SUN BLUEPRINTS APPROVED

The Solaris™ Security Toolkit - Internals

Updated for Toolkit version 0.3

By Alex Noordergraaf - Enterprise Engineering and Glenn Brunette - Sun Professional Services

Sun BluePrints™ OnLine - June 2001

Sun microsystems
ENTERPRISE ENGINEERING

http://www.sun.com/blueprints

Sun Microsystems, Inc.
901 San Antonio Road
Palo Alto, CA 94303 USA
650 960-1300 fax 650 969-9131

Part No.: 816-1467-10
Revision 01, 06 / 12 / 01
Edition: June 2001

DOCUMENTATION IS PROVIDED "AS IS" AND ALL EXPRESS OR IMPLIED CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE OR NON-INFRINGEMENT, ARE DISCLAIMED, EXCEPT TO THE EXTENT THAT SUCH DISCLAIMERS ARE HELD TO BE LEGALLY INVALID.

CETTE PUBLICATION EST FOURNIE "EN L'ETAT" ET AUCUNE GARANTIE, EXPRESSE OU IMPLICITE, N'EST ACCORDEE, Y COMPRIS DES GARANTIES CONCERNANT LA VALEUR MARCHANDE, L'APTITUDE DE LA PUBLICATION A REPONDRE A UNE UTILISATION PARTICULIERE, OU LE FAIT QU'ELLE NE SOIT PAS CONTREFAISANTE DE PRODUIT DE TIERS. CE DENI DE GARANTIE NE S'APPLIQUERAIT PAS, DANS LA MESURE OU IL SERAIT TENU JURIDIQUEMENT NUL ET NON AVENU.

The Solaris™ Security Toolkit - Internals

Updated for Toolkit version 0.3

Overview

This is the fourth and final article in a four-part series discussing the Solaris™ Security Toolkit, common referred to by its executable name `jass`, as a mechanism to secure Solaris™ Operating Environment (Solaris OE) systems.

This article provides an in-depth discussion of the directories and scripts used by the Toolkit to harden and minimize Solaris OE systems. Each directory included in the Toolkit's distribution is discussed, as to both the purpose of that directory, and the individual files or scripts.

Recommendations are also made on how the functionality of the Toolkit can be extended.

Update

This Sun BluePrints™ OnLine article has been updated to reflect changes in the newly released version (0.3) of the Solaris Security Toolkit for the Solaris OE. The documentation for this release has been re-written into four parts:

- *Quick Start* focuses on the minimal set of required information to get the Toolkit up and running. Setup and configuration requirements for the Toolkit are quite different, depending on whether it is being run in standalone or JumpStart™ modes, so this article will discuss each method.

- *Release Notes* discusses the changes and enhancements included in the new release.
- *Installation, Configuration, and Usage Guide* focuses on installation, configuration, and usage information not contained in the *Quick Start* guide.
- *Internals* focuses on the actual components of the Toolkit. Each of the internal scripts are individually discussed.

Supported Solaris OE Versions

The current release of the Toolkit works with the Solaris 2.5.1, 2.6, 7, and 8 OE. Scripts which contain OS specific instructions will detect which version of the Solaris OE is being used, and will only run tasks appropriate for that release.

Toolkit Architecture

The Toolkit is made up of a number of directories. Directory structure is based on the recommendations made in the Sun BluePrints OnLine article *Building a JumpStart™ Infrastructure* (April 2001) available at:

```
http://www.sun.com/blueprints/0401/BuildInf.pdf
```

The following directories are in the Toolkit:

- `Documentation`
- `Drivers`
- `Files`
- `Finish`
- `OS`
- `Packages`
- `Patches`
- `Profiles`
- `Sysidcfg`

Each directory is discussed in more detail. Where appropriate, each script, configuration file, or sub-directory is discussed individually. Suggestions are also made on how to modify and add scripts.

Documentation Directory

This directory contains Sun BluePrints Online documentation discussing security recommendations for the Toolkit. These documents may also be accessed at:

```
http://www.sun.com/blueprints/browsesubject.html#security
```

Drivers Directory

The files in the Drivers directory contain configuration information specifying what finish scripts will be executed and what files will be installed as a result of the Toolkit's execution. Finish scripts called by the individual driver files are located in the $JASS_HOME_DIR/Finish directory. Similarly, files installed by the driver files are located under the $JASS_HOME_DIR/Files directory.

Driver Script Creation

All driver scripts have three parts:

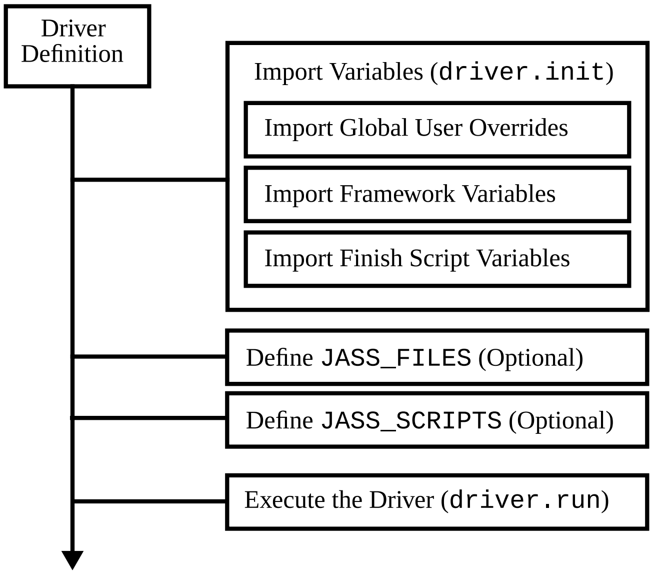
- The first part sets the directory path and calls the driver.init script. The driver.init script calls the finish.init and user.init scripts, which should contain all site-specific configuration information. The driver.init script then sets those environment variables not site-specific and not defined by the finish.init and user.init scripts. All subsequent Toolkit scripts use these environment variables.

Note – The Toolkit will not overwrite site-specific variable assignment.

- The second part defines the JASS_FILES and JASS_SCRIPTS environment variables. The JASS_FILES variable defines those files which will be copied from the Files directory to the client. The JASS_SCRIPTS variable defines what scripts will be executed on the client. Each of the finish scripts available in the Toolkit will be discussed later in this article.
- The final component is the driver.run script. This script processes the contents of the JASS_FILES and JASS_SCRIPTS environment variables. Based on the definition of these variables, the driver.run script copies files to the client and executes the selected Finish scripts.

A flow chart of these three parts looks like the following:

FIGURE 1 illustrates the driver control flow.


Driver Definition
Import Variables (driver.init)
Import Global User Overrides
Import Framework Variables
Import Finish Script Variables
Define JASS_FILES (Optional)
Define JASS_SCRIPTS (Optional)
Execute the Driver (driver.run)

All of the environment variables from the various `.init` files are imported first. Once this is complete, the driver script moves on to part two, which is the definition of `JASS_FILES` and `JASS_SCRIPTS`. The definition of these are optional; either a single environment can be defined, or both, or neither. Part three of the driver script calls `driver.run` to perform the tasks defined by the `JASS_FILE` and `JASS_SCRIPTS` environment variables.

The following code is from an excerpt demonstrating all three driver script parts:

```
DIR="‘/bin/dirname $0‘"

export DIR
. ${DIR}/driver.init

JASS_FILES="
               /etc/cron.d/cron.allow
               /etc/default/ftpd
               /etc/default/telnetd
"

JASS_SCRIPTS="
               install-at-allow.fin
               remove-unneeded-accounts.fin
"
. ${DIR}/driver.run
```

This sample script sets and exports the `$DIR` environment variable so that the scripts will recognize the starting directory. Next, the `$JASS_FILES` environment variable is defined as containing those files which will be copied from the `$JASS_HOME_DIR/Files` directory onto the client. The `$JASS_SCRIPTS` environment variable is then defined with those finish scripts which will be actually run by the Toolkit. Finally, the execution of the Toolkit is started by calling the `driver.run` script. Once called, `driver.run` will copy the files specified by `$JASS_FILES`, and run the scripts specified by `$JASS_SCRIPTS`.

Driver Script Listing

The following files are in the `Drivers` directory:

- `audit.driver`
- `config.driver`
- `driver.funcs`
- `driver.init`
- `driver.run`
- `finish.init`
- `hardening.driver`
- `hardening-jumpstart.driver`
- `install-iPlanetWS.driver`
- `secure.driver`
- `undo.driver`
- `undo.funcs`
- `undo.run`

- `user.init.SAMPLE`
- `user.run.SAMPLE`

The remainder of this section discusses these critical scripts in more detail.

audit.driver

This driver script calls all Toolkit print routines with the exception of the `print-jass-environment.fin` and `print-jumpstart-environment.fin` scripts. The print routines included with the Toolkit can be used to verify certain parts of the systems configuration after a JumpStart installation or standalone Toolkit run. The scripts included are useful when certain types of files, such as set-UID or set-GID binaries, need to be catalogued. Other print scripts included with the Toolkit will list any `rhost` files on the system, files that are not owned by a valid userid on the system, or files which can be written to by any user.

config.driver

This driver script implements a mechanism to separate scripts which perform system configuration tasks from security specific scripts. Because of this separation mechanism, machines with different security requirements can still share the same base Solaris OE configuration driver.

Following is an excerpt from the `config.driver` script included with the Toolkit:

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

. ${DIR}/driver.init

JASS_FILES="
                /.cshrc
"

JASS_SCRIPTS="
                set-root-password.fin
                set-term-type.fin
"

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

This script performs several tasks. First, it calls the driver.init script. Then, it sets both the JASS_FILES and JASS_SCRIPTS environment variables. Once these environment variables are set, the driver.run script is called. The driver.run script completes the installation of the specified files and the execution of all configuration-specific scripts. In the previous example, the .cshrc file contained in $JASS_HOME_DIR/Files directory will be copied to /.cshrc.

driver.funcs

With the release of Toolkit version 0.3, functions common to the driver.run were also needed by the undo.driver file. So as to not duplicate these functions into separate files, a separate driver.funcs file contains function definitions available to other scripts.

driver.init

The first script executed by any driver script must be driver.init. The driver.init script, in combination with the user.init script, sets the environment variables on which the Finish scripts depend. Each of these variables is discussed in the *The Solaris™ Security Toolkit - Installation, Configuration, and Usage Guide: Updated for Toolkit version 0.3* Sun BluePrints OnLine article. Refer to the Bibliography for the URL.

driver.run

This script is the core of the Toolkit. All previously defined environment variables are used by the driver.run script as it:

- verifies the configuration.
- mounts the file systems to the JumpStart client (JumpStart mode only).
- copies the files specified by the JASS_FILES environment variable.
- runs scripts specified by the JASS_SCRIPTS environment variable.
- unmounts the file systems from the JumpStart client (JumpStart mode only).

Each of these functions are described in more detail.

Note – The user.run script can be used to enhance or override functionality defined by the driver.run script.

Verify Configuration

The first task of the `driver.run` script is verification of the Toolkit configuration by checking the following environment variables:

- `JASS_FINISH_DIR`
- `JASS_UNAME`
- `JASS_STANDALONE`
- `JASS_PATCH_MOUNT`

If these variables are not set, the verification process fails and the installation exits.

Mount Filesystems

If the Toolkit is being used in JumpStart mode, the script calls an internal subroutine called `mount_filesystems`. This routine mounts the following directories onto the JumpStart client:

- `JASS_PACKAGE_MOUNT`, which is mounted onto `JASS_PACKAGE_DIR`
- `JASS_PATCH_MOUNT`, which is mounted onto `JASS_PATCH_DIR`

If other file system mount points are required, the `user.run` script can be used to implement them. This is a JumpStart mode specific routine and is not executed during standalone Toolkit runs.

Copy Files

After the mounts have completed successfully, the script copies over all files specified in the `JASS_FILES` environment variable (which can be set in any driver script) to the client. This copy mechanism is useful if many Solaris OE configuration files need to be replaced during a system installation.

Note – The file copy functionality is performed first, so that the files will be available for any finish script use.

Execute Scripts

After the previous scripts have been executed, the finish scripts listed in the `JASS_SCRIPTS` environment variable are executed in sequence. The output of these finish scripts are processed in one or more of the following ways:

a. Logged to the file specified by the `jass-execute -o` option. If a file is not specified, the output will be directed to standard output. This option is only available in standalone mode.

b. Logged into the /var/sadm/system/logs/finish.log file on the JumpStart client during JumpStart installations. The /var/sadm/system/logs/finish is the standard log file used by any JumpStart command run on the client.This option is only available in JumpStart mode.

c. Logged to the file /var/opt/SUNWjass/run/<timestamp>/jass-install.log. The timestamp is a fully qualified time parameter of the form YYYYMMDDHHMMSS. This value is constant for each execution of the Toolkit and represents the time at which the run was started. For example, a run started at 1:30 p.m. on April 1, 2001 would be represented by the value 20010401133000. These Toolkit log files are generated during every Toolkit run.

Unmount Filesystems

After all Finish scripts for the particular driver have been run, the driver.run script unmounts all filesystems mounted during the *Mount Filesystems* process (described in a previous section), then exits gracefully. At this point the JumpStart client reboots.

This is a JumpStart mode specific routine and is not executed during standalone Toolkit runs.

finish.init

This script provides a central location for the definition of finish script environment variables. Most finish scripts have the option to use either a hard-coded value, or an environment variable defined in either finish.init or user.init. Site-specific modifications should be made in user.init to simplify migration to new Toolkit releases. For a detailed description of all the environment variables in this file, refer to *The Solaris™ Security Toolkit - Installation, Configuration, and Usage Guide* Sun BluePrints OnLine article. Refer to in the Bibliography for URL.

hardening.driver

Most of the security specific scripts included in the Toolkit are listed in the hardening.driver script. This script, similar to the config.driver script, defines both files and scripts to be run by the driver.run script. Some scripts, which implement functionality not commonly required, are not included in this driver.

These Toolkit scripts implement all the recommendations made in the Sun BluePrints OnLine article *Solaris™ Operating Environment Security - Updated for Solaris 8 Operating Environment* (April 2001) available from:

`http://www.sun.com/blueprints/0401/ossec-updt1.pdf`

hardening-jumpstart.driver

This driver provides a set of scripts which can be used to harden a JumpStart server. This driver is not referenced by any other driver in the Toolkit. Its only purpose is to provide a listing of what finish scripts can be executed and still have a functioning JumpStart server.

install-iPlanetWS.driver

This driver calls the `minimize-iPlanetWS.fin` script first presented in the Sun BluePrints OnLine article *Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible and Secure Application Installation Methodology - Updated for Solaris 8 Operating Environment* (November 2000). The script removes all Solaris OE packages not required to successfully install and run the iPlanet™ Web Server software. The script has been updated to include support for the Solaris 8 OE. The following are the contents of the driver script:

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

. ${DIR}/driver.init

. ${DIR}/config.driver

JASS_SCRIPTS="
minimize-iPlanetWS.fin
install-iPlanetWS.fin
"
. ${DIR}/driver.run

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

If a JumpStart client is built using this driver script, it must be listed in the `rules` file. This script performs all the actions specified by the `config.driver` and `hardening.driver scripts`, in addition to the minimization functionality in the `minimize-iPlanetWS.fin` and `install-iPlanetWS.fin` scripts.

secure.driver

The following are the contents of the `secure.driver` script included with the Toolkit:

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

. ${DIR}/driver.init

. ${DIR}/config.driver

. ${DIR}/hardening.driver

# This is a sample driver to contain
# code for checking the status of
# various system attributes.
#
# . ${DIR}/audit.driver
```

This script is provided as a ready-to-use mechanism implementing all the hardening functionality in the Toolkit. The script performs the initialization tasks required, then calls the `config.driver` and `hardening.driver` scripts. This configures the system and performs all the hardening tasks specified in the `hardening.driver` script. In addition, the `audit.driver` script is listed, but commented out. If the additional functionality of that script is desired, it should be uncommented. The `secure.driver` script should be the default script used in the `rules` file for client installation.

undo.driver

This driver implements the undo feature. This driver is quite straightforward and only contains the following:

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

. ${DIR}/driver.init

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

When called by `./jass-execute -u`, this driver initializes itself much the same way as any other driver—by calling `driver.init`, and then passing control to a different driver—`undo.driver`, in this case.

undo.funcs

As with all the other files in the `Drivers` directory ending with `funcs`, this script contains functions associated with the undo Toolkit option, but which can be used by other drivers.

undo.run

This script is the core of the Toolkits undo functionality. It performs the following tasks:

- imports needed functions from `driver.funcs` and `undo.funcs`.
- verifies that all of the initialization scripts have been run.
- reads any user-defined functions from `user.run`.
- prints identifying information about the undo run to the log file and console.
- executes the `undo_ops` function to perform the undo.

This script is called by `jass-execute` when the `-u` option is specified.

user.init.SAMPLE

This sample script provides a mechanism to specify Toolkit user functions. This script should be used to override any default environment variables and addition of site-specific or organization-specific Toolkit information, thereby minimizing future Toolkit migration issues.

This script provides default values for the `PACKAGE_MOUNT` and `PATCH_MOUNT` environment variables. These variables must be modified for the specific JumpStart server and directory paths required.

For details on each of the environment variables specified in this script, refer to *The Solaris™ Security Toolkit - Installation, Configuration, and Usage Guide* Sun BluePrints OnLine article of this series.

Note – This script is distributed as a `.SAMPLE` file so that it will not overwrite any user-defined variables when upgrading to a newer release of the Toolkit.

user.run.SAMPLE

As with user.init, this script should be used to add any site-specific or organization-specific information into the Toolkit to avoid migration issues. The user.run script should contain all site-specific and organization-specific overrides for the driver.run script.

Note – This script is distributed as a .SAMPLE file so that it will not overwrite any user-defined scripts when upgrading to a newer release of the Toolkit.

Files Directory

The Files directory is used in conjunction with the JASS_FILES environment variable and the driver.run script. This directory stores files that will be copied to the JumpStart client.

The JASS_FILES Environment Variable and Files Directory Setup

The JASS_FILES environment variable is used to specify the complete Solaris OE path of files stored in the $JASS_HOME_DIR/Files directory. This environment variable can be used in the following ways:

1. The first option is to specify the file which will be copied from the Toolkit to the client. For example, the following is defined in the hardening.driver script:

```
JASS_FILES=”
        /etc/motd
”
```

   By defining the JASS_FILES environment variable to include this file, the /etc/motd file on the client will be replaced by the $JASS_HOME_DIR/Files/etc/motd file from the Toolkit distribution. Any file can be copied in this manner by simply including it in the Files directory, and adding it to the JASS_FILES definition in the appropriate driver script.

2. The second option is to specify host-specific files. This is done by creating files in the `Files` directory of the following form:

```
/etc/syslog.conf.$HOSTNAME
```

   In this scenario, the `$JASS_HOME_DIR/Files/etc/syslog.conf` file will only be copied to a system with a hostname that matches `$HOSTNAME`. When there is both an `syslog.conf` and `syslog.conf.$HOSTNAME,` the host-specific file will have precedence.

3. The third option is the specify OS release-specific files. This feature can be used by creating files in the `Files` directory with the following form:

```
/etc/syslog.conf+$OS
```

   The `$OS` variable should mirror the output produced by the `uname -r` command. If OS version 5.8 was being secured, then a file with the name of `$JASS_HOME_DIR/Files/etc/syslog.conf+5.8` would be copied. This file would not be copied to any other OS release. OS specific files have precedence over generic files, but not over host-specific files.

4. The final option is to have the `JASS_FILES` variable specify a directory. When used, the entire directory contents are copied to the JumpStart client. If the `JASS_FILES` variable contains the following line:

```
/etc/rc2.d
```

   then the entire contents of the `$JASS_HOME_DIR/Files/etc/rc2.d` directory on the JumpStart server will be copied to the JumpStart client.

Files Directory Listing

The following are in the Files directory:

- `/etc/issue`
- `/etc/motd`
- `/etc/notrouter`
- `/etc/nsswitch.conf`
- `/etc/syslog.conf`
- `/etc/default/ftpd`
- `/etc/default/sendmail`
- `/etc/default/telnetd`
- `/etc/dt/config/Xaccess`

- /etc/init.d/inetsvc
- /etc/init.d/nddconfig
- /etc/init.d/set-tmp-permissions
- /etc/rc2.d/S00set-tmp-permissions
- /etc/rc2.d/S07set-tmp-permissions
- /etc/rc2.d/S70nddconfig
- /etc/security/audit_class
- /etc/security/audit_control
- /etc/security/audit_event
- /sbin/noshell

The remainder of this section discusses these files.

/etc/issue and /etc/motd

These files are based on U.S. government recommendations. They provide users legal notice that their activities may be monitored. If an organization has specific legal banners, they can be installed into these files.

/etc/notrouter

This file disables IP forwarding between interfaces on the system by creating an /etc/notrouter file. Once the JumpStart client is rebooted, the client will no longer function as a router, regardless of the number of network interfaces.

/etc/nsswitch.conf

This is an nsswitch.conf file configured so that a system will use files for name resolution. It is a copy of the /etc/nsswitch.files shipped with Solaris 8 OE.

/etc/syslog.conf

This modified /etc/syslog.conf file is installed to perform additional logging. It serves as a placeholder for organizations to add in their own centralized log server (or servers) so that proactive log analysis can be done.

/etc/default/ftpd

This file enables the feature available in Solaris OE versions 7 and 8 to change the default FTP banner. The banner is changed by adding a BANNER entry to the /etc/default/ftpd file. The /etc/default/ftpd file included in the Toolkit creates a generic *Authorized Access Only* entry, which denies FTP version information to potential attackers.

/etc/default/sendmail

This sendmail configuration file was released with the Sun BluePrints OnLine article titled *Solaris™ Operating Environment Security - Updated for Solaris 8 Operating Environment* (April 2001). This article is available in the Documentation directory of the Toolkit or refer to the Bibliography. With the release of Solaris 8 OE, a sendmail configuration file can be used to run sendmail in queue mode, instead of running it through cron (as was previously the case). This script is copied onto the system being hardened by the disable-sendmail.fin script only when on a Solaris 8 OE system.

/etc/default/telnetd

This file enables the feature available in Solaris OE versions 7 and 8 to change the default TELNET banner. The banner is changed by adding the BANNER entry to the /etc/default/telnetd file. The /etc/default/telnetd file included in the Toolkit creates a generic *Authorized Access Only* entry, which denies TELNET version information to potential attackers.

/etc/dt/config/Xaccess

This file disables all remote access, whether directed or broadcast, to any X server running on this system. Depending on the environment the Toolkit will be used in and the X support requirements, this file may not be appropriate.

/etc/init.d/nddconfig and /etc/rc2.d/S70nddconfig

These files copy over the nddconfig and S70nddconfig startup scripts required to implement the settings described in the Sun BluePrints OnLine article *Solaris™ Operating Environment Network Settings for Security: Updated for Solaris 8 Operating Environment* (December 2000) available at:

```
http://www.sun.com/blueprints/1200/network-updt1.pdf
```

/etc/init.d/set-tmp-permissions, /etc/rc2.d/S00set-tmp-permissions and /etc/rc2.d/S07set-tmp-permissions

The purpose of these scripts is to set the correct permissions on the /tmp and /var/tmp directories when the system is rebooted. If an inconsistency is found, it will be displayed to standard output and logged via SYSLOG. This script is installed into /etc/rc2.d twice to permit this check to be performed both before and after the mountall command is run from S01MOUNTFSYS. This helps ensure that both the mount point and the mounted filesystem have the correct permissions and ownership.

/etc/init.d/inetsvc

This file replaces the default /etc/init.d/inetsvc with a minimized version containing only those commands required for the configuration of the network interfaces. The minimized script has only four lines, as compared to the 256 lines of the Solaris 8 OE version. The minimized inetsvc script is as follows:

```
#!/bin/sh

/usr/sbin/ifconfig -au netmask + broadcast +
/usr/sbin/inetd -s -t &
```

Although this script has been used successfully by a variety of Sun customers, it has no support for the DHCP or BIND servers. Therefore, this file should only be used in environments that use static IP assignment.

/etc/security/audit_class, /etc/security/audit_control and /etc/security/audit_event

These three configuration files for the Solaris OE Auditing subsystem, also referred to as the Basic Security Module, were released with the Sun BluePrints OnLine article title *Auditing in the Solaris™ 8 Operating Environment* (February 2001). Refer to the Bibliography for the URL. This article is also in the Documentation directory of the Toolkit.

/sbin/noshell

This script is leveraged from the Titan security toolkit and is used to track access attempts to any accounts which have been locked using this script. These log messages are of the format:

```
Attempted access by ${USER} on host ${HOSTNAME}
```

This script is used by the `disable-system-accounts.fin` script.

`Finish` Directory

The `Finish` directory contains the scripts which perform system modifications and updates during installation.

Finish Script Creation

When installing with a JumpStart server, the finish scripts run from a memory-resident mini-root running on the JumpStart client. The mini-root contains almost all of the Solaris OE functions. When creating finish scripts, it is sometimes necessary to execute commands using the `chroot` command.

Many of these limitations are not present during a standalone Toolkit installation.

To simplify portability and configuration issues, the environment variables defined in the various `.init` scripts are used throughout the Toolkit. If additional variables are required, they should be added as environment variables to the `user.init` and `user.run` scripts.

Note – The default environment variables values used by finish scripts are defined in the `finish.init` script.

Finish Script Listing

Each of the scripts in the `Finish` directory is briefly discussed in this section. The scripts fall into the following categories:

- `Disable`

- `Enable`
- `Install`
- `Minimize`
- `Print`
- `Remove`
- `Set`
- `Update`

Individual scripts in each category are discussed. For additional background or justifications of the scripts, see the previously published Sun BluePrints OnLine security articles referenced in the Bibliography.

Disable Finish Scripts

The following `disable finish` scripts are discussed in this section:

- `disable-apache.fin`
- `disable-asppp.fin`
- `disable-autoinst.fin`
- `disable-automount.fin`
- `disable-core-generation.fin`
- `disable-dhcp.fin`
- `disable-dmi.fin`
- `disable-dtlogin.fin`
- `disable-ipv6.fin`
- `disable-keyserv-uid-nobody.fin`
- `disable-ldap-client.fin`
- `disable-lp.fin`
- `disable-mipagent.fin`
- `disable-nfs-client.fin`
- `disable-nfs-server.fin`
- `disable-nscd-caching.fin`
- `disable-power-mgmt.fin`
- `disable-preserve.fin`
- `disable-remote-root-login.fin`
- `disable-rhosts.fin`
- `disable-rpc.fin`
- `disable-sendmail.fin`
- `disable-slp.fin`
- `disable-snmp.fin`
- `disable-spc.fin`
- `disable-syslogd-listen.fin`
- `disable-system-accounts.fin`
- `disable-uucp.fin`
- `disable-vold.fin`
- `disable-wbem.fin`

disable-apache.fin

This script prevents the Apache web server shipped with Solaris OE 8 from starting. The one startup and four kill scripts are all disabled.

disable-asppp.fin

This script disables all the Asynchronous PPP (asppp) startup and shutdown scripts (three kill scripts and one startup script) in the /etc/rc directories.

disable-autoinst.fin

This script disables the startup scripts used to re-initialize or re-install the system, including S30sysid.net, S71sysid.sys, and S72autoinstall. These startup scripts will never be used in a JumpStart environment and should be disabled to help prevent an intruder from reconfiguring the system.

disable-automount.fin

This script disables all the automounter startup and shutdown scripts. Five shutdown scripts and one startup script are disabled.

disable-core-generation.fin

This script disables the creation of core files by adding the appropriate command to the /etc/system file.

disable-dhcp.fin

This script disables the DHCP server included in Solaris OE version 8.

disable-dmi.fin

This script disables the DMI startup and shutdown scripts. Four shutdown scripts and one startup script are disabled.

disable-dtlogin.fin

This script disables all the CDE startup and shutdown scripts. One startup script and three shutdown scripts are disabled.

disable-ipv6.fin

This script disables the IPv6 network interfaces created by default on Solaris 8 OE by removing the associated hostname files in /etc.

disable-keyserv-uid-nobody.fin

This script disables secure RPC access to user nobody by adding the -d option to the keyservd daemon startup command in the /etc/init.d/rpc file.

disable-ldap-client.fin

This script disables the LDAP client daemons included with Solaris OE version 8. One startup and three kill scripts are disabled.

disable-lp.fin

This script disables all lp startup and shutdown scripts. There are a total of six scripts for the subsystems. Additionally, all lp access to the cron subsystem is removed by adding lp to the /etc/cron.d/cron.deny file, and removing all lp commands in the /var/spool/cron/crontabs directory. This functionality is distinct from the update-cron-deny.fin script, because the lp packages may or may not be installed on a system. In addition, the lp subsystem may be necessary, while the functions removed by the cron-deny-update.fin script are not.

disable-mipagent.fin

This script disables the Mobile IP (MIP) agents included in Solaris OE version 8. One startup and four scripts are disabled.

disable-nfs-client.fin

This script disables the NFS client startup scripts. Three kill scripts and one startup script are disabled.

disable-nfs-server.fin

This script disables the NFS server startup scripts. Seven kill scripts and one startup script are disabled.

disable-nscd-caching.fin

This script modifies the nscd.conf file to disable caching for passwd, group, and hosts by changing the value of the enable_cache option to no in the /etc/nscd-caching.conf file.

Note – Care should be taken when using the disable-nscd-caching.fin script in NIS and NIS+ environments, as nscd may be required.

---

disable-power-mgmt.fin

This script disables the auto power shutdown option on Sun SPARC™ hardware platforms by creating a /noautoshutdown file. This script also disables the four scripts used for startup and shutdown of the powerd daemon.

disable-preserve.fin

This script disables the /etc/init.d/PRESERVE startup script.

disable-remote-root-login.fin

This script disallows direct root logins. Even though this has been the default for the Solaris OE since the final update of 2.5.1, it should still be verified to ensure correct configuration.

disable-rhosts.fin

This script disables rhosts authentication for rlogin and rsh by modifying the Pluggable Authentication Module (PAM) configuration in /etc/pam.conf.

disable-rpc.fin

This script disables the three kill and one startup scripts for Remote Procedure Calls (RPC).

disable-sendmail.fin

This script disables the sendmail daemon startup and shutdown scripts, and adds an entry to the cron subsystem which executes sendmail once an hour for Solaris OE versions 2.5.1, 2.6, and 7. For Solaris 8 OE, the /etc/default/sendmail file is installed, which implements similar functionality. This method of purging outgoing mail is more secure than having the daemon running continually.

disable-slp.fin

This script disables all Service Location Protocol (SLP) startup and shutdown scripts. There are four scripts for the subsystem.

disable-snmp.fin

This script disables the startup and shutdown scripts for the default Solaris OE SNMP daemons.

disable-spc.fin

This script disables all SunSoft™ Print Client startup and shutdown scripts. There are six scripts for the subsystem.

disable-syslogd-listen.fin

This script prevents the syslogd daemon from accepting SYSLOG messages from other systems on the network. This option has been added to Solaris OE version 8, and is enabled by adding the -t option to the syslogd startup script. Even after using this option, processes on the local system can still use SYSLOG.

disable-system-accounts.fin

This script disables system accounts and enables logging of access attempts. Disabled accounts are those with a UID of less then 100 or greater then 60,000, with the exception of root and sys. Access attempt logging is implemented by creating an /sbin/noshell script, which denies access to the disabled account and logs the attempt (via SYSLOG) as an authentication error. Within the minimized Solaris OE, the logged accounts include daemon, bin, adm, lp, uucp, nobody, and noaccess.

disable-uucp.fin

This script disables the UUCP startup script. In addition, the nuucp system account and all uucp crontab entries are removed.

disable-vold.fin

This script prevents the volume management service from starting by disabling the run-control startup and kill scripts.

disable-wbem.fin

This script disables the Web Based Enterprise Management (WBEM) daemons from starting on Solaris OE version 8. One startup and three kill scripts are disabled.

Enable Finish Scripts

The following enable finish scripts are discussed in this section:

- enable-32bit-kernel.fin
- enable-bsm.fin
- enable-ftp-syslog.fin
- enable-inetd-syslog.fin
- enable-priv-nfs-ports.fin
- enable-process-accounting.fin
- enable-rfc1948.fin
- enable-stack-protection.fin

enable-32bit-kernel.fin

This script sets the boot-file variable in the EEPROM of Sun SPARC systems to the value of /kernel/unix. This forces the system to boot using a 32-bit kernel. It is useful for products that can run on the Solaris 7 OE or later, but must run in 32-bit only mode, such as Checkpoint's Firewall-1. This script is intended for sun4u systems.

enable-bsm.fin

This script performs all the necessary tasks involved in enabling the Basic Security Module (BSM) on a Solaris OE system in a lights-out data center environment. This includes:

- Running bsmconv script

- Removing the L1A (STOP-A) disable option, which the bsmconv script added to /etc/system
- Editing the /etc/security/audit_control file created by bsmconv; and
- Adding the audit_warn alias to the sendmail aliases file (if not there already)

After the system is rebooted, the BSM subsystem is enabled and logging begins.

enable-ftp-syslog.fin

This script forces the in.ftpd daemon to log all FTP access attempts through the SYSLOG subsystem. This option is enabled by adding the -l option to the in.ftpd command in the /etc/inetd.conf file.

enable-inetd-syslog.fin

This script enables logs of all incoming connection requests for service by the inetd daemon. When logging is enabled, inetd logs the source IP address, source TCP address, and service name through SYSLOG. Logging is enabled by adding the -t option to the inetd startup script in /etc/init.d/inetsvc.

enable-priv-nfs-ports.fin

This script sets the kernel variable nfssrv:nfs_portmon to 1, which restricts NFS requests to privileged ports only. After setting the variable in the /etc/system file, only NFS requests from ports less than 1024 are accepted.

enable-process-accounting.fin

This script will enable Solaris OE process accounting if the required Solaris OE packages are installed on the system.

enable-rfc1948.fin

This script enables RFC 1948 unique-per-connection ID sequence number generation by setting the variable TCP_STRONG_ISS to 2 in the /etc/default/inetinit file.

enable-stack-protection.fin

This script enables the stack protection and logging included in all Solaris OE releases since version 2.6. These options are enabled by adding the following two commands to the /etc/system file:

- set noexec_user_stack = 1
- set noexec_user_stack_log = 1

After the two variables are set, the system denies attempts to execute the stack directly, and logs any stack execution attempt through SYSLOG. This facility is enabled to protect the system from common buffer overflow attacks.

Install Finish Scripts

The following install finish scripts are discussed in this section:

- install-at-allow.fin
- install-fix-modes.fin
- install-ftpusers.fin
- install-iPlanetWS.fin
- install-jass.fin
- install-loginlog.fin
- install-newaliases.fin
- install-openssh.fin
- install-recommended-patches.fin
- install-sadmind-options.fin
- install-security-mode.fin
- install-shells.fin
- install-strong-permissions.fin
- install-sulog.fin

install-at-allow.fin

This script restricts the at command execution by creating an empty at.allow file in /etc/cron.d. An empty at.allow file forces the system to check the at.deny file for unauthorized at users. All users who require at access must now be added to the at.allow file. This script should be used in conjunction with the update-at-deny.fin script.

install-fix-modes.fin

This script both copies the fix-modes package (created by Casper Dik, see reference in the Bibliography) from the Toolkit to the client, and executes the script. The fix-modes package must first be acquired from: either

```
http://www.sun.com/blueprints/tools
```

or

```
ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz
```

Once downloaded, it must be compiled and installed on the JumpStart server in:

```
$JASS_HOME_DIR/Packages/FixModes.tar.Z
```

install-ftpusers.fin

Solaris OE versions prior to Solaris 8 OE do not create an ftpusers file by default. The file included in the Toolkit contains entries for default system accounts including root, daemon, sys, bin, adm, lp, smtp, uucp, nuucp, listen, nobody, noaccess, and nobody4.

install-iPlanetWS.fin

This script performs basic installation tasks for the iPlanet web server, and was first presented in the Sun BluePrints Online article *Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible and Secure Application Installation Methodology - Updated for Solaris 8 Operating Environment* (November 2000) available at:

```
http://www.sun.com/blueprints/1100/minimization-updt1.pdf
```

install-jass.fin

The purpose of this script is to automate the installation of the Toolkit software onto a system where the Toolkit is being run. This is recommended so that the Toolkit will be available to be re-run after patch installations on the client. The installation is performed by installing the Toolkit package distribution with the Solaris OE command pkgadd. The Toolkit package installs, by default, in /opt/SUNWjass.

install-loginlog.fin

This script creates the /var/adm/loginlog file which is used by the system to log unsuccessful login attempts. The failed logins are logged after the number of failed logins has been exceeded. The number of failed logins permitted is specified in the RETRIES variable set in the /etc/default/login configuration file. See also the set-login-retries.fin finish script.

install-newaliases.fin

This script checks to see if the /usr/bin/newaliases file is present. If not, and /usr/lib/sendmail is present, it links /usr/bin/newaliases to /usr/lib/sendmail. This file is part of the SUNWnisu package and is sometimes not installed on minimal builds.

install-openssh.fin

This script installs the OpenBSD version of OpenSSH into /opt/OBSDssh. The installation is based on having a Solaris OE package stream formatted package called OBSDssh.pkg in the $JASS_PACKAGE_DIR directory. An upcoming Sun BluePrints OnLine article, scheduled to be published in July 2001, will describe how to create this package.

install-recommended-patches.fin

This script installs applicable patches from the $JASS_HOME_DIR/Patches directory on the JumpStart server. The appropriate *Recommended and Security Patch Clusters* must be downloaded and extracted to the $JASS_HOME_DIR/Patches directory for the script to execute properly.

install-sadmind-options.fin

This script adds the options specified in the $JASS_SADMIND_OPTIONS Toolkit environment variable to the sadmind daemon entry in /etc/inet/inetd.conf.

install-security-mode.fin

This script displays the current status of the Open Boot PROM security mode. This script does not set the EEPROM password directly, as it is not possible to script the setting of the EEPROM password during a JumpStart installation. The output of the script provides instructions on how to set the EEPROM password.

`install-shells.fin`

This script creates the `/etc/shells` file that is used to restrict access to the system. The Solaris OE function *getusershell(3C)* is the primary user the `/etc/shells` file to determine valid shells on the system.

Note – This script will only add the shell to the file if the shell exists on the system and does not already exist in the file.

`install-strong-permissions.fin`

This script changes a variety of permissions to restrict group and user access on the system. In addition, it sets the permissions on the `/etc/security` directory to 0750 from the default value of 0755. By denying access to users not in the `sys` group, users have less access to information on the `BSM` subsystem.

`install-sulog.fin`

This script creates the `/var/adm/sulog` file, which enables logging of all `su` attempts.

Minimize Finish Script

The following minimize finish script is discussed in this section:

```
minimize-iPlanetWS.fin
```

`minimize-iPlanetWS.fin`

This script implements the Solaris OE minimization procedure as described in the updated Sun BluePrints OnLine article *Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible and Secure Application Installation Methodology - Updated for Solaris 8 Operating Environment* (November 2000) available at:

```
http://www.sun.com/blueprints/1100/minimization-updt1.pdf
```

Print Finish Scripts

The following print finish scripts are discussed in this section:

- print-jass-environment.fin
- print-jumpstart-environment.fin
- print-rhosts.fin
- print-sgid-files.fin
- print-suid-files.fin
- print-unowned-objects.fin
- print-world-writeable-objets.fin

print-jass-environment.fin

This script prints out all the environment variables used in the Toolkit. It is included for diagnostic purposes.

print-jumpstart-environment.fin

This script prints out all the environment variables used by the JumpStart server during a system installation. It is included for diagnostic purposes.

print-rhosts.fin

This script will list all the .rhosts and hosts.equiv files contained in any directory under the JASS_ROOT_DIR directory. The results will be displayed on standard output unless the JASS_RHOSTS_FILE variable is defined. If this variable is defined, then all of the results will be written to that file.

print-sgid-files.fin

This script will print all files in any directory under the JASS_ROOT_DIR directory with set group ID permissions. The results will be displayed on standard output unless the JASS_SGID_FILE variable is defined. If this variable is defined, all of the results will be written to that file.

print-suid-files.fin

This script will print all files in any directory under the JASS_ROOT_DIR directory with set user ID permissions. The results will be displayed on standard output unless the JASS_SUID_FILE variable is defined. If this variable is defined, all of the results will be written to that file.

print-unowned-objects.fin

This script will list all objects on a system, starting from JASS_ROOT_DIR, which do not have correct ownerships. This includes files, directories, etc. that do not have a valid user or group assigned to them. The results will be displayed on standard output unless the JASS_UNOWNED_FILE variable is defined. If this variable is defined, then all of the results will be written to that file.

print-world-writeable-objets.fin

This script will list all world writeable objects on a system, starting from JASS_ROOT_DIR. The results will be displayed on standard output unless the JASS_WRITEABLE_FILE variable is defined. If this variable is defined, then all of the results will be written to that file.

Remove Finish Script

The following remove finish script is discussed in this section:

- remove-unneeded-accounts.fin

remove-unneeded-accounts.fin

This script removes unused Solaris OE accounts from the /etc/passwd and /etc/shadow files using the passmgmt command. This script removes the smtp, nuucp, listen, and nobody4 accounts, based on the JASS_ACCT_REMOVE variable.

Set Finish Scripts

The following set finish scripts are discussed in this section:

- set-ftpd-umask.fin
- set-login-retries.fin
- set-power-restrictions.fin
- set-rmmount-nosuid.fin
- set-root-password.fin
- set-sys-suspend-restrictions.fin
- set-system-umask.fin
- set-term-type.fin
- set-tmpfs-limit.fin
- set-user-password-reqs.fin
- set-user-umask.fin

set-ftpd-umask.fin

This script adds an umask value, defined within the Toolkit as $JASS_FTPD_UMASK, to the /etc/default/ftpd file to be used by the in.ftpd(1M) daemon.

set-login-retries.fin

This script modifies the RETRIES variable in the /etc/default/login file to three, from the default value of five, based on the JASS_LOGIN_RETRIES variable. By reducing the logging threshold, additional information may be gained. The previously discussed install-loginlog.fin script enables the logging of failed login attempts.

set-power-restrictions.fin

This script alters the configuration of /etc/default/power to restrict user access to power management functions using the JASS_POWER_MGT_USER and JASS_CPR_MGT_USER variables.

set-rmmount-nosuid.fin

This script modifies the /etc/rmmount.conf file, so that setuid executables on removable media will no longer execute with setuid privileges.

set-root-password.fin

This script automates setting the root password by setting the password to an initial value as defined by JASS_ROOT_PASSWORD. The password used in this script should only be used during the installation and must be changed immediately after the JumpStart installation process has successfully completed. This script sets the root password to be ‘t00lk1t’.

Note – This script will only execute during a JumpStart software installation. It will not execute when the Toolkit is invoked from the command line.

set-sys-suspend-restrictions.fin

This script alters the configuration of /etc/default/sys-suspend to restrict user access to suspend and resume functionality based on the JASS_SUSPEND_PERMS variable.

set-system-umask.fin

This script creates startup scripts for each run level, which in turn, set the system UMASK properly to 022 for Solaris OE versions prior to 8. For Solaris OE version 8, the CMASK variable in /etc/default/init is verified to have a value of 022.

set-term-type.fin

This script sets a default terminal type of vt100 to avoid issues with systems not recognizing dtterm. This script is intended mainly for use on systems that do not have graphical consoles and are generally accessed over a terminal console or other serial link.

set-tmpfs-limit.fin

This script installs a limit on the disk space that can be used as part of a tmpfs filesystem. This limit can help prevent memory exhaustion. The usable space is limited by default in this script to 512 megabytes.

set-user-password-reqs.fin

This script enables more strict password requirements by enabling:

- Password aging
- Minimum intervals between password changes
- Increasing the password minimum length

This script is recommended for systems with non-privileged user access.

Note – Take care to ensure the root account is not inadvertently locked when running this script on restricted access servers.

set-user-umask.fin

This script adds an updated UMASK value of 022 in the /etc, /etc/skel, and /etc/default/login files, and to the startup files for all default shells.

Note – A more restrictive UMASK of 077 may be more appropriate for highly sensitive systems.

Update Finish Scripts

The following update finish scripts are discussed in this section:

- update-at-deny.fin
- update-cron-allow.fin
- update-cron-deny.fin
- update-cron-log-size.fin
- update-inetd-conf.fin

update-at-deny.fin

This script adds system accounts in /etc/passwd to the /etc/cron.d/at.deny file. All accounts in /etc/passwd are added to this file. When used in conjunction with the install-at-allow.fin file, no access will be permitted to the at subsystem.

update-cron-allow.fin

This script updates the /etc/cron.d/cron.allow file to restrict access to the cron subsystem. Only one account, root, is included in the new cron.allow file. No other system accounts are added by default. The root account will be the only account able to utilize the cron functionality. To add additional accounts, use the JASS_CRON_ALLOW variable.

update-cron-deny.fin

This script updates the /etc/cron.d/cron.deny file by adding every user with a UID less than 100 or greater than 60000 (except the root and sys accounts) to it. In addition, the crontab entries for uucp and adm are removed from the system crontab.

Depending on the packages installed, some modifications to this finish script may be required, because it has been written to run against minimized systems. This minimized system is described in the Sun BluePrints OnLine article *Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible and Secure Application Installation Methodology - Updated for Solaris 8 Operating Environment* (November 2000) available at:

```
http://www.sun.com/blueprints/1100/minimization-updt1.pdf
```

In a minimized Solaris OE installation, only the uucp and adm crontab entries need to be removed.

update-cron-log-size.fin

This purpose of this script is to adjust the LIMIT parameter in the /etc/cron.d/logchecker script. By default, that script will rotate the CRON log file, /var/cron/log, after it exceeds a size of 0.5 MBytes. This script now sets the LIMIT parameter to the value specified by the $JASS_CRON_LOG_SIZE environment variable. By default, this variable is set to 20480 or 10 MBytes.

update-inetd-conf.fin

This script disables all default Solaris OE entries in the /etc/inetd.conf file. The services are disabled after the script inserts a "#" at the start of each line. All services included in the base OS are disabled in Solaris OE versions 2.5.1 forward. Additional services installed by unbundled or third party software are not disabled.

OS Directory

This directory contains only Solaris OE images. These will be used by the JumpStart software installation process as the source of the client installation, and to provide the add_install_client and rm_install_client scripts, which add new clients to the JumpStart environment. The installation naming convention recommended is Solaris_*os version_4 digit year_2 digit month of CD release*. For example, the Solaris 8 Operating Environment CD, dated April 2001, would have a directory name of Solaris_8_2001-04. By separating updates and releases of the Solaris OE, very fine control can be maintained for testing and deployment purposes.

Release 0.3 of the Toolkit has been updated to support both Trusted Solaris™ Software and Solaris OE (Intel Platform Edition) software versions in this directory. The Trusted Solaris directory name should be in the following format *Trusted*_Solaris_*os version_4 digit year_2 digit month of CD release*. For the Trusted Solaris software release dated December of 2000, the directory name would be: Trusted_Solaris_8_2000-12. Solaris OE (Intel Platform Edition) should use the following format: *Solaris_os version_4 digit year_2 digit month of CD release_ia.* For the Solaris OE (Intel Platform Edition) release dated April, 2001 the directory name would be: Solaris_8_2001-04_ia.

The add_client script has been updated to parse these additional directory names.

Packages Directory

This directory contains software packages which can be installed with a finish script. For example, the iPlanet Web Server software package could be stored in the Packages directory so the appropriate finish script can install the software as required.

Several finish scripts included in the Toolkit perform software installation and basic configuration functions. Some of these functions were described in the preceding finish script section. The Toolkits scripts which will install software from the Packages directory include:

- install-fix-modes.fin
- install-iPlanetWS.fin
- install-jass.fin
- install-openssh.fin

Patches Directory

This directory should contain *Recommended and Security Patch Clusters for Solaris;* these required clusters must be downloaded and extracted into this directory from http://sunsolve.sun.com. A directory should be created for each of the Solaris OE versions being used. There may be several directories, including 2.5.1_Recommended and 2.6_Recommended within the Patches directory. These patch clusters are extracted in the Patches directory, which allows the patch installation script to run without extracting the patch clusters for each system installation.

Version 0.3 of the Toolkit has been updated to support Solaris OE (Intel Platform Edition) patch clusters. The supported naming convention for these patch clusters is the same as made available through SunSolve OnLine^{SM} service. This format is *Solaris release_x86_Recommended*. The Solaris OE (Intel Platform Edition) patch cluster for Solaris 8 OE would be in a directory named: 8_x86_Recommended.

`Profiles` Directory

This directory contains all of the profiles. Profiles are files that contain configuration information used by JumpStart software to determine Solaris OE clusters for installation (for example, `Core`, `End User`, `Developer`, or `Entire Distribution`), disk layout, and installation type to perform (e.g. standalone). These files are listed in the `rules` file to define how specific systems or groups of systems are built.

Profile Creation

Profiles are only used during JumpStart mode executions. The required and optional contents of profiles are discussed in the Sun BluePrints OnLine article *Building a JumpStart™ Infrastructure* (April 2001). For additional information on profiles, refer to the *Creating the JumpStart* `profile` *file* section of that article, listed in the Bibliography.

Profile Configuration Files

A variety of standard JumpStart profiles have been included with the Toolkit:

- `32-bit-minimal.profile`
- `end-user.profile`
- `entire-distribution.profile`
- `minimal-iPlanetWS-Solaris26.profile`
- `minimal-iPlanetWS-Solaris7-32bit.profile`
- `minimal-iPlanetWS-Solaris7-64bit.profile`
- `minimal-iPlanetWS-Solaris8-32bit.profile`
- `minimal-iPlanetWS-Solaris8-64bit.profile`

Most of the profiles supplied with the Toolkit have been customized for the laboratory environment. Therefore, these profiles should be viewed as samples requiring individual site modifications. These files should not be modified, as updates to the Toolkit may included updated versions. When making changes, create copies of these sample files specific to the local environment. This will simplify the migration to new Toolkit releases.

Sysidcfg Directory

Similar to the previous Profiles Directory section, sysidcfg files are only used during JumpStart mode installations to automate Solaris OE installations, by providing the required installation information. A separate directory tree stores OE-specific information.

Each Solaris OE has a separate directory and uses a naming scheme similar to that used by the OS directory. For each release there is a directory named: Solaris_*OE Version*. The Toolkit includes sample sysidcfg files for Solaris 2.5.1 through 8 OE, which are in the following directories:

- Solaris_2.5.1
- Solaris_2.6
- Solaris_7
- Solaris_8

For additional information on sysidcfg files, refer to the Sun BluePrint OnLine article *Building a JumpStart™ Infrastructure* (April 2001), listed in the Bibliography and also available in the Documentation directory of the Toolkit.

Version Control

Maintaining version control for all files and scripts in the Toolkit environment is critical for two reasons. First, one of the goals of this environment is to be able to re-create a system installation. This will be impossible without having a snapshot of all file versions used during the installation. Secondly, because these scripts are performing security functions—which is a critical process for many organizations—extreme caution should be exercised to ensure only appropriate and tested changes are implemented.

One Source Code Control System (SCCS) version control package is contained in the Solaris OE SUNWsprot package. Other version control software available from freeware and commercial vendors can also manage version information. Whichever version control product is used, it is important that a process *be in place* to manage updates and capture version information for future system re-creation.

Conclusion

This article reflects the changes made to Toolkit version 0.3. Focusing on Toolkit internals, this article discusses the directory structure, files, and scripts. Guidelines are provided for adding new scripts. Recommendations on Toolkit changes required when moving to different environments are also discussed. Additional information on the Toolkit and JumpStart installations are referenced in the Bibliography.

Bibliography

Solaris Advanced Installation Guide, Sun Microsystems,
`http://docs.sun.com`

Dik, Casper, *fix-modes tool*,
`ftp://ftp.wins.uva.nl/pub/solaris/fix-modes.tar.gz`

Noordergraaf, Alex, *Building a JumpStart™ Infrastructure*, Sun BluePrints OnLine, April 2001,
`http://sun.com/blueprints/0401/BuildInf.pdf`

Noordergraaf, Alex, *Solaris™ Operating Environment Minimization for Security: A Simple, Reproducible and Secure Application Installation Methodology - Updated for the Solaris 8 Operating Environment*, Sun BluePrints OnLine, November 2000,
`http://sun.com/blueprints/1100/minimization-updt1.pdf`

Noordergraaf, Alex and Brunette, Glenn, *The Solaris™ Security Toolkit - Installation, Configuration, and Usage Guide: Updated for Toolkit version 0.3*, Sun BluePrints OnLine, June 2001,
`http://sun.com/blueprints/0601/jass_conf_install-v03.pdf`

Noordergraaf, Alex and Brunette, Glenn, *The Solaris™ Security Toolkit - Quick Start: Updated for Toolkit version 0.3*, Sun BluePrints OnLine, June 2001,
`http://sun.com/blueprints/0601/jass_quick_start-v03.pdf`

Noordergraaf, Alex and Brunette, Glenn, *The Solaris™ Security Toolkit - Release Notes: Updated for Toolkit version 0.3*, Sun BluePrints OnLine, June 2001,
`http://sun.com/blueprints/0601/jass_release_notes-v03.pdf`

Noordergraaf, Alex and Watson, Keith, *Solaris™ Operating Environment Security: Updated for the Solaris 8 Operating Environment*, Sun BluePrints OnLine, April 2001,
`http://sun.com/blueprints/0401/security-updt1.pdf`

Osser, William and Noordergraaf, Alex, *Auditing in the Solaris™ 8 Operating Environment*, Sun BluePrints OnLine, February 2001,
`http://sun.com/blueprints/0201/audit_config.pdf`

Powell, Brad, et. al., *Titan Toolkit*,
`http://www.fish.com/titan`

Watson, Keith and Noordergraaf, Alex, *Solaris™ Operating Environment Network Settings for Security: Updated for the Solaris 8 Operating Environment*, Sun BluePrints OnLine, December 2000,
`http://sun.com/blueprints/1200/network-updt1.pdf`

Author's Bio: Alex Noordergraaf

Alex Noordergraaf has more than nine years experience in the area of Computer and Network Security. As a Senior Security Architect in the Enterprise Engineering group of Sun Microsystems, he is developing, documenting, and publishing security best practices through the Sun BluePrints OnLine program. Articles completed include recommendations on Solaris OE Security settings, Solaris OE Minimization, and Solaris OE Network settings.

Prior to his role in Enterprise Engineering, he was a Senior Security Architect with Sun Professional Services, where he worked with many Fortune 500 companies on projects that included Security Assessments, Architecture Development, Architectural Reviews, and Policy/Procedure review and development. In addition to providing billable services to customers, he developed and delivered an Enterprise Security Assessment methodology and training curriculum to be used worldwide by the Sun Professional Services™ organization. *His customers have included major telecommunication firms, financial institutions, ISPs, and ASPs.*

Author's Bio: Glenn Brunette

Glenn Brunette has more than eight years experience in the areas of computer and network security. Glenn currently works in the Sun Professional Services organization, where he is the Lead Security Architect for the Northeastern USA region. In this role, he works with many Fortune 500 companies to deliver tailored security solutions such as assessments, architecture design and implementation, as well as policy and procedure review and development. His customers have included major financial institutions, ISPs, New Media, and government organizations.

In addition to billable services, Glenn works with the Sun Professional Services Global Security Practice and Enterprise Engineering group on the development and review of new security methodologies, best practices, and tools.