Global Information Assurance Certification Paper

Building a Secure Sun JumpStart Environment Using the Solaris Security Toolkit, Step-by-Step

Mahrlon Willis
July 2, 2004

GIAC Certified UNIX Security Administrator
Practical Assignment
Option 1, version 2.0

Contents Page

Introduction

Building a secure Sun JumpStart environment using the Solaris Security Toolkit, Step-by-Step.

The primary objective of this practical is to demonstrate the steps required to develop a secure environment for building, configuring and hardening Solaris servers using the Sun JumpStart architecture and the Solaris Security Toolkit.

This document will describe in detail how to build and harden a Solaris system to be used as the foundation of the Sun JumpStart environment. The Sun JumpStart environment is used to build Solaris systems for deployment in a corporate enterprise environment.

The Solaris Operating Environment (OE) is a widely used UNIX operating system that serves as the platform for a variety of application across the Internet. Many corporations have made significant investments in deploying Solaris servers across their enterprises to run mission critical business applications. This document will explain the rationale for using JumpStart to develop a repeatable process for building and hardening Solaris servers in order to better protect the corporation's assets.

This document will provide an overview of the JumpStart concept, its major components and how they are configured, operated and maintained. It will demonstrate how to position the JumpStart server and clients on the network to ensure the integrity of the build process. It will discuss how the JumpStart server itself is made secure using the Security tool kit in stand-alone mode, which is a process that can be used to harden existing servers. It will then show how the toolkit is integrated into the JumpStart environment to be used to harden new servers as they are being built. It will demonstrate the steps required to validate and test the configuration to confirm that the process is working correctly. This document will also demonstrate the necessary steps to maintain the server after it's placed into production.

Intended audience

The intended audience is anyone responsible for administering, building or securing the Solaris OE in an enterprise environment. It is recommended that the UNIX System Administrator (SA) has a minimum of six months to one year experience managing multiple enterprise class Solaris servers in a networked environment. It is assumed that the administrator has experience installing Solaris and third party software, and is comfortable editing configuration files. The individual should also be knowledgeable in UNIX shell scripting.

Server Specifications

Purpose of the Server

The purpose of the JumpStart server is to perform Solaris Operating System installations on Sun hardware. The Sun JumpStart environment is a two-tier client, server architecture made up of a server that provides access to the Solaris software distribution over the network for installation onto a hardware platform (the client). The JumpStart server contains the software distribution, customized installation and configuration scripts and customized configuration files that are used to build and configure a server. In addition the Sun recommended patch clusters and supplemental software packages required for the build process are housed on the JumpStart server. The entire process is managed by editing a series of customization scripts that perform a fully automated, unattended installation of the OS over the network.

Hardware Requirements

The JumpStart server's hardware requirements are relatively simple. The key components are ample storage to house the required versions of the Solaris OS and the respective recommended patch clusters and supplemental software. It is also required that the network segment that the server and clients are on be isolated by a router or firewall to protect the client systems while they are being built. Since the installations all take place over the network a 100 Mbit speed or faster network segment is recommended however it is not required. The server will house the software requirements for Solaris 9, to be used for all new server builds.

The hardware for the JumpStart server is as follows:

Sun Microsystems Netra T1
1 500 MHz UltraSparc II processor
1 Gb RAM
2 18Gb 10,000 RPM SCSI hard drives
1 dual port 10/100/1000 Mbit Ethernet card

Operating System and Supplemental Software Packages

The operating system will be Solaris 9. The Supplemental software packages will include the following:

- **Solaris Security Toolkit v4.0** (a.k.a. JASS, **J**umpStart™ **A**rchitecture and **S**ecurity **S**cripts). The Solaris Security Toolkit is the primary mechanism used to harden the JumpStart server's operating system and all newly built systems prior to deployment. The Sun Security Toolkit is a utility developed by Sun Microsystems for the purpose of securing the Solaris Operating environment. The utility can be downloaded from http://www.sun.com/solutions/blueprints/tools/index.html. The toolkit is made up of a series of scripts and configuration files that can be run on a

system to harden it. The toolkit can be incorporated into the JumpStart architecture and run on a system as its being built or it can be used in stand-alone mode to harden servers that are already in production as well. We will discuss both methods over the course of this document.

- **SunSSH v1.0** – SSH will be used for secure remote management and file transfers. SSH has long been considered the standard for secure remote administration. Effectively replacing the much less secure telnet, FTP and "r" commands (rlogin, rcp, and rsh) whose transmission of passwords in clear text over the network and weak authentication pose a significant risk. Clear text password transmission allows the possibility of someone using a network packet sniffer like TCPdump or snoop to harvest users IDs and passwords. SSH establishes a secure communication channel between the client and server using strong encryption and authentication. SSH can negotiate an encrypted connection using 3DES, and up to 256 bit AES, Blowfish, and twofish algorithms, eliminating the risk of having passwords compromised by eavesdropping on a session connection using a network sniffer. In addition, SSH can provide strong authentication by using native public key exchange and also has the capability to integrate with multi factor authentication methods such as Radius, PKI, SecureID tokens, Smart cards and Kerberos.

- **Sun TCP Wrappers (SUNWtcpd)** – TCP Wrappers will be used to provide additional access control and logging for network services running on the system. TCP Wrappers is a freely available tool developed by Wietse Z. Venema (ftp://ftp.porcupine.org/pub/security/index.html). TCP Wrappers is now included in the Solaris 9 distribution as the SUNWtcpd package by Sun Microsystems. TCP Wrappers runs as the tcpd service daemon, and is used to provide access control and logging for most network services that run as daemons including those spawned from /etc/inetd.conf as stream,nowait. TCP Wrappers does this by interrogating the network connection request destined for the network service daemon it's wrapping, against its /etc/hosts.allow and /etc/hosts.deny access control files. The /etc/hosts.allow file is used to explicitly allow network connections to specified network services, while the /etc/hosts.deny files is used to deny access. A connection request is received by tcpd on behalf of a protected service, and logs the request to SYSLOG. The request is then compared first against the access control files. If the request is allowed it is passed to the appropriate service daemon for processing. If the request is denied it is dropped by tcpd and logged. We will use TCP Wrappers to log all network connections made using tftp and SSH on our trusted network and deny and log all other connection attempts.

- **MD5** – MD5 is an algorithm that takes an arbitrary length input stream and computes a unique 128-bit hash value or digital "fingerprint" of that

stream. The MD5 program is commonly used to create a digital signature for a file in order to validate its integrity. It is used by many Security based utilities such as SSH to validate file transfers and is required in order to use MD5-Tool which we will use to produce a digital hash set of critical system files on our Solaris systems. Sun Microsystems provides a MD5 package for Solaris as a free download from http://www.sun.com/solutions/blueprints/tools/index.html.

- **MD5 Tool v1.1** – Used for data integrity checking, intrusion detection, and change control. The MD5 Tool is a UNIX utility that uses the MD5 hash algorithm to calculate digital signature values for files on a system. It is available for download from http://razor.bindview.com/tools/index.shtml. This tool will allow you to create a MD5 hash signature database of any files on your system and then allow you to monitor them for changes. This will be used to verify the integrity of the Solaris OE Distribution media and other files on our JumpStart server.

- **FixModes** – Used for file permissions control. FixModes is a tool written by Capser Dik of Sun Microsystems and can be downloaded from http://www.sun.com/solutions/blueprints/tools/index.html. It is used to regulate file permissions on a Solaris system by removing group and world write permissions on the files, directories and devices listed in the /var/sadm/install/contents excluding those listed in exceptions.h. This will be used to set and maintain file permissions on the JumpStart server and all newly built systems.

- **CIS Benchmark tool for Solaris v1.2.0** – Used for vulnerability assessment and build testing. The Center for Internet Security provides a series of free tools to benchmark the security posture of Windows and LINUX/UNIX operating systems including Solaris against their standard for security. The Benchmark tool is installed on the host and when executed performs a series of checks against the system to assess how secure it is. It can be downloaded from the CIS website at http://www.cisecurity.org/bench_solaris.html.

Network and System Services

The JumpStart server will be required to run several network services in order to perform its designated role. We will discuss them in detail in this section.

In order to install operating systems on client machines over a network the JumpStart server must have both the tftp (trivial file transfer protocol) and Sun RPC NFS (network file system) service available. In addition the server will also need to have the SSH service available in order to remotely manage the server.

Trivial File Transfer Protocol or tftp is a User Defined Protocol (UDP) and as its name suggests is very simple in design. Its primary function is to transfer data files between two hosts. It does not have the capacity to interrogate filesystems on the source or destination system or even authenticate users or hosts that negotiate a network connection. The tftp service is defined in the /etc/services configuration file. The tftpd daemon is configured in inetd.conf and called by the inetd daemon on Solaris systems.

Secure Shell (SSH) will be required on the JumpStart server as the only means of remote administration access. As stated earlier, SSH is a much more secure alternative to telnet, ftp and shell services. Those traditional services are limited in their native authentication methods and transmit all data in clear text. SSH has long been considered the de-facto standard for secure remote administration and should be considered as an alternative in any situation where confidentiality is a concern. Solaris 9 now includes SunSSH v1.0.

The SSH protocol runs as a network daemon called sshd that listens for network connection requests on TCP port 22. It is initiated by the /etc/rc3.d/S89sshd script and is terminated by the /etc/rc2.d/K03sshd script. SSH can successfully negotiate a connection based on several authentication methods. When the SSH software is installed am RSA and DSA public and private key pair are generated, this key uniquely identifies this host to any client attempting to establish a session. The keys are typically stored in the /etc/ssh, and the private key is named ssh_host_XXX_key and the public key is called ssh_host_XXX_key.pub. Where XXX = dsa or rsa. The files are owned by root and the file access permissions on the private key is read/write by the owner only. The file access permissions on the public key is read/write by the owner and read only by group and world.

The client system maintains a repository of the remote host's public keys in the .ssh/known_hosts file in the user's home directory. If there is no key present for the host the user is prompted to either add it to the host repository, or establish a one-time connection using the key. If the key is added the client will not be prompted when attempting future connections. If the key is not added the user will be prompted again the next time they connect to the server using SSH. Most importantly, if the key presented is different than the key in the repository the user is warned and can terminate the connection attempt. If the key matches the client then generates a random 256-bit session key and sends it to the server to establish encryption and authenticate.

There are two primary configuration files that are used to control the SSH server and client functionality. The /etc/ssh/sshd_config is used to configure the sshd server, and the /etc/ssh/ssh_config is used to configure the client. The JumpStart server will be configured to use ID and password authentication and public key authentication and will use TCP Wrappers for access control.

SunRPC services / NFS (Network File System) will be used as a means to provide client systems access to the OS distribution media and JASS toolkit during the build process. The Sun NFS protocol provides the capability to share data on disk between multiple systems seamlessly over the network. A typical example of this is when a user has an account on several systems and is able to access the same "home" directory regardless of what system they are logging in to. This is possible because the users home directory is "shared" and mounted on the other systems using NFS. NFS access is controlled by the RPCbind service, which can communicate over both TCP and UDP protocols. When an RPC service such as NFS starts it registers its program numbers and listening address with the rpcbind service agent, which maintains a table of available services. The rpcbind service agent maps the service to a port and listens for connection requests for the services it has mapped, in our case NFS. The client then makes a RPC call to rcpbind using the NFS program number to determine the address where the request should be sent. There are several RPC services that are configured to run out of inetd.conf. We will disable all of them on the JumpStart server since they will not be required. The NFS service however, is controlled by startup and shutdown scripts for the client and server portions. The startup script for the server is /etc/rc3.d/S15nfs.server, the shutdown script is /etc/rc1.d/K28nfs.server. The NFS client will be disabled on the JumpStart server during JASS hardening.

Risk Analysis & Mitigation Plan

Since the JumpStart server will be used to build and configure every Solaris system that will be deployed on our network, including our DMZ, it is critical that the OS distribution media images, patch clusters, and control scripts remain tamper proof to ensure their integrity. If any of the image data were to be compromised or the configuration scripts tampered with it could result in incorrectly configured servers. If someone maliciously planted a back door, Trojan or a counterfeit configuration file in the build images every server deployed would be immediately compromised. It's for this reason that the critical files on the JumpStart server will be digitally fingerprinted using MD5-Tool v1.1. In addition, each server will have a vulnerability assessment done to validate that no additional services or vulnerabilities have been introduced as a result of the build process and that the server was adequately patched.

Remote access to the server will be limited to the Build Center team and will be restricted to Secure Shell (SSH), since Telnet, ftp and the shell services will be disabled. TCP Wrappers, which is now included in Solaris 9, will be enabled and configured to provide an additional level of access control and logging.

In order for the JumpStart server to perform OS installs it will require two high risk network services, Trivial File Transfer Protocol or tftp and the Sun RPCbind service in order to mount NFS filesystems. Tftp is required so that the client machine can boot a "mini" Solaris OS from the JumpStart server over the

network. This will provide the client with enough of a system configuration so that it can access the local area network and obtain the rest of the installation configuration information from the JumpStart server via an NFS mounted filesystem.

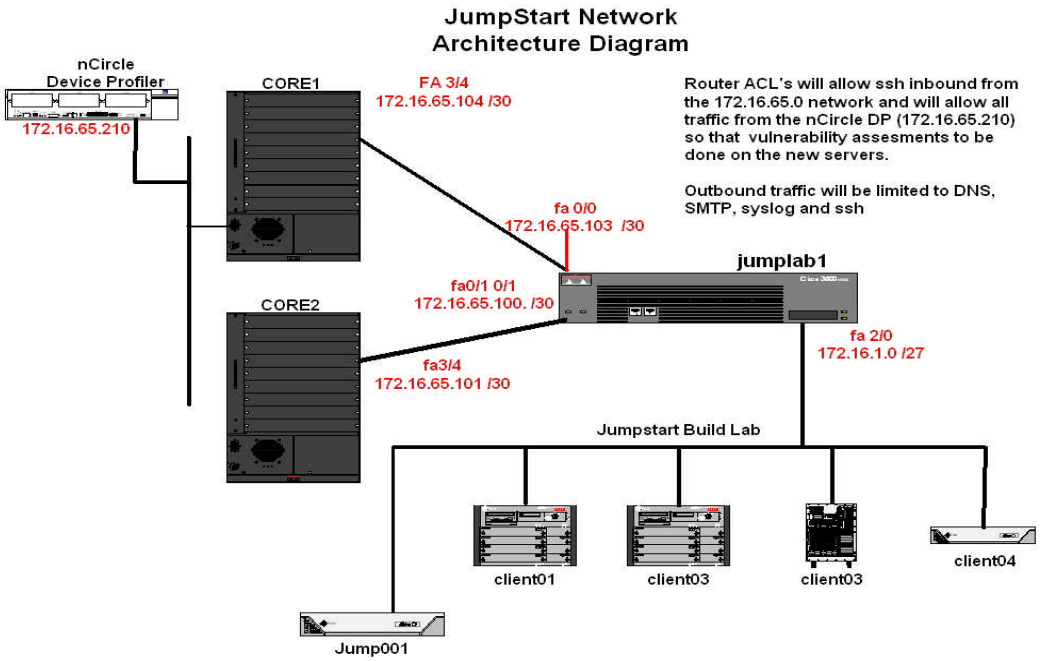
Because of the simple nature of tftp, and the lack of authentication required to establish a connection, it poses a significant security exposure. There are many documented vulnerabilities and several known exploits that use tftp as the primary means of compromise. A remote attacker can gain access to any file on a system with world read/write access, which is why it's critical to control and monitor file permissions on our system. The Nimda virus which is touted as one of the most damaging worms to ever hit the internet, and compromised thousands of hosts used tftp as one of its modes of propagation. This CERT advisory typifies the significance of the risks associated with tftp - http://www.cert.org/advisories/CA-1991-18.html. The tftp protocol is most certainly a network service that would be disabled on most servers of any type on a production network, and in fact, tftp will be disabled on all servers built by JumpStart by default (unless we are building another JumpStart server of course). However it is required for use on our JumpStart server and it will require that significant compensating controls be put in place to ensure its security and integrity.
The tftp service will be configured to run as the user nobody, an account which does not have elevated privileges, and the tftp root directory will be restricted to the /tftpboot directory using chroot( ) to prevent unauthorized access to other data. These controls will be put in place during the JASS hardening process.

Sun RPCbind and NFS are another set of services that pose a significant security risk. Many exploits and vulnerabilities are attributed to these services and specific controls will need to put in place. Here is a CERT advisory for NFS http://www.cert.org/advisories/CA-1994-15.html.

Steps must also be taken to secure access to the NFS mounted filesystem. Especially since we will be required to export the entire /jumpstart directory, which contains sensitive information related to the server builds. The directory will be shared in read-only mode and the /etc/system file modified to enable restricted NFS port access. The server will then only accept NFS requests from hosts over privileged ports that are below 1024.

Another security measure will be to limit network access to the Build Center lab by creating a private VLAN behind a CISCO router. The Access Control List (ACL) on the router will allow only SSH inbound from a specific subnet of one internal network. This will allow the server to be remotely managed. The router will also allow all traffic from the nCircle Device Profiler so that ongoing vulnerability assessments can be done to validate the new server builds. The router ACL's will allow DNS, syslog, SMTP and SSH outbound from the private network so the server can do name resolution, email status reports to

administrators, forward syslog data to the central log server and perform secure file transfers. The diagram below shows the network architecture for the Build Center lab.



Lastly in terms of risk mitigation, we will minimize the number of Solaris packages installed on the JumpStart server. We will be using the End User install cluster, plus four additional packages not included in the cluster that will add system accounting functionality and online manual pages. The End User cluster contains about 300 packages as compared to over 500 included in the entire Solaris distribution. This is significant because it reduces the amount disk space used by the OS. It reduces system resource utilization by preventing unnecessary services from running. It minimizes exposure to security exploits by eliminating vulnerabilities associated with the unnecessary services. And finally, it reduces the amount of software that will need to be patched.

The Jumpstart server will also receive regularly scheduled patch maintenance to ensure the software we're running stays as up to date as possible, and protected against the latest vulnerabilities.

Solaris Security Toolkit (JASS) Overview

The Solaris Security toolkit or JASS is a mechanism developed by Sun Microsystems for securing the Solaris operating environment. It was developed over several years by Sun's Enterprise Server Products and Professional Services organizations and is based on general best practices and customer experiences. The process of hardening a server involves:

- Determining the appropriate OS software or packages that are required for the server to function in its designated role

- Modifying several system configuration parameters to enhance performance and eliminate vulnerabilities
- Disabling unnecessary network services that may allow unauthorized access or denial of service (DoS)
- Implementing the proper level of access controls to ensure data integrity
- Displaying the proper banners and disclaimers to deter unauthorized access and position the corporation to enforce its security policies.

All of these tasks if done manually, require a considerable amount of resources and are extremely repetitive if a large number of servers are being rolled out on a regular basis. In addition, the manual hardening would introduce non uniformity in that different individuals might use their own unique approach ultimately resulting in deviations across systems, and of course the unlikely chance of human error. JASS is designed to allow those tasks to be automated and repeated within a framework that has enough flexibility built in to allow for customization. JASS integrates with the JumpStart architecture to allow a standardized, seamless process for installing, configuring and hardening a Solaris OS.

The JASS architecture is comprised of a series of scripts that control the hardening process and when integrated with JumpStart can apply the hardening directly at server build time. The major components of JASS are Profile, Driver, Finish scripts and configuration files. These components will be discussed in more detail in the next section.

The JASS toolkit can be downloaded from Sun's website in the form of a package and is installed using the pkgadd command. When installed, it creates the following directory structure:

JASS Home Directory – Typically /opt/SUNWjass

- *Documentation* – Contains a README file with references to additional Sun documentation about JASS
- *Drivers* – Contains configuration files that specify the sequence and details of how the systems are hardened. These files can be used as templates to custom build a hardening process for a specific installation. The Drivers call other scripts that set up the environment that the hardening scripts execute in.
- *Files* – This directory contains specific configuration files that will be installed on the system during the hardening process. The two main sub-directories are /etc and /sbin. This directory also contains the .cshrc and .profile files
- *Finish* – This directory contains the finish scripts that actually perform the hardening and are called by the Driver scripts. The Finish scripts are grouped into the following categories. The variables for all finish scripts are defined in the /jumpstart/Drivers/finish.init file (see sample file in appendix A).

Category	*Function*
Disable	Disables OS services
Enable	Enables OS Services or components
Install	Installs OS components or configuration settings
Minimize	Removes packages to minimize the OS
Print	Prints JASS environmental variable information
Remove	Removes unneeded accounts
Set	Sets system parameters and banners
Update	Updates specific configuration files

- *OS* – Contains the Solaris OS images
- *Packages* – Contains software packages that can be installed by a finish script
- *Patches* – Contains Sun Recommended and Security patch clusters
- *Profiles* – Contains JumpStart profile files that contain the configuration information for building servers, such as package cluster information, Disk layout and installation type.
- *Sysidcgf* – Contains configuration files used during JumpStart mode installation

The Solaris Security toolkit also maintains a data repository where it saves the data from each run and maintains information about which files were modified. This information is retained for future runs and allows the modifications to be reversed with the undo feature in the event they need to be rolled back.

Integrating JASS with Jumpstart

The Sun JumpStart environment is used to build and configure Solaris system using an automated repeatable process. JumpStart can leverage the JASS toolkit to implement system hardening at build time. This is done by manually installing the toolkit and moving the components into the JumpStart environment. These steps will be detailed later in section 7.

In order for a client to be built by the JumpStart server the characteristics for the client must be configured in several places. The network location or identity of the client is defined by creating entries in the /etc/ethers and /etc/hosts files on the JumpStart server. The entry in /etc/ethers defines the Ethernet (a.k.a. MAC) address of the client, and the /etc/hosts entry defines the IP address. Additional system build characteristics about the client are defined using the Profile and Sysidcfg files. The Profile file tells JumpStart if this will be an initial install or an upgrade, what OS cluster and additional packages to install, and the specifics about the disk layout. The Sysidcfg file tells JumpStart the locale information, the time zone and basic network interface and name service type information that the

client will need. Once this information is established JumpStart can build the system.

When the OS is loaded JumpStart will invoke JASS in the form of the Driver files. The secure.driver file establishes the environmental variables and calls the subsequent config and hardening driver files. The config.driver file will implement some basic configuration such as installing the recommended patches (install-recommended.fin finish script) and terminal information (set-term-type.fin finish script). The config.driver also calls the driver.run script that dose some further variable initialization and calls the install-template.fin finish script. The install-template.fin special purpose script is automatically run by the driver.run script and should not be included manually in the config or hardening driver scripts. This script is executed prior to any other hardening script.

install-template.fin
This special purpose script is called by the driver.run script when the config.driver and hardening.driver scripts are executed. This script copies files from the /jumpstart/Files directory as defined by the JASS_FILES variable. The following files will be copied to our system

/.cshrc
/.profile
/etc/dt/config/Xaccess
/etc/init.d/inetsvc
/etc/init.d/nddconfig
/etc/init.d/set-tmp-permissions
/etc/issue
/etc/motd
/etc/notrouter
/etc/syslog.conf
/etc/nsswitch.conf
/etc/resolv.conf
/etc/rc2.d/S00set-tmp-permissions
/etc/rc2.d/S07set-tmp-permissions
/etc/rc2.d/S70nddconfig

NDD Configuration
A file of note is the /etc/init.d/nddconfig file. This file will implement network driver parameters at boot time that will make the system more resistant to certain network attacks. The nddconfig script implements the following network driver settings:

arp_cleanup_interval=60000
This option determines the period of time the Address Resolution Protocol (ARP) cache maintains entries. ARP attacks may be effective with the default interval.

Shortening the timeout interval should reduce the effectiveness of such an attack. The default value is 300000 milliseconds (5 minutes).

ip_forward_directed_broadcasts=0
This option determines whether to forward broadcast packets directed to a specific net or subnet, if that net or subnet is directly connected to the machine. If the system is acting as a router, this option can be exploited to generate a great deal of broadcast network traffic. Turning this option off will help prevent broadcast traffic attacks. The default value is 1 (true).

ip_forward_src_routed=0
This option determines whether to forward packets that are source routed. These packets define the path the packet should take instead of allowing network routers to define the path. The default value is 1 (true).

ip_ignore_redirect=1
This option determines whether to ignore Internet Control Message Protocol (ICMP) packets that define new routes. If the system is acting as a router, an attacker may send redirect messages to alter routing tables as part of sophisticated attack (man in the middle attack) or a simple denial of service. The default value is 0 (false).

ip_respond_to_address_mask_broadcast=0
This options determines whether to respond to ICMP netmask requests which are typically sent by diskless clients when booting. An attacker may use the netmask information for determining network topology or the broadcast address for the subnet. The default value is 0 (false).

ip_respond_to_echo_broadcast=0
This option determines whether to respond to ICMP broadcast echo requests (ping). An attacker may try to create a denial of service attack on subnets by sending many broadcast echo requests to which all systems will respond. This also provides information on systems that are available on the network. The default value is 1 (true).

ip_respond_to_timestamp=0
This option determines whether to respond to ICMP timestamp requests which some systems use to discover the time on a remote system. An attacker may use the time information to schedule an attack at a period of time when the system may run a cron job (or other time-based event) or otherwise be busy. It may also be possible predict ID or sequence numbers that are based on the time of day for spoofing services. The default value is 1 (true).

ip_respond_to_timestamp_broadcast=0
This option determines whether to respond to ICMP broadcast timestamp requests which are used to discover the time on all systems in the broadcast

range. This option is dangerous for the same reasons as responding to a single timestamp request. Additionally, an attacker may try to create a denial of service attack by generating many broadcast timestamp requests. The default value is 1 (true).

ip_send_redirects=0
This option determines whether to send ICMP redirect messages which can introduce changes into remote system's routing table. It should only be used on systems that act as routers. The default value is 1 (true).

ip_strict_dst_multihoming=1
This option determines whether to enable strict destination multihoming. If this is set to 1 and ip_forwarding is set to 0, then a packet sent to an interface from which it did not arrive will be dropped. This setting prevents an attacker from passing packets across a machine with multiple interfaces that is not acting a router. The default value is 0 (false).

ip_def_ttl=255
This option sets the default time to live (TTL) value for IP packets. Normally, this should not be altered from the default value. Changing it to a different value may fool some OS "fingerprinting" tools such as queso or nmap. The default value is 255.

tcp_conn_req_max_q0=4096
This option sets the size of the queue containing unestablished connections. This queue is part of a protection mechanism against SYN flood attacks. The queue size default is adequate for most systems but should be increased for busy servers. The default value is 1024.

tcp_conn_req_max_q=1024
This option sets the maximum number fully established connections. Increasing the size of this queue provides some limited protection against resource consumption attacks. The queue size default is adequate for most systems but should be increased for busy servers. The default value is 128.

tcp_smallest_anon_port=32768
tcp_largest_anon_port=65535
udp_smallest_anon_port=32768
udp_largest_anon_port=65535
These options define the upper and lower bounds on ephemeral ports. Ephemeral (means short-lived) ports are used when establishing outbound network connections. Defaults values:
tcp_smallest_anon_port=32768
tcp_largest_anon_port=65535
udp_smallest_anon_port=32768
udp_largest_anon_port=65535

tcp_smallest_nonpriv_port=1024
udp_smallest_nonpriv_port=1024
These options define the start of nonprivileged TCP and UDP ports. The nonprivileged port range normally starts at 1024. Any program that attempts to bind a nonprivileged port does not have to run as root.
Defaults values:
tcp_smallest_nonpriv_port=1024
udp_smallest_nonpriv_port=1024

ip_ire_arp_interval=60000
This option determines the period of time at which a specific route will be kept, even if currently in use. ARP attacks may be effective with the default interval. Shortening the time interval may reduce the effectiveness of attacks.
The default interval is 1200000 milliseconds (20 minutes).

tcp_extra_priv_ports_add="6112"
These options define additional TCP and UDP privileged ports outside of the 1-1023 range. Any program that attempts to bind the ports listed here must run as root. This prevents normal users from starting server processes on specific ports. Multiple ports can be specified by quoting and separating them with spaces. This will provide additional security for nfsd by preventing unprivileged user connection attempts. Defaults values:
tcp_extra_priv_ports: 2049 (nfsd) 4045 (lockd)

tcp_rev_src_routes=0
This option determines whether the specified route in a source routed packet will be used in returned packets. TCP source routed packets may be used in spoofing attacks, so the reverse route should not be used.
The default value is 0 (false).

Building the Driver scripts for Hardening
Here we will identify the Driver scripts and their contents. There are three driver scripts that are used to harden a system:

- The Secure Driver – The secure is used to control the other two drivers. First it calls the Driver.init file, which sets all of the JASS environmental variables. It then calls the associated config and hardening drivers.
- Configuration or Config driver – This script performs the basic system configuration primarily the installation of the Sun Recommended patches.
- Hardening driver – Installs configuration files onto the system from the toolkit Files directory and tells JASS which Finish scripts to execute to perform configuration and hardening of the system.

The following finish scripts are called by the hardening driver and will perform the actual system hardening. The following list details the scripts, the functions they perform, and the impact they have on the overall security of the system. The

scripts have been categorized into three classifications based on the areas of the operating environment that they impact.

1. **Access control** – These finish scripts secure how the system and its components are accessed and provide additional controls in those areas.

2. **System Security** – These finish scripts provide additional controls around the core services and components of the operating system.

3. **Network Security** – These finish scripts provide additional security and controls around specific network services that the system provides.

As the individual scripts are executed by jass the original system configuration files that get modified are saved. The files are renamed by adding a ".JASS.YYYYMMDDHHMM" extension to the original file. Start up scripts in the /etc/rc*.d directories are prefixed with an underscore ("_") in addition to the JASS appending. This naming convention is so that the changes can be reversed by the undo function of jass. It also allows the administrators to examine what specific changed were made by comparing the differences between the files.

Access Control Hardening Finish Scripts

disable-dtlogin.fin

This script prevents any windowing environment from starting on the system at boot time by renaming the K10dtlogin shutdown scripts in /etc/rcS, 0 and 1.d and the S99dtlogin startup script in /etc/rc2.d. There have been several documented exploits against the CDE windowing software. The JumpStart server will not require a Graphical Display so the windowing environment can be disabled.

disable-remote-root-login.fin

This script will prevent anyone from remotely logging into the system directly using the root userid by modifying the CONSOLE variable in the /etc/default/login. Remote root logins pose a security exposure because it limits the ability to audit the use of the root privilege. Forcing users to log on with their own userid and using the su command to elevate their privileges will allow root access to be audited in the /var/adm/sulog and /var/adm/messages files.

disable-ssh-root-login.fin

This script will provide the same functionality as the previous script but for SSH access. This script will set the PermitRootLogin line to no in the /etc/ssh/sshd_config file.

disable-rhosts.fin

This script will disable authentication for the rlogin and rsh remote access commands by commenting out the entries for the commands in the Pluggable Authentication Module configurations file /etc/pam.conf.

enable-inetd-syslog.fin
This script will configure the Internet Service Daemon (inetd) to log all incoming TCP connection requests to syslog for any service that the inetd daemon is listening for. This is done by modifying the ENABLE_CONNECTION_LOGGING variable to YES in the /etc/default/inetd configuration file. The logging of network connections requests is important to for monitoring both authorized and unauthorized system access. All control logs are forwarded to a central log server and monitored daily.

enable-tcpwrappers.fin
This script will enable the use of TCP Wrappers to be used for access control. TCP Wrappers is used to restrict access to TCP services controlled by the inetd daemon that are defined as stream,nowait. This script modifies the /etc/default/inetd configuration file by setting the ENABLE_TCPWRAPPERS parameter to YES. It also installs sample /etc/hosts.allow and /etc/hosts.deny control files to be used as templates for controlling access to services protected by TCP Wrappers.

install-at-allow.fin
This script will install the at.allow file in /etc/cron.d. All users that will require access to the at facility must be added to the at.allow file. Restricting access to the at facility can prevent users from unauthorized scheduling of programs that can run on a system. Limiting access to this facility will increase the overall security of the system.

install-loginlog.fin
This script will create the /var/adm/loginlog file that will be used to log all unsuccessful login attempts. Failed login attempts are logged after the value of the RETRIES variable that is set in /etc/default/login files has been exceeded. This value is set to three.

install-sulog.fin
This script created a /var/adm/sulog file that will be used to log all su attempts on the system.

Note: *The displaying of banners is considered a best practice for securing systems. Banners provide no capability to prevent unauthorized access, but an effective banner should inform anyone accessing the system of the access policy of the organization and provide a warning of legal recourse if there is unauthorized access. The presence of a banner could help establish a legal foundation in the event that a system is compromised and an organization wishes to take legal action against those responsible. This is synonymous to a "No Trespassing" sign on physical property.*

The only remote access services that will be enabled on the server will be SSH. As mentioned before, telnet, ftp and Common Desktop Environment (CDE) will be disabled, however, the software packages were not removed from the system and could be enabled if needed. The set-banner-dtlogin.fin, set-banner-ftp.fin and set-banner-telnet.fin scripts will be run to preemptively banner these services in the event they are enabled at a later time.

set-banner-dtlogin.fin
This script installs a banner for the dtlogin service to be displayed after a successful login using CDE. The /etc/dt/config/Xsession.d/0050.warning file is modified to display the contents of the /etc/motd file, which would contain the appropriate banner message.

set-banner-ftpd.fin
This script installs the banner for the ftp service. This banner will be displayed prior to authentication and will show the contents of the /etc/ftpd/banner.msg file. The feature is enabled by setting the BANNER variable in the /etc/default/ftpd configuration file.

set-banner-telnet.fin
This script installs the telnet banner that will display the banner defined by the BANNER parameter in the /etc/default/telnetd file prior to authentication when a connection attempt is made.

set-banner-sshd.fin
This script installs the SSH banner that will display the contents of the /etc/issue file prior to authentication when a connection attempt is made. This is done by setting the BANNER parameter to /etc/issue in the /etc/ssh/sshd_config file.

set-login-retries.fin
This script modifies the /etc/default/login configuration file in order to control the number of failed login attempts that are allowed before the login utility exits (RETRIES=3). It also is used to determine how many failed login attempts are required before a failed login attempt message is sent to syslog (SYSLOG_FAILED_LOGINS=0). Setting RETRIES to three will allow three unsuccessful password entry attempts before dropping the TTY line and terminating the login process, meaning that the terminal session will need to be restarted. Setting the syslog failed logins to zero will log every occurrence of the previous event in /var/log/authlog.

set-user-password-reqs.fin
This script implements an organizations password policy regarding password aging, interval between password changes and password length on the system. This is accomplished by modifying the /etc/default/passwd file and implementing the following parameters

MINWEEK=1 – The minimum amount of time before a user can change their password
MAXWEEKS=8 – The amount of time in weeks that the system will require a password change.
WARNWEEKS=1 – The amount of time in weeks that the user will be warned of a pending password change.
PASSLENGTH=8 – The minimum length that the password can be.

The JASS variables that define these values are JASS_AGING_MINWEEKS, JASS_AGING_MAXWEEKS, JASS_AGING_WARNWEEKS and JASS_PASSLENGTH.

update-at-deny.fin
This script restricts the access to the at batch scheduling facility on the server. This is done by populating the /etc/cron.d/at.deny with the user accounts that will be denied access to the facility. This script will populate the file with the accounts that currently exist in the /etc/passwd file. As a security best practice it is common to deny all access to system facilities and explicitly allow access for those who require it.

update-cron-deny.fin
This script restricts the access to the cron batch scheduling facility on the server. This is done by populating the /etc/cron.d/cron.deny with the user accounts that will be denied access to the facility. This script will populate the file with the accounts that currently exist in the /etc/passwd file if the userid is less than 100 or greater than 60000 with the exception of root. As a security best practice it is common to deny all access to system facilities and explicitly allow access for those who require it.

update-cron-allow.fin
This script allows access to the cron batch scheduling facility on the server. This is done by populating the /etc/cron.d/cron.allow with the root user accounts. As a security best practice it is common to deny all access to system facilities and explicitly allow access to those who require it. As a result root will be the only account authorized to use the cron facility

System Security Hardening Finish Scripts

Note: *As a security best practice, all non-essential services will be disabled as a result of the following disable finish scripts. As a result, the system will more secure because the potential exploits that may take advantage of the non-essential services will be ineffective. Another benefit will be that the system will incur less performance overhead in terms of memory and CPU usage since the non-essential processes will not be running and consuming system resources. In addition, the set, install and enable finish scripts will implement additional*

security controls on services that are required to enhance the overall security of the system.

disable-autoinst.fin
This script will disable the run-control scripts that allow automatic configuration of the system. The sys-unconfig run-control script can be used to reconfigure or restore a systems configuration, including network parameters. This functionality is not required and will be disabled to prevent the unauthorized automatic reconfiguration of the systems network parameters.

disable-automount.fin
This script will disable NFS automount service. The automount service will respond to mount and umount requests from the autofs file system. This service will not be required as the /jumpstart directory will be the only NFS filesystem shared by the server and will explicitly mounted all the time.

disable-dmi.fin
This script will disable the Desktop Management Interface (DMI) on the system. This is a non-essential service and will be prevented by renaming the startup and shutdown scripts in /etc/rc*.d K07dmi in /etc/rcS, 0, 1, 2.d, K77dmi in /etc/rc2.d and S77dmi in /etc/rc3.d.

disable-kdc.fin
This script will disable Kerberos Key Distribution Service by preventing the service from starting. This is done by renaming the startup ans shutdown scripts in /etc/rc*.d K28kdc.master and K28kdc in /etc/rcS, 0, 1 and 2.d and S13kdc.master and S14kdc in /etc/rc3.d. The KDC service is not required for this system.

disable-lp.fin
This script will disable the line printer service and remove the user's access to the cron service. This is done independent of the update-cron-deny.fin script in the event that the lp service is required.

disable-nscd-caching.fin
This script will disable Name Service Caching for the passwd, group,hosts and ipnodes entries. This is done by setting the ttl value for those entries to zero in the /etc/nscd.conf configuration file. In doing so it reduces the chance of spoofing attacks by reducing the amount of data that the name service will cache.

disable-preserve.fin
This script will prevent the moving of saved files that are recovered by a file editing session that was interrupted by a system crash or abnormal termination. The scrip renames the S80PRESERV and S89PRESERV start up scripts in /etc/rc2.d.

disable-power-mgmt.fin
This script will prevent the power management services from starting up. These services will allow the system to remove power from devices such as monitors, disk drives or even the entire system in an effort to reduce power consumption. Typically server class machines will disable this functionality because availability is a requirement. The services are disabled by renaming the K37power script in /etc/rcS, 0 and 1.d, the K85power script in /etc/rc1.d and the S85power script in/etc/rc2.d. This scripr will also create a /noautoshutdown file to prevent the system from prompting for the power management status at the first reboot after an installation.

disable-system-accounts.fin
This script will disable unused system accounts and enables logging of access attempts to them. This is done by creating a /sbin/noshell script which denies access to the account and logs access attempts to SYSLOG. The /sbin/noshell is used as the default shell for the accounts defined by the JASS_ACCT_DISABLE variable.

disable-vold.fin
This script will disable the Volume Management (vold) service from starting. The vold daemon manages the mounting and un-mounting of removable media such as CD-ROM disks. This is done by renaming the K35volmgt scripts in /etc/rcS, 0, and 1.d directories, and the S92volmgt script in /etc/rc2.d and the S81volmgt script in /etc/rc3.d. The mounting and un-mounting of all removable media will dome manually on this server.

enable-coreadm.fin
This script will enable the coreadm functionality on the system. Core file are created as a result of programs terminating abnormally and are used to diagnose the cause of the application crash. They do however, contain contents of memory, which has the potential to include sensitive or confidential data. Thus the importance of managing core files on a system. Coreadm will store global and init core files by pattern in /var/core and tag the file accordingly based on a pattern specified by the JASS_CORE_PATTERN variable.

enable-process-accounting.fin
This script will enable the process accounting facility on the system if the SUNWaccr and SUNWaccu packages are installed. The process accouting facility maintains statistical information about every process that runs on the system. This data can be useful for performance monitoring and tuning, and incident response. The script will validate that the required packages are installed and the /etc/rc3.d/S99acct startup script is in place.

enable-rfc1948.fin

This script will enable support for rcf1948 unique per connection TCP sequence number generation which can help prevent IP spoofing attacks based on sequence number spoofing. The script sets the TCP_STRONG_ISS to 2 in the /etc/default/inetinit file.

enable-stack-protection.fin
This script will enable program stack protection and exception logging. This is done by adding the noexec_user_stack and noexec_user_stack_log kernel parameters to the /etc/system configuration file. Enabling program stack protection will reduce the systems vulnerability to common buffer overflow attacks by denying attempts to execute the program stack directly. The system will also log any program stack execution attempts to SYSLOG.

install-shells.fin
This script will add user shells listed in the JASS_SHELLS variable to the /etc/shells file. The file is used by the getusershell function to determine valid shells on the system.

remove-unneeded-accounts.fin
This script will disable unused Solaris accounts from the /etc/shadow and /etc/passwd files. This is done by using the passmgnt command for the accounts defined by the JASS_ACCT_REMOVE variable (smtp, listen, and nobody4).

set-power-restrictions.fin
This script will restrict access to the power management functionality, thereby controlling access to the suspend/resume and power management utilities.

set-root group.fin
This script will modify the root users primary group from the default of 1 (other) to 0 (root). This prevents root from sharing a common user group with non-privileged users. The value is defined by the JASS_ROOT_GROUP variable in the /jumpstart/Drivers/finish.init.

set-rmmount-nosuid.fin
This script will disable the mounting of Set-UID files from removable media such ad CD-ROM's. Set-UID files pose a significant risk to a system because the programs execute with elevated privileges, setuid(root) programs execute as root! If someone were to install a CD-ROM and mount it with set-UID binaries they could potentially compromise the system. Solaris 9 restricts this functionality by default. This script verifies that the "mount * hsfs udfs ufs nosuid" entry exists in /etc/rmmount.conf and will add it if it is not there.

set-system-umask.fin
This script will set a system umask so that the system's run control scripts will execute with a safe file creation. This is done by setting the CMASK variable to

022 in /etc/default/init. Using a strong umask will prevent critical files from being writable by any user.

set-users-umask.fin
This script add the a umask of 022 to ther following user startup files, which will prevent files created by users from being world writable.

/etc/.login
/etc/profile
/etc/skel/local.cshrc
/etc/skel/local.login
/etc/skel/local.profile
/etc/default/login

update-cron-log-size.fin
This script will set the size maximum limit for the /var/log/cron log file to 10240k. This provides the capacity to log more information. This is done by using the logadm utility and adding the –s 10240k to the /var/log/cron entry. The log size is specified by the JASS_CRON_LOG_SIZE environmental variable in the /jumpstart/Drivers/finish.init file.

install-fix-modes.fin
This script will install and run the FixModes script, which is used to tighten file permissions on a Solaris system.

install-md5.fin
This script will install the md5 software, which is used to create digital signatures of system objects.

Network Security Hardening Finish Scripts

Note: *As a security best practice, all non-essential network services will be disabled as a result of the following disable finish scripts. As a result, the system will more secure because the potential exploits that may take advantage of the non-essential network services will be ineffective. Network services pose a significant risk to the system because they can be exploited remotely. Even though our system is on an isoladed network and protected by a router with restrictive Access Control List as a best practice the servers network services will be hardened to provide an additional layer of security.*

disable-directory.fin
This script will prevent the Suns One directory server from starting by renaming the K41directory script in /etc/rcS,0 and1.d and the S72directory startup script in /etc/rc2.d. The Sun One directory service provides central storage repository that can be used to store everything from access privileges and can be used for

authentication and authorization. This service will not be required and will be disabled.

disable-ipv6.fin
This script will disable IPv6 on specific network interfaces by removing the /etc/hostname6.* file for each interface. This script will also prevent the in.ndpd service from running.

disable-ldap-client.fin
This script will prevent the LDAP (Lightweight Directory Access Protocol) client network services from starting by renaming the K41ldap.client shutdown scripts in /etc/rcS, 0, and 1.d and the S71ldap.client startup script in /etc/rc2.d. LDAP client services will not be required.

disable-nfs-client.fin
This script will prevent the NFS (Network File System) client network services from starting by renaming the K41nfs.client and K75nfs.client shutdown scripts in /etc/rc0.d, the K80nfs.client shutdown script in rc1.d and the S73nfs.client startup script in /etc/rc2.d. NFS client services will not be required.

disable-ppp.fin
This script will prevent the PPP (Point –to-Point Protocol) network service from starting by renaming the K50pppd shutdown script in /etc/rcS, 0, and 1.d, and the S47pppd startup script in /etc/rc2.d. the PPP network services will not be required.

disable-mipagent.fin
This script will disable the mobile IP (MIP) network service from starting by renaming the K06mipagent shutdown scripts in /etc/rcS, 0, 1, and 2.d, and the S80mipagent startup script in /etc/rc3.d. The network services will not be required.

disable-sendmail.fin
This script will disable the sendmail daemon from starting by renaming the K36sendmail shutdown script in /etc/rcS, 0, and 1.d, the K57sendmail shutdown script in /etc/rc0, and 1.d and the S88sendmail startup script in /etc rc2.d. The script also add an entry to root's crontab file to purge the outgoing mail queue hourly. This script will leave the mail services in a state where outbound mail only will be processed. Outboubd mail will be required to send audit reports of various system components to systems administrators.

disable-samba.fin
This script will prevent the Samba file and print sharing facility from starting by renaming the K03samba shutdown scripts in /etc/rcS, 0, 1 and 2.d and the S90samba startup script in /etc/rc3.d. Samba services will not be required.

disable-snmp.fin
This script will disable the SNMP (Simple Network Management Protocol) network services from starting by renaming the K07snmpdx shutdown scripts in /etc/rcS, 0, 1, and 2.d, the K76snmpdx shutdown script in rc2.d and the S76snmpdx startup script in /etc/rc3.d. SNMP is another server that has an abundance of known vulnerabilities and exploits associated with it. Since this service is not required it will be disabled.

disable-slp.fin
This script will disable the SLP (Service Location Protocol) network service by renaming the K41slp shutdown scripts in /etc/rcS, 0 and 1.d, and the S72slp startup script in /etc/rc2.d. SLP network services will not be required.

disable-syslogd-listen.fin
This script will prevent the SYSLOG daemon from accepting remote log messages. This is done by setting the LOG_FROM_REMOTE variable to NO in the /etc/default/syslogd file.

disable-uucp.fin
This script will prevent the UNIX-to-UNIX copy (UUCP) facility from starting by renaming the S70uucp script in /etc/rc2.d. This script will also remove the nuucp system account and the uucp crontab entries in the /var/spool/cron/crontabs directory. UUCP services will not be required on the system.

disable-xserver-listen.fin
This script will disable the X11 server from listening on port 6000 for network connection requests. This is done by adding the –nolistentcp option to the configuration line in the /etc/dt/config/Xservers file. The X11 service is another historically exploitable service that is not required and will be disabled.

disable-wbem.fin
This script will disable the WBEM (Web-based enterprise management) network services from starting by renaming the K36wbem shutdown scripts in /etc/rcS, 0, and 1.d, the S90wbem startup script in /etc/rc2.d. The WBEM services are not require and will be disabled.

enable-priv-nfs-ports.fin
This script will enable restricted NFS port access by adding the “set nfssrv:nfs_portmon=1” paremeter to the /etc/system file. This will configure NFS to only accept connection requests from privileged system ports less than 1024.

update-inetd-conf.fin
This script will disable all services started bt the inetd (Internet Services daemon) that are defined by the JASS_SVCS_DISABLE variable in /jumpstart/Drivers/finish.init (See sample finish.init file in appendix A for the complete list) Some of the services include are chargen, comsat, daytime,

discard, dtspc, echo, exec, finger, fs, ftp, login, name, netstat, rexd, rquotad, rstatd, rusersd, shell, sprayd, systat, talk, telnet, tftp, time, ufsd, uucp, and walld.

Note: *This script disables the tftp service. JumpStart requires the tftp service but it can safely be disabled here because it will be re-enabled by JumpStart later by defining the JASS_SVCS_ENABLE="tftp" variable in the jumpstart-secure-new.driver file. JumpStart will also ensure that it is enabled when the JumpStart client is added using the add-client command in step 7.7.*

Step-by-Step Hardening

Note: *Since the server will not have a monitor, keyboard or mouse directly attached, it will be necessary to run the install from a local serially attached ascii terminal, or Laptop running the appropriate terminal emulation software such as Windows Hyperterminal or by connecting the server to a terminal server.*

1. **System Configuration**
   - **1.1.** Boot from the most current Solaris 9 CD Distribution
   - **1.2.** Choose the appropriate language support and terminal type
   - **1.3.** Provide the Network Configuration Information
     - **1.3.1.** Select “yes” for the server being networked
     - **1.3.2.** Hostname
     - **1.3.3.** No DHCP
     - **1.3.4.** Enter IP address information
     - **1.3.5.** Enter “yes” for Subnet and provide the appropriate mask
     - **1.3.6.** Select “no” for IPv6 support
     - **1.3.7.** Specify a default route
   - **1.4.** Specify that the server will not use Kerberos security
   - **1.5.** Select “none” for the type of name directory service
   - **1.6.** Enter the appropriate Geographic and Time zone information
   - **1.7.** Select an Initial install

2. **Minimized OS Install**
   - **2.1.** Choose an “initial” install
   - **2.2.** Select “stand-alone” server
   - **2.3.** Select “end-user” package cluster
   - **2.4.** Configure the select the boot disk and provide the disk layout information
     - **2.4.1.** The 18Gb boot disk layout is
       10Gb / (this slice contains /opt)
       4Gb /var (for logging)
       1Gb swap (the size of physical ram)
       2Gb /export/home (for users home directories)
       The second disk layout

18 Gb /JumpStart (for JumpStart and JASS Installation)

2.5. Select "continue" when prompted to ignore mounting external file systems

2.6. Select to "reboot" automatically after install

3. Installing Additional Solaris Packages

This process will install the terminfo database, system accounting, and manual pages, which are not included in the end-user package cluster.

NOTE: *System Accounting will be enabled and configured during JASS hardening by the enable-process-accounting.fin script. Since the volume manager daemon is running on the system by default she CD-Rom should mount automatically. If that is the case the mount command will not be necessary in the following step and the mount point for the CD-ROM will be /cdrom/cdrom0*

Insert disk two of the Solaris 9 Software CD

 # mount –r –F hsfs /dev/dsk/c0t0d0s0 /mnt
 # cd /mnt/sol_9_sparc/s0/Solaris_9/Products
 # pkgadd –d . SUNWter SUNWaccr SUNWaccu SUNWman
 # cd /
 # umount /mnt

4. Adding User Accounts and Preparing JASS

Note*: The following steps must be done on a secure trusted network that is not directly connected to the internet. Connecting the server to a hostile network before it is fully patched and hardened could lead to the system being compromised. Download all packages in the next section from a secure server on your internal network using SSH (sftp or scp).*

4.1. Set the root password and login as root
(Remember: you must connect to the server over a serially connected terminal, not over the network. Direct root logins over the network are disabled by default in Solaris 9 so you will not be able to access the server that way until the next step is completed.)

4.2. Create a user account for yourself on the system using the useradd command

 # useradd myaccount –s /bin/bash –G sysadmin

4.3. Set the password for the account you just created

 # passwd myaccount
New Password: ******
Re-enter new Password: *******
passwd: password successfully changed for myaccount
#

4.4. Download the following software packages and recommended patch cluster from a secure trusted host using sftp

NOTE: *Initial connections to hosts using SSH will produce the following dialog in order to populate the local host database with known host's public keys.*

```
# cd /opt
# sftp myaccount@securehost1
The authenticity of host 'securehost1' can't be established.
DSA key fingerprint in md5 is:
2d:1c:24:96:75:39:76:65:88:c6:48:11:de:bd:5d:46
Are you sure you want to continue connecting(yes/no)?yes
Warning: Permanently added 'myhost1,10.1.0.60' (DSA) to the list of
known hosts.
myaccount@securehost1's password: *********
sftp > cd Solaris9_Installs
sftp > get SUNBEmd5.pkg.Z              # Sun md5 package
sftp > get SUNWjass-4.0.1.pkg.Z        # Sun JASS Toolkit package v4
sftp > get SUNBEfixm.pkg.Z             # Sun FixModes package
sftp > get 9_Recommended.Zip           # Sun Recommended patches
sftp > get cis-solaris.tar.Z           # CIS benchmark tool v1.2.0
sftp > get md5-tool-1.1.gz             # Razor MD5-Tool v1.1
sftp > quit
#
```

4.5. Install the JASS Toolkit package downloaded from the previous step as follows: **Note:** *JASS installs into /opt/SUNWjass.*

```
# uncompress SUNWjass-4.0.1.pkg.Z
# mv SUNWjass-4.0.1.pkg.Z /opt
# cd /opt
#  pkgadd -d SUNWjass-4.0.1.pkg

The following packages are available:
  1  SUNWjass     Solaris Security Toolkit 4.0.1
       (Solaris) 4.0.1

Select package(s) you wish to process (or 'all' to process
all packages). (default: all) [?,??,q]: all
```

4.6. Uncompress the Sun FixModes and MD5 Packages and move them into the JASS Toolkit Packages directory

NOTE: *The FixModes and MD5 packages will be installed by the JASS toolkit.*

 # uncompress SUNBEfixm.pkg.Z
 # uncompress SUNBEmd5.pkg.Z
SUNBEmd5.pkg.Z: not in compressed format

Note: *The SUNBEmd5 package is already uncompressed but downloads with a .Z extension from Sun's website. Trying to uncompress the file will result in the above error message. This step can be omitted and the file can be moved to the appropriate directory with the correct extension as shown below for installation. The state of the file can be confirmed by using the "file" command. Sun also provides the MD5 sum values for the files. You should always validate the files using MD5 on another system prior to using them.*

md5 value for md5-sparc binary is
e4cb81d8ac18bcac085f84e401e00646

 # file SUNBEmd5.pkg.Z
SUNBEmd5.pkg.Z: English text
#
If results above confirm that the package is already uncompressed. A compressed file will display the following results:
 # file SUNBEmd5.pkg.Z
SUNBEmd5.pkg.Z: compressed data block compressed 16 bits
#
 # mv SUNBEfixm.pkg /opt/SUNWjass/Packages
 # mv SUNBEmd5.pkg.Z opt/SUNWjass/Packages/SUNBEmd5.pkg

4.7. Move the Recommended Patch cluster to the Toolkit Patches directory and Uncompress it

 # mv 9_Recommended.zip /opt/SUNWjass/Patches
 # cd /opt/SUNWjass/Patches
 # gunzip 9_Recommended.zip

4.8. Uncompress and install and run the CIS benchmark tool v1.2.0

# uncompress cis-solaris.tar.Z	**# uncompress distribution**
# tar xvf cis-solaris.tar	**# un-tar distribution**
# cd /opt/cis	**#**
# pkgadd -d CISscan all	**# installs the package**
# /opt/CIS/cis-scan	**# executes the script**
#	

5. Hardening The Server With The JASS Toolkit

5.1. Modifying Driver Scripts

Copy an existing Secure, Configure, and Hardening script to use as a template and make the appropriate modifications (see appendix C for modified files)

```
# cp Jumpstart-secure.driver Jumpstart-secure-new.driver
# cp Jumpstart-config.driver Jumpstart-config-new.driver
# cp Jumpstart-hardening.driver Jumpstart-hardening-new.driv
```

5.2. Execute the JASS toolkit in stand-alone mode. The –o option saves the output in a log file for review

NOTE: *Remember this step will harden the server to the specification designated in the jumpstart-hardening-new.driver file. This includes applying the Recommended patch cluster, installing and running FixModes, and installing MD5. Harding also includes disabling all services started in inetd.conf except for tftp. At this point you should be sure ssh is functioning properly because telnet and ftp will be disabled.*

```
# cd /opt/SUNWjass
# ./jass-execute –o jassrun.log –d Jumpstart-secure-new.driver
```

Partial output from jassrun.log

```
================================================================
jumpstart-secure-new.driver: Driver started.
================================================================


================================================================
JASS Version:  4.0.1
Node name:     jump001
Host ID:       830bec2b
Host address:  176.16.1.123
MAC address:   0:3:ba:b:ec:2b
OS version:    5.9
Date:          Wed Jun 02 21:59:31 EDT 2004
================================================================


================================================================
jumpstart-secure-new.driver: Finish script: install-templates.fin
================================================================

Copying personalized files.

[NOTE] Copying /.cshrc from /opt/SUNWjass/Files/.cshrc.
[NOTE] Copying /.profile from /opt/SUNWjass/Files/.profile.

================================================================
.
.
```

```
.
.
================================================================
jumpstart-secure-new.driver: Finish script: install-recommended-
patches.fin
================================================================

Installing the Solaris Recommended and Security patch cluster.

Installing the patches from the directory,
//opt/SUNWjass/Patches/9_Recommended.


================================================================
Installing Patch  : 114008-01
Patch Synopsis    : SunOS 5.9: cachefsd Patch
  Root Directory : /
 Source Location : ///opt/SUNWjass/Patches/9_Recommended
        Options:
================================================================


Checking installed patches...
Verifying sufficient filesystem capacity (dry run method)...
Installing patch packages...

Patch number 114008-01 has been successfully installed.
See //var/sadm/patch/114008-01/log for details

Patch packages installed:
 SUNWcsr
 SUNWcsu
.
.
.
.
================================================================
jumpstart-secure-new.driver: Finish script: update-inetd-conf.fin
================================================================

Updating status of network services in //etc/inet/inetd.conf.

Disabling those services listed in JASS_SVCS_DISABLE.

[NOTE] Copying /etc/inet/inetd.conf to
/etc/inet/inetd.conf.JASS.20040630225000
```

```
Disabling service, ftp (/usr/sbin/in.ftpd).
Disabling service, telnet (/usr/sbin/in.telnetd).
Disabling service, name (/usr/sbin/in.tnamed).
Disabling service, talk (/usr/sbin/in.talkd).
Disabling service, finger (/usr/sbin/in.fingerd).
Disabling service, rquotad/1 (/usr/lib/nfs/rquotad).
Disabling service, rusersd/2-3 (/usr/lib/netsvc/rusers/rpc.rusersd).
Disabling service, sprayd/1 (/usr/lib/netsvc/spray/rpc.sprayd).
Disabling service, walld/1 (/usr/lib/netsvc/rwall/rpc.rwalld).
Disabling service, shell (/usr/sbin/in.rshd).
Disabling service, login (/usr/sbin/in.rlogind).
Disabling service, exec (/usr/sbin/in.rexecd).
Disabling service, comsat (/usr/sbin/in.comsat).
Disabling service, time (internal).
Disabling service, echo (internal).
Disabling service, discard (internal).
Disabling service, daytime (internal).
Disabling service, chargen (internal).
Disabling service, rstatd/2-4 (/usr/lib/netsvc/rstat/rpc.rstatd).
Disabling service, 100068/2-5 (/usr/dt/bin/rpc.cmsd).
Disabling service, 100083/1 (/usr/dt/bin/rpc.ttdbserverd).
Disabling service, 100221/1 (/usr/openwin/bin/kcms_server).
Disabling service, fs (/usr/openwin/lib/fs.auto).
Disabling service, 100232/10 (/usr/sbin/sadmind).
Disabling service, 100235/1 (/usr/lib/fs/cachefs/cachefsd).
Disabling service, printer (/usr/lib/print/in.lpd).
Disabling service, 100234/1 (/usr/lib/gss/gssd).
Disabling service, dtspc (/usr/dt/bin/dtspcd).
Disabling service, 100150/1 (/usr/sbin/ocfserv).
Disabling service, 100134/1 (/usr/lib/krb5/ktkt_warnd).
Disabling service, 100229/1 (/usr/sbin/rpc.metad).
Disabling service, 100230/1 (/usr/sbin/rpc.metamhd).
Disabling service, 100242/1 (/usr/sbin/rpc.metamedd).
Disabling service, 100155/1 (/usr/lib/smedia/rpc.smserverd).

Enabling those services listed in JASS_SVCS_ENABLE.

Enabling service, tftp (/usr/sbin/in.tftpd).

================================================================
jumpstart-secure-new.driver: Driver finished.
================================================================
```

6. Configure TCP Wrappers

Note: *TCP Wrappers is now included in Solaris 9. It was enabled by the enable-tcpwrappers.fin Finish script when JASS hardened the system. The script creates the sample TCPWrapper control files /etc/hosts.allow and /etc/hosts.deny.*

6.1. Modify TCPWrappers control files

hosts.allow
sshd: 172.16.65., 172.16.1. # Explicitly allows ssh from internal nets
in.tftpd: 172.16.1. # Explictly allows tftp from build lab net

hosts.deny
ALL: ALL # Denys everything else - This is the default

7. Configuring JumpStart and Integrating JASS

7.1. Mount the Solaris 9 distribution CD labeled 1 of 2

Note: *This will need to be done manually since volume management was disabled during hardening.*

 # mount –r –F hsfs /dev/dsk/c0t2d0s0 /cdrom

7.2. Copy the OS distribution from the CD to the jumpstart OS directory

NOTE: *The naming convention on the target directory should reflect the build number so that multiple builds of the same OS version can be installed. The example below represents Solaris 9 build 4/01*

 # cd /cdrom/sol_9_sparc/s0/Solaris_9/Tools
 # ./setup_install_server /jumpstart/OS/Solaris_9_401

When the following message is displayed disk one is complete

Verifying target directory…
Calculating the required disk space for the Solaris 9 product
Copying the CD image to disk…
Install Server Setup Complete

The disk one can now be removed and disk two mounted
 # cd /
 # umount /cdrom
 # mount –r –F hsfs /dev/dsk/c0t2d0s0 /cdrom
 # ./add_to_install_server .jumpstart/OS/Solaris_9_401

7.3. Copy the Jass toolkit to the Jumpstart root directory and create a user.init file

 # cd /opt/JUNWjass

 # cp –r * /jumpstart
 # cd /jumpstart/Drivers
 # cp user.init.SAMPLE user.init

7.4. Modify the JASS_PACKAGE_MOUNT and JASS_PATCH_MOUNT variables to be the IP address of the JumpStart server in the /user.init file created in the previous step.

7.5. Modify /etc/dfs/dfstab to share the JumpStart root directory as follows and use the share command to make the share available read only and use the dfshares command to verify

share –F nfs –o ro,anon=0 –d "Jumpstart Root Directory" /jumpstart
 # shareall
 # dfshares

RESOURCE	**SERVER ACCESS**	**TRANSPORT**
Jump001:/jumpstart	**jump001 -**	**-**

#

7.6. In order for the JumpStart server to perform client installations the client must be identified to the server by entering the client's Ethernet or MAC in the /etc/ethers file and the client's IP address in the/etc/hosts file.

 # touch /etc/ethers
 # chmod 644 /etc/ethers
Add the following entry in /etc/ethers
8:27:32:25:8f:3f client01

Add the following entry to the /etc/hosts file
172.16.1.20 client01

7.7. Add the jumpstart client to JumpStart as follows.

 # cd /jumpstart
 # ./add-client client01 Solaris_9_401 sun4u jump001

making /tftpboot
enabling tftp in /etc/initd.conf
starting rarpd
starting bootparmd
starting nfsd's
starting nfs mount
updating /etc/bootparams
copying to tftpboot

7.8. Modify the /jumpstart/rules file to instruct JumpStart how to configure the client

 # cd /jumpstart
Edit the rules file and add the following entry

hostname client01 – Profile/ jumpstart-new.profile Drivers/jumpstart-secure-new.driver
hostname = rule_keyword used to identify system attributes used in the rule_value to match a system with a corresponding value client01 = The hostname of the system being built Profile= The file that contains the Solaris configuration information for the client which would include disk layouts, Solaris cluster information and other specific information. Drivers = The driver file that contains the finish script files that JumpStart will run to harden and configure the system.

7.9. Verify the rules, profile and configuration files and scripts as follows.

 # cd /jumpstart
 # ./check

Validating rules…
Validating profile Profiles/jumpstart-new.profile…
The custom JumpStart configuration is ok.

7.10. Initiate the install from the console of the client machine

ok boot net - install
Resetting ...

Netra T1 200 (UltraSPARC-IIe 500MHz), No Keyboard
OpenBoot 4.0, 1024 MB memory installed, Serial #51112709.
Ethernet address 0:3:ba:b:eb:5, Host ID: 830beb05.

Executing last command: boot net - install
Boot device: /pci@1f,0/pci@1,1/network@c,1 File and args: - install
2aa00
Booting the 32-bit OS …

SunOS Release 5.9 Version Generic_112233-11 64-bit
Copyright 1983-2003 Sun Microsystems, Inc. All rights reserved.
Use is subject to license terms.
Configuring /dev and /devices
Using RPC Bootparams for network configuration information.
Configured interface eri0
Using sysid configuration file 176.16.1.2/jumpstart/sysidcfg
The system is coming up. Please wait.
Sarting remote procedure call (RPC) services: sysidns done.
Starting Solaris installation program …
Searching for JumpStart directory …
Checking rules.ok file …
Using profile: Profiles/jumpstart-new.profile.
Using finish script: Drivers/jumpstart-secure-new.driver
.
.
.

8. Installing and Configuring MD5-Tool

8.1. Unzip and un-tar the software package

#gunzip md5-tool-1.1.gz
 # mv md5-tool.1.1 md5-tool.1.1.tar
 # tar xvf md5-tool-1.1.tar

8.2. Modify the following Configuration files

mail_to Add email address that reports will be sent to
one address per line

localapps Add additional filesystems that need to be monitored to existing list

```
/jumpstart/*            # Everything in /jumpstart base dir
/jumpstart/Drivers/*    # Drivers files
/jumpstart/Profiles*    # Profile files
/jumpstart/Finish/*     # Finish Scripts
/jumpstart/Sysidcfg/*# Sysidcfg files
/etc/ssh/*              # SSH Configuration files and host keys
/etc/dfs/*              # NFS shares
/etc/default/*          # Default system settings
/.ssh/*                 # Root SSH keys directory
/export/home/*/.ssh/* # Users SSH Keys directory
```

Modify the following variables in the health, check_sys and baseliner scripts

```
MD5=/opt/SUNBEmd5/md5       # the location of the MD5 binary
```

TOOLDIR=/opt/md5-tool-1.1 # root directory for md5-tool

8.3. Run the baseliner script to create an MD5 hash database

 # /opt/md5-tool-1.1/baseliner

NOTE: *When baseliner completes it will create two files, md5-baseline and suid-baseline. Md5-baseline contains the md5 hash values for each file specified in the localapps configuration file modified in step 8.2. the suid-baseline contains the file names of every suid file on the system. The tool will then email the MD5 sums of the two files to the address in the mail_to script. As an extra precaution the two files are secure copied to another system nightly for safe keeping in step 11. Baseliner will need to be rerun when files are intentionally modified to establish a new baseline to check against.*

8.4. Modify root's crontab file to run the health and check_sys scripts nightly. The reports from these scripts will be emailed to the distribution list in mail_to.

58 23 * * * /opt/md5-tool-1.1/health > /dev/null 2>&1
00 01 * * * /opt/md5-tool-1.1/check_sys > /dev/null 2>&1

9. Configure Sendmail to send mail outbound only

Edit sendmail /etc/mail/submit.cf configuration file and modify

Change D{MTAHost} localhost to D{MTAHost} [your.mail.relay.server]

10. Configure Log Maintenence

10.1. Modify the logadm configuration file /etc/logadm.conf to rotate additional security logs

Add the Following entries
/var/adm/loginlog –A 60d
/var/adm/sulog –A 60d
/var/log/authlog –A 60d

These entries will rotate the logs and delete any logs older that 60 days.

11. Backup Critical Files

NOTE: *In order to script this function there must be a key exchange between the systems using a key with a null pass phrase. This is required to eliminate the need for interactive authentication.*

11.1. Create a public / private key pair with a null pass phrase for the root userid.

bash-2.05# ssh-keygen -t dsa
Generating public/private dsa key pair.
Enter passphrase(empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in id_dsa.
Your public key has been saved in id_dsa.pub.
The key fingerprint is:
md5 1024 ba:cb:95:32:7e:23:b6:2c:e4:ac:83:af:6a:49:1f:f1 root@jump001
bash-2.05#

11.2. Push the public key to the backup host

 # sftp bkuphost
Connecting to bkuphost...
root@bkuphost's password:
sftp > pwd
Remote working directory: /
sftp > cd /.ssh
sftp > put id_dsa.pub
sftp > quit

11.3. Log on to the backup server and add the public key to the /.ssh/authorized_keys

#cd /.ssh
 # cat id_dsa.pub >> authorized_keys

11.4. In order to maintain a backup copy of the customized driver files, and md5tool database they will be moved to another system by creating a tar file and transferring the tar file using secure copy (scp) with the following script.

jumpstart_server_bkup.sh

```
#!/bin/sh
md5="/opt/SUNBEmd5/md5"
BACKUP_DIR="/jumpstart/jumpstart_bkups"
DATE="`date '+%m%d%y'`"
cd /jumpstart
tar cf $BACKUP_DIR/jumpstart_Drivers.tar ./Drivers
tar cf $BACKUP_DIR/jumpstart_Profiles.tar ./Profiles
tar cf $BACKUP_DIR/jumpstart_Sysidcfg.tar ./Sysidcfg
cd /opt
tar cf $BACKUP_DIR/md5-tool_backup.tar ./md5-tool-1.1
cd $BACKUP_DIR
```

$md5 jumpstart_Drivers.tar jumpstart_Profiles.tar –
\ jumpstart_Sysidcfg.tar md5-tool_backup.tar > js_md5_sums.out
tar cvf $BACKUP_DIR/jump001_backup.tar ./*
compress $BACKUP_DIR/jump001_backup.tar
scp -pr $BACKUP_DIR/jump001_backup.tar.Z -
\root@bkup001:$BACKUP_DIR/$DATE/

11.5. Add the following entry to root's crontab file on the JumpStart server to run the script at 2:00 am Monday through Friday.

0 2 * * * /jumpstart/jumpstart_server_bkup.sh > /dev/null 2>&1

12. Reboot the server

 # init 6

Ongoing Maintenance Procedures

In order to make sure our server continues to operate smoothly we will schedule the following maintenance program.

- Ongoing Patch Management
- System Backups
- Data integrity compliance
- Ongoing Compliance monitoring and vulnerability assessments
- Auditing JumpStart policies

Patch Maintenance Schedule: All Solaris servers in the environment are required to be patched in accordance with corporate policy which states: Systems are required to be no more than the current Sun Recommended & Security Patch release minus one. And that all systems will be at the current patch cluster release within 30 days of when the internal cluster testing is complete. Once a new cluster is released and tested internally, the patch cluster will be installed on the JumpStart server and positioned in the /JumpStart/Patches directory to be installed on all new servers. The exception would be if a significant threat were to become eminent to a specific vulnerability in our environment the appropriate patch would be tested and deployed in a timeframe dictated by the Threat Management Team. The Threat Management Team's responsibility is to continually assess the threat to the environment based on the most recent intelligence information.

System Backups: Much of the data that resides on the server is static. The customized components of JumpStart however, will be archived and copied to another backup server on the intranet using scp nightly. In addition to the customized JumpStart configuration files it is important to archive and export the

MD5 Tool hash databases. This is an extra precaution in the event that the system is tampered with. The archive files will be retained for thirty days. If the data on the server is lost or compromised in any way the server can be rebuilt within a few hours and the customized data can easily be restored. Based on the server's role this is an acceptable window for recovery. The impact to operations is that new servers will not be able to be built while the JumpStart server is being recovered.

MD5 Tool Report Analysis: As stated previously, the integrity of the data maintained on the server is considered very important since every Solaris server on our network is imaged from the JumpStart Server. For that reason, both file authenticity and access permissions and must be checked periodically to ensure that unauthorized access or modifications to the image data are prevented. The MD5 Tool jobs are run nightly and reports are email to administrators for review daily. Any abnormalities are addressed immediately. As with any production environment, all changes to a system require remediation thorough the corporate change control system. All changes to the system including patching, installing new OS images, modifying and installing any configuration scripts and files, etc. must be authorized, documented and scheduled through the corporate change control facility.

Ongoing Vulnerability Assessment and Compliance Monitoring: In order to minimize the effort required to administer and secure the Solaris environment, a decision was made early on to standardize and maintain a consistent server image and build methodology. The development of standardized environment begins with the classification of each server regarding its role or function on the network (i.e. web server or database server), and most importantly, the sensitivity of the data contained on the server and its impact to the business. In other words, if the data is compromised or lost, or the application that the server runs is unavailable, will the business suffer a significant loss? As one would assume the server roles are closely associated with the network services and application that they make available. With JumpStart we are able to harden the servers based on their function by customizing profiles for each role. In the cases where a unique application requires a variation we can use one of our existing profiles as a template to work from and test for which a new standard can be developed if necessary.

Another advantage to standardization is that it makes it easier to do compliance monitoring, auditing, policy enforcement, and vulnerability assessments. In a standardized environment things that deviate from the standard tend to draw attention more quickly, especially once a baseline of the environment has been established. Currently we perform ongoing vulnerability assessments of our Solaris environment using the nCircle Device Profiler (DP). The DP will automatically detect, scan and associate a score to a server and retain the profile in a database. If the server's configuration changes and a new vulnerability is detected in a subsequent scan, it will by reflected in the score for that server.

nCircle can be configured to alert an administrator based on the delta in the score and the appropriate action can be taken. An example of the nCircle reports will be seen later in the System Testing and Verification section.

JASS Auditing: The JASS toolkit also includes an auditing function that will check the systems current setting against the secure.driver script that was used to harden the system. This will be scheduled to run periodically and the results will be emailed to the system administrators and compliance monitoring groups for review. This is a sample of the output report from the audit run on our JumpStart server.

 # jass-execute –a JumpStart-secure-new.driver –V 2 –o /var/adm/jass_audit_runs/[logfile name]

update-inetd-conf	[PASS] File /etc/netconfig was found.
update-inetd-conf	[NOTE] Service tftp was found in both DISABLE and ENABLE lists.
update-inetd-conf	[PASS] Service ftp is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service telnet is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service name is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service talk is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service uucp does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service smtp does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service finger is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service systat is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service netstat is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service rquotad is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service rusersd is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service sprayd is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service walld is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service rexd is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service shell is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service login is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service exec is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service comsat is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service time is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service echo is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service discard is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service daytime is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service chargen is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service 100087 does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service rwalld does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service rstatd is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100068 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100083 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100221 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service fs is disabled in /etc/inet/inetd.conf.

update-inetd-conf	[NOTE] Service ufsd does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100232 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100235 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service 536870916 does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service kerbd does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service printer is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100234 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service dtspc is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service xaudio does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service 100146 does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service 100147 does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100150 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100134 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100229 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100230 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100242 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service 300326 does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100232 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100068 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100083 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100221 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100235 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service 100155 is disabled in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service uuidgen does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service sun-dr does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service kshell does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service klogin does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[NOTE] Service eklogin does not exist in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Service tftp is enabled in /etc/inet/inetd.conf.
update-inetd-conf	[PASS] Audit Check Total : 0 Error(s)

System Testing and Verification

In order to confirm that the JumpStart server is in compliance with corporate standards, and that our hardening scripts are functioning correctly, we will run the following battery of tests.

CIS Benchmark Tool
The script executes and produces its output in /opt/CIS/cis-ruler-log.[date-time-stamp]. The file can be examined by entering the following commands

 # egrep "^Positive" /opt/CIS/cis-ruler-log.20040429-14:43:48.6116|more
 # egrep "^Negative" /opt/CIS/cis-ruler-log.20040429-14:43:48.6116|more

The respective commands filter the negative and positive results. Positive conditions represent good security measures in effect, while negative conditions

represent potentially unfavorable security conditions. While negative conditions exist they may be acceptable for the role of the server.

Running the tail command against the CIS output file will display the summarized score of the tools findings with ten being the most secure and zero being the least. The following summaries are before and after results.

#tail /opt/CIS/cis-ruler-log.20040429-14:43:48.6116

CIS Benchmark Results *before* Hardening

```
Preliminary rating given at time: Mon Apr 26 17:38:22 2004

      Preliminary rating = 2.91 / 10.00
Positive: 6.7 No non-standard world-writable files.
Positive: 6.8 No non-standard SUID/SGID programs found.
Ending run at time: Mon Apr 26 17:38:38 2004

      Final rating = 3.16 / 10.00
```

CIS Benchmark Results *after* Hardening

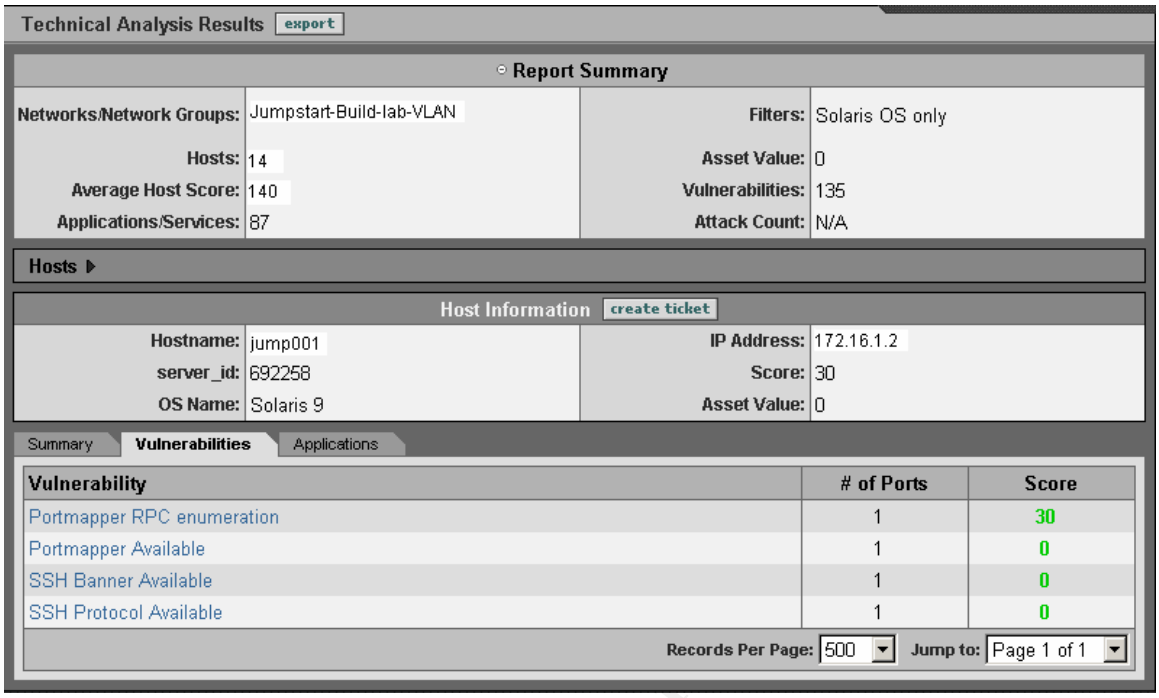
```
Preliminary rating given at time: Thu Apr 29 14:43:49 2004

      Preliminary rating = 6.20 / 10.00
Positive: 6.7 No non-standard world-writable files.
Positive: 6.8 No non-standard SUID/SGID programs found.
Ending run at time: Thu Apr 29 14:43:52 2004

      Final rating = 6.46 / 10.00
```

The results show that the server's score doubled after the hardening was applied, resulting in a more secure system.

<u>nCircle Vulnerability Assessments</u>

The nCircle Device Profiler (DP) will automatically detect and scan hosts as they come on to the network. We have modified the access control lists of the router between the Jumpstart Build Lab and our internal network to allow the DP to proactively scan the lab network. This will allow the DP to scan and assess both the JumpStart server and the clients it builds and hardens.

This report shows the state of the JumpStart server after the hardening has been completed. The server received a score of thirty, which will be archived. Since the server should not require any additional services and its configuration should remain static, nCircle will be configured to email the system administrator if the score deviates in future scans.



Technical Analysis Results export

Report Summary

Networks/Network Groups:	Jumpstart-Build-lab-VLAN	Filters:	Solaris OS only
Hosts:	14	Asset Value:	0
Average Host Score:	140	Vulnerabilities:	135
Applications/Services:	87	Attack Count:	N/A

Hosts

Host Information create ticket

Hostname:	jump001	IP Address:	172.16.1.2
server_id:	692258	Score:	30
OS Name:	Solaris 9	Asset Value:	0

Summary | Vulnerabilities | Applications

Vulnerability	# of Ports	Score
Portmapper RPC enumeration	1	30
Portmapper Available	1	0
SSH Banner Available	1	0
SSH Protocol Available	1	0

Records Per Page: 500 Jump to: Page 1 of 1

The following report shows the nCircle scan results of the same server prior to hardening. Note the significant difference in the score, 7519 before versus 30 after. The nCircle scoring values are weighted based on the type of vulnerability and its impact to the system and overall security. A higher overall score represents a less secure system. A server with only a few high-risk vulnerabilities could score higher that a server with several low risk exposures.

Report Summary

Hosts	469	Vulnerabilities	135
Asset Value	0	Applications/Services	87
Average Host Score	1760	Attack Count	0

Host Summary

Hostname	IP Address	Owner	Asset Value	Score	OS
jump001	172.16.1.2			7519	Solaris 9

Operating System

OS Name
Solaris 9

Applications

Application	Port
Sendmail 8.12.x	587
Sendmail 8.12.x	25
Solaris Telnet	23
Sun RPC service over TCP	32771
Sun RPC service over TCP	32773
Sun RPC service over TCP	32774
Sun RPC service over TCP	32775
Sun RPC service over TCP	32776
Sun RPC service over TCP	32777
Sun RPC service over TCP	32778
Sun RPC service over TCP	32780
Sun RPC service over UDP	111
Sun SSH	22
SunOS LPD	515
Undetermined Application.	32779
Undetermined Application. Common applications on this port are: font-service	7100
Undetermined Application. Common applications on this port are: lockd	4045
Undetermined Application. Common applications on this port are: who, rlogin	513
WU-FTPd	21
chargen	19
daytime	13
dtspc on Sun	6112
echo tcp	7
echo udp	7
finger	79
rexec	512
rpc.nlockmgr	111
rpc.rstatd	111
rpc.rusers	111
rpc.sadmind	111
rpc.sprayd	111
rpc.statd	111
rpc.ttdbservd	111
rpc.walld	111
rshell	514
sadmind Remote Admin Service	32772
snmp	161

Nmap port scans

Another method used to validate our hardening results is to see what network services are being made available by the system using nmap. Nmap is a free port scanner that can quickly assess the state of network services. It can be downloaded from www.sunfreeware.com. Nmap must be installed and run from another system in order to get an external assessment of our server. The following is a before and after assessment of our server:

Pre Hardening nmap port scan results

```
# nmap 3.50 scan initiated Mon Apr 26 16:16:14 2004 as: nmap -sS -O -sV
Interesting ports on jump001 (172.16.1.2):
(The 1630 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE         VERSION
7/tcp     open  echo
9/tcp     open  discard?
13/tcp    open  daytime         Sun Solaris daytime
19/tcp    open  chargen
21/tcp    open  ftp             Solaris ftpd
22/tcp    open  ssh             SunSSH 1.0 (protocol 2.0)
23/tcp    open  telnet          Sun Solaris telnetd
25/tcp    open  smtp            Sendmail 8.12.2+Sun/8.12.2
37/tcp    open  time
79/tcp    open  finger          Sun Solaris fingerd
111/tcp   open  rpcbind         2-4 (rpc #100000)
512/tcp   open  exec
513/tcp   open  rlogin
514/tcp   open  shell?
515/tcp   open  printer         Solaris lpd
587/tcp   open  smtp            Sendmail 8.12.2+Sun/8.12.2
4045/tcp  open  nlockmgr        1-4 (rpc #100021)
6112/tcp  open  dtspc?
7100/tcp  open  font-service    Sun Solaris fs.auto
32771/tcp open  ttdbserverd     1 (rpc #100083)
32772/tcp open  sometimes-rpc7?
32773/tcp open  metad           1 (rpc #100229)
32774/tcp open  metamhd         1 (rpc #100230)
32775/tcp open  rpc.metamedd    1 (rpc #100242)
32776/tcp open  rusersd         2-3 (rpc #100002)
32777/tcp open  status          1 (rpc #100024)
32778/tcp open  dmispd          1 (rpc #300598)
32779/tcp open  sometimes-rpc21?
32780/tcp open  snmpXdmid       1 (rpc #100249)
Device type: general purpose
Running: Sun Solaris 9
OS details: Sun Solaris 9
Uptime 0.008 days (since Mon Apr 26 16:07:17 2004)
```

```
# Nmap run completed at Mon Apr 26 16:18:49 2004 -- 1 IP address (1 host up)
scanned in 155.313 seconds
```

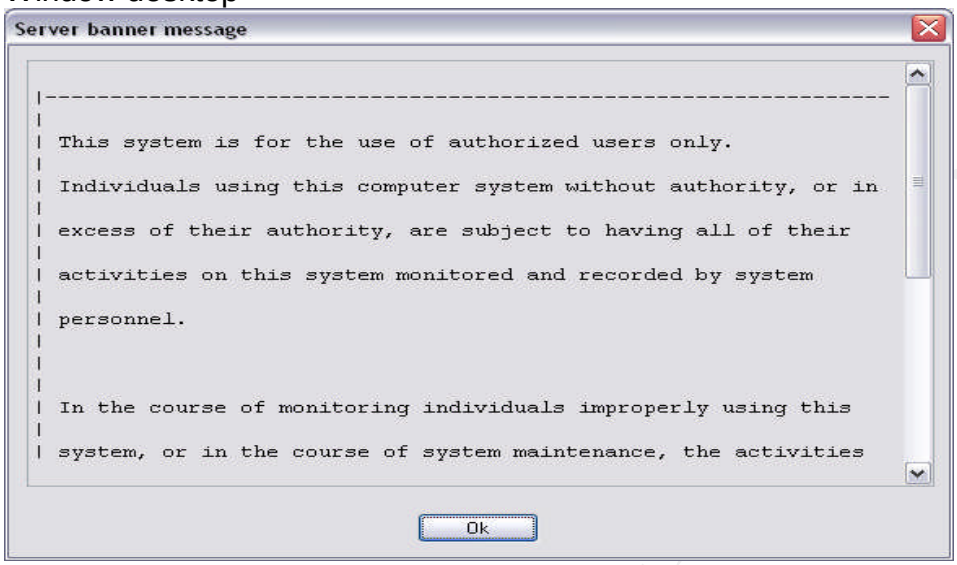
Post Hardening nmap port scan results

```
# nmap -sS -sU -sV -O 172.16.1.2

Starting nmap 3.50 ( http://www.insecure.org/nmap/ ) at 2004-04-29 22:47 EDT
Interesting ports on jump001 (171.16.1.2):
(The 3132 ports scanned but not shown below are in state: closed)
PORT      STATE SERVICE        VERSION
22/tcp    open  ssh            SunSSH 1.0 (protocol 2.0)
69/udp    open  tftp?
111/tcp   open  rpcbind        2-4 (rpc #100000)
111/udp   open  rpcbind        2-4 (rpc #100000)
32771/udp open  sometimes-rpc6?
Device type: general purpose
Running: Sun Solaris 9
OS details: Sun Solaris 9 with TCP_STRONG_ISS set to 2
Uptime 1.313 days (since Wed Apr 28 15:20:19 2004)

Nmap run completed -- 1 IP address (1 host up) scanned in 231.285 seconds
```

As expected the results are vastly different. After applying the JASS hardening the only TCP ports available are SSH which is required for remote administration and rpcbind(NFS) and tftp for JumpStart. Also note that by using the –O switch we successfully fingerprinted the OS as Solaris 9. The –sV switch probes open ports and attempts to determine service and application names and versions. Note the level of detail regarding the application and version results on the "before" run.

Banner Audit

Another compliance requirement is the validation of Banners
The following banner is displayed when attempting an ssh connection from a Window desktop



System Logging and Access Control

The following log entries validate that TCP Wrappers and SYSLOG are configured correctly and are monitoring network connections and logging them. The entries were taken from the central log server and show both tftp and ssh entries

Failed tftp connection attempts

Jun 3 19:45:35 jump001 in.tftpd[17613]: [ID 947420 auth.warning] refused connect from 10.209.32.84
Jun 3 19:45:41 jump001 in.tftpd[17614]: [ID 947420 auth.warning] refused connect from 10.209.32.84

Successful SSH connection via password

Jun 3 10:40:20 jump001 sshd[16917]: [ID 800047 auth.info] Failed publickey for myuserid from 172.16.1.20 port 2302 ssh2
Jun 3 10:40:20 jump001 sshd[16917]: [ID 800047 auth.info] Failed none for myuserid from 172.16.1.20 port 2302 ssh2
Jun 3 10:40:33 jump001 sshd[16917]: [ID 800047 auth.info] Accepted password for myuserid from 172.16.1.20 port 2302 ssh2

TCP Wrappers dropping a SSH session attempt from un-trusted host “badhost”

Jun 3 20:13:12 badhost sshd[456]: [ID 550680 auth.error] warning: /etc/hosts.allow, line 14: can't verify hostname: gethostbyname(HIGTEKWZT496) failed
Jun 3 20:13:12 badhost sshd[456]: [ID 947420 auth.warning] refused

```
connect from 10.209.97.188
```

Active System Processes

The following results show the difference in active processes before and after hardening

Note that the before hardening display shows several daemons and services active that will be disabled after hardening.

ps –ef before hardening

```
sh-2.05# ps –ef
    UID   PID  PPID  C    STIME TTY      TIME CMD
   root     0     0  0   Jun 25 ?       0:18 sched
   root     1     0  0   Jun 25 ?       0:00 /etc/init -
   root     2     0  0   Jun 25 ?       0:00 pageout
   root     3     0  0   Jun 25 ?      18:57 fsflush
   root   257     1  0   Jun 25 ?       0:00 /usr/lib/saf/sac -t 300
 nobody   718   140  0   Jun 26 ?       0:00 fs
   root   719   140  0   Jun 26 ?       0:00 rpc.ttdbserverd
   root    47     1  0   Jun 25 ?       0:00 /usr/lib/sysevent/syseventd
   root    55     1  0   Jun 25 ?       0:00 /usr/lib/picl/picld
   root   119     1  0   Jun 25 ?       0:00 /usr/sbin/rpcbind
   root   140     1  0   Jun 25 ?       0:00 /usr/sbin/inetd -s
 daemon   155     1  0   Jun 25 ?       0:00 /usr/lib/nfs/statd
   root   164     1  0   Jun 25 ?       0:00 /usr/lib/autofs/automountd
   root   238     1  0   Jun 25 ?       0:33 /usr/lib/snmp/snmpdx -y -c
/etc/snmp/conf
   root   175     1  0   Jun 25 ?       0:00 /usr/sbin/syslogd
   root   156     1  0   Jun 25 ?       0:00 /usr/lib/nfs/lockd
   root   198     1  0   Jun 25 ?       0:00 /usr/lib/power/powerd
   root   183     1  0   Jun 25 ?       0:00 /usr/sbin/cron
   root   195     1  0   Jun 25 ?       0:01 /usr/sbin/nscd
   root   213     1  0   Jun 25 ?       0:00 /usr/sbin/vold
   root   211     1  0   Jun 25 ?       0:00 /usr/sadm/lib/wbem/cimomboot start
   root   209     1  0   Jun 25 ?       0:00 /usr/lib/utmpd
   root   262     1  0   Jun 25 ?       0:00 /usr/lib/ssh/sshd
   root   247     1  0   Jun 25 ?       0:00 /usr/lib/dmi/dmispd
   root   232     1  0   Jun 25 ?       0:04 /usr/lib/sendmail -bd -q15m
  smmsp   235     1  0   Jun 25 ?       0:02 /usr/lib/sendmail -Ac -q15m
   root   726   140  0   Jun 26 ?       0:00 sadmind
   root   261   257  0   Jun 25 ?       0:00 /usr/lib/saf/ttymon
   root   246     1  0   Jun 25 ?       0:01 /usr/dt/bin/dtlogin -daemon
   root   271   238  0   Jun 25 ?       7:27 mibiisa -r -p 32793
   root   250     1  0   Jun 25 ?       0:00 /usr/lib/dmi/snmpXdmid -s jumo001
   root  2985     1  0   Jun 29 console  0:00 /usr/lib/saf/ttymon -g -h -p jump01
```

```
console login:  -T sun -d /dev/console -
l
   root   720   140  0   Jun 26 ?       0:00 rpc.metad
   root   721   140  0   Jun 26 ?       0:00 rpc.metamhd
   root   722   140  0   Jun 26 ?       0:00 rpc.metamedd
   root   728   140  0   Jun 26 ?       0:00 rpc.rstatd
  daemon   727   140  0   Jun 26 ?       0:00 rpc.cmsd
   root  3538   262  0 19:22:14 ?       0:01 /usr/lib/ssh/sshd
   root  3607  3606  0 19:28:51 pts/2    0:00 sh -i
 myuserid  3540  3538  0 19:22:19 pts/1    0:00 -bash
   root  3554  3540  0 19:23:32 pts/1    0:00 -bash
   root  3610  3607  1 19:29:03 pts/2    0:00 ps -ef
```

ps –ef after hardening

```
    UID   PID  PPID  C    STIME TTY      TIME CMD
   root     0     0  0 19:40:10 ?       0:18 sched
   root     1     0  0 19:40:10 ?       0:00 /etc/init -
   root     2     0  0 19:40:10 ?       0:00 pageout
   root     3     0  0 19:40:10 ?       0:01 fsflush
   root   351     1  0 19:40:36 ?       0:00 /usr/lib/saf/sac -t 300
   root   382   347  0 19:54:28 ?       0:01 /usr/lib/ssh/sshd
   root    53     1  0 19:40:27 ?       0:00 /usr/lib/sysevent/syseventd
   root    61     1  0 19:40:28 ?       0:00 /usr/lib/picl/picld
   root   261     1  0 19:40:33 ?       0:00 /usr/sbin/rpcbind
   root   287     1  0 19:40:34 ?       0:00 /usr/sbin/syslogd
   root   295     1  0 19:40:34 ?       0:00 /usr/sbin/cron
   root   302     1  0 19:40:34 ?       0:00 /usr/sbin/nscd
   root   280     1  0 19:40:33 ?       0:00 /usr/sbin/inetd -s -t
   root   354   351  0 19:40:36 ?       0:00 /usr/lib/saf/ttymon
   root   313     1  0 19:40:35 ?       0:00 /usr/lib/utmpd
   root   352     1  0 19:40:36 console  0:00 /usr/lib/saf/ttymon -g -h -p
jump001 console login:  -T sun -d /dev/console -
l
   root   347     1  0 19:40:36 ?       0:00 /usr/lib/ssh/sshd
   root   461   434  0 20:18:19 pts/1    0:00 ps -ef
 myuserid   384   382  0 19:54:33 pts/1    0:00 -bash
  smmsp   337     1  0 19:40:36 ?       0:00 /usr/lib/sendmail -Ac -q15m
   root   338     1  0 19:40:36 ?       0:00 /usr/lib/sendmail -bd -q15m
   root   434   384  0 20:10:15 pts/1    0:00 -bash
```

MD5 Tool Integrity Checking

The MD5 Tool runs a nightly check of files against its baseline. If no files have been modified no notifications are emailed. The data below is that MD5-Tool detected that a file was modified in the /jumpstart/Drivers directory.

MD5 Report shows problems!
1477c1477 < MD5 (/jumpstart/Drivers/jumpstart-hardening-new.driver) = c9145fbdcf175d803a9ec092f917cccd --- > MD5 (/jumpstart/Drivers/jumpstart-hardening-new.driver) = ce6bb028dfcbba808e6f3a7fe6905909

The message displays the baseline MD5 value and the current value of the modified file.

Appendix A: Sample finish.init Script used to establish environmental variables for the finish scripts called by the JumpStart-hardening-new.drivers to harden the Jumpstart Server.

```
#
# Copyright (c) 2000-2003 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident      "@(#)finish.init  3.8   03/04/11    SMI"
#
# This file contains all of the variable definitions used by any of the
# default JASS Finish scripts.  These variables serve as the default
values
# that are used by the JASS Toolkit.  All of these values can be
modified
# using the "user.init" file.  Any variables that are defined in the
# "user.init" file will not be replaced by this script.
#
# To augment these values (rather than replace them), this file can be
# sourced by the "user.init" file and then the variables can be
extended
# in the normal shell fashion (i.e., VAR="${VAR} new_value").
Furthermore,
# more complex functions can be included into "user.init" to provide
# advanced filtering and assignment capabilities.

#
================================================================================
=====
# Define the JASS Finish script specific variables.
#
================================================================================
=====

#
================================================================================
=====
# JASS_KILL_SCRIPT_DISABLE
#
# This variable contains a boolean value that determines whether the
"kill" # run control scripts (in a given Finish script) will be
disabled.  Note # that the "start" run control scripts are always
disabled.  Some
# administrators prefer to have the "kill" scripts left in place so
that
# any services that may be started will be properly terminated during a
# system shutdown/reboot.
#
================================================================================
=====

if [ -z "${JASS_KILL_SCRIPT_DISABLE}" ]; then
   JASS_KILL_SCRIPT_DISABLE="1"
fi
export JASS_KILL_SCRIPT_DISABLE
```

```
#
================================================================================
=====
# JASS_PASSWD
#
# This variable contains a filename value that specifies the location
of
# the JumpStart client's password file.  This variable is used by many
of # the variables below.
#
================================================================================
=====

if [ -z "${JASS_PASSWD}" ]; then
   JASS_PASSWD="${JASS_ROOT_DIR}/etc/passwd"
fi
export JASS_PASSWD

#
================================================================================
=====
# JASS_ACCT_DISABLE
#
# This variable contains a (possibly empty) list of users to be
disabled
# as part of the disable-system-accounts.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_ACCT_DISABLE}" ]; then
   JASS_ACCT_DISABLE="
      daemon bin adm lp uucp nuucp nobody smtp listen noaccess nobody4
smmsp"
fi
export JASS_ACCT_DISABLE

#
================================================================================
=====
# JASS_ACCT_REMOVE
#
# This variable contains a (possibly empty) list of users to be removed
# from the system as part of the remove-uneeded-accounts.fin Finish
script.
#
================================================================================
=====

if [ -z "${JASS_ACCT_REMOVE}" ]; then
   JASS_ACCT_REMOVE="smtp listen nobody4"
fi
export JASS_ACCT_REMOVE
```

```
#
================================================================================
=====
# JASS_AGING_MINWEEKS
#
# This variable contains a numeric value specifying the minimum number
# of weeks that must pass before a user can change their password.
This
# variable is used in the set-user-password-reqs.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_AGING_MINWEEKS}" ]; then
   JASS_AGING_MINWEEKS="1"
fi
export JASS_AGING_MINWEEKS

#
================================================================================
=====
# JASS_AGING_MAXWEEKS
#
# This variable contains a numeric value specifying the maxinum number
# of weeks a password remains valid before it must be changed by the
user.
# This variable is used in the set-user-password-reqs.fin Finish
script.
#
================================================================================
=====

if [ -z "${JASS_AGING_MAXWEEKS}" ]; then
   JASS_AGING_MAXWEEKS="8"
fi
export JASS_AGING_MAXWEEKS

#
================================================================================
=====
# JASS_AGING_WARNWEEKS
#
# This variable contains a numeric value specifying the number of weeks
# before a password expiration that the user is warned.  This variable
is # used in the set-user-password-reqs.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_AGING_WARNWEEKS}" ]; then
   JASS_AGING_WARNWEEKS="1"
fi
export JASS_AGING_WARNWEEKS

#
================================================================================
=====
```

```
# JASS_AT_ALLOW
#
# This variable contains a (possibly empty) list of users to be added
to the
# /etc/cron.d/at.allow file.  Note that a user will not be added to
this file
# if it already exists in /etc/cron.d/at.deny or if it does not exist
in
# ${JASS_PASSWD}.  This variable is used by the install-at-allow.fin
# Finish script.
#
================================================================================

if [ -z "${JASS_AT_ALLOW}" ]; then
   JASS_AT_ALLOW=""
fi
export JASS_AT_ALLOW

#
================================================================================
# JASS_AT_DENY
#
# This variable contains a (possibly empty) list of users to be added
to
# the /etc/cron.d/at.deny file.  Note that a user will not be added to
this
# file if it already exists in /etc/cron.d/at.allow or if it does not
exist
# in ${JASS_PASSWD}.  This variable is used by the update-at-deny.fin
# Finish script.
#
================================================================================

if [ -z "${JASS_AT_DENY}" ]; then
   JASS_AT_DENY="
      `awk -F: '{ print $1 }' ${JASS_PASSWD}`"
fi
export JASS_AT_DENY


#
================================================================================
# JASS_BANNER_DTLOGIN
#
# This variable contains a string value that is the file which contains
# the dtlogin banner for CDE connections.
# For more information, refer section discussing the BANNER variable in
the
# dtlogin(1M) manual page.  This variable is used by the
# set-banner-dtlogin.fin Finish script.
#
================================================================================
```

```
if [ -z "${JASS_BANNER_DTLOGIN}" ]; then
   JASS_BANNER_DTLOGIN="/etc/motd"
fi
export JASS_BANNER_DTLOGIN

#
================================================================================
=====
# JASS_BANNER_FTPD
#
# This variable contains a string value that will be used as a banner
# displayed to a user prior to authenticating to this service.  For
more
# information, refer section discussing the BANNER variable in the
# in.ftpd(1M) manual page.  This variable is used by the set-banner-
ftpd.fin
# Finish script.
#
================================================================================
=====

if [ -z "${JASS_BANNER_FTPD}" ]; then
   JASS_BANNER_FTPD="\"Authorized Use Only\""
fi
export JASS_BANNER_FTPD

#
================================================================================
=====
# JASS_BANNER_TELNETD
#
# This variable contains a string value that will be used as a banner
# displayed to a user prior to authenticating to this service.  For
more
# information, refer section discussing the BANNER variable in the
# in.telnetd(1M) manual page.  This variable is used by the
# set-banner-telnetd.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_BANNER_TELNETD}" ]; then
   JASS_BANNER_TELNETD="\"Authorized Use Only\""
fi
export JASS_BANNER_TELNETD

#
================================================================================
=====
# JASS_BANNER_SENDMAIL
#
# This variable contains a string value that will be used as a banner
that
# is displayed when a remote host connects to port 25.
# For more information, refer to the sendmail(1M) manual page.
# This variable is used by the set-banner-sendmail.fin Finish script.
```

```
#
================================================================================
=====

if [ -z "${JASS_BANNER_SENDMAIL}" ]; then
   JASS_BANNER_SENDMAIL="Mail Server Ready"
fi
export JASS_BANNER_SENDMAIL

#
================================================================================
=====
# JASS_BANNER_SSHD
#
# This variable contains a string for the banner file that ssh will
# display on a successful login.
# For more information, refer to the sshd(1M) manual page.
# This variable is used by the set-banner-sshd.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_BANNER_SSHD}" ]; then
   JASS_BANNER_SSHD="/etc/issue"
fi
export JASS_BANNER_SSHD

#
================================================================================
=====
# JASS_CORE_PATTERN
#
# This variable contains the path and core dump file pattern for
# coreadm.  This variable is used by the enable-coreadm.fin Finish
# script.
#
================================================================================
=====

if [ -z "${JASS_CORE_PATTERN}" ]; then
   JASS_CORE_PATTERN="/var/core/core.%f.%p.%n.%u.%g.%t"
fi
export JASS_CORE_PATTERN

#
================================================================================
=====
# JASS_CPR_MGT_USER
#
# This variable contains a string value that will be used to define
# what user(s) will be permitted to perform checkpoint resume
functions.
# This default value for this variable is "-" which indicates that only
# the 'root' account will be able to perform these management
functions.
# For more information, refer to the /etc/default/power file.  This
# variable is used in the set-power-restrictions.fin Finish script.
```

```
#
================================================================================
=====

if [ -z "${JASS_CPR_MGT_USER}" ]; then
   JASS_CPR_MGT_USER="-"
fi
export JASS_CPR_MGT_USER

#
================================================================================
=====
# JASS_CRON_ALLOW
#
# This variable contains a (possibly empty) list of users to be added
to the
# /etc/cron.d/cron.allow file.  Note that a user will not be added to
this
# file if it already exists in /etc/cron.d/cron.deny or if it does not
exist
# in ${JASS_PASSWD}.  This variable is used by the update-cron-
allow.fin
# Finish script.
#
================================================================================
=====

if [ -z "${JASS_CRON_ALLOW}" ]; then
   JASS_CRON_ALLOW="root"
fi
export JASS_CRON_ALLOW

#
================================================================================
=====
# JASS_CRON_DENY
#
# This variable contains a (possibly empty) list of users to be added
to the
# /etc/cron.d/cron.deny file.  Note that a user will not be added to
this
# file if it already exists in /etc/cron.d/cron.allow or if it does not
exist
# in ${JASS_PASSWD}.  This variable is used by the update-cron-
allow.fin
# Finish script.
#
================================================================================
=====

if [ -z "${JASS_CRON_DENY}" ]; then
   JASS_CRON_DENY="
      `awk -F: '($3 < 100) || ($3 > 60000) { print $1 }' ${JASS_PASSWD}
|\
          grep -vw root`"
fi
export JASS_CRON_DENY
```

```
#
================================================================================
=====
# JASS_CRON_LOG_SIZE
#
# This variable contains a numeric value representing the maximum size
(in
# blocks) of /var/cron/log file.  If the file exceeds this maximum
limit,
# it will be moved to /var/cron/olog as part of the
/etc/cron.d/logchecker
# script.  The default value for this script was 1024 (0.5MB).  This
value
# is changed to be 20480 (10.0MB).  This variable is used by the
# update-cron-log-size.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_CRON_LOG_SIZE}" ]; then
   JASS_CRON_LOG_SIZE="20480"
fi
export JASS_CRON_LOG_SIZE

#
================================================================================
=====
# JASS_FIXMODES_DIR
#
# This variable contains the path name where the Fix Modes program, if
# found will be installed by the install-fix-modes.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_FIXMODES_DIR}" ]; then
   JASS_FIXMODES_DIR="/opt"
fi
export JASS_FIXMODES_DIR

#
================================================================================
=====
# JASS_FIXMODES_OPTIONS
#
# This variable contains a (possibly empty) list of options to be
passed to
# the Fix Modes program upon execution by the install-fix-modes.fin
Finish
# script.
#
================================================================================
=====

if [ -z "${JASS_FIXMODES_OPTIONS}" ]; then
   JASS_FIXMODES_OPTIONS=""
```

```
fi
export JASS_FIXMODES_OPTIONS

#
================================================================================
=====
# JASS_MD5_DIR
#
# This variable contains the path name where the MD5 executables, if
# found will be installed by the install-md5.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_MD5_DIR}" ]; then
   JASS_MD5_DIR="/opt"
fi
export JASS_MD5_DIR

#
================================================================================
=====
# JASS_FTPUSERS
#
# This variable contains a (possibly empty) list of users to be added
to the
# /etc/ftpusers file.  This variable is used by the install-
ftpusers.fin
# Finish script.
#
================================================================================
=====

if [ -z "${JASS_FTPUSERS}" ]; then
   JASS_FTPUSERS="
      `awk -F: '($3 < 100) || ($3 > 60000) { print $1 }'
${JASS_PASSWD}`"
fi
export JASS_FTPUSERS

#
================================================================================
=====
# JASS_FTPD_UMASK
#
# This variable contains a numeric (octal) value to be used for the
# default file creation mask used by the in.ftpd(1M) daemon.  This
# variable is used in the set-ftpd-umask.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_FTPD_UMASK}" ]; then
   JASS_FTPD_UMASK="022"
fi
export JASS_FTPD_UMASK
```

```
#
================================================================================
=====
# JASS_LOGIN_RETRIES
#
# This variable contains a numeric value specifying the number of
# consequtive failed logins that can occur for a user before the
# login process logs the failure and terminates the connection.
# This variable is used in the set-login-retries.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_LOGIN_RETRIES}" ]; then
   JASS_LOGIN_RETRIES="3"
fi
export JASS_LOGIN_RETRIES

#
================================================================================
=====
# JASS_PASS_LENGTH
#
# This variable contains a numeric value specifying the minimum length
# of a user's password.  The current range is between 1 and 8.  This
# variable is used in the set-user-password-reqs.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_PASS_LENGTH}" ]; then
   JASS_PASS_LENGTH="8"
fi
export JASS_PASS_LENGTH

#
================================================================================
=====
# JASS_POWER_MGT_USER
#
# This variable contains a string value that will be used to define
# what user(s) will be permitted to perform power management functions.
# This default value for this variable is "-" which indicates that only
# the 'root' account will be able to perform power management
functions.
# For more information, refer to the /etc/default/power file.  This
variable
# is used in the set-power-restrictions.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_POWER_MGT_USER}" ]; then
   JASS_POWER_MGT_USER="-"
fi
export JASS_POWER_MGT_USER
```

```
#
================================================================================
=====
# JASS_REC_PATCH_OPTIONS
#
# This variable contains a string value that specifies options to be
# passed to the patchadd/installpatch command when installing the
# Solaris Recommended/Security Patch Cluster.  The default is to pass
# no arguments.  For information on the available options, refer to the
# patchadd(1M) manual page or the installpatch program code.  This
variable
# is used by the install-recommended-patches.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_REC_PATCH_OPTIONS}" ]; then
   JASS_REC_PATCH_OPTIONS=""
fi
export JASS_REC_PATCH_OPTIONS

#
================================================================================
=====
# JASS_RHOSTS_FILE
#
# This variable contains a filename value that specifies the location
where
# the print-rhosts.fin Finish script will store its output.  If no
value is
# specified, then the script will print its output to "standard
output".
#
================================================================================
=====

if [ -z "${JASS_RHOSTS_FILE}" ]; then
   JASS_RHOSTS_FILE=""
fi
export JASS_RHOSTS_FILE

#
================================================================================
=====
# JASS_ROOT_GROUP
#
# This variable contains a numeric value that will be used as root's
# primary group identifier value.  By default, this value is set to
# "0" (root).  This will override the OS shipped default value of "1"
# (other).  This variable is used in the set-root-group.fin Finish
script.
#
================================================================================
=====

if [ -z "${JASS_ROOT_GROUP}" ]; then
   JASS_ROOT_GROUP="0"
```

```
fi
export JASS_ROOT_GROUP

#
================================================================================
=====
# JASS_ROOT_PASSWORD
#
# This variable contains a string value that will be used as root's
# encrypted password by the set-root-password.fin Finish script.  Note
that
# this script will only operate when the system is running from
miniroot
# during a JumpStart installation to prevent root's password from being
# accidentally overwritten.  From more information on this default
password,
# refer to the JASS documentation.
#
================================================================================
=====

if [ -z "${JASS_ROOT_PASSWORD}" ]; then
   JASS_ROOT_PASSWORD="JdqZ5HrSDYM.o"
fi
export JASS_ROOT_PASSWORD

#
================================================================================
=====
# JASS_SADMIND_OPTIONS
#
# This variable contains a string value that specifies the options that
# will be used with the "sadmind" daemon that is executed from the
"inetd"
# process.  By default, a value of "-S 2" is used to enable strong
# authentication (AUTH_DES) when communicating with clients.  This
variable
# is used in the install-sadmind-options.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_SADMIND_OPTIONS}" ]; then
   JASS_SADMIND_OPTIONS="-S 2"
fi
export JASS_SADMIND_OPTIONS

#
================================================================================
=====
# JASS_SENDMAIL_MODE
#
# This variable contains a string value that specifies the options that
# will be used by /usr/lib/sendmail for its mode.  For example, if the
# daemon should accept incoming SMTP connections, then the string "-bd"
# should be used.  If the daemon should only perform queue processing,
then
```

```
# the empty string ("") should be used.  This variable is used in the
# disable-sendmail.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_SENDMAIL_MODE}" ]; then
   JASS_SENDMAIL_MODE="\"\""
fi
export JASS_SENDMAIL_MODE

#
================================================================================
=====
# JASS_SGID_FILE
#
# This variable contains a filename value that specifies the location
where
# the print-sgid-files.fin Finish script will store its output.  If no
value
# is specified, then the script will print its output to "standard
output".
#
================================================================================
=====

if [ -z "${JASS_SGID_FILE}" ]; then
   JASS_SGID_FILE=""
fi
export JASS_SGID_FILE

#
================================================================================
=====
# JASS_SHELLS
#
# This variable contains a list of shells to be added to the
/etc/shells
# file.  The default shells for each version of the Solaris Operating
# Environment are included below.  This variable is used by the
# install-shells.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_SHELLS}" ]; then

   # These shells are by default found in Solaris 2.5.1 to Solaris 7

   JASS_SHELLS="
      /usr/bin/sh     /usr/bin/csh     /usr/bin/ksh
      /usr/bin/jsh    /bin/sh          /bin/csh
      /bin/ksh        /bin/jsh         /sbin/sh
      /sbin/jsh"

   # This is to handle special cases by OE.
```

```
   case ${JASS_OS_REVISION} in
      5.8 | 5.9)
         JASS_SHELLS="${JASS_SHELLS}
            /bin/bash       /bin/pfcsh       /bin/pfksh
            /bin/pfsh       /bin/tcsh        /bin/zsh
            /usr/bin/bash   /usr/bin/pfcsh   /usr/bin/pfksh
            /usr/bin/pfsh   /usr/bin/tcsh    /usr/bin/zsh"
      ;;
   esac
fi
export JASS_SHELLS

#
================================================================================
=====
# JASS_SHELL_DISABLE
#
# This variable contains a filename value that specifies the location
of the
# shell to be used to disable accounts.  By default, this variable
contains
# the "/sbin/noshell" program, supplied by the Toolkit.  This variable
is # offered to allow user's to supply their own.  This variable is
used by the
# disable-system-accounts.fin Finish script.  Note that if this file
does # not already exist, the Toolkit's copy_files function will be
called in an
# attempt to install the new shell.
#
================================================================================
=====

if [ -z "${JASS_SHELL_DISABLE}" ]; then
   JASS_SHELL_DISABLE="/sbin/noshell"
fi
export JASS_SHELL_DISABLE

#
================================================================================
=====
# JASS_SUID_FILE
#
# This variable contains a filename value that specifies the location
where
# the print-suid-files.fin Finish script will store its output.  If no
value
# is specified, then the script will print its output to "standard
output".
#
================================================================================
=====

if [ -z "${JASS_SUID_FILE}" ]; then
   JASS_SUID_FILE=""
fi
export JASS_SUID_FILE
```

```
#
================================================================================
=====
# JASS_SUSPEND_PERMS
#
# This variable contains a string value that will be used to define
# what user(s) will be permitted to perform system suspend/resume
functions.
# This default value for this variable is "-" which indicates that only
# the 'root' account will be able to perform these management
functions.
# For more information, refer to the /etc/default/sys-suspend file.
This # variable is used in the set-sys-suspend-restrictions.fin Finish
script.
#
================================================================================
=====

if [ -z "${JASS_SUSPEND_PERMS}" ]; then
   JASS_SUSPEND_PERMS="-"
fi
export JASS_SUSPEND_PERMS

#
================================================================================
=====
# JASS_SVCS_DISABLE
#
# This variable contains a (possibly empty) list of services (from the
# /etc/inet/inetd.conf file) to be disabled as part of the
# update-inetd-conf.fin Finish script.
#
# Baseline from Solaris 2.5.1:
#    100068,  100083,  100221,  100232,  chargen, comsat, daytime,
#    discard, discard, echo,    exec,    finger,  fs,     ftp,
#    login,   name,    netstat, rexd,    rquotad, rstatd, rusersd,
#    shell,   sprayd,  systat,  talk,    telnet,  tftp,   time,
#    ufsd,    uucp,    walld
# Added in Solaris 2.6:
#    100235,  dtspc,   kerbd,   printer, xaudio
# Added in Solaris 7:
#    100234
# Added in Solaris 8:
#    100134, 100146, 100147, 100150, sun-dr
# Added in Solaris 9:
#    100155,  uuidgen
# Added in Solaris 10:
#    kshell, klogin, eklogin
#
#
================================================================================
=====

if [ -z "${JASS_SVCS_DISABLE}" ]; then
   JASS_SVCS_DISABLE="
      ftp     telnet name      talk    uucp    smtp
      tftp    finger systat    netstat rquotad rusersd sprayd walld
```

```
      rexd    shell  login     exec    comsat  time    echo   discard
daytime
      chargen 100087 rwalld    rstatd  100068  100083  100221 fs
ufsd       100232  100235 536870916 kerbd   printer 100234  dtspc
xaudio
      100146  100147 100150    100134  100229  100230  100242
      300326  100232 100068    100083  100221  100235  100155 uuidgen
sun-dr
      kshell  klogin eklogin"
fi
export JASS_SVCS_DISABLE

#
================================================================================
=====
# JASS_SVCS_ENABLE
#
# This variable contains a (possibly empty) list of services (from the
# /etc/inet/inetd.conf file) to be enabled (or remain enabled) as part
# of the update-inetd-conf.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_SVCS_ENABLE}" ]; then
   JASS_SVCS_ENABLE=""
fi
export JASS_SVCS_ENABLE

#
================================================================================
=====
# JASS_TMPFS_SIZE
#
# This variable contains a string value representing the amount of
space
# to be allocated to the /tmp (tmpfs) filesystem.  This value should be
# set to be large enough to handle your current /tmp needs.  This
variable # is used in the set-tmpfs-limit.fin Finish script.
#
================================================================================
=====

if [ -z "${JASS_TMPFS_SIZE}" ]; then
   JASS_TMPFS_SIZE="512m"
fi
export JASS_TMPFS_SIZE

#
================================================================================
=====
# JASS_UMASK
#
# This variable contains a numeric (octal) value to be used for both
the
# system and user default file creation masks.  This variable is used
in
```

```
# the set-system-umask.fin and set-user-umask.fin Finish scripts.
#
================================================================================
=====

if [ -z "${JASS_UMASK}" ]; then
   JASS_UMASK="022"
fi
export JASS_UMASK

#
================================================================================
=====
# JASS_UNOWNED_FILE
#
# This variable contains a filename value that specifies the location
where
# the print-unowned-objects.fin Finish script will store its output.
If no # value is specified, then the script will print its output to
# "standard output".
#
================================================================================
=====

if [ -z "${JASS_UNOWNED_FILE}" ]; then
   JASS_UNOWNED_FILE=""
fi
export JASS_UNOWNED_FILE

#
================================================================================
=====
# JASS_WRITABLE_FILE
#
# This variable contains a filename value that specifies the location
where
# the print-world-writable-objects.fin Finish script will store its
output.
# If no value is specified, then the script will print its output to
# "standard output".
#
================================================================================
=====

if [ -z "${JASS_WRITABLE_FILE}" ]; then
   JASS_WRITABLE_FILE=""
fi
export JASS_WRITABLE_FILE

#-------------------------------------------------------------------------
-------
```

Appendix B: JumpStart Profile and Driver Scripts

jumpstart-new.profile

```
# All rights reserved.
#
#ident   "@(#)end-user.profile     2.5       02/02/21          SMI"
#

# install_type MUST be first
install_type    initial_install

# start with the minimal required number of packages
cluster         SUNWCuser

# To support the Process Accounting
package         SUNWaccr        add
package         SUNWaccu        add

# want to define how the disk is used - not use defaults
#
# General Purpose Server.  Given the size of today's hard
# drives, providing 500 MB for logs, electronic mail,
# and print queues in addition to 1 GB for swap space
# should not be a problem.  The significant amount of swap
# space can also help protect against certain denial of
# service attacks.
#
# partitioning    explicit
# filesys         rootdisk.s3     500     /var
# filesys         rootdisk.s1     1000    swap
# filesys         rootdisk.s0     free    /

partitioning        explicit
filesys             rootdisk.s1         1024     swap
filesys             rootdisk.s3     2048    /var
filesys             rootdisk.s4     4096    /opt
filesys             rootdisk.s0         free       /

filesys             c1t1d0s0          free       /jumpstart

# install system as standalone
system_type     standalone
```

jumpstart-secure-new.driver

```
#!/bin/sh
#
# Copyright (c) 2000-2003 by Sun Microsystems, Inc.
# All rights reserved.
#
```

```
#ident   "@(#)JumpStart-secure.driver    1.4       03/04/21          SMI"
#
# The purpose of this driver is to act as a wrapper calling the
# "configuration" and "hardening" scripts.

DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init

# Ensuring that the following services remain enabled in
# this configuration: tftp

JASS_SVCS_ENABLE="tftp"

. ${DIR}/JumpStart-config-new.driver
. ${DIR}/JumpStart-hardening-new.driver
```

jumpstart-config-new.driver

```
#!/bin/sh
#
# Copyright (c) 2000-2003 by Sun Microsystems, Inc.
# All rights reserved.
#
#ident      "@(#)jumpstart-hardening.driver     1.5   03/05/14    SMI"
#
# The purpose of this driver is to implement the Solaris Operating
Environment
# hardening recommendations that are documented in the Sun BluePrints
Online
# articles (Solaris Operating Environment Security and Solaris OE
Network
# Settings for Security).
#

DIR="`/bin/dirname $0`"
export DIR

. ${DIR}/driver.init

JASS_FILES="
            /etc/resolv.conf
            /etc/nsswitch.conf
            /etc/dt/config/Xaccess
            /etc/dt/config/Xsession.d/0050.warning
            /etc/init.d/inetsvc
            /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
```

```
"

# Note: install-strong-permissions.fin and install-fix-modes.fin are
# generally always the last Finish scripts to run as their effects
# could be undone by Finish scripts that would follow them.  The
# install-fix-modes.fin script is generally run first.  If additional
# tightening is required, install-strong-permissions.fin can be used.
#
# Note: install-security-mode.fin is not included in the script list
# since it requires manual intervention.  For more information, refer
# to the script source code.

JASS_SCRIPTS="
#           Access Control Hardening Finish Scripts

            disable-dtlogin.fin
            disable-remote-root-login.fin
            disable-ssh-root-login.fin
            disable-rhosts.fin
            enable-inetd-syslog.fin
            enable-tcpwrappers.fin
            disable-apache.fin
            install-at-allow.fin
            install-loginlog.fin
            install-sulog.fin
            set-banner-dtlogin.fin
            set-banner-ftpd.fin
            set-banner-sshd.fin
            set-banner-telnetd.fin
            set-login-retries.fin
            set-user-password-reqs.fin
            update-at-deny.fin
            update-cron-allow.fin
            update-cron-deny.fin
#           System Security Hardening Finish Scripts

            disable-autoinst.fin
            disable-automount.fin
            disable-dmi.fin
            disable-dtlogin.fin
            disable-kdc.fin
            disable-lp.fin
            disable-nscd-caching.fin
            disable-preserve.fin
            disable-power-mgmt.fin
            disable-system-accounts.fin
            disable-vold.fin
            enable-coreadm.fin
            enable-process-accounting.fin
            enable-rfc1948.fin
            enable-stack-protection.fin
            install-shells.fin
            remove-unneeded-accounts.fin
            set-power-restrictions.fin
            set-root-group.fin
            set-rmmount-nosuid.fin
            set-system-umask.fin
```

```
            set-user-umask.fin
            update-cron-log-size.fin
            install-md5.fin
            install-fix-modes.fin
#           Network Security Hardening Finish Scripts

            disable-directory.fin
            disable-ipv6.fin
            disable-ldap-client.fin
            disable-mipagent.fin
            disable-nfs-client.fin
            disable-ppp.fin
            disable-samba.fin
            disable-sendmail.fin
            disable-slp.fin
            disable-snmp.fin
            disable-spc.fin
            disable-syslogd-listen.fin
            disable-uucp.fin
            disable-xserver-listen.fin
            disable-wbem.fin
            enable-priv-nfs-ports.fin
            update-inetd-conf.fin
"

. ${DIR}/driver.run
```

Appendix C: SunSSH sshd _config file

```
# Copyright (c) 2001 by Sun Microsystems, Inc.
# All rights reserved.
#
# ident     "@(#)sshd_config  1.3   01/10/08 SMI"
#
# Configuration file for sshd(1m)

# Protocol versions supported
#
# The sshd shipped in this release of Solaris has support for major
versions
# 1 and 2.  It is recommended due to security weaknesses in the v1
protocol
# that sites run only v2 if possible. Support for v1 is provided to
help sites
# with existing ssh v1 clients/servers to transition.
# Support for v1 may not be available in a future release of Solaris.
#
# To enable support for v1 an RSA1 key must be created with ssh-
keygen(1).
# RSA and DSA keys for protocol v2 are created by /etc/init.d/sshd if
they
# do not already exist, RSA1 keys for protocol v1 are not automatically
created.

# Uncomment ONLY ONE of the following Protocol statements.

# Only v2 (recommended)
Protocol 2

# Both v1 and v2 (not recommended)
#Protocol 2,1

# Only v1 (not recommended)
#Protocol 1

# Listen port (the IANA registered port number for ssh is 22)
Port 22

# The default listen address is all interfaces, this may need to be
changed
# if you wish to restrict the interfaces sshd listens on for a multi
homed host.
# Multiple ListenAddress entries are allowed.

# IPv4 only
#ListenAddress 0.0.0.0
# IPv4 & IPv6
ListenAddress ::

# Port forwarding
AllowTcpForwarding no
```

```
# If port forwarding is enabled, specify if the server can bind to
INADDR_ANY.
# This allows the local port forwarding to work when connections are
received
# from any remote host.
GatewayPorts no

# X11 tunneling options
X11Forwarding no
X11DisplayOffset 10

# The maximum number of concurrent unauthenticated connections to sshd.
# start:rate:full see sshd(1) for more information.
# The default is 10 unauthenticated clients.
#MaxStartups 10:30:60

# Banner to be printed before authentication starts.
Banner /etc/issue

# Should sshd print the /etc/motd file and check for mail.
# On Solaris it is assumed that the login shell will do these (eg
/etc/profile).
PrintMotd no
CheckMail no

# KeepAlive specifies whether keep alive messages are sent to the
client.
# See sshd(1) for detailed description of what this means.
# Note that the client may also be sending keep alive messages to the
server.
KeepAlive yes

# Syslog facility and level
SyslogFacility auth
LogLevel info

#
# Authentication configuration
#

# Host private key files
# Must be on a local disk and readable only by the root user (root:sys
600).
HostKey /etc/ssh/ssh_host_rsa_key
HostKey /etc/ssh/ssh_host_dsa_key

# Default Encryption algorithms and Message Authentication codes
Ciphers     aes128-cbc,blowfish-cbc,3des-cbc
MACS  hmac-sha1,hmac-md5

# Length of the server key
# Default 768, Minimum 512
ServerKeyBits 768

# sshd regenerates the key every KeyRegenerationInterval seconds.
# The key is never stored anywhere except the memory of sshd.
```

```
# The default is 1 hour (3600 seconds).
KeyRegenerationInterval 3600

# Ensure secure permissions on users .ssh directory.
StrictModes yes

# Length of time in seconds before a client that hasn't completed
# authentication is disconnected.
# Default is 600 seconds. 0 means no time limit.
LoginGraceTime 600

# Maximum number of retries for authentication
# Default is 6. Default (if unset) for MaxAuthTriesLog is MaxAuthTries
/ 2
MaxAuthTries      6
MaxAuthTriesLog   3

# Are logins to accounts with empty passwords allowed.
# If PermitEmptyPasswords is no, pass PAM_DISALLOW_NULL_AUTHTOK
# to pam_authenticate(3PAM).
PermitEmptyPasswords no

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

# Use PAM via keyboard interactive method for authentication.
# Depending on the setup of pam.conf(4) this may allow tunneled clear
text
# passwords even when PasswordAuthentication is set to no. This is
dependent
# on what the individual modules request and is out of the control of
sshd
# or the protocol.
PAMAuthenticationViaKBDInt yes

# Are root logins permitted using sshd.
# Note that sshd uses pam_authenticate(3PAM) so the root (or any other)
user
# maybe denied access by a PAM module regardless of this setting.
# Valid options are yes, without-password, no.
#PermitRootLogin yes

# sftp subsystem
Subsystem   sftp  /usr/lib/ssh/sftp-server


# SSH protocol v1 specific options
#
# The following options only apply to the v1 protocol and provide
# some form of backwards compatibility with the very weak security
# of /usr/bin/rsh.  Their use is not recommended and the functionality
# will be removed when support for v1 protocol is removed.

# Should sshd use .rhosts and .shosts for password less authentication.
IgnoreRhosts yes
RhostsAuthentication no
```

```
# Rhosts RSA Authentication
# For this to work you will also need host keys in
/etc/ssh/ssh_known_hosts.
# If the user on the client side is not root then this won't work on
# Solaris since /usr/bin/ssh is not installed setuid.
RhostsRSAAuthentication no

# Uncomment if you don't trust ~/.ssh/known_hosts for
RhostsRSAAuthentication.
#IgnoreUserKnownHosts yes

# Is pure RSA authentication allowed.
# Default is yes
RSAAuthentication yes
PermitRootLogin no
```

References

Noordergraaf, Alex and Brunette, Glenn Sun Blueprints: Securing System with the Solaris Security Toolkit. Sun Microsystems Press, A Prentice Hall Title. Saddle River, NJ. June 2003

Noordergraaf, Alex and Watson, Keith. Solaris Operating Environment Security. Sun Microsystems, Inc. Santa Clara, CA January 2003.

Noordergraaf, Alex and Brunette, Glenn. JumpStart Architecture and Security Scripts for Solaris. Sun Microsystems, Inc. Santa Clara, CA September 2000

Sun Microsystems
www.sun.com/blueprints

Solaris Scripts and Tools;
http://www.sun.com/solutions/blueprints/tools/index.html

Solaris 9 /etc/init.d/nddconfig script comments
Sun Microsystems, Inc. Santa Clara, CA

CERT Coordination Center. CERT/CC Advisories
http://www.cert.org/advisories/CA-1994-15.html
http://www.cert.org/advisories/CA-1991-18.html

Internet FAQ Archives
http://www.faqs.org/faqs/
RCF1094, NFS
RCF 1057, Remote Procedure Call (RPC)
RFC1350, TFTP

The Internet Engineering Task force
http://www.ietf.org
RCF1094, NFS
RCF 1057, Remote Procedure Call (RPC)
RFC1350, TFTP

BindView Razor
Razor Tools
http://razor.bindview.com/tools/index.shtml

Center for Internet Security
CIS Benchmark tool for Solaris
http://www.cisecurity.org/bench_solaris.html

Sun Freeware
www.sunfreeware.com

The SANS Institute. Solaris Security Step by Step Version 2.0
Drafted and Edited by Hal Pomeranz, Deer Run Associates, Copyright 2001.

Wietse Z. Venema
TCP Wrappers README file
ftp://ftp.porcupine.org/pub/security/index.html