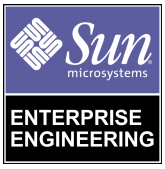
JumpStart™ Architecture and Security Scripts for the Solaris™ Operating Environment - Part 1

Updated for Toolkit version 0.2

By Alex Noordergraaf - Enterprise Engineer and Glenn Brunette - SunPS

Sun BluePrints™ OnLine - November 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.: 806-6370-10
Revision 02, November 2000

Please
Recycle

JumpStart™ Architecture and Security Scripts for the Solaris™ Operating Environment - Part 1

Updated for Toolkit version 0.2

Update

This Sun BluePrints™ OnLine article has been updated to reflect changes in the newly released version (0.2) of the JumpStart™ Architecture and Security Scripts (“JASS“ Toolkit) for the Solaris™ Operating Environment.

Parts 1, 2, and 3 of the *JumpStart™ Architecture and Security Scripts for the Solaris™ Operating Environment - Updated for version 0.2* articles are available at:

- *Part 1* – `http://www.sun.com/blueprints/1100/jssec-updt1.pdf`
- *Part 2* – `http://www.sun.com/blueprints/1100/jssec2-updt1.pdf`
- *Part 3* – `http://www.sun.com/blueprints/1100/jssec3-updt1.pdf`

Introduction

This is the first article of a three part series that introduces the JumpStart Architecture and Security Scripts (“JASS“ Toolkit) for the Solaris Operating Environment (Solaris OE). The Toolkit focuses on Solaris OE 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 OE 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.

Information contained in this article is based on recommendations made in the following published Sun BluePrints™ OnLine articles:

- *Solaris Operating Environment Minimization for Security* published December 1999
- *Solaris Operating Environment Network Settings for Security* published December 1999
- *Solaris Operating Environment Network Settings for Security - Updated for Solaris 8* published November 2000
- *Solaris Operating Environment Security* published January 2000

This article provides a step by step analysis of the installation and configuration of a JumpStart server to familiarize readers with JumpStart—which is the primary installation mechanism of the Toolkit.

Problem

The time-to-market time frame for many businesses is being eroded at breakneck speed. This is especially true in today's Internet driven economy—consequently, there is less time to perform all tasks critical for the security of the infrastructure.

Manually dealing with security issues for each server on an individual basis is extremely time consuming. Tools have been developed to address these issues in both the freeware and commercial arenas; however, many of the tools can only be used at the individual server level, and generally have to be run manually following the installation and configuration of a server.

A process is needed that will automatically install the operating system and configure all necessary security functions. The technology required to automatically install the operating environment has been available for the Solaris product since version 2.1—this is called JumpStart technology and is currently used by many organizations to automate operating environment installation and configuration. However, not all organizations are using the JumpStart framework to optimize the security features of their installations. This Toolkit has been developed to assist organizations that currently use the JumpStart product to enhance their installations, and to assist organizations not using the JumpStart product to start using it.

An important justification for this framework is to improve server baseline security. By having the process and technology available, it will be possible to ensure that every server has the necessary modifications performed on them.

An automated and non-interactive installation process has additional important benefits. By using the Toolkit, a process can be developed that captures and communicates knowledge. This process is critical when training new staff as well as for capturing updates and publicizing information to other staff members. The JumpStart environment can be used to help implement updates to the environment—either by re-building the entire system from scratch with new updates, or by installing the new software directly onto the system. Other benefits include the simplification of system reconstruction due to major hardware failures and replacements.

JumpStart

This section provides a high-level overview of the JumpStart software—and details instructions for configuring a JumpStart server and client—including applicable configuration files for each step. Additional information on configuring the JumpStart software can be found in the Bibliography.

The JumpStart software provides a means of installing groups of identical systems automatically. A JumpStart system installation is begun by booting the JumpStart client via a network. The JumpStart client will broadcast a request over the network asking for configuration information—the local JumpStart server replies to these requests after verifying it has been instructed to boot, configure, and install the Solaris OE on the system. However, before this is possible a JumpStart server must be installed and configured.

The remainder of this section provides step by step instructions on how to install and configure a JumpStart server and client running the Solaris 8 OE. Each step in configuring the server and client shows the commands and associated output. Explanations of the JumpStart configuration files and options are also provided. However, this section only discusses the JumpStart options used by the Toolkit—for a complete listing of JumpStart options and commands refer to the Bibliography for other JumpStart documentation.

JumpStart Server Installation

The scenario being discussed in this article consists of two systems. One is the JumpStart server, and the other is the client. The server is named *server01* and the client is named *client01*.

The *Solaris Advanced Installation* guide (`http://docs.sun.com`) recommends creating a separate directory or partition for the JumpStart directory. The directory is named `/jumpstart` in the Toolkit. Within this directory (or partition) all other directories required by the Toolkit should be created.

Any required Solaris image(s) should be copied into the `/jumpstart/OS` directory. The naming convention recommended for OS specific sub directories is:

```
Solaris_x.x<version #>_<2 digit>month-year
```

The installation process used in this article is based on a the Solaris 8 Operating Environment CD dated March 2000; therefore, the directory should be named `Solaris_8.0_03-00`. By creating different directories to store separate updates and releases of the Solaris OE, fine grained control can be maintained for testing and deployment purposes.

The installation is begun by running the `setup_install_server` command from the Solaris CD. The following procedure uses the Solaris 8 OE.

Note – The JumpStart server installation process for earlier versions of the Solaris OE requires only one CD.

To create a Solaris 8 JumpStart server—insert the first Solaris 8 Software CD (labeled 1 of 2) into the CD ROM drive and enter the following commands:

```
# pwd
/cdrom/sol_8_sparc/s0/Solaris_8/Tools
# ./setup_install_server /jumpstart/OS/Solaris_8.0_03-00
```

The above command produces the following output:

```
Verifying target directory...
Calculating the required disk space for the Solaris_8 product
Copying the CD image to disk...
Install Server setup complete
```

The first CD of the Solaris 8 OE is now installed. Insert the second CD into the CD ROM drive and enter the following commands:

```
# pwd
/cdrom/sol_8_sparc_2/Solaris_8/Tools
# ./add_to_install_server /jumpstart/OS/Solaris_8.0_03-00
```

The previous command produces the following output:

```
The following Products will be copied to /jumpstart/OS/
Solairs_8.0_03-00/Solaris_8/Product:

Solaris_2_of_2

If only a subset of products is needed enter Control-C
and invoke ./add_to_install_server with the -s option.

Checking required disk space...

Copying the Early Access products...
41990 blocks

Processing completed successfully.
```

After the Solaris 8 OE software is installed on the JumpStart server, the `/jumpstart` directory must be made available to the JumpStart clients through NFS. Therefore, the following line should be added to the `/etc/dfs/dfstab` file:

```
share -F nfs -o ro,anon=0 -d “Jumpstart Directory“ /jumpstart
```

Enter the following command to execute the share command listed above:

```
# shareall
```

JumpStart Client Configuration

For a JumpStart installation to be performed successfully, the JumpStart server must know the Ethernet hardware address (MAC) and IP addresses of the JumpStart client(s). This information is provided to the JumpStart server through a naming service such as NIS+ or NIS—or through the use of the `/etc/hosts` and `/etc/ethers` files. This information will be used by the `add_install_client` JumpStart script to create an entry in the `/etc/bootparams` file for that client. To simplify this example, the `/etc/ethers` and `/etc/hosts` files will be used for this procedure.

Create an `/etc/ethers` file, and add a line similar to the following:

```
8:0:20:82:d8:8f client01
```

Add the following line (for the JumpStart client) to the`/etc/hosts` file:

```
10.0.0.30 client01
```

Note! - The JumpStart server, *server01*, uses an IP address of 10.0.0.20.

Finally, the JumpStart client *client01* is added with following command:

```
# pwd
/jumpstart/OS/Solaris_8.0_03-00/Solaris/Tools
# ./add_install_client -c server01:/jumpstart \
-p server01:/jumpstart client01 sun4m
```

The above command produces the following output:

```
making /tftpboot
enabling tftp in /etc/inetd.conf
starting rarpd
starting bootparamd
starting nfsd's
starting nfs mount
updating /etc/bootparams
copying inetboot to /tftpboot
```

Note how the `add_install_client` command will start any services required by the JumpStart server to function correctly (which were not running when `add_install_client` was run). For example, if the NFS server on the JumpStart server has not been started, it will be started by the `add_install_client` command.

The JumpStart server is now configured to supply a client with an IP address and the Solaris OE. However, until a profile configuration file and `rules` file are created, the JumpStart server does not know what components of the Solaris OE to offer the client; therefore, an automated JumpStart installation will not be possible. Although an automated installation is not possible, an interactive Solaris installation may be performed.

Rules Definition

The JumpStart software uses rules to determine how a JumpStart client will be built. The `rules` file is a text based configuration file that contains a rule for each group of systems (or single system), and contains information on configuring and installing the Solaris OE. Each rule defines a system based on its attributes—rules must be located in the JumpStart directory on the JumpStart server.

The `rules` file is created by a system administrator and should contain the rules for all different types of systems to be installed in the environment. The following is a sample rule in a `rules` file:

```
hostname www - Profiles/inet.profile -
```

A rules file entry has five fields. The syntax of the rules files must follow this convention:

```
rule_keyword rule_value begin profile finish
```

A `rule` file entry must contain *at least* a rule_keyword, a rule_value, and a profile. In addition, Begin and Finish scripts can be included—which will be executed by the JumpStart server before (or after) the Solaris OE is installed.

The Toolkit sample rules file, `rules.SAMPLE`, only uses four of the five available fields. The four fields are:

- **rule_keyword** – This field is used to define system attributes used in the rule_value, to match a system with a corresponding value. The sample `rules.SAMPLE` file provided in the Toolkit uses the keyword `any` to match all client systems.
- **rule_value** – The value of this field is the corresponding value of the rule_keyword. The `rules.SAMPLE` file uses a hyphen in this field which is required when using the rule_keyword `any`.
- **profile** – This field points to a separate file that contains specific Solaris OE configuration information for a client. This configuration information may include disk layouts, Solaris OE cluster specifics—whether the JumpStart will be an initial installation or upgrade—and other relevant information. The `rules.SAMPLE` file uses a hyphen in this field so that the JumpStart server performs a default configuration.
- **finish** – The value contained in this field is an executable Bourne shell script which will be run after the Solaris OE installation is completed. In the Toolkit, this script is normally a `Driver` script which calls other scripts in the Toolkit. The `rules.SAMPLE` file calls the `nomatch.beg` script to display information about the JumpStart client.

There are additional options available in the rules file than those described above. For additional information, refer to the Bibliography for other JumpStart reference material.

A basic `rules` entry will be used for the simple JumpStart environment described in this article. The `any` argument in the `rules` file will be used by a JumpStart client not matching another rule previously listed in the `rules` file. If we added just this entry to the `rules` file, all JumpStart clients defined on the server could be installed using this entry. To implement the `any` argument, a `rules` file should be created in the `/jumpstart` directory by including only the following entry:

```
any - - Profiles/basic.profile -
```

The above entry was used in the rules file for the examples described throughout this article.

Profile Definition

A `rules` file must specify a `profile`—this defines how a Solaris OE system is to be installed and configured. The `profile` will contain profile keywords and the corresponding value for each keyword. Each `profile` keyword is used to define a specific component of the Solaris OE installation and configuration process.

The following is a sample profile named `basic.profile`:

```
# install_type MUST be listed first
install_type    initial_install

# start with the minimal required number of packages
cluster         SUNWCreq

# define how the disk is laid out
partitioning    default
```

The example above is a minimal profile. All profiles must contain *at least* the `install_type` keyword as indicated above. The other keywords listed are not required because they have default values that will be used if no explicit definition is made. However, as this profile is part of a security Toolkit, it is strongly recommended that the values are specified. Several sample profiles are included in the Toolkit for reference purposes.

The `rule` file being used for the JumpStart environment described in this article uses the sample profile above (`Profiles/basic.profile`) to define which components will be installed on the JumpStart client. Based on this profile, the following actions will be performed:

1) `install_type initial_install`: A new Solaris OE will be installed (as opposed to an upgrade).

2) `cluster SUNWCreq`: The Solaris OE cluster `SUNWCreq` will be installed (which only includes the minimal number of packages required by the Solaris product). If this variable is not specified, the `SUNWCuser` cluster or End User cluster will be installed.

3) `partitioning default`: By specifying `default`, the system will configure the hard drive using the Solaris OE requirements. If the `partitioning` keyword is not specified in the profile, the drive will be partitioned *as if* the `partitioning default` was specified.

By convention, all `Profiles` are stored in the `/jumpstart/Profiles` directory of the Toolkit. These files are grouped by system function—for example, all web servers will use the same profile. The goal is to have an identical physical configuration, disk layout, and OS installation. This will simplify the hardening process and streamline administration and management.

The elements used in the `basic.profile` are the most common—for additional information refer to the Bibliography.

Finish Script Definition

The final field used in the Toolkit `rules` file is the Finish script. The script listed in this field will be called by the JumpStart software after the OS installation has been completed. To enable the execution of multiple scripts, a driver script is used—the driver script does not perform any tasks other than calling additional scripts to run on the system. An example of a rules files entry using a Finish script named `Drivers/bp-iplanet.driver` would look as follows:

```
any - - Profiles/basic.profile Drivers/bp-iplanetes.driver
```

By convention, the driver scripts are kept in the `Drivers` directory. Additional information on driver scripts will be included in Parts 2 and 3 of this series.

Check

The `rules` file, profile configuration files, and scripts require validation after creation or modification—they are validated by running the `check` script which creates a `rules.ok` file (if no errors are detected). The `rules.ok` file is used by the JumpStart server to install the Solaris OE. The `check` script is located on the JumpStart server in the directory:

```
/jumpstart/OS/Solaris_8.0_03-00/Solaris_8/Misc/jumpstart_sample
```

This script should be copied to the base JumpStart directory of the Toolkit, `/jumpstart`, and then executed as follows:

```
# pwd
/jumpstart
# ./check
```

This command will generate the following output:

```
Validating rules...
Validating profile Profiles/basic.profile...
The custom JumpStart configuration is ok.
```

At this point the JumpStart client, *client01*, is ready to be JumpStarted as a JumpStart client. This is accomplished by booting the system from the `ok` prompt by entering the following command:

```
ok> boot net - install
```

Configuring the `sysidcfg` file

To fully automate an installation, all required information (i.e. netmask, locale, timeserver, etc.) must be available to the installation process. This information is provided through the `sysidcfg` configuration file, or a naming service such as NIS+. Additional information on how to implement these options is available in the JumpStart articles referenced in the Bibliography.

The `add_install_client` command used previously, includes the `-p` option. This option will direct the JumpStart client to use the `sysidcfg` file from the `/jumpstart` directory on the JumpStart server.

Note – The `-p` option is not available in Solaris OE version 2.5.1

We used the following `sysidcfg` file to fully automate the installation:

```
system_locale=en_US
timezone=US/Eastern
network_interface=le0   {netmask=255.255.255.0
                        protocol_ipv6=no}
terminal=vt100
security_policy=NONE
root_password=DcwyMAx8TwtL2
name_service=NONE
timeserver=localhost
```

Note the above `sysidcfg` file contains keywords specific to the Solaris 8 OE which will not work with any previous versions. Both the `network_interface` and `security_policy` keywords are specific to the Solaris 8 OE.

Note – If any errors or unknown variables are found in the `sysidcfg` file, all information in the file will be ignored by JumpStart

---

JumpStart Client Installation

The initial JumpStart client boot messages (using the `sysidcfg` file) are as follows:

```
ok boot net - install
Resetting ...

Sun Ultra 1 SBus (UltraSPARC 167MHz), No Keyboard
OpenBoot 3.1, 128 MB memory installed, Serial #8575119.
Ethernet address 8:0:20:82:d8:8f, Host ID: 8082d88f.

Rebooting with command: boot net - install
Boot device: /sbus/ledma@e,8400010/le@e,8c00000 File and args: -
install
2aa00
Booting the 32-bit OS ...

SunOS Release 5.8 Version Generic 32-bit
Copyright 1983-2000 Sun Microsystems, Inc.  All rights reserved.
whoami: no domain name
Configuring /dev and /devices
Using RPC Bootparams for network configuration information.
Configured interface le0
Using sysid configuration file 10.0.0.20:/jumpstart/sysidcfg
The system is coming up.  Please wait.
Starting remote procedure call (RPC) services: sysidns done.
Starting Solaris installation program...
Searching for JumpStart directory...
Using rules.ok from 10.0.0.20:/jumpstart.
Checking rules.ok file...
Using profile: Profiles/basic.profile
Using finish script: Drivers/bp-iplanetes.driver
Executing JumpStart preinstall phase...
Searching for SolStart directory...
Checking rules.ok file...
Using begin script: install_begin
Using finish script: patch_finish
Executing SolStart preinstall phase...
Executing begin script "install_begin"...
Begin script install_begin execution completed.
```

Toolkit (Parts 2 and 3)

The following articles will present detailed information on Toolkit features—site specific information, environment variables, configuration, and installation details. Additionally, we will dissect the scripts used to harden systems, and provide a guide for adding new scripts to the architecture. Recommendations on which changes may be required for various JumpStart environments will be evaluated to simplify the process of porting the Toolkit scripts and JumpStart environment configuration to other locations.

Parts 2 and 3 of this series are available at:

- `http://www.sun.com/blueprints/1100/jssec2-updt1.pdf`
- `http://www.sun.com/blueprints/1100/jssec3-updt1.pdf`

Conclusion

This article highlighted the problem that drove the development of the Toolkit and has provided an overview and background information on the architecture and functions of the JumpStart software. This software has been designed to provide a mechanism to help install groups of systems automatically and identically. Additionally, the Toolkit includes scripts to harden and minimize systems using the Solaris OE.

Further information on how to setup and configure the JumpStart software was referenced throughout this article and 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, *JumpStart Architecture and Security Scripts for the Solaris Operating Environment - Part 1*, Sun BluePrints OnLine, July 2000,
`http://www.sun.com/blueprints/0700/jssec.pdf`

Noordergraaf, Alex, *JumpStart Architecture and Security Scripts for the Solaris Operating Environment - Part 2*, Sun BluePrints OnLine, August 2000,
`http://www.sun.com/blueprints/0800/jssec2.pdf`

Noordergraaf, Alex, *Solaris Operating Environment Minimization for Security: Updated for Solaris 8*, Sun BluePrints OnLine, November 2000,
`http://www.sun.com/blueprints/1100/minimization-updt1.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 Network Settings for Security*, Sun BluePrints OnLine, December 1999,
`http://www.sun.com/blueprints/1299/network.pdf`

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

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

Acknowledgements

I would like to thank Keith Watson for his input and assistance in the development and testing of the Toolkit.

Keith Watson reviewed the end result of the Toolkit and provided many recommendations on required changes, and also developed some elegant `Finish` script implementations.

Author's Bio: Alex Noordergraaf

Alex Noordergraaf has over 9 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: Solaris OE Minimization for Security, Solaris OE Network Settings, and Solaris OE Security.

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.

Author's Bio: Glenn Brunette

Glenn Brunette has over 8 years experience in the areas of computer and network security. Glenn currently works with in the Sun Professional Services organization where he is the Lead Security Architect for the North Eastern USA region. In this role, Glenn 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, ISP, New Media, and government organizations.

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