

# **Sun<sup>TM</sup> Shared Shell**

**White Paper  
February 2007**

## Table of Contents

<b>1 Overview</b>	<b>3</b>
<b>2 Terminology</b>	<b>3</b>
<b>3 Architecture</b>	<b>4</b>
3.1 Initiator System	5
3.2 Target System	5
3.3 Shared Shell Server Infrastructure	5
3.3.1 TCP/IP Port Requirements	5
3.4 Encryption Processes and Algorithms	7
3.5 Session Logging	8
3.5.1 Windows Operating Systems	8
3.5.2 MAC OS X	8
3.5.3 The Solaris™ Operating System, Linux, and Other UNIX® Operating Systems	8
3.6 Session Log Contents	8
<b>4 Removing the Shared Shell Client</b>	<b>9</b>

## 1 Overview

Sun™ Shared Shell (Shared Shell) is a collaborative service tool that allows, under your control, Sun Support Services engineers to remotely view and diagnose your Sun products. It is a Java™ application that provides secure, shared, remote access to a shell session that both you and Sun Support Services engineers view at the same time. You can also invite additional participants to join the conference as needed. But more important, you are always in control of who participates, who has access to the remote access session, and each participant's shell access level. To use Shared Shell with a Sun Support Services engineer, you must have a SunSpectrum<sup>SM</sup> Support service contract.

All communication between your system and Sun is secured by industry-standard Secure Sockets Layer (SSL) encryption. Shared Shell also provides secure file transfer (which you control) and a chat interface for sending text messages between participants.

The [Shared Shell User's Guide](#) provides the details on using Shared Shell as well as system requirements.

This white paper explains the technical details of Shared Shell so that you can determine its suitability for use in your network and computing environment.

## 2 Terminology

The following terms apply to Shared Shell:

- **Initiator** — The person and Shared Shell client that *initiates* a session. This is the Shared Shell client that makes a local connection to the target system (that is being diagnosed) and invites and approves all other participants.
- **Participant** — A person and Shared Shell client that *joins* a session using an invitation key provided by the initiator.
- **Initiator system** — The system from which the Shared Shell session is launched. Further details about the initiator system are available in section 3.1, *Initiator system*.
- **Target system** — The system being diagnosed, which is initially accessed by a local connection by the conference initiator. Further details about the target system are available in section 3.2, *Target system*.

### 3 Architecture

Figure 1 illustrates the configuration and communication routes of a typical Shared Shell session.

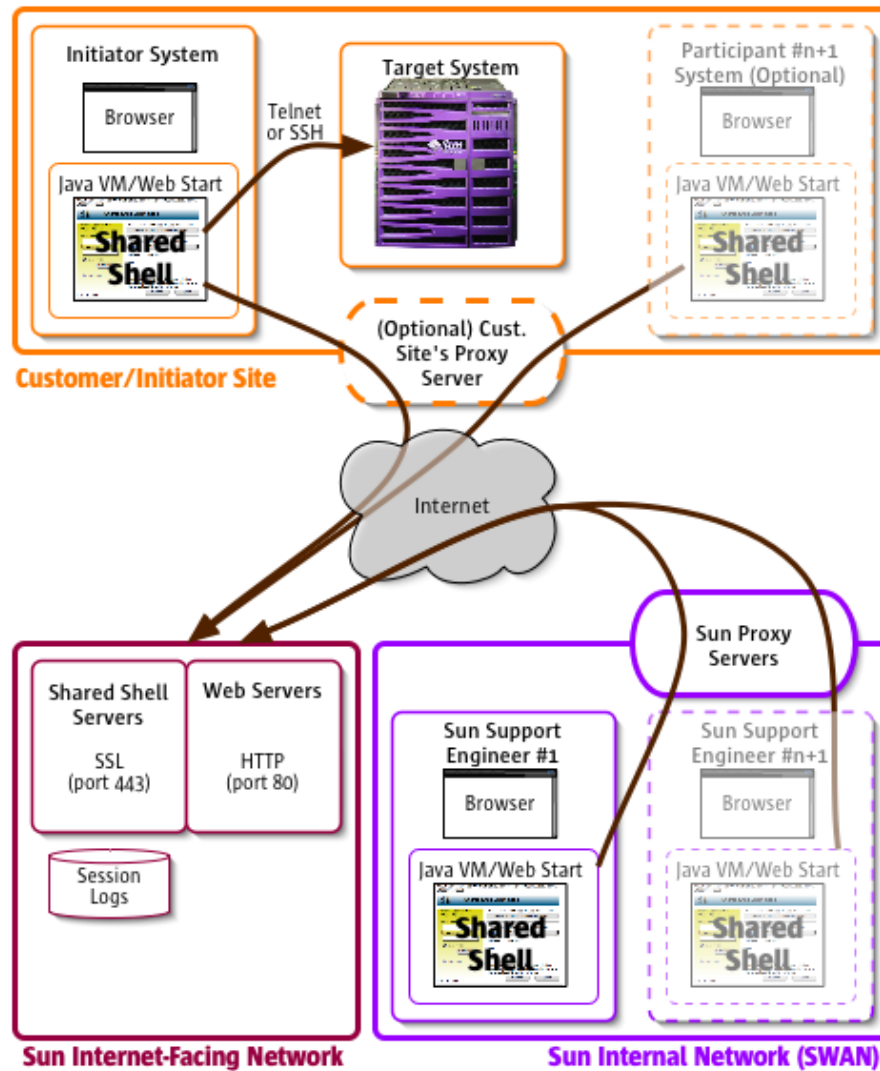


Figure 1: Shared Shell Architecture

### 3.1 Initiator system

The initiator uses a Web browser to access the Shared Shell Web server and initiate the launch of Shared Shell. The Shared Shell client application is launched and automatically updated by Java Web Start software. The initiator site might have a proxy server through which all Internet communication must flow. Shared Shell can use an HTTPS or SOCKS5 proxy server.

### 3.2 Target system

The target system is the system being diagnosed. It must accept connections using Telnet or Secure Shell (SSH). Only the initiator has a connection to the target system. The target system can be a Sun server, a terminal server, a console server, a Linux system, a storage array, or any other device that supports Telnet or SSH access.

*Note: The target system and the initiator system could be the same system.*

### 3.3 Shared Shell server infrastructure

The Web server hosts the launch page, user documentation, help files, all the files needed for launch by Java Web Start software (`jnlp`, `jar`, `icon`), and so on.

The Shared Shell server is the rendezvous point for communication between a Shared Shell initiator and all the participants. This server is responsible for user authentication by means of Common Web Protocol (CWP) and for conference management, security management, and session logging. CWP authentication service is the common authentication service used across sun.com.

The session log database is a record of all sessions, all participants, all shell output, and all chat messages, and is kept for reference in support cases and as an audit trail.

#### 3.3.1 TCP/IP port requirements

The Shared Shell initiator system does not require any open inbound ports. All Internet communications are initiated through outbound connections from the client.

Several outbound ports and connections are needed for Shared Shell. These connections can be direct to the Internet or they can be through a customer-provided proxy server.

The initiator and all participants need to use their Web browser to connect to the Shared Shell launch page, which is [sun.com/sharedshell](http://sun.com/sharedshell).

This launch page uses a normal HTTP connection to port 80.

If the Java Virtual Machine is configured correctly, it starts and automatically downloads all the components it needs through Java Web Start software when a user clicks the Launch button. These downloads are done over normal HTTP connections to port 80.

After Shared Shell starts, it makes connections only if a user clicks the Launch button from the launch page or the Test button from the Proxy Configuration dialog. Once Shared Shell is started, a connection is made from the client to the Shared Shell server over TCP/IP port 443, optionally using a proxy server and whatever port the proxy server uses. This connection is a long-lived connection over which all traffic from the client to the Shared Shell server is sent and received. Though the connection uses port 443 for firewall-friendliness purposes, it is not an HTTPS connection.

Once the Shared Shell server launches, the CWP Web service is queried for authentication, communicating using Simple Object Access Protocol (SOAP) Web service calls on an HTTPS connection to port 443 on the identity.sun.com server. After successful user authentication with the Shared Shell server, the initiator system attempts to connect to the system being diagnosed, either by Telnet or SSH.

Telnet connections default to port 23, but can be set to any valid port number by the initiator. Secure shell connections default to port 22, but can be set to any valid port number by the initiator. If all these connections are successful, a terminal window appears on the initiator's system. The initiator can begin interacting with the target system being diagnosed.

If an error occurs in communication with the Shared Shell server — either with user authentication or with connecting to the target system — all connections are shut down and the initiator is returned to the launch page. Once this is complete, the initiator then uses Conference->Invite to generate a nine-character alphanumeric string called an invitation key. The initiator should then communicate the invitation key to the first participant, typically by phone or some other secure out-of-band channel, such as a secure Instant Messenger session. To add any additional participants, the initiator can then repeat the process.

All participants must launch Shared Shell and communicate with the Shared Shell server in a similar fashion. If they access the Internet through a proxy server, they must specify their own proxy configuration settings. For example, a Sun Support Services engineer on Sun's internal network needs to specify an HTTPS proxy server such as `webcache` on port 8080. A participant connects only to the Shared Shell server without making any other connections.

### 3.4 Encryption processes and algorithms

One of the first elements the initiator sees is the Java Web Start Permissions dialog. This dialog indicates that the application has been signed by Sun Microsystems, Inc. and its VeriSign-managed certificate.

All Java classes and other resources used by Shared Shell are contained within a signed .jar file. Java code signing helps ensure the integrity of all the classes, files, and other items that are downloaded. Any attempts to modify the file will break the signature. More information about Java code signing is available at: [java.sun.com/docs/books/tutorial/security/sigcert/index.html](http://java.sun.com/docs/books/tutorial/security/sigcert/index.html).

All communication between the Shared Shell client and the Shared Shell server is conducted over an SSL version 3 or Transport Layer Security (TLS) version 1 connection. By default, the Java platform chooses the strongest encryption algorithm and key size available on both ends of the connection. In most cases, due to export restrictions, this is AES-128 or 3DES encryption for the packets. Both of these encryption algorithms are widely published, believed to be secure, and commonly used for secure banking transactions and so on. When needed, the SHA1 Secure Hash Algorithm is used for message integrity instead of the older and possibly less secure MD5 algorithm.

For user authentication, all communication between the Shared Shell server and the CWP Identity Web service is also made over an SSL version 3 or TLS version 1 connection. The system chooses the most secure available algorithm, key size, and hash algorithm.

If the initiator chooses to make a Telnet connection to the target system, that connection is unencrypted, but the connection between the Shared Shell initiator and the target system is a local connection. At many sites, the local area network (LAN) is considered secure enough to permit this traffic. If the initiator launches Shared Shell on the target system and makes the shell connection to `localhost` or `127.0.0.1`, the clear text traffic and passwords never leave the machine, providing good security even with an insecure protocol such as Telnet.

If the initiator chooses to make an SSH connection to the system being diagnosed, that connection is fully encrypted and no passwords are sent without being encrypted. For a discussion of the confidentiality, authentication, and integrity provided by the SSH protocol, go to [ietf.org/rfc/rfc4251.txt](http://ietf.org/rfc/rfc4251.txt).

The invitation key is not encrypted. However, it is based on a secure random number with 45 bits of randomness. The invitation key is made of nine alphanumeric characters and is not case-sensitive. The letters B, I, O, and Z are not used, giving 32 letter or digit values (two to the fifth power).

In addition, invitation keys are transient and expire after five minutes if not used. Invitation keys expire immediately once used.

## 3.5 Session logging

Shared Shell logs session information on the initiator system and on the Shared Shell server. Each Shared Shell session is logged by the initiator, if at all possible. The default location for these logs depends on the platform. The log location can be changed using the Preferences menu.

Participant systems do not log the Shared Shell session.

### 3.5.1 Windows operating systems

On Windows systems, the default location for logging Shared Shell information is the following:

```
%HOMEDRIVE%%HOMEPATH%\Sun Shared Shell\logs
```

For United States English installations of Windows, this location is typically the following:

```
C:\Documents and Settings\User Name\Sun Shared Shell\logs
```

The default location might be different depending upon the locale or the administrative settings.

### 3.5.2 MAC OS X

On Macintosh systems, the default location for logging Shared Shell information is the following:

```
$HOME/Library/Application Support/Sun Shared Shell/logs
```

### 3.5.3 The Solaris™ Operating System, Linux, and other UNIX® operating systems

On UNIX® systems, the default location for logging Shared Shell information is the following:

```
$HOME/SunSharedShell/logs
```

Debug logs are also stored in this directory.

## 3.6 Session log contents

Session terminal logs contain only output, not input, from the terminal. Therefore, passwords are not logged unless they are echoed to the screen. As long as the remote system and all utilities used are careful to turn off terminal echo before getting passwords, no passwords are stored in the Shared Shell logs.

For example, users of Oracle SQL Plus should not specify passwords on the command line while in a Shared Shell session. To start SQL Plus so that the user is prompted for the password without the password being echoed to the terminal or stored in the Shared Shell logs, use the following command:

- `sqlplus system@SID`

All chat messages are logged on the Shared Shell server, as are all chat messages received by the initiator on the client side.

The same terminal output is logged on the Shared Shell server as is logged on the initiator's end. The Shared Shell server also tracks usage information for all sessions, such as the identity of the participants and the number and size of packets sent and received.

The session logs are available to Sun Services personal on an as-needed basis. Generally, the logs are not available to other Sun personal.

## 4 Removing the Shared Shell Client

Java Web Start software is not really an installer, but it does cache files (such as `ssa.jar`) locally to provide fast startup on subsequent launches of the application.

On Windows systems, you should be able to remove the Shared Shell application from the Java control panel. You can find the application by using the following path:

Start -> Control Panel -> Java -> General tab -> Temporary Internet Files section -> Settings... -> View Applications

On Mac OS X systems, use one of the following tools to remove the Shared Shell application:

- `/Applications/Utilities/Java/J2SE5.0/Java Cache Viewer.app`
- `/Applications/Utilities/Java/Java Web Start.app`

On other UNIX platforms, run the `javaws` utility for the Java Runtime Environment or the Java Development Kit.

To remove the Shared Shell from any platform, display all applications that are managed by the Java Web Start software. Then highlight Shared Shell and click Remove or Delete.

If you also want to delete the session log files, see the Session Logging section for the location of the log files.

**Sun Microsystems, Inc.** Network Circle, Santa Clara, CA 95054 USA **Phone** 1-650-960-1300 or 1-800-555-9SUN **Web** sun.com

THIS PRODUCT CONTAINS CONFIDENTIAL INFORMATION AND TRADE SECRETS OF SUN MICROSYSTEMS, INC. USE, DISCLOSURE OR REPRODUCTION IS PROHIBITED WITHOUT THE PRIOR EXPRESS WRITTEN PERMISSION OF SUN MICROSYSTEMS, INC.

Use is subject to license terms. This distribution may include materials developed by third parties.

© 2007 Sun Microsystems, Inc. All rights reserved. Sun, Sun Microsystems, the Sun logo, Java, Solaris, and SunSpectrum are trademarks or registered trademarks of Sun Microsystems, Inc. in the United States and other countries. UNIX is a registered trademark in the United States and other countries, exclusively licensed through X/Open Company, Ltd. SSH is a registered trademark of SSH Communications Security Corp in the United States and/or other countries. Information subject to change without notice.

U.S. Government Rights-Commercial use. Government users are subject to the Sun Microsystems, Inc. standard license agreement and applicable provisions of the FAR and its supplements.

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.

02/07 SunWIN #496790