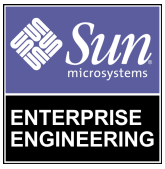
JumpStart™ Architecture and Security Scripts for the Solaris™ Operating Environment - Part 2

By Alex Noordergraaf - Enterprise Engineering

Sun BluePrints™ OnLine - August 2000



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.: 806-6475-10
Revision 02, August 2000

Please
Recycle

JumpStart™ Architecture and Security Scripts for the Solaris™ Operating Environment - Part 2

Overview

This is the second article in a three part series discussing the JumpStart Architecture and Security Scripts tool (Toolkit) version 0.1 as a method of securing systems using the Solaris Operating Environment. The first article is available from:

```
http://www.sun.com/blueprints/0700/jssec.pdf
```

The first article presented a detailed overview of the JumpStart product, and provided step-by-step instructions for installing and configuring a JumpStart client and server.

This article continues with an overview of the configuration files, directories, and scripts used by the Toolkit to harden Solaris systems.

Philosophy

The goal of the Toolkit is to build a secured Solaris system. This is accomplished by executing scripts during the JumpStart installation process. After the installation is completed the system will be patched, configured, and secured.

Because of the modular and flexible architecture of directories and scripts, a system administrator is able to define and implement Solaris Operating Environment security modifications down to a granular level during system installation.

The Toolkit focuses on Solaris Operating Environment security modifications to harden and minimize a system. Hardening is the modification of Solaris configurations to improve the security of the system. Minimization is the removal of unnecessary Solaris packages from the system which will reduce the number of components that have to be patched and made secure—reducing the number of components has the potential to reduce entry points to an intruder.

Note – Configuration modifications for performance enhancements and software configuration are not addressed by the Toolkit.

The Toolkit was designed to harden systems during installation—this is achieved by using the JumpStart technology as a mechanism for running the Toolkit scripts. As discussed in the first JumpStart Architecture and Security Scripts tool article, JumpStart technology provides a method of installing the Solaris Operating Environment over a network, and also has the ability to run scripts on the JumpStart client during installation. This function is used to run the scripts in the Toolkit during the installation process automatically.

Additionally, the Toolkit has been designed to run outside the JumpStart framework—this allows the Toolkit to be used on systems that require security modifications and/or updates but cannot be taken out of service to re-install the OS from scratch.

The Toolkit was built with a modular framework. Customers with existing JumpStart installations will benefit from the Toolkit's ability to integrate into their existing JumpStart architecture. For customers who do not currently use the JumpStart product, the flexibility of the Toolkit's framework will enhance their ability to start using the JumpStart product efficiently.

Supported Versions

The current release of the Toolkit works with Solaris Operating Environment versions 2.5.1, 2.6, 7, and 8. The Toolkit scripts will automatically detect which version of the Solaris software is installed, and will only run tasks specific to that version.

Toolkit Architecture

The main components of the architecture comprises eight directories:

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

Drivers Directory

Contains all driver scripts—`Driver` scripts are the scripts listed in the `rules` files that call all other scripts during security modifications. The `Driver` scripts determine which security modifications will be made to each system by calling the appropriate `Finish` scripts. The `Finish` scripts perform the actual modifications to the Solaris Operating Environment on the JumpStart clients.

Files Directory

Stores files to be copied to the JumpStart client—the Files directory is used in conjunction with an environment variable and driver scripts to select and copy files to the JumpStart client.

Finish Directory

Contains the `Finish` scripts that perform system modifications and updates during installation. `Finish` scripts have been written to perform various tasks such as patch and software installation. These scripts will be discussed further in Part 3 of this series.

OS Directory

This directory must contain only Solaris Operating Environment files. These files will be used by the JumpStart server (over the network) to build the client. Different Solaris Operating Environment releases should be stored in this directory in separate sub directories. The sub directories should use the naming convention recommended in Part 1 to enable fine grained control for testing and deployment purposes.

Packages Directory

Contains software packages that will be installed by the `Finish` scripts. For example, the OpenSSH software could be stored in the `Packages` directory so the appropriate `Finish` script can install and configure the software as required.

Patches Directory

Contains the Recommended and Security Patch Clusters (in addition to individual patches). Create sub directories within the `Patches` directory for each of the Solaris versions being used. The patch clusters should be extracted into the individual sub directories—this will allow the patch installation script to run without having to first extract the patch cluster for each system installation.

Profiles Directory

Contains all profiles—a profile is a file that contains configuration information used by the JumpStart software to determine which Solaris cluster to install (`Core`, `End User`, `Developer`, or `Entire Distribution`), the disk layout to use, and the type of installation to perform (e.g., standalone). These configuration files are used to define how specific systems, or groups of systems are built.

Sysidcfg Directory

Contains directories with OS and host specific `sysidcfg` files. Due to the OS specific nature of the sysidcfg file, a generic version can no longer be used for all Solaris Operating Environment releases. The sub directories should use a naming convention similar to that recommend for the OS directory in Part 1. The installation convention used is `Solaris_x.x<version #>`. The `sysidcfg` files for the Solaris Operating Environment version 2.6 should be stored in a sub-directory named `Solaris_2.6`.

Script Development Framework

The JumpStart software determines which Solaris cluster type to install, specifies disk partitioning, and calls all scripts that are to be executed based on the information specified in the `rules` file. Additionally it provides a robust framework for developing scripts to configure Solaris systems.

The Toolkit architecture includes additional configuration information that enables scripts to be used in different environments. All variables used in the scripts are maintained in a configuration file—this configuration file is imported by a driver script which will then make the variables available to all subsequent scripts.

Toolkit Configuration

To simplify the migration of the JumpStart environment between sites, specific configuration information is kept in a configuration file. This file is stored in the `Drivers` directory and contains the following variables:

- FILES_DIR
- FINISH_DIR
- JASS_SUFFIX
- PACKAGE_DIR
- PACKAGE_MOUNT
- PATCH_DIR
- PATCH_MOUNT
- ROOT_DIR
- SI_CONFIG_DIR
- STANDALONE
- UNAME
- USER_DIR

Only these twelve variables need to be verified when moving the JumpStart environment from one site to another. The function of each variable is as follows:

FILES_DIR

The FILES_DIR variable points to the location of the Files directory on the JumpStart Server. This directory contains files which can be copied to the JumpStart client.

Any files to be copied are specified in the FILES_LIST variable—these will be copied to the client during installation. The FILES_LIST variable is set by individual drivers and not in the configuration file. There are several methods available for copying files using this variable which will be covered in Part 3 of this series.

FINISH_DIR

The convention used by the Toolkit is to store all Finish scripts in the directory named Finish. However, for flexibility, the FINISH_DIR environment variable has been included for those organizations who require Finish scripts to be stored in different locations. This environment variable is defined as FINISH_DIR and should not normally require modification.

JASS_SUFFIX

The JASS_SUFFIX is used by the Toolkit to determine which suffixes must be appended onto backup copies of files. By default this is set to JASS.

PACKAGE_DIR and PACKAGE_MOUNT

The PACKAGE_DIR and PACKAGE_MOUNT environment variables specify where software packages are stored.

The PACKAGE_DIR variable specifies where to NFS mount the PACKAGE_MOUNT directory. Normally, the PACKAGE_DIR variable will not require modification because this is a transient mount-point used only during the JumpStart installation

The PACKAGE_MOUNT variable identifies the location on the JumpStart server by hostname and directory path. The hostname and complete path are required because this directory will be NFS mounted to the JumpStart client during installation. Because a hostname or IP address is specified in the value of the environment variable, it will *always* require modification.

PATCH_DIR and PATCH_MOUNT

The PATCH_DIR and PATCH_MOUNT variables specify the location of the Patches directory on the JumpStart server.

The PATCH_DIR variable specifies the directory where the Patch directory will be mounted during a JumpStart installation and does not usually require modification.

The PATCH_MOUNT variable specifies the JumpStart server hostname and complete path of the Patch directory, therefore, the PATCH_MOUNT variable will require modification for each site.

ROOT_DIR

This variable defines the root directory of the file system. For JumpStart installations this will always be /a. However, when using the Toolkit directly and not through JumpStart, this variable must be changed to /.

SI_CONFIG_DIR

This variable defines the directory used on the JumpStart server that holds all other required directories and files. This directory should be a disk partition, and is usually named /jumpstart. The partition must be large enough to hold all JumpStart information. Although the scripts themselves are relatively small, the OS, Patches, and Packages directories can be quite large.

STANDALONE

This variable informs the Toolkit that it is being run directly and not through the JumpStart software. Because of this, the mount points will be used directly and not mounted through NFS.

UNAME

This variable is used as a global environment variable specifying the OS of the JumpStart client being built. This variable is set by the driver.init script through the use of the uname -r command and exported so all other scripts can access it.

USER_DIR

This variable specifies a user-defined override file. This file can be used by an administrator to modify the default Toolkit settings without first having to modify the core Toolkit scripts.

Limitations

JumpStart is an extremely powerful tool, however, it does have limitations and restrictions. For instance, while booting, a JumpStart client will load a Solaris Operating Environment mini-root and run all subsequent commands from this memory based operating system. The operating system being installed is mounted on the mini-root through the mountpoint `/a`. However, many of the required commands can only be run on the disk-based OS and not from the memory resident mini-root. These commands and scripts must be called through the `chroot` command. By using the `chroot` command, the commands and scripts can be run on the newly installed OS image of the client system.

Version Control

Maintaining version control for all files and scripts in the JumpStart 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.

The Source Code Control System (SCCS) used for version control is contained in the Solaris `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.

Toolkit (Part 3)

The following article will present detailed information on the Toolkit installation and configuration. Additionally, the scripts contained in the Toolkit will be individually listed and discussed. Recommendations on which changes may be necessary to port the Toolkit to another environment will also be made.

Conclusion

This article provided an overview of the Jumpstart Architecture and Security Scripts toolkit. As part of the overview, the design philosophy of the Toolkit was also reviewed. Additionally, the architecture and framework of the Toolkit was discussed. As part of the architecture discussion, the directory structures and their functions were described.

Bibliography

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

Noordergraaf, Alex, *Jumpstart Architecture and Security Scripts Toolkit - Part 1*, Sun BluePrints OnLine, July 2000,
`http://www.sun.com/blueprints/0700/jssec.pdf`

Noordergraaf, Alex and Watson, Keith, *Solaris Operating Environment Minimization for Security*, Sun BluePrints OnLine, December 1999,
`http://www.sun.com/blueprints/1299/minimization.pdf`

Noordergraaf, Alex and Watson, Keith, *Solaris Operating Environment Security*, Sun BluePrints OnLine, January 2000,
`http://www.sun.com/blueprints/0100/security.pdf`

Acknowledgements

I would like to thank Glenn Brunette and Keith Watson for their input and assistance in the development and testing, and for their many hours spent reviewing and editing.

Glenn Brunette was of tremendous help in implementing and testing configuration changes.

Keith Watson reviewed the end result of the Toolkit and made recommendations on needed changes, and also developed some elegant Finish scripts.

Author's Bio: Alex Noordergraaf

Alex Noordergraaf has over 9 years experience in the area of Computer and Network Security. As a Senior Staff Engineer 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 Security settings, Solaris Minimization, and Solaris 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 SunPS. His customers have included major telecommunication firms, financial institutions, ISPs, and ASPs.