creation of /etc/hosts.deny and /etc/hosts.allow is handled with the pkgadd installation scripts for TCP wrappers.

Installation of ANDIrand.

ANDIrand[69] is an implementation of /dev/random for Solaris that is required for OpenSsh. The source distribution is set up to completely compile and create a Solaris pkg format binary just by typing make. The resulting ANDIrand.pkg package file is installed on the SMB server by typing "pkgadd –d ANDIrand.pkg"

Installation and configuration of OpenSSL.

The installation of OpenSSL, as with other packages, is accomplished via Solaris pkgadd. The installation from source, without a suitably pkg'ed set of binaries, is relatively straightforward. The documentation associated with OpenSSL discusses compiling the binaries from source. It should be noted that older versions of OpenSSL expose a bug in the Solaris 8 development environment, in particular, problems with the automatic initialization of static variables. Version 0.9.6c works around these problems by explicitly initializing the variables to NULL. There is no configuration of OpenSSL that needs to be done.

Actually, in retrospect, OpenSSL is not required on the SMB server if only static OpenSSL libraries were used to create OpenSSH and Stunnel. In this case, OpenSSL is not required on the SMB server. For this particular SMB server, the OpenSSL pkg was installed because it sets up the file system namespace that was used by the Swat/Stunnel configuration.

Installation and Configuration of OpenSsh

The compilation of OpenSsh is relatively straight forward, consisting of "./configure" followed by a "make". The only things that need to be mentioned is that for this SMB server, the OpenSsh software was built using the ANDIrand /dev/random implementation and the libwrap libraries provided by TCP wrappers. Installation is slightly more complicated if done by hand since multiple config files and "key" files need to be generated and installed. (This is normally handled by the "make install" command.) All of this is covered in the SSSS guide[70]. Since the OpenSsh binaries have been packaged into a Solaris pkg, all the configuration files that are installed by "make install" are installed by pkgadd and the postinstall script that is part of the OpenSsh pkg that has been locally developed. The key part of the OpenSsh installation is the configuration of the sshd_config file.

Contents of /etc/sshd_config

```
Port 22
Protocol 2,1
#ListenAddress 0.0.0.0
#ListenAddress ::
HostKey /etc/ssh_host_key
HostKey /etc/ssh_host_rsa_key
HostKey /etc/ssh_host_dsa_key
ServerKeyBits 768
LoginGraceTime 600
KeyRegenerationInterval 3600
PermitRootLogin yes
#
# Don't read ~/.rhosts and ~/.shosts files
IgnoreRhosts no
IgnoreUserKnownHosts yes
StrictModes yes
X11Forwarding yes
X11DisplayOffset 10
PrintMotd no
KeepAlive yes

# Logging
SyslogFacility AUTH
LogLevel INFO

RhostsAuthentication no
#
# You will also need host keys in /etc/ssh_known_hosts
RhostsRSAAuthentication yes
```

```
# similar for protocol version 2
HostbasedAuthentication no
#
RSAAuthentication yes

# To disable tunneled clear text passwords, change to no
PasswordAuthentication yes
PermitEmptyPasswords no

# Uncomment to disable s/key passwords
#ChallengeResponseAuthentication no

Banner /etc/issue

Subsystem      sftp      /opt/local/src/openssh-3.0.2p1/libexec/sftp-server
```

The key contents of the /etc/sshd_config file are the ones that permit the tripwire server to access the system without a password or passphrase. The "IgnoreRhosts no" entry allows the use of .shosts files. The "RhostsAuthentication no" entry disables .rhosts but keeps .shosts functioning. The "IgnoreUserKnownHosts yes" is used to restrict use of the .shosts file to only the users that can modify /etc/ssh_known_hosts (in this case, root).

For compatibility reasons, the system is still set to accept Ssh protocol 1 in spite of its problems. This would have been phased out if it were not for some infrastructure issues that will be resolved in the near future. With the ListenAddress option, the Ssh daemon could be configured to listen only on the internal interface, providing more protection. However, the decision was made to listen on all interfaces (and prevent inappropriate acess with hosts.deny and the host based firewall) to keep the configuration consistent with other systems that will be built from the jumpstart infrastructure. Finally, note the use of the Banner option to specify the banner used when users connect (Ssh Protocol 2 only).

Contents of /etc/ssh_known_hosts

Getting the Tripwire server to access the system and run nightly tripwire system integrity checks requires that the system have login access to the SMB server. This is accomplished with a combination of the /etc/ssh_known_hosts file and a /.shosts file on the SMB server. The /etc/ssh_known_hosts file is used by the OpenSsh server on the SMB server to verify the identity of the Tripwire server. The contents of the /etc/ssh_known_hosts file is:

```
TRIP-Name <tripwire host public key>
```

Contents of /.shosts

An /.shosts file is created with permissions 0400 to allow the tripwire server to access the system and run tripwire nightly. The contents of the file is:

```
TRIP-Name root
```

Installation and configuration of Stunnel.

The compilation of Stunnel consists of the usual combination of ./configure and make commands. The make install command copies the binaries to the installation directory. For this system, the stunnel software was packaged as a pkg file and installed with pkgadd. There are no configuration files so installation from the source distribution by hand is relatively simple. This package depends on OpenSSL for libraries.

Installation and Configuration of Samba.

The compilation of Samba is the usual combination of the commands ./configure and make. As with all the other packages, it is installed in this system with pkgadd since it has been locally converted to a package. (Note that the pkg system includes installation of /etc/init.d startup scripts that start smbd and nmbd on server boot.) The /etc/printers.conf file was also populated with network accessible printers. The critical part of Samba is getting the security components in the configuration file set properly. (A general discussion on configuring Samba is outside the scope of this document.) The relevant items in the smb.conf file that are of interest are associated with handling the dual interfaces, restricting access and encrypting passwords. The "hosts allow" directive can be used to prevent off site systems from accessing the server. For this SMB server, the hosts allow is set to allow all systems in the corporate subnet , **SUBNETsite-IP**. The "interfaces" directive is used to specify the interface from which the server will accept requests. In this case, only the "outside" interface is set to accept SMB service requests. The `security=user` setting tells SMB to require username/password authentication to access shares. The `encrypt passwords = yes` option informs Samba to require encrypted passwords when clients authenticate with the server. However, for this to work, a local smbpasswd file is required on the server. This latter file is initially created "by hand" and is populated "by hand" with Samba users. Once created, maintenance of passwords is accomplished via the swat program that is part of the Samba distribution.

```
hosts allow = SUBNETsite-IP 127.
interfaces = SMBOut-IP/32
security = user
encrypt passwords = yes
```

Installation and Configuration of the SWAT password management server

Since the swat binary is part of the Samba distribution, installing Samba results in the installation of swat. However, since swat is not run as a stand alone daemon, the inetd server on the SMB server needs to be configured to start up swat. This is done by editing the /etc/services file and the /etc/inetd.conf file.

Add the following entry to /etc/services. This maps the swat port to the numerical TCP port 901.

```
      swat 901/tcp
```

Add the following entry to /etc/inetd.conf

```
swat stream    tcp  nowait    root /usr/sbin/stunnel swat\
-p /opt/local/src/openssl-0.9.6c/certs/stunnel.pem -l\
/usr/sbin/swat
```

As before, the \ represents line breaks that are added only
for formatting purposes. In reality, the entry is one line in the file. The entry assumes that stunnel has been installed in /usr/sbin, the openssl certificate has been installed in /opt/local/src/openssl-0.9.6c/certs/stunnel.pem and Samba binary for swat is in /usr/sbin. This entry allows users to securely change their password via a web browser[71]. The stunnel.pem certificate needs to be obtained from a CA (Certificate Authority).

Installation of second Ethernet interface

The installation and configuration of the second ethernet interface is critical to get straight since the firewall rules are tied to iprb0 and iprb1. The /etc/hostname.iprb0 is set to the inside IP and /etc/hostname.iprb1 is set to the outside IP. /etc/hostname.iprb0 is shown below:

SMBIn-Name

And /etc/hostname.iprb1 is shown below:

SMBOut-Name

Static routes for dual NIC configuration

With dual interfaces and multiple subnets accessible from each interface, a static routing table needs to be set up so that IP packets are routed to the right interface. This is done with a start up script that adds routes to the routing table. The content of the startup script is :

```
#! /sbin/sh
```

```
#
# Static routes
#

case "$1" in
'start')
     echo "Setting static routes for SMB Gateway\n"
     /usr/sbin/route add net SUBNETIn01-IP GATEWAYIn-IP
     /usr/sbin/route add net SUBNETIn02-IP GATEWAYIn-IP
     /usr/sbin/route add net SUBNETIn03-IP GATEWAYIn-IP
     /usr/sbin/route add net SUBNETIn04-IP GATEWAYIn-IP
     echo ""
;;
'stop')
     echo "$0: No stop script"
;;
'*')
     echo "Usage: $0 {start|stop}"
     ;;
esac
```

This startup script needs to be linked into the /etc/rc*.d/ directories. In this case, there is a symlink to the script from /etc/rc2.d/S99static-routes.

Installation and configuration of the host based firewall.

The compilation of ip-filters is trivial for Solaris. Issuing the command "make solaris" builds the binaries. Issuing the command "make package" in the SunOS5 directory will create the pkg. Installing the software reduces to issuing the command "pkgadd –d ipf.pkg". The hard part in installing the host based firewall is setting up the firewall rules.

The firewall helps protect the server from direct assault from the network. The rules used on the firewall are fairly simple and can be tightened. For simplicity, a stateless configuration is used. The firewall configuration file, /etc/opt/ipf/ipf.conf, is shown in Tables 5, 6, 7, 8, and 9. Note that the configuration file for ip-filter is a little counter intuitive in that processing is not done on a first match basis unless the rule specifies "quick". Instead, the processing is basically the last match.

Table 5 Rules for all interfaces

```
#
# The following are the firewall rules
#
block in log quick from any to any with ipopts
block in log quick proto tcp from any to any with short
```

The rules in Table 5 immediately reject all packets with ip options set and any tcp packets that are deemed to be "malicious" short fragments[72].

Table 6 iprb0 outgoing rules

```
#
# iprb0 (inside) branch, outgoing
#
pass out on iprb0 all head 250
block out all group 250
pass out from SMBIn-IP/32 to SUBNETIn00-IP/MASKIn00 group 250
pass out from SMBIn-IP/32 to SUBNETIn02-IP/MASKIn01 group 250
pass out from SMBIn-IP/32 to SUBNETIn02-IP/MASKIn02 group 250
pass out from SMBIn-IP/32 to SUBNETIn03-IP/MASKIn03 group 250
pass out from SMBIn-IP/32 to SUBNETIn04-IP/MASKIn04 group 250
```

Table 6 are the rules governing outbound packets from the inside interface, iprb0. The rules state that all outbound packets are dropped by default unless they have the source IP address of the SMB server and a destination adddress that is in one of the internal subnets.

Table 7 iprb0 incoming rules

```
#
# iprb0 (inside) branch, incoming
#
pass in on iprb0 all head 200
block in all group 200
pass in from SUBNETIn00-IP/MASKIn00 to SMBIn-IP/32 group 200
pass in from SUBNETIn01-IP/MASKIn01 to SMBIn-IP/32 group 200
pass in from SUBNETIn02-IP/MASKIn02 to SMBIn-IP/32 group 200
pass in from SUBNETIn03-IP/MASKIn03 to SMBIn-IP/32 group 200
pass in from SUBNETIn04-IP/MASKIn04 to SMBIn-IP/32 group 200
```

Table 7 are the rules governing inbound packets to the inside interface, iprb0. The rules state that all inbound packets are dropped by default unless they have a source IP address that is in one of the internal subnets and they have a destination address that is the ip address of the internal interface to the SMB server.

Table 8 iprb1 outgoing rules

```
#
# iprb1 (outside) branch, outgoing
#
pass out on iprb1 all head 350
pass out quick on iprb1 proto icmp from SMBOut-IP/32 to any group 350
pass out quick from SMBOut-IP/32 port=137 to SUBNETsite-IP/MASKSite group 350
pass out quick from SMBOut-IP/32 port=138 to SUBNETsite-IP/MASKSite group 350
pass out quick from SMBOut-IP/32 port=139 to SUBNETsite-IP/MASKSite group 350
```

```
pass out quick from SMBOut-IP/32 port=901 to SUBNETsite-IP/MASKSite group 350
pass out quick from SMBOut-IP/32 to SUBNETsite-IP/MASKSite port=137 group 350
pass out quick from SMBOut-IP/32 to SUBNETsite-IP/MASKSite port=138 group 350
pass out quick from SMBOut-IP/32 to SUBNETsite-IP/MASKSite port=138 group 350
pass out quick from SMBOut-IP/32 to SUBNETsite-IP/MASKSite port=515 group 350
pass out quick from SMBOut-IP/32 to any port = 25 group 350
block out all group 350
```

Table 8 shows the firewall rules for outgoing packets from the external interface. The first pass out quick directive allows ICMP packets to leave the external interface. The next three pass out quick directives allow SMB response messages from the SMB server to reach desktops in the corporate subnet. The fifth pass out quick directive allows SWAT responses to reach desktop systems. The sixth, seventh and eighth pass out quick directives allow the SMB server to send packets to other netbios servers. The ninth pass out quick directive allows the system to send packets to printers in the corporate subnet. The final pass out quick directive allows the SMB server to send packets to any SMTP server. The block out directive blocks remaining outgoing packets that do not satisfy any of the previous directives.

Table 9 iprb1 incoming rules

```
#
# iprb1 (outside) branch, incoming
#
pass in on iprb1 all head 300
pass in quick on iprb1 proto icmp from any to SMBOut-IP/32 group 300
pass in quick from SUBNETsite-IP/MASKSite to SMBOut-IP/32 port = 137 group 300
pass in quick from SUBNETsite-IP/MASKSite to SMBOut-IP/32 port = 138 group 300
pass in quick from SUBNETsite-IP/MASKSite to SMBOut-IP/32 port = 139 group 300
pass in quick from SUBNETsite-IP/MASKSite to SMBOut-IP/32 port = 901 group 300
pass in quick from SUBNETsite-IP/MASKSite port = 137 to SMBOut-IP/32 group 300
pass in quick from SUBNETsite-IP/MASKSite port = 138 to SMBOut-IP/32 group 300
pass in quick from SUBNETsite-IP/MASKSite port = 139 to SMBOut-IP/32 group 300
pass in quick from SUBNETsite-IP/MASKSite port = 515 to SMBOut-IP/32 group 300
pass in quick from any port = 25 to SMBOut-IP/32 group 300
block in all group 300
```

Table 9 shows the firewall rules for incoming packets to the external interface. The first pass in quick directive allows ICMP packets to enter the external interface. The next three pass in quick directives allow SMB messages from the desktops in the corporate subnet to reach the SMB server. The fifth pass in quick directive allows SWAT messages to reach the SMB server. The sixth, seventh and eighth pass in quick directives allow the SMB server to receive packets from other netbios servers. The ninth pass in quick directive allows the system to receive packets from printers in the corporate subnet. The final pass in quick directive allows the SMB server to receive packets from a SMTP server (This is the return channel of a TCP connection when the SMB server is sending

mail to the SMTP server). The block in directive blocks remaining inbound packets that do not satisfy any of the previous directives.

As is evident from the description of the firewall contents, it is clear that a significant amount of tightening can be applied. The first pass would be to isolate access on the internal interface to the NFS servers, system administration server, the tripwire server, and the NIS/DNS/NTP server. This would significantly enhance the security of the server by isolating the system from the interactive linux systems. However, as mentioned before, more tempting and obvious targets are available to intruders if they have reached the Linux farm inside the firewall. A second pass would be to tighten the rules with respect to sendmail to include only the corporate mailhost. A third pass would be to tighten the rules for printer access.

Configuration for Serial Console support.

In order to configure the system to use the serial port as a console, the following commands were issued at the command line:

```
eeprom output-device=ttya
eeprom input-device=ttya
```

Log rotation script

This is the log rotation script given in the appendix of the SSSS guide[73]. This was installed onto the system via a Solaris pkg.

Tripwire run.

Configuration and installation of tripwire are covered in the SANS "UNIX Security Tools & Their Uses"[74] and in the documentation that comes with the tripwire source. As with the other software installed here, the tripwire binaries and configuration files were packed into Solaris pkg format so all that was done here was a pkgadd. With tripwire installed, it is time to run tripwire to create a database of checksums of the installed files. This is done with ./tripwire –initialize. The resulting tripwire database is moved to the tripwire server, where it is archived and used nightly to run integrity checks on the SMB system

Archive of selected files

Since the SMB server contains no data, there is very little that cannot be recreated. Given the rapid reinstall capabilities, less than 2 hours from a clean system, the decision was made to only archive a few selected files, in particular the following files are archived.

- /etc/ssh_host_key and /etc/ssh_host_key.pub

- /etc/ssh_host_dsa_key and /etc/ssh_host_dsa_key.pub
- /etc/ssh_host_rsa_key and /etc/ssh_host_rsa_key.pub
- The smbpasswd file.
- The stunnel.pem certificate.
- The smb.conf file.

The closest thing to a proper backup of the system is a copy of the entire root disk to the second disk drive that is attached to the system. (This can either be done with a dd or a ufsdump and ufsrestore to the second disk. The command "dd if=/dev/rdsk/c0t0d0p0 of=/dev/rdsk/c1t0d0p0 bs=1024k" will do the job.) Note that all the third party software is archived on the data center "software build" system. As a result, there is no need to archive them on the SMB server. (Experience at the data center is that the server is completely reloaded from scratch every 6 months, so rapid building of a new server is considered to be more important than being able to recreate an existing system.) However, it should be noted that without consistent and up to date archives of the system, forensic analysis (i.e., post intrusion investigation) becomes more difficult.

At this point, the system is ready to be placed on the production networks. Once on the network, ypinit –c needs to be run to enable NIS service. Note that in the initialization process the NIS servers are specified, so the system will accept NIS information from only the NIS servers. With NIS configured, tripwire needs to be run again to update the database and the database needs to be copied to the tripwire server.

At this point, the system if ready for production.

JASS and SAN Solaris Security Step by Step

At this point, a quick comparison between JASS and the SAN Solaris Security Step by Step Version 2.0 guide is provided. This is done by listing the important Solaris Security Step by Step steps and the corresponding JASS/Jumpstart step.

Solaris Security Step by Step step	**Jumpstart/JASS equivalent**
1.3.1 Set root password	set-root-password.fin
1.3.2 /etc/defaultrouter	set by Jumpstart sysidcfg file
1.3.3 /etc/notrouter	set by JASS
1.3.4 /etc/resolv.conf	set by Jumpstart sysidcfg file
1.3.5 /etc/nsswitch.conf	set by Jumpstart and JASS
2.1.2 Auto Config	disable-autoinst.fin
2.1.3 NFS	disable-nfs-server.fin disable-nfs-client.fin (not used here)
2.1.4 RPC	disable-rpc.fin (not used for this system)
2.1.5 nscd	disable-nscd-caching.fin (Only selected caching is disable not ncsd itself)
2.1.6 LDAP	disable-ldap-client.fin

2.1.7 Sendmail	disable-sendmail.fin
2.1.8 Preserve	disable-preserve.fin
2.2.1 umask	Default in Solaris 8
2.2.2 newinetsvc	JASS inetsvc (not used here)
2.2.3 devfsadm	No equivalent
2.2.7 syslog	disable-syslog-listen.fin
2.3.1 netconfig	JASS nddconfig
2.3.4 stack protection	enable-stack-protection.fin
2.3.5 user resource limits	No equivalent
2.3.6 NFS ports	enable-priv-nfs-ports.fin
2.4.1 NFS config files	No equivalent
2.4.2 Empty crontabs	No equivalent
2.5.1 mount options	No equivalent
2.5.2 root mount options	No equivalent
2.5.3 logging	No equivalent
2.5.4 rmmount.conf	set-rmmount-nosuid.fin
2.6.1 syslog.conf	JASS syslog.conf
2.6.2 authlog	No equivalent (Locally written script used)
2.6.3 loginlog	install-loginlog.fin
2.6.5 root crontab	No equivalent
2.6.6 /etc/default/cron	Default in Solaris 8
2.6.7 /etc/init.d/perf	No equivalent (Locally written script used)
2.6.8 sys crontab	No equivalent (Locally written script used)
2.7.1 Enable BSM	enable-bsm.fin
2.7.1 Config BSM	JASS config files (Not used here)
2.8.1 /etc/passwd	remove-unneeded-accounts.fin
2.8.2 null shell	disable-system-accounts.fin
2.8.3 ftpusers	install-ftpusers.fin
2.8.4 pam.conf	disable-rhosts.fin
2.8.5 empty files	No equivalent
2.8.6 at/crontab	install-at-allow.fin update-at-deny.fin update-cron-allow.fin update-cron-deny.fin
2.9.1 issue/motd	JASS issue/motd
2.9.2 telnetd	set-banner-telnetd.fin
2.9.4 ftpd	set-banner-ftpd.fin
2.9.5 oem-banner	No equivalent
2.10.2 sendmail crontab	disable-sendmail.fin
2.11.1 eeprom security	install-security-mode.fin
2.11.2 /etc/default/login	set-login-retries.fin (similar) set-user-umask.fin
2.11.4 inetint	enable-rfc1948.fin
2.11.5 password expire	set-user-password-reqs.fin
2.11.6 /etc/inittab	No equivalent

As can be seen from the table, a substantial amount of what is recommended in the Solaris Security Step by Step Guide Version 2.0 is covered in some way by the JASS toolkit (Version 0.3.3). However, for the installation of the SMB server, some of the hardening scripts that are in the JASS toolkit were not run.

Ongoing maintenance

For operating system updates, the include patchdiag and patchcheck tools from Sun (Available to support contract customers only.) are used to determine the patch "level" of the system. Unfortunately there is no process in place that automatically downloads the latest patchdiag.xref file (This file contains a database of released patches) and runs the patchdiag commands to see what patches need to be installed. Thus, the process must be done by hand. The installation of required patches is definitely a manual process. An issue to be resolved is the need to verify the JASS configuration after a patch install. A cursory reading of the JASS documentation suggests that it is capable of being used in this mode. However, insufficient time was available to verify this and document the procedure.

Assuming sufficient manpower, the appropriate maintenance procedure for the system would be the following:

- Subscription to Sun's Security Bulletin mailing list to obtain Security bulletins directly from Sun's Security Coordination Team[75].
- Weekly looks at the SunSolve Recommended and Security Patches web page[76] for updated patches.
- Weekly download of patchdiag.xref patch database and run of patchdiag or patchcheck.
- Weekly looks at the Samba web page[77] for updates to Samba.
- Weekly look at the Stunnel web page[78] for updates to Stunnel.
- Weekly look at the OpenSSL web page[79] for updates to OpenSSL.
- Weekly look at the OpenSsh web page[80] for updates to OpenSsh.
- Weekly look at the IP-Filter web page[81] for updates to IP-Filter.
- Weekly look at the ANDIrand web page[82] for updates to /dev/random.
- Daily look at the nightly tripwire logs.
- Daily look at the output of syslog.
- Daily look at the SMB log files and log directory.
- Continuous monitoring of the rstatd output. (via tools like perfmeter)
- With sufficient time and effort, additional tightening can be achieved. The most obvious place to start is with the firewall rules.
- Cross training of additional administrator (assuming any are available) on all the processes needed to recreate the system.

For updates of the third party software, the procedure would be to compile the application, build the Solaris pkg, install and test on a test system and then roll out on the production server with pkgrm/pkgadd.

For intrusion detection, the nightly tripwire scans from the tripwire server are the main line of defense. Tripwire scans are issued from a cron job on the tripwire server and send the results to all relevant administrators at the facility. Note that the tripwire database file that is used is stored on the Tripwire server, not the SMB server.

```
cat database | ssh root@SMBIn-Name 'tripwire – q –loosedir –c /etc/tw.config –d –" |
mailx –s "SMB Server Tripwire" administrators@DNSite
```

Where database is the tripwire database and tripwire and mailx are assumed to be in roots path. Note that with the above command, the integrity checking has been loosened up (loosedir). Also, notice that the tw.config is vulnerable since it resides on the system. A more secure method would be to copy the tw.config file over with the database file.

A secondary line of defense is provided by network profiling being made by the corporate cyber security group. Log file monitoring on the syslog server is currently done manually, however, given the increasing number of systems logging to the syslog server (as servers are replaced with Jumpstart/JASS installed replacements) automating this process will quickly become a high priority task.

Configuration testing and checking.

Rudimentary testing of proper functionality was accomplished by utilizing the SMB server from various Windows based clients. Tasks included reading and writing to SMB shares and accessing printers through the SMB server from Windows desktops. The fact that the system is currently in production without problems is another indication that the SMB server portion of the system is functioning. Nightly tripwire mailings and observed output in the /var/adm/messages file on the Syslog server show that the tripwire and syslog functionality are working. Views of the log files also shows that process accounting are functioning properly. Contents of the /var/log and /var/adm directories show that the log rotation scripts and cron jobs are functioning properly also.

An NMAP[83] port scan of the SMB server's external interface from the external subnet shows that the only ports open are the SMB ports (netbios-*) and the password changing port (samba-swat), as is expected.

```
> nmap SMBOut-Name

Starting nmap V. 2.54BETA30 ( www.insecure.org/nmap/ )
Interesting ports on rsmb00.rhic.bnl.gov (130.199.80.234):
(The 1545 ports scanned but not shown below are in state:
filtered)
```

```
Port        State       Service
137/tcp     closed      netbios-ns
138/tcp     closed      netbios-dgm
139/tcp     open        netbios-ssn
901/tcp     open        samba-swat


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

An NMAP port scan of the SMB server's internal interface from the external subnet (with the hardware firewall in an open configuraton) should show no open ports if the configuration of the host based firewall is correct. Unfortunately it is not possible to run such a scan. However, cursory attempts to access known services on the system from the external subnet fail (e.g., rpcinfo –p **SMBIn-IP**) which suggests that the host base firewall rules are working.

An NMAP port scan from outside the corporate firewall to the external interface should show that there is no open ports if the configuration of the host based firewall is correct. However, available resources and the state of the corporate firewall make this extremely difficult to test.

One way to test the firewall rules more completely would be to set the system up on a test network that duplicates the production topology. In this configuration, it is possible to run NMAP scans on both interfaces from inside the corporate network and from the internet.

Conclusion

In this document, a secure SMB server has been configured that fits within the network and security profile of the organization. With sufficient manpower, additional tightening can be accomplished (removal of NIS, tightening of the firewall, as outlined previously). However, given that the SMB server is a component in a facility, the emphasis was placed on creating an installation and configuration infrastructure that would allow for the rapid deployment of secure systems. This makes it possible to bring all the systems up to a more secure baseline. (The infrastructure has already been used to build and install desktop systems and new Ssh gateway systems).

---

[1] Tridgell, Andrew and the Samba Team, http://www.samba.org.

[2] Sun Microsystems, Solaris 8 Advanced Installation Guide", Sun Microsystems, July 2001, http://docs.sun.com/ab2/coll.780.2/SPARCINSTALL/@Ab2PageView/5?Ab2Lang=C&Ab2Enc=iso-8859-1. p. 28.

[3] Sun Microsystems Enterprise Engineering and Professional Services Team, "Security: Solaris™ Security Toolkit (JASS)", Sun Microsystems, 2002, http://www.sun.com/security/jass.

[4] Sun Microsystems, "Solaris ™ OE Hardware Compatibility List", Sun Microsystems, http://www.sun.com/software/intel/hcl/index.html.
[5] ibid.
[6] Sun Microsystems Enterprise Engineering and Professional Services Team ,op.cit.
[7] Tridgell, Andrew and the Samba Team, "Samba Download", http://us2.samba.org/samba/download.html.
[8] The OpenBSD Project, "OpenSsh", http://www.openssh.com.
[9] Maier, Andreas, "Solaris /dev/random", http://www.cosy.sbg.ac.at/~andi/
[10] Venema, Weitse and Dik, Casper, "TCP Wrappers", ftp://ftp.porcupine.org/pub/security/index.html.
[11] Kim, Gene, ftp://coast.cs.purdue.edu/pub/tools/unix/ids/tripwire/
[12] The OpenSSL Project, "OpenSSL", http://www.openssl.org.
[13] Trojnara, Michal, "Stunnel Universal SSL Wrapper", http://www.stunnel.org.
[14] Reed, Darren, "IP-Filter", http://coombs.anu.edu.au/~avalon/ip-filter.html.
[15] Sun Microsystems, Solaris Patch diagnostic tool, available to Contract Support customers only, http://sunsolve.sun.com.
[16] Sun Microsystems, Solaris Patch diagnostic tool, available to Contract Support customers only, http://sunsolve.sun.com.
[17] Pomeranz, Hal, "Solaris Security Step by Step, Version 2.0", SANS Institute, 2001, p. 36.
[18] Krieger, Markus, "Swat and SSL", http://us2.samba.org/samba/docs/swat_ssl.html.
[19] Sun Microsystems Enterprise Engineering and Professional Services Team, op.cit.
[20] Solaris 8 Advanced Installation Guide", op.cit.
[21] Sun Microsystems Enterprise Engineering and Professional Services Team, op.cit.
[22] "Solaris™ 8 Advanced Installation Guide", op.cit, p.129-182.
[23] "Solaris™ 8 Advanced Installation Guide", ibid.
[24] Available at http://sunsolve.sun.com.
[25] Free Software Foundation, "GNU C compiler", http://www.gnu.org/software/gcc/gcc.html.
[26] Sunfreeware.com, http://www.sunfreeware.com.
[27] Sun Microsystems Enterprise Engineering and Professional Services team, op.cit.
[28] Sun Microsystems Enterprise Engineering and Professional Services team, ibid.
[29] SunSolve Online, Sun Microsystems, http://sunsolve.sun.com/pub-cgi/show.pl?target=home
[30] "Solaris™ 8 Advanced Installation Guide", op.cit, p.47-58.
[31] Stevens, W. R., "TCP/IP Illustrated, Volume 1". Addison-Wesley Publishing Co, 1997, pg . 65.
[32] "Solaris™ 8 Advanced Installation Guide", op.cit, p.152-180.
[33] Pomeranz, Hal, op.cit., p. 7.
[34] Noordergraff, Alex, and Brunette, Glen, "Solaris Security Toolkit – Internals", Sun Microsystems, June 2001, http://www.sun.com/blueprints/0601/jass_internals-v03.pdf
[35] Mills, David, "Notes on setting up a NTP subnet", http://www.eecis.udel.edu/~ntp/ntp_spool/html/notes.htm.
[36] Noordergraff, Alex, and Brunette, Glen, op.cit.
[37] Pomeranz, Hal, op.cit., p. 18.

[38] Watson, Keith and Noordergraf, Alex, "Solaris Operating Environment Network Settings for Security", Sun Microsystems, http://www.sun.com/blueprints/1200/network-updt1.pdf.
[39] Pomeranz, Hal, op.cit., p. 10.
[40] Pomeranz, Hal, ibid., p. 14.
[41] Pomeranz, Hal, ibid., p. 6.
[42] Pomeranz, Hal, ibid., p. 11.
[43] Pomeranz, Hal, ibid., p. 7.
[44] Pomeranz, Hal, ibid., p. 6.
[45] Pomeranz, Hal, ibid., p. 7.
[46] Pomeranz, Hal, ibid., p. 7.
[47] Pomeranz, Hal, ibid., p. 17.
[48] Pomeranz, Hal, ibid., p. 7.
[49] Pomeranz, Hal, ibid., p. 20.
[50] Sendmail Consortium, http://www.sendmail.org.
[51] Pomeranz, Hal, ibid., p. 9.
[52] Pomeranz, Hal, ibid., p. 17.
[53] Pomeranz, Hal, ibid., p. 11.
[54] Pomeranz, Hal, op.cit., p. 21.
[55] Pomeranz, Hal, ibid., p. 11.
[56] Sun Microsystems, "Solaris Tunable Parameters Reference Manual", Sun Microsystems, July 2000, p. 57.
[57] Pomeranz, Hal, ibid., p. 15.
[58] Pomeranz, Hal, ibid., p. 14..
[59] Pomeranz, Hal, ibid., p. 36.
[60] Pomeranz, Hal, ibid., p. 14.
[61] Pomeranz, Hal, ibid., p. 16.
[62] Pomeranz, Hal, ibid., p. 19.
[63] Pomeranz, Hal, ibid., p. 18.
[64] Dik, Caspar, Sun Microsystems, http://www.sun.com/security/jass.
[65] Pomeranz, Hal, ibid., p. 21.
[66] "Solaris 8 Application Packaging Developers Guide", Sun Microsystems, http://docs.sun.com/ab2/coll.45.13/PACKINSTALL/@Ab2TocView?Ab2Lang=C&Ab2Enc=iso-8859-1.
[67] "Solaris™ 8 Advanced Installation Guide", op.cit, p.144-151.
[68] Pomerantz, Hal, "SANS Institute 6.2 UNIX Security Tools & Their Uses", Deer Run Associates, 2001, p. 95.
[69] Maier, Andreas, op.cit.
[70] Pomeranz, Hal, op.cit., p. 24-25.
[71]Krieger, Markus, op.cit.
[72] Reed, Darren, "IP Filter Examples", http://coombs.anu.edu.au/~avalon/examples.html#frags
[73] Pomeranz, Hal, "Solaris Security Step by Step", SANS Institute, 2001, p. 36.
[74] Pomerantz, Hal, "UNIX Security Tools & Their Uses", Deer Run Associates, 2001, p. 128.

[75] Sun Security Coordination Team, "SunSolve Online Security Information", Sun Microsystems, http://sunsolve.sun.com/pub-cgi/show.pl?target=security/sec"
[76] Sun Microsystems, "Patches Recommended and Security Patches", Sun Microsystems, http://sunsolve.sun.com.
[77] Tridgell, Andrew and the Sambas Team, op.cit.
[78] Trojnara, Michal, op.cit.
[79] The OpenSSL Project, op.cit.
[80] The OpenBSD Team, op.cit.
[81] Reed, Darren, op.cit.
[82] Maier, Andreas, op.cit.
[83] NMAP Network Security Scanner, http://www.insecure.org/nmap/.