Sun Fire™ V20z and Sun Fire V40z Servers — Server Management Guide

Sun Microsystems, Inc.
www.sun.com

Part No. 817-5249-12
December 2004, Revision A

Submit comments about this document at: `http://www.sun.com/hwdocs/feedback`

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.

LA DOCUMENTATION EST FOURNIE «EN L'ÉTAT» ET TOUTES AUTRES CONDITIONS, DÉCLARATIONS ET GARANTIES EXPRESSES OU TACITES SONT FORMELLEMENT EXCLUES, DANS LA MESURE AUTORISÉE PAR LA LOI APPLICABLE, Y COMPRIS NOTAMMENT TOUTE GARANTIE IMPLICITE RELATIVE À LA QUALITÉ MARCHANDE, À L'APTITUDE À UNE UTILISATION PARTICULIÈRE OU À L'ABSENCE DE CONTREFAÇON.

Contents

Preface

This guide explains how to manage the Sun Fire™ V20z and Sun Fire V40z servers.

How This Book Is Organized

Chapter 1 provides an overview of the ways in which a user can manage the servers.

Chapter 2 describes how to manage the servers through the Intelligent Platform Management Interface (IPMI).

Chapter 3 describes how to manage the servers through the Simple Network Management Protocol (SNMP).

Chapter 4 provides further management information, such as how to enable scripting capability, Console Redirection over Serial and Serial-over-LAN.

Appendix A contains an overview of the server management commands that you can use to manage the server. Following appendixes describe each command type in detail.

Appendix B contains detailed descriptions of Access commands.

Appendix C contains detailed descriptions of Diagnostics commands.

Appendix D contains detailed descriptions of Inventory commands.

Appendix E contains detailed descriptions of IPMI commands.

Appendix F contains detailed descriptions of Platform commands.

Appendix G contains detailed descriptions of Sensor commands.

Appendix H contains detailed descriptions of Service Processor (sp) commands.

Related Documentation

Application	Title	Part Number
Safety notices and international compliance certification statements	*Sun Fire V20z and Sun Fire V40z Servers Safety and Compliance Guide*	817-5251-xx
Safety information	*Important Safety Information for Sun Hardware Systems*	816-7190-xx
Hardware installation and initial configuration	*Sun Fire V20z and Sun Fire V40z Servers Installation Guide*	817-5246-xx
Operating system installation	*Sun Fire V20z and Sun Fire V40z Servers Linux Operating-System Installation Guide*	817-5250-xx
Service and Diagnostics	*Sun Fire V20z and Sun Fire V40z Servers User Guide*	817-5248-xx
Release notes and updated information	*Sun Fire V20z and Sun Fire V40z Servers Release Notes*	817-5252-xx

Accessing Sun Documentation

You can view, print, or purchase a broad selection of Sun documentation, including localized versions, at:

`http://www.sun.com/documentation`

Third-Party Web Sites

Sun is not responsible for the availability of third-party web sites mentioned in this document. Sun does not endorse and is not responsible or liable for any content, advertising, products or other materials that are available on or through such sites or resources. Sun will not be responsible or liable for any actual or alleged damage or loss caused by or in connection with the use of or reliance on any such content, goods or services that are available on or through such sites or resources.

Contacting Sun Technical Support

If you have technical questions about this product that are not answered in this document, go to:

`http://www.sun.com/service/contacting`

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can submit your comments by going to:

`http://www.sun.com/hwdocs/feedback`

Please include the title and part number of your document with your feedback:

Sun Fire V20z and Sun Fire V40z Servers—Server Management Guide, part number 817-5249-12

CHAPTER 1

Introduction

Overview

Strong server-management capabilities are crucial to maintaining mission-critical servers. Advance notification of problems and rapid diagnosis and correction are critical functions to an environment in which a few servers bear the bulk of the workload. The Sun Fire™ V20z and Sun Fire V40z servers and their extensive server-management capabilities lower costs by reducing failure and by potentially eliminating hands-on management.

This document describes how to perform remote management on the Sun Fire V20z and Sun Fire V40z servers.

The Sun Fire V20z server is an AMD Opteron processor-based, enterprise-class one-rack-unit (1U), two-processor (2P) server. The Sun Fire V40z server is also an AMD Opteron processor-based server, but is a three-rack-unit (3U), four-processor (4P) server.

These servers include an embedded Service Processor (SP), flash memory, RAM, a separate Ethernet interface and server-management software. They come equipped with superior server-management tools for greater control and minimum total cost of ownership. You can use the command-line interface (CLI), SNMP integration with third-party frameworks, or IPMI to configure and manage the platform with the SP. The dedicated SP provides complete operating-system independence and maximum availability of server management.

Acronyms

TABLE 1-1 defines the acronyms found in this document.

TABLE 1-1 Acronyms

Acronym	**Explanation**
ACPI	Advanced Configuration and Power Interface
ARP	Address Resolution Protocol
BMC	Baseboard Management Controller
CRU	Customer-Replaceable Unit
DPC	Direct Platform Control
FRU	Field-Replaceable Unit
grub	Grand Unified Bootloader
IPMI	Intelligent Platform Management Interface
KCS	Keyboard Controller Style
KVM	Keyboard, video and mouse
LAN	Local Area Network
LILO	Linux Loader
LOM	Lights Out Management
MIB	Management Information Base
RMCP	Remote Management Control Protocol
SDR	Sensor Data Record
SEL	System Event Log
SNMP	Simple Network Management Protocol
SOL	Serial Over LAN
SP	Service Processor
SSU	System Setup Utility
SunMC	Sun Management Center
UART	Universal Asynchronous Receiver/Transmitter
UDP	User Datagram Protocol
WAN	Wide Area Network

Server Management

There are several options for remotely managing a Sun Fire V20z or Sun Fire V40z server:

- Lights Out Management (LOM) through IPMItool
- Simple Network Management Protocol (SNMP)

Service Processor

The Sun Fire V20z and Sun Fire V40z servers include a dedicated chipset for complete operating-system independence and maximum availability of server-management functions. This chipset, called Service Processor (SP), is an embedded PowerPC chip providing the following:

- Environmental monitoring of the platform (such as temperatures, voltages, fan speeds and panel switches)
- Alert messages when problems occur
- Remote control of server operations (boot, shutdown and reboot of the server's operating system, turning the server's power on and off, stopping the server's boot process in BIOS, and upgrading the BIOS)

Note – In this document, you might see references to a Baseboard Management Controller (BMC). A BMC is a dedicated IPMI controller. The SP found in these servers is a general-purpose, embedded CPU that contains software to emulate a BMC.

FIGURE 1-1 shows the back panel of the Sun Fire V20z server.

FIGURE 1-2 shows the back panel of the Sun Fire V40z server.

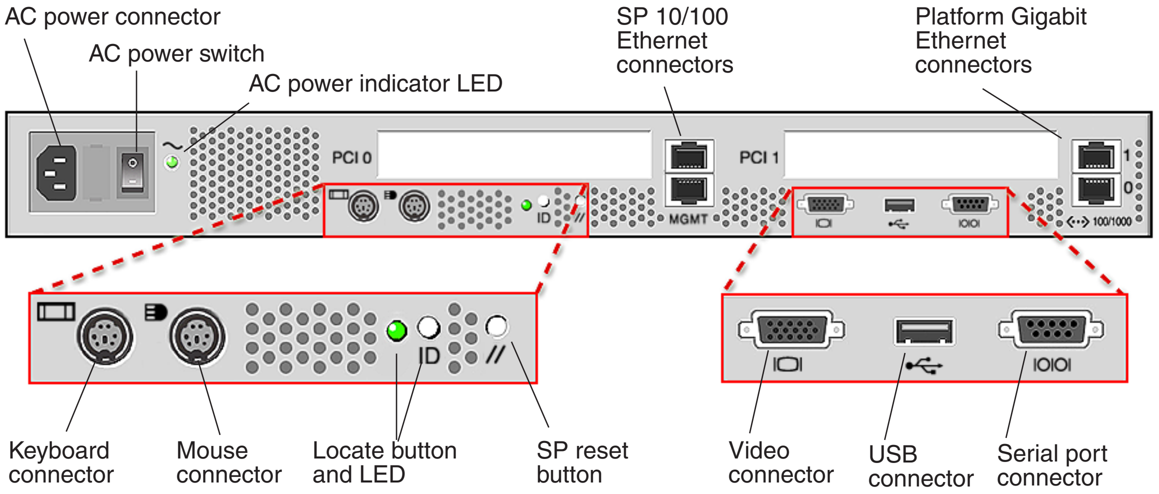

FIGURE 1-1 Sun Fire V20z Back Panel

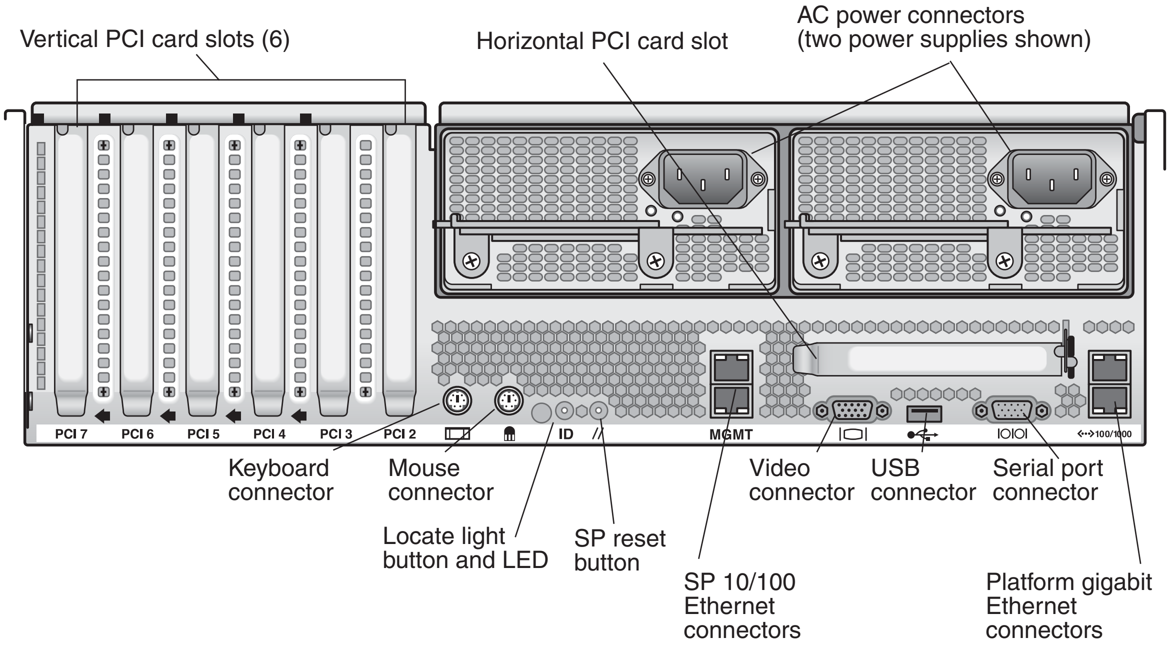
Vertical PCI card slots (6)
Horizontal PCI card slot
AC power connectors (two power supplies shown)
PCI 7
PCI 6
PCI 5
PCI 4
PCI 3
PCI 2
ID
MGMT
100/1000
Keyboard connector
Mouse connector
Locate light button and LED
SP reset button
Video connector
USB connector
Serial port connector
SP 10/100 Ethernet connectors
Platform gigabit Ethernet connectors

FIGURE 1-2 Sun Fire V40z Back Panel

Server-Management Interfaces

These servers include local and remote server-management capabilities through the SP; the SP supports four server-management interfaces:

- IPMI using a Keyboard Controller Style (KCS) interface and an IPMI kernel driver (in-band)
- IPMI over local area network (LAN) (out-of-band)
- SNMP integration with third-party SNMP management consoles
- Command-line-interface (CLI) LOM

Command Line Interface

Server-management capabilities are available from the command line.

See Appendix A for a list of server-management commands that you can use with these servers, as well as a description, the command format, a list of arguments and a list of return codes for each command.

SSH and Scripting Capabilities

A system administrator can log in to the Service Processor using SSH and issue commands, or more commonly, write a shell script that remotely invokes these operations.

The server-management commands enable you to efficiently manage each area of the server. From the command line, you can write data-driven scripts that automate the configuration of multiple machines. For example, a central management system can cause many servers to power on and boot at a specified time, or when a specific condition occurs.

For more information about scripting, see "Further Management Information" on page 55.

SNMP Integration

SNMP management provides remote access by SNMP-compliant entities to monitor the health and status of the server. The SP sends SNMP alerts to external management functions when warranted.

For more information about SNMP, refer to "SNMP Server Management" on page 41.

The diagram in FIGURE 1-3 illustrates the communications paths for the different server-management options.

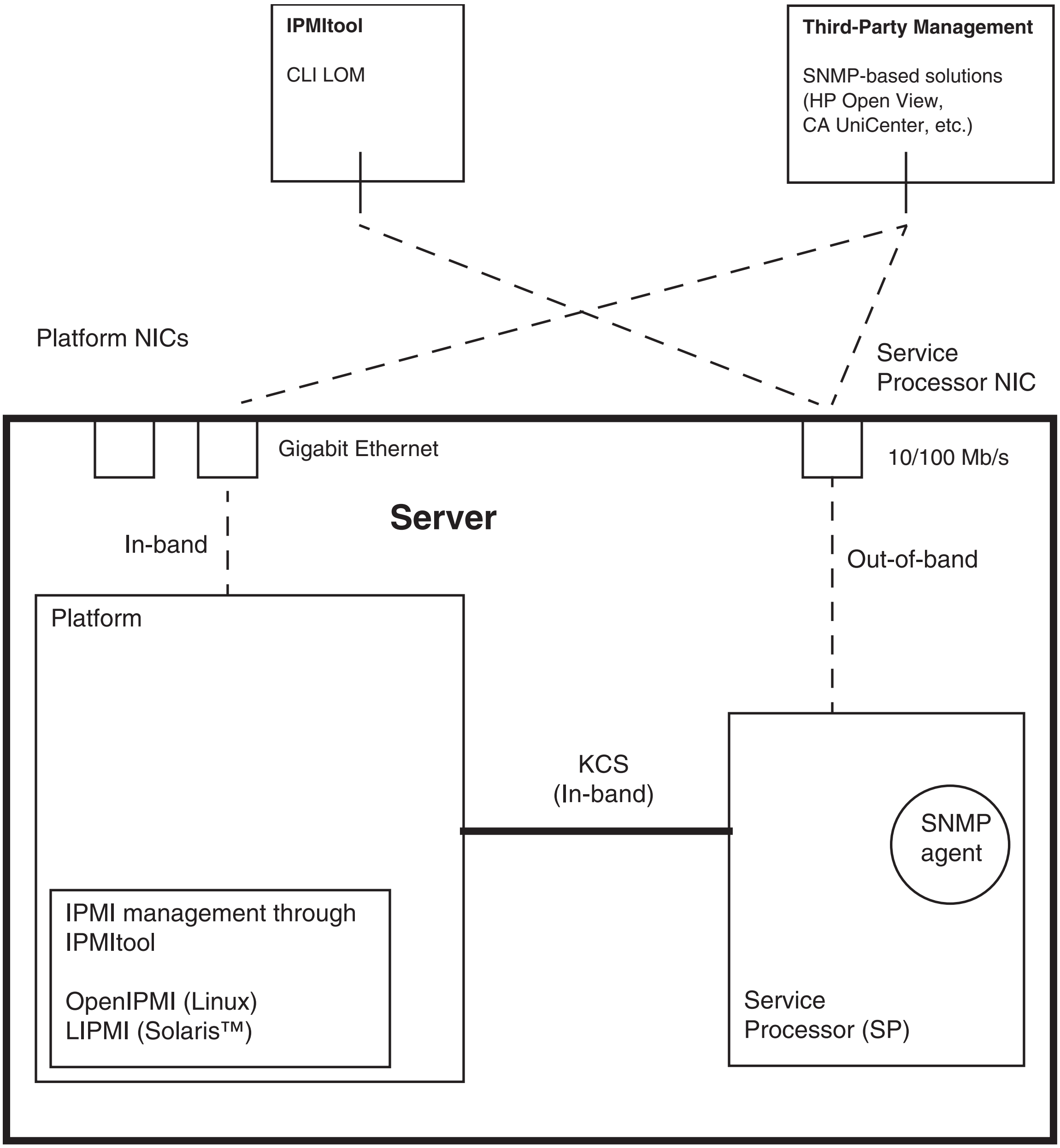

FIGURE 1-3 Diagram of the Server-Management Options

Operator Panel

You can use the operator panel to configure network settings for the SP. See FIGURE 1-4 or FIGURE 1-5 for the operator panel location on your server.

Note – The SP defaults to Dynamic Host Configuration Protocol (DHCP) networking if the operator panel is not interactively engaged on the first power-up.

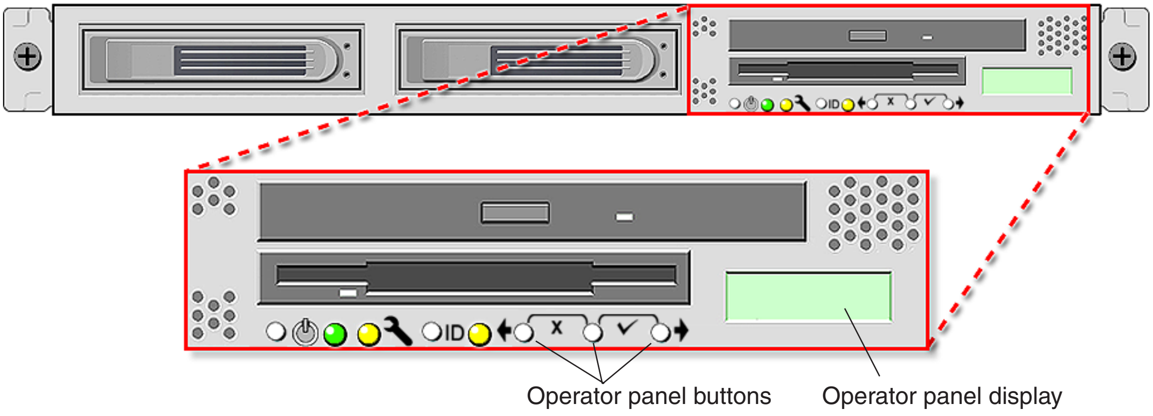

FIGURE 1-4 Sun Fire V20z Server Operator Panel and Buttons

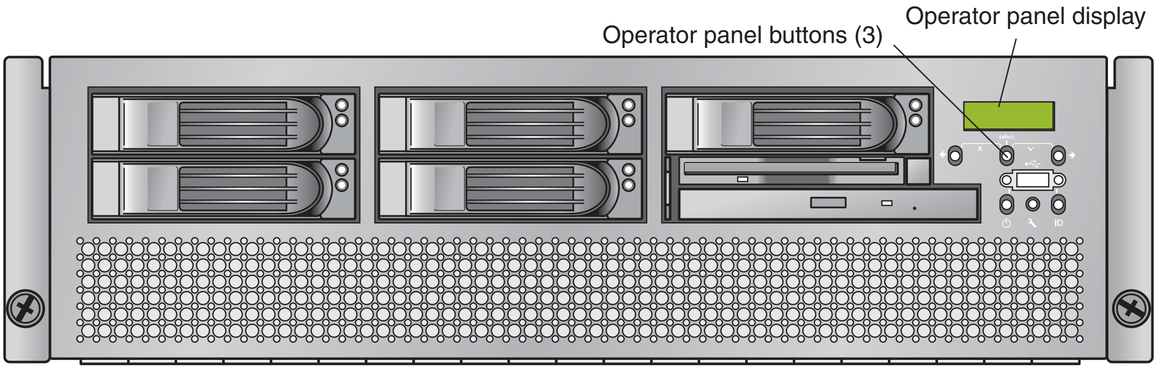

FIGURE 1-5 Sun Fire V40z Server Operator Panel and Buttons

The operator panel displays information on the LCD display in two lines; you respond to prompts or initiate actions using the following buttons:

TABLE 1-2 Operator Panel Buttons

Buttons	Function
◀●	Back/No
●	Select
●▶	Forward/Yes
⌜✓⌝ ● ●▶	Enter
⌜✕⌝ ◀● ●	Cancel

If a menu or data-entry screen displays for more than 30 seconds with no action taken, the menu or data entry is cancelled and the display returns to the idle/background state.

For every action that you confirm, feedback displays on the panel to indicate success, failure, or that the action has been initiated.

The Back and Forward buttons automatically scroll, repeating the action as long as the button is held down. After holding the button down a few seconds, auto scrolling begins and rapidly increments or decrements the value.

User Groups

Administrators can define several different user groups, or types, on the server. Capabilities of the different user types are defined in TABLE 1-3.

For example, when you log in to the system the first time using the setup account, the first thing you must do is set up the initial manager account so that other user accounts can be managed. (see "Creating the Initial Manager Account" on page 14 for details)

TABLE 1-3 User Types

User Type	Capability
monitor	Read-only access for sensor data and log displays.
admin	All capabilities except user account management and SP field upgrades
manager	All capabilities except SP field upgrade
service	SP field upgrades

Initial Setup of the Service Processor

This procedure describes the steps for the initial setup of the SP.

Part I: Assigning Network Settings to the SP

This section contains two alternate methods you can use to define SP network settings:

- "Assigning SP Network Settings Using DHCP" on page 10
- "Assigning Static SP Network Settings" on page 12

Note – As an alternative, if no DHCP server or physical access is available, you can configure the SP using IPMItool in conjunction with an IPMI kernel driver. To configure your server for IPMI, perform the correct procedures for your operating system in "Part III: Enabling IPMI Access on the Server" on page 15, then "Part IV: Enabling IPMI LAN Access" on page 17.

Assigning SP Network Settings Using DHCP

The following procedure describes how to set the SP network settings using DHCP from the Operator Panel. If your network does not use DHCP, or you want to assign a static IP address to the SP, follow the instructions in "Assigning Static SP Network Settings" on page 12.

Note – This procedure assumes that you have cabled the server and powered it on as described in the Sun Fire V20z and Sun Fire V40z Servers Installation Guide. At least of the server's SP ports must be connected to a LAN.

---

1. **Press any operator panel button on the server front panel (see** FIGURE 1-6**).**

   The LCD panel displays the first menu option:

   ```
   Menu:
   Server Menu
   ```

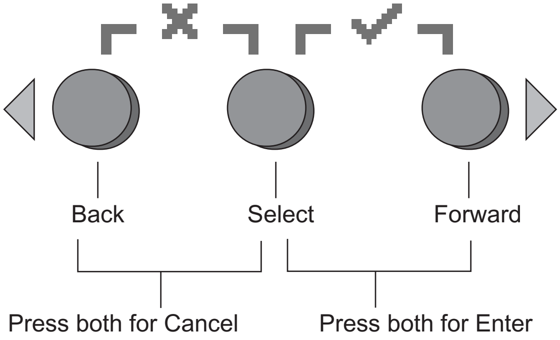

FIGURE 1-6 Operator Panel Buttons

2. **Press the Forward button until you reach the SP menu:**

```
Menu:
SP menu
```

3. **Press the Select button to display the SP menu options.**

```
SP Menu:
Set SP IP info?
```

4. **Press the Select button.**

   The following prompt appears with the default response:

```
SP use DHCP?
No
```

5. **Press the Forward button to change to `Yes`, then press the Select button.**

6. **Press the Select button at the confirmation prompt.**

```
SP use DHCP:
Yes?
```

   The server attempts to contact a DHCP server for an IP address. Once a DHCP server is contacted, the LCD panel displays the default SP settings. The SP address is configured and the server is ready for use.

7. **Continue with "Part II: Securing the Service Processor" on page 14 for instructions on creating the initial manager account.**

Note – A prompt appears that asks if you want to perform autoconfiguration. As an alternative to configuring an SP manually, you can run autoconfiguration, which replicates the configuration of one SP to another. Refer to "Autoconfiguring the SP (Optional Method)" on page 24for instructions on autoconfiguration.

Assigning Static SP Network Settings

Follow these steps to set the SP network settings using a static IP address. You must specify a subnet mask and default gateway. This example uses the following sample settings:

IP Address: 10.10.30.5
Subnet Mask: 255.255.255.0
Default Gateway: 10.10.30.254

1. **Press any operator panel button on the server front panel (see** FIGURE 1-6**).**

   The LCD panel displays the first menu option:

   ```
   Menu:
   Server Menu
   ```

2. **Press the Forward operator panel button until you reach the SP menu:**

   ```
   Menu:
   SP menu
   ```

3. **Press the Select operator panel button to display the SP menu options.**

   ```
   SP Menu:
   Set SP IP info?
   ```

4. **Press the Select operator panel button. The following prompt displays with the default response:**

   ```
   SP use DHCP?
   No
   ```

5. **Press the Select operator panel button.**

   The LCD displays as follows:

   ```
   SP IP Address:
   0.0.0.0
   ```

6. **With the cursor in the first field, increase or decrease the value using the Back and Forward operator panel buttons.**

   This field can hold a value between 0 and 255.

   ```
   SP IP Address:
   10.0.0.0
   ```

7. **After reaching your desired value, press the Select operator panel button to advance the cursor to the next field.**

   ```
   SP IP Address:
   10.0.0.0
   ```

   Note – The Back and Forward operator panel buttons automatically scroll, repeating the action as long as the button is held down.

   ---

8. **Repeat Step 6 and Step 7 for each field until the desired IP address is displayed, then use the Enter button combination to save the IP Address.**

   The process continues to the next network setting, the Subnet Mask. The LCD displays as follows:

   ```
   SP netmask:
   255.255.255.0
   ```

9. **Edit the subnet mask setting in the same manner as you did for the IP address. When finished, use the Enter button combination to save the subnet mask.**

   The process continues to the next network setting, the default gateway. The LCD displays as follows:

   ```
   SP IP Gateway
   10.10.30.1
   ```

10. **Edit the default gateway setting in the same manner as you did for the IP address and the subnet mask. When finished, use the Enter button combination to save the default gateway.**

    The LCD displays the following confirmation prompt:

    ```
    Use new IP data:
    Yes?
    ```

11. **Press the Select operator panel button to use the new data, or use the Cancel button combination to disregard.**

    The SP address is now configured and the server is ready for use.

    Note – A prompt appears that asks if you want to perform autoconfiguration. As an alternative to configuring an SP manually, you can run autoconfiguration, which replicates the configuration of one SP to another. Refer to "Autoconfiguring the SP (Optional Method)" on page 24 for instructions on autoconfiguration.

    ---

12. **Continue with "Part II: Securing the Service Processor" on page 14.**

Part II: Securing the Service Processor

After you install the server and configure the SP, you must create the initial manager account to secure and access the server. You can then perform initial configuration of the server and create additional user accounts.

Caution – The SP must be secured with a user name and password when the server is first deployed. Failure to secure the SP can expose the server to security and availability risks through the SP network interface.

Creating the Initial Manager Account

A setup account is included with each server. This setup account has no password. When you log in to the SP the first time using the setup account, you are prompted to define the initial manager account with a password and an optional public key.

Log in to the setup account and create the initial manager account by following this procedure:

1. **Using an SSHv1 or SSHv2 client, connect to the IP address of the SP.**

2. **Authenticate as the user *setup* with no password required.**

   ```
   # ssh spipaddress -l setup
   ```

3. **Follow the on-screen prompts to create the initial manager account.**

   After you create the initial manager account, the setup account is deleted and you are logged out of the server. You can then log in using the new manager account, from which you can create other user accounts.

Note – If you are prompted for a password, this indicates that the SP has already been secured with an account. If you do not know the management user name and password, you can reset the SP from the operator panel by navigating to the `SP` menu and selecting the `Use defaults` option. Note that all current settings for users and networks will be lost and the SP will reboot.

Note – The IP address, user name and password that you configure are referred to in subsequent examples as the *spipaddr*, *spuser* and *sppasswd*.

Part III: Enabling IPMI Access on the Server

This section contains two alternate procedures: one for a Linux-based server and one for a Solaris-based x86 server. Use the procedure that corresponds to your OS:

- "Enabling IPMI Access on a Linux-Based Server (In-Band)" on page 15
- "Enabling IPMI Access on a Solaris-Based x86 Server (In-Band)" on page 17

Enabling IPMI Access on a Linux-Based Server (In-Band)

1. **Log in to the server and authenticate as the user *root*.**

2. **Install the custom OpenIPMI Linux kernel driver from the Sun Fire V20z and Sun Fire V40z Servers Documentation and Support Files CD. The drivers are located in the CD directory `/support/sysmgmt/`.**

   Browse to the OS variant installed on your server. The options are:

   - redhat/rhel3 for Red Hat Enterprise Linux, version 3 (32-bit mode uses the architecture type "i386"; 64-bit mode uses architecture type "x86_64")
   - suse/sles8 for SUSE Enterprise Linux, version 8 (32-bit mode uses the architecture type "i386"; 64-bit mode uses architecture type "x86_64")
   - suse/suse9 for SUSE 9 Professional

3. **Ensure that the kernel-source RPM is already installed on your distribution by running the command:**

   ```
   # rpm -qvi kernel-source
   ```

   If this utility reports that the kernel-source software package is not installed, install the kernel-source RPM that is current for your installed Linux distribution.

   - On SUSE distributions, install the kernel-source RPM by running the command:

     ```
     # yast2
     ```

   - On RedHat distributions, download the current kernel-source RPM to a temporary directory (such as `/tmp`). Install the package by running the command:

     ```
     # rpm -ivh /tmp/kernel-source*.rpm
     ```

4. **Install the OpenIPMI Linux kernel driver RPM.**

   a. **Browse to the OS variant installed on your server. The options are:**

   - redhat/rhel3 for Red Hat Enterprise Linux, version 3 (32-bit mode uses the architecture type "i386"; 64-bit mode uses architecture type "x86_64")
   - suse/sles8 for Suse Enterprise Linux, version 8 (32-bit mode uses the architecture type "i386"; 64-bit mode uses architecture type "x86_64")
   - suse/suse9 for Suse 9 Professional

b. **Install the OpenIPMI RPM file by running the command:**

```
# rpm -ivh openipmi*.rpm
```

Note – The kernel driver will be compiled using the kernel-source code during installation.

---

5. **Install IPMItool.**

   IPMItool is the command-line-interface (CLI) server-management client.

   - If the installed Linux distribution uses the 32-bit “i386” architecture, run the following command:

     ```
     # rpm -ivh ipmitool*.i386.rpm
     ```

   - If the installed Linux distribution uses the 64-bit “x86_64” architecture, run the following command:

     ```
     # rpm -ivh ipmitool*.x86_64.rpm
     ```

6. **Test the IPMI kernel device driver and client application by running the following command:**

   ```
   # ipmitool -I open chassis status
   ```

   Successful output should look similar to the following:

   ```
   "
   System Power: on
   Power Overload: false
   Power Interlock: inactive
   Main Power Fault: false
   Power Control Fault: false
   Power Restore Policy: unknown
   Last Power Event:
   Chassis Intrusion: inactive
   Front-Panel Lockout: inactive
   Drive Fault: false
   Cooling/Fan Fault: false
   "
   ```

Note – On a subsequent reboot, the IPMI kernel driver may have to be loaded with the following command:

```
# modprobe ipmi_kcs_drv
```

Note – If you upgrade your Linux kernel, refer to “Upgrading the Linux Kernel” on page 19.

Enabling IPMI Access on a Solaris-Based x86 Server (In-Band)

1. **Log in to the server and authenticate as the user root.**
2. **Run the following command to install the LIPMI Solaris x86 kernel driver and the IPMItool management control application.**

   These files are located on the Documentation and Support Files CD in the `/support/sysmgmt/solaris9` directory.

   ```
   # pkgadd -d ./
   ```

   Confirm installation of all packages when prompted.

3. **Reboot the server.**

Part IV: Enabling IPMI LAN Access

This section contains three alternate procedures: two in-band procedures and one out-of-band procedure. Use the procedure that corresponds to your OS:

- "Enabling IPMI LAN Access on a Linux-Based Server (In-Band)" on page 17
- "Enabling IPMI LAN Access on a Solaris-Based x86 Server (In-Band)" on page 18
- "Alternate Method for Enabling IPMI LAN Access (Out-of-Band)" on page 18

Enabling IPMI LAN Access on a Linux-Based Server (In-Band)

1. **If the server is powered off, boot the local OS.**
2. **Log in to the server and authenticate as the user *root*.**
3. **Load the OpenIPMI kernel device driver (as installed in Step 3 of "Enabling IPMI Access on a Linux-Based Server (In-Band)" on page 15).**

   ```
   # modprobe ipmi_kcs_drv
   ```

4. **Using the following commands in IPMItool, configure the network setting for the SP.**

   Note – For more information on the syntax for IPMItool commands, refer to "Syntax" on page 30.

   ```
   # ipmitool -I open lan set 6 ipaddr ipaddr
   # ipmitool -I open lan set 6 netmask netmask
   # ipmitool -I open lan set 6 defgw ipaddr gwipaddr
   # ipmitool -I open lan set 6 password ipmipasswd
   ```

Enabling IPMI LAN Access on a Solaris-Based x86 Server (In-Band)

1. **If the server is powered off, boot the local OS.**
2. **Log in to the server and authenticate as the user *root*.**
3. **Using IPMItool, configure the network setting for the SP by using the following commands.**

Note – For more information on the syntax for IPMItool commands, refer to "Syntax" on page 30.

```
# ipmitool -I lipmi lan set 6 ipaddr ipaddr
# ipmitool -I lipmi lan set 6 netmask netmask
# ipmitool -I lipmi lan set 6 defgw ipaddr gwipaddr
# ipmitool -I lipmi lan set 6 password ipmipasswd
```

Alternate Method for Enabling IPMI LAN Access (Out-of-Band)

1. **Using an SSHv1 client or SSHv2 client, log in to the IP address of the SP.**
2. **Authenticate as the newly created management user (see "Part II: Securing the Service Processor" on page 14").**

```
# ssh spipaddr -l spuser
```

3. **Enable IPMI LAN access and assign a password when prompted.**

```
# ipmi enable channel lan
# exit
```

Note – This password will be referred to as *ipmipasswd* in subsequent examples.

---

4. **Using IPMItool, test the IPMI LAN access.**

```
# ipmitool -I lan -H spipaddr -P ipmipasswd chassis status
```

Upgrading the Linux Kernel

Upgrading the installed Linux kernel to a newer version requires you to recompile the upgraded IPMI kernel device driver.

1. **Install the kernel-source RPM that matches the version of the upgraded kernel binary RPM package.**

2. **Log in to the server and authenticate as the user *root.***

3. **Change to the following directory:**

   ```
   # cd /usr/src/kernel-modules/openipmi
   ```

4. **Recompile the module by running the following commands:**

   ```
   # make clean
   # make
   # make install
   ```

5. **Re-test the IPMI kernel device driver and client application by running the following command:**

   ```
   # ipmitool -I open chassis status
   ```

   Successful output should look similar to the following:

   ```
       "
       System Power: on
       Power Overload: false
       Power Interlock: inactive
       Main Power Fault: false
       Power Control Fault: false
       Power Restore Policy: unknown
       Last Power Event:
       Chassis Intrusion: inactive
       Front-Panel Lockout: inactive
       Drive Fault: false
       Cooling/Fan Fault: false
       "
   ```

Note – On a subsequent reboot, the IPMI kernel driver may have to be loaded with the following command:

```
# modprobe ipmi_kcs_drv
```

Daisy-Chaining the Servers

You can interconnect multiple servers in a daisy-chain configuration by using the SP connectors to form a management LAN as shown in FIGURE 1-7. This figure also shows how the servers are connected to external LANs using the platform gigabit connectors.

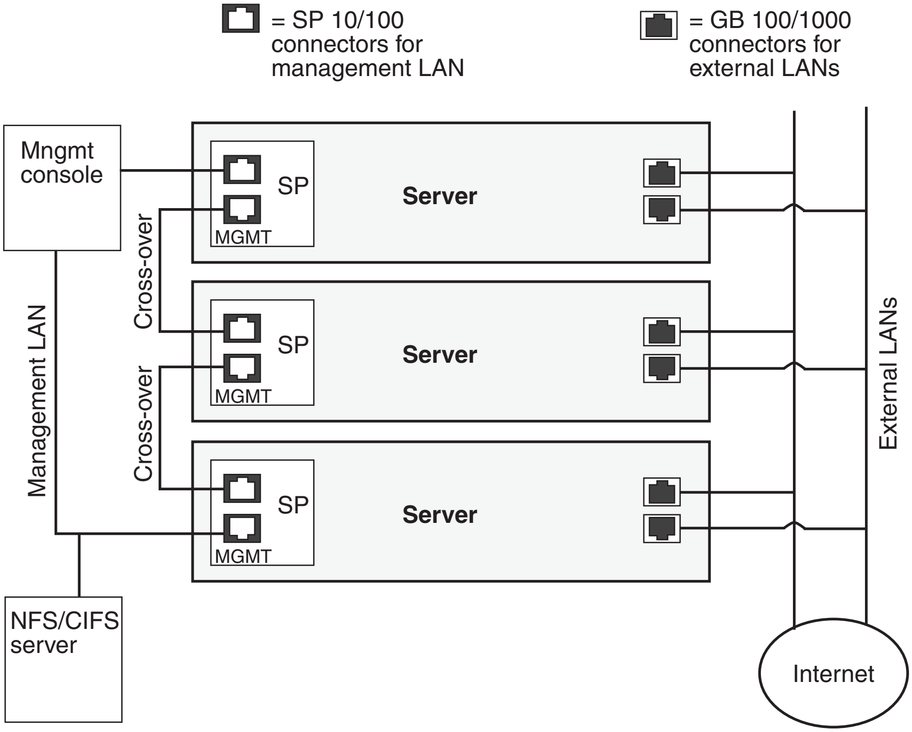

FIGURE 1-7 Daisy Chain Architecture

To interconnect the servers, you must use an RJ-45 cross-over cable. Cables can be connected to either the top or bottom SP port. To configure servers in a daisy chain, connect the first and last server in the chain to different switches.

Managed spanning-tree capable switches are required to redundantly connect both the top and bottom of the chain. If the switch is not capable of spanning-tree discovery, then only connect either to the top or the bottom of the chain, but not both.

Site Integration

When deploying your server, ensure that you determine the best integration strategy for your environment.

These servers include network connections for the service processor (SP) that are separate from network connections for the platform. This allows you to configure the server so that the SP is connected to an isolated, management network and is not accessible from the production network.

Updating the SP Software

Note – For complete information about the menu options available through the operator panel, refer to the *Sun Fire V20z and Sun Fire V40z Servers User Guide*.

If you attempt to update the SP software using the operator panel when the IP address for the SP has not been set, the update fails. Ensure that the IP address has been set prior to attempting an update. For more information, refer to the *Sun Fire V20z and Sun Fire V40z Servers Installation Guide*.

Refer to "Operator Panel" on page 7 for general orientation and usage of the operator panel.

Note – Prior to executing this procedure, you must start the Java™ Update Server. Refer to "Updating the Service Processor Base Component" on page 23 for details about starting the Java Update Server.

To update the SP software:

1. **When the LCD displays the Service Processor information (as shown in the following example), press any button.**

   ```
   123.45.67.89
   OS running
   ```

   The LCD displays the first menu option:

   ```
   Menu:
   Server Menu
   ```

2. **Press the Forward button until you reach the SP menu.**

   ```
   Menu:
   SP menu
   ```

3. **Press Select or Enter to display the SP menu's options.**

   ```
   SP Menu:
   Set SP IP info?
   ```

4. **Press the Forward button until you reach the Update SP Flash menu option.**

   ```
   SP Menu:
   Update SP Flash?
   ```

5. **Press Select or Enter.**

6. **A string of 0s displays with the cursor at the left digit. Use the Forward and Back buttons to increment or decrement a digit.**

Note – You are prompted for an IP address. If you attempt to update the SP software using the operator panel when the IP address for the SP has not been set, the update fails.

Note – If you need to supply a port address, it can be any number between 0 and 65535. The leading 0s are removed.

See Step 3 in "Updating the Service Processor Base Component" on page 23 for more information.

---

7. **Press Select to move to the next digit.**

8. **Press Select when finished to return to the left-most column.**

9. **Press the button combination for Enter.**

Updating the Service Processor Base Component

To update the SP base component:

1. **Start the spupdate server on a machine with a Java Runtime Environment (JRE) by running the following command:**

   ```
   # java -jar spupdate.jar -f filename [ -p port ]
   ```

   The `spupdate.jar` file is located in the `spupdate` folder of the Network Share Volume (NSV).

   In this command, *filename* is an SP `.image` file located in `sw_images/sp/spbase/`*version*. This `sw_images` directory contains an SP base `.image` file for each version available.

   By default, the server uses port number 52708. If this port number is already in use, specify another port using the optional `-p` flag.

   The update server does not start if the file is not found in the specified path. Otherwise, the server is ready to receive update requests from any SP. The update server can simultaneously accept multiple update requests from different SPs.

2. **Log in to the SP by running the following command:**

   ```
   # ssh spipaddr -l spuser
   ```

3. **Run the sp command to start the update process on the SP:**

   ```
   # sp update flash all {-i | --ipaddress} IPADDRESS [{-p | --port} PORT]
   ```

   Note – This command includes the optional `-p` flag to denote that the server is running on a port other than the default port. This command pings the update server to see if the update server is up and running. If successful, your connection is closed when the SP reboots and the update process begins.

   Refer to Appendix A for more information about the `sp` commands.

4. **Monitor progress of the update process on the server.**

   Messages display as the installation process progresses. When complete, the SP reboots with the new version installed.

Autoconfiguring the SP (Optional Method)

Autoconfiguration replicates the majority of configuration files from an SP that has already been configured to another SP, so that the two servers have identical configurations, except for the host name and IP address.

For example, after you configure a single SP (set up users, hosts, certificates, mounts and so on), you then run autoconfiguration on each additional SP so that the settings are identical. In addition, if you modify the configuration of one SP, you can update all of them by re-running autoconfiguration on each one. (For this reason, set the IP address of the autoconfigure server to x.x.x.1.)

Note – Autoconfiguration does not merge configurations, it overwrites the existing configuration.

Note – Autoconfiguration does not work across different server platforms. That is, you cannot configure a Sun Fire V40z service processor using settings on a Sun Fire V20z service processor.

You can start autoconfiguration either when you are prompted at the completion of setting the IP address of the SP, or by selecting Autoconfigure from the SP menu option on the operator panel at any time.

To perform autoconfiguration of an SP, follow these steps:

1. **On the operator panel, press the Forward or Back buttons until the following prompt shows** `Yes`**.**

   ```
   SP Auto Setup?
   No
   ```

   For instructions on setting an IP address, refer to the *Sun Fire V20z and Sun Fire V40z Server Installation Guide*.

2. **Press the Select button.**

   The SP attempts to locate an IP address.

   - If the SP successfully locates an IP address, the following prompt appears, displaying an IP address for this SP:

     ```
     Setup Server IP:
     x.x.x.1
     ```

     Where *x.x.x* is the first three octets of the SP IP address. For example, if the address is 10.10.30.19, the address that displays in the prompt appears as 10.10.30.1.

     In this case, press the Select operator panel button to start autoconfiguration.

- If the SP does not locate an IP address, the following message appears:

  ```
  Unable to get
  SP IP address
  ```

  In this case, you must manually enter an IP address before you press the Select operator panel button to start autoconfiguration.

3. **Wait until the autoconfiguration is complete, at which point the SP automatically reboots.**

   The following message displays when autoconfiguration is running.

   ```
   SP AutoConfigure
   in progress
   ```

Note – If the autoconfiguration is unsuccessful, a failure message displays. Press any button to clear it.

Determining SP and Platform Network MAC Addresses

Use the following commands if you need to determine the MAC address of your server's SP or platform:

```
# ssh spipaddress -l spusername sp get mac
```

```
# ssh spipaddress -l spusername platform get mac
```

CHAPTER **2**

IPMI Server Management

Server manufacturers today have to re-invent how each new server manages itself. The hardware and software design for one server does not necessarily work with another. Every server supplier provides basic monitoring and data collection functions but no two do it exactly the same. These proprietary implementations for manageability only complicate the problem.

The standardization of server-based management, called Intelligent Platform Management Interface (IPMI), provides a solution. IPMI allows you to interconnect the CPU and devices being managed. It allows for:

- Easy replication of the monitoring functions from server to server
- Support for a reasonably large number of monitoring devices
- Common driver-level access to management instrumentation
- More cost-effective implementations
- Increased scalability of the server management functions

IPMI is an industry-standard, hardware-manageability interface specification that provides an architecture defining how unique devices can all communicate with the CPU in a standard way. It facilitates platform-side server management and remote server-management frameworks, by providing a standard set of interfaces for monitoring and managing servers.

With IPMI, the software becomes less dependent on hardware because the management intelligence resides in the IPMI firmware layer, thereby creating a more intelligently managed server. The IPMI solution increases server scalability by distributing the management intelligence closer to the devices that are being managed.

Baseboard Management Controller

In order to perform autonomous platform-management functions, the processor runs embedded software or firmware. Together, the processor and its controlling firmware are referred to as the Baseboard Management Controller (BMC), which is the core of the IPMI structure. Tightly integrating an IPMI BMC and management software with platform firmware facilitates a total management solution.

The BMC is a service processor integrated into the motherboard design, providing a management solution independent of the main processor. The monitored server can communicate with the BMC through one of three defined interfaces, which are based on a set of registers shared between the platform and the BMC.

Note – In these servers, the SP has software that emulates a BMC.

The BMC is responsible for:

- Managing the interface between server management software and platform management hardware
- Interfacing to the system sensors, such as fan speed and voltage monitors
- Providing access to the system event log
- Providing autonomous monitoring, event logging and recovery control
- Acting as a gateway between the management software and the IPMB/ICMB
- Monitoring the system watchdog timer
- Facilitating the remote-management tasks, even when the main server hardware is in an inoperable state

The BMC provides the intelligence behind IPMI. In these servers, the SP serves as the BMC, providing access to sensor data and events through the standard IPMI interfaces.

Manageability

IPMI defines a mechanism for server monitoring and recovery implemented directly in hardware and firmware. IPMI functions are available independent of the main processors, BIOS and operating system.

IPMI monitoring, logging and access functions add a built-in level of manageability to the platform hardware. IPMI can be used in conjunction with server-management software running under the OS, which provides an enhanced level of manageability.

IPMI provides the foundation for smarter management of servers by providing a methodology for maintaining and improving the reliability, availability and serviceability of expensive server hardware.

IPMI Compliance and LAN Channel Access

The server supports IPMI through the SP software version 2.0 and later. These servers meet compliance standards for IPMI version 1.5.

The IPMI implementation on these servers also support LAN channel access. (Refer to the IPMI specification version 1.5 for details.) The LAN channel access is disabled by default. To enable it, use the `ipmi enable channel` command and specify the ID of the channel to enable for the LAN Interface, as follows.

Note – This ID is case-sensitive and must be lowercase.

```
# ssh spipaddr -l spuser ipmi enable channel {sms | lan}
```

For more information about enabling or disabling the IPMI channel, refer to Appendix E.

Usernames and Passwords

Operator and administrator-level access over the LAN channel requires a valid user ID and password. These servers come preconfigured with an administrator-level user with a null user ID. However, you can re-add the anonymous user at a later time if you wish. You can configure both the user ID and password to be null.

Note – For security reasons, the LAN channel access is disabled by default.

Note – IPMI user identities are in no way associated with user accounts defined for server-management capabilities. Refer to “Initial Setup of the Service Processor” on page 10 for more information about these server-management user accounts.

Lights Out Management (LOM)

On these servers, Lights Out Management is performed through IPMItool, a utility for controlling IPMI-enabled devices.

Description

IPMItool is a simple command-line interface (CLI) to servers that support the Intelligent Platform Management Interface (IPMI) v1.5 specification. It provides the ability to:

- Read the Sensor Data Record (SDR) and print sensor values
- Display the contents of the System Event Log (SEL)
- Print information about Field Replaceable Units (FRUs)
- Read and set LAN configuration parameters
- Perform chassis power control

Originally written to take advantage of IPMI-over-LAN interfaces, IPMItool is also capable of using a system interface, as provided by a kernel device driver such as OpenIPMI.

Further Information

- For up-to-date information about IPMItool, visit:

  `http://ipmitool.sourceforge.net/`

- For more information about the IPMI specification, visit:

  `http://www.intel.com/design/servers/ipmi/spec.htm`

- For more information about the OpenIPMI project (MontaVista IPMI kernel driver), visit:

  `http://openipmi.sourceforge.net/`

Syntax

The syntax used by IPMItool is as follows:

ipmitool [**-ghcvV**] **-I lan -H** *address* [**-P** *password*] *expression*

ipmitool [**-ghcvV**] **-I open** *expression*

Options

TABLE 2-1 lists the options available for IPMItool.

TABLE 2-1 Options for IPMItool

Option	Description
-h	Provides help on basic usage from the command line.
-c	Makes the output suitable for parsing, where possible, by separating fields with commas instead of spaces.
-g	Attempts to make IPMI-over-LAN communications more robust.
-V	Displays the version information.
-v	Increases the amount of text output. This option may be specified more than once to increase the level of debug output. If given three times, you receive hexdumps of all incoming and outgoing packets.
-I *interface*	Selects the IPMI interface to use. The possible interfaces are LAN or open interface.
-H *address*	Displays the address of the remote server; it can be an IP address or host name. This option is required for the LAN interface connection.
-P *password*	Displays the password for the remote server; the password is limited to a maximum of 16 characters. The password is optional for the LAN interface; if a password is not provided, the session is not authenticated.

Expressions

TABLE 2-2 lists the expressions and parameters available for IPMItool.

Note – For each of these expressions, the beginning command is always **`ipmitool`**, followed by the expression and parameter(s).

Note – The sol command is not supported in these servers, but you can enable a Serial-over-LAN feature. See "Serial Over LAN" on page 70.

TABLE 2-2 Expressions and Parameters for IPMItool *(1 of 4)*

Expression	Parameter	Sub-parameter	Description and examples

help			Can be used to get command-line help on IPMItool commands. It may also be placed at the end of commands to get help on the use of options.
			EXAMPLES:
			`ipmitool -I open help`
			`Commands:  chassis, fru, lan, sdr, sel`
			`ipmitool -I open chassis help`
			`Chassis Commands:  status, power, identify, policy, restart_cause`
			`ipmitool -I open chassis power help`
			`Chassis Power Commands: status, on, off, cycle, reset, diag, soft`

raw	*netfn*	*cmd data*	Allows you to execute raw IPMI commands (for example, to query the POH counter with a raw command).
			EXAMPLE:
			`ipmitool -I open raw 0x0 0x1`
			`RAW REQ (netfn=0x0 cmd=0x1 data_len=0)`
			`RAW RSP (3 bytes)`
			`60 00 00`

TABLE 2-2 Expressions and Parameters for IPMItool *(2 of 4)*

Expression	Parameter	Sub-parameter	Description and examples

chaninfo	*channel*		Displays information about the selected channel. If no channel is specified, the command displays information about the channel currently being used.
			EXAMPLES:
			`ipmitool -I open chaninfo`
			`Channel 0xf info:`
			`  Channel Medium Type: System Interface`
			`  Channel Protocol Type: KCS`
			`  Session Support: session-less`
			`  Active Session Count: 0`
			`  Protocol Vendor ID: 7154`
			`ipmitool -I open chaninfo 7`
			`Channel 0x7 info:`
			`  Channel Medium Type: 802.3 LAN`
			`  Channel Protocol Type: IPMB-1.0`
			`  Session Support: multi-session`
			`  Active Session Count: 1`
			`  Protocol Vendor ID: 7154`
			`  Alerting: enabled`
			`  Per-message Auth: enabled`
			`  User Level Auth: enabled`
			`  Access Mode: always available`

userinfo	*channel*
	Note: Channels 6 and 7 are not supported on Sun Fire V20z servers.
			Displays information about configured user information on a specific LAN channel.
			EXAMPLE:
			`ipmitool -I open userinfo 6`
			`Maximum User IDs      : 4`
			`Enabled User IDs      : 1`
			`Fixed Name User IDs   : 1`
			`Access Available      : call-in / callback`
			`Link Authentication   : disabled`
			`IPMI Messaging        : enabled`

chassis	status		Returns information about the high-level status of the server chassis and main power subsystem.
	identify	*interval*	Controls the front panel identification light. The default value is 15 seconds. Enter “0” to turn it off.
	restart_cause		Queries the chassis for the cause of the last server restart.

TABLE 2-2 Expressions and Parameters for IPMItool *(3 of 4)*

Expression	Parameter	Sub-parameter	Description and examples
power			Performs a chassis control command to view and change the power state.
	status		Shows the current status of the chassis power.
	on		Powers on the chassis.
	off		Powers off chassis into the *soft off* state (S4/S5 state). NOTE: This command does not initiate a clean shutdown of the operating system prior to powering off the server.
	cycle		Provides a power-off interval of at least 1 second. No action should occur if chassis power is in S4/S5 state, but it is recommended to check the power state first and then only issue a power-cycle command if the server power is on or in a lower sleep state than S4/S5.
	reset		Performs a hard reset.
lan	print	*channel*	Prints the current configuration for the given channel.
	set	*channel parameter*	Sets the given parameter on the given channel.
		ipaddr *x.x.x.x*	Sets the IP address for this channel.
		netmask *x.x.x.x*	Sets the netmask for this channel.
		macaddr *xx:xx:xx:xx:xx:xx*	Sets the MAC adddress for this channel.
		defgw ipaddr *x.x.x.x*	Sets the default gateway IP address.
		defgw macaddr *xx:xx:xx:xx:xx:xx*	Sets the default gateway MAC address.
		bakgw ipaddr *x.x.x.x*	Sets the backup gateway IP address.
		bakgw macaddr *xx:xx:xx:xx:xx:xx*	Sets the backup gateway MAC address.
		password *pass*	Sets the null user password.
		user	Enables the user-access mode.
		access [on|off]	Sets the LAN-channel-access mode.

		ipsrc *source*	Sets the IP address source. As a source, you can indicate:
			none = unspecified
			static = manually configured static IP address
			dhcp = address obtained by BMC running DHCP
			bios = address loaded by BIOS or system software

TABLE 2-2 Expressions and Parameters for IPMItool *(4 of 4)*

Expression	Parameter	Sub-parameter	Description and examples
		arp respond [on | off]	Sets the BMC-generated ARP responses.
		arp generate [on | off]	Sets the BMC-generated gratuitous ARPs.
		arp interval [seconds] s	Sets the interval for the BMC-generated gratuitous ARPs.
		auth *level*,... *type*,...	This command sets the valid authtypes for a given auth level. Levels can be: `callback`, `user`, `operator`, `admin` Types can be: `none`, `md2, md5`
fru	print		Reads all inventory data for the Customer Replaceable Units (CRUs) and extracts such information as serial number, part number, asset tags and short strings describing the chassis, board or product.
sdr	list		Reads the Sensor Data Record (SDR) and extracts sensor information, then queries each sensor and prints its name, reading and status.
sel	info		Queries the BMC for information about the system event log (SEL) and its contents.
	clear		Clears the contents of the SEL. The `clear` command cannot be undone.
	list		Lists the contents of the SEL.

IPMI Linux Kernel Device Driver

The IPMItool application utilizes a modified MontaVista OpenIPMI kernel device driver found on the Sun Fire V20z and Sun Fire V40z Servers Documentation and Support Files CD. The driver has been modified to use an alternate base hardware address and modified device IO registration.

This driver must be compiled and installed from the Documentation and Support Files CD.

The following kernel modules must be loaded in order for IPMItool to work:

1. `ipmi_msghandler`

   The message handler for incoming and outgoing messages for the IPMI interfaces.

2. `ipmi_kcs_drv`

   An IPMI Keyboard Controller Style (KCS) interface driver for the message handler.

3. `ipmi_devintf`

   Linux-character-device interface for the message handler.

To force IPMItool to use the device interface, you can specify it on the command line:

```
# ipmitool -I open [option...]
```

Installing and Compiling the Driver

To install and compile this kernel device driver, see “Initial Setup of the Service Processor” on page 10.

LAN Interface for the BMC

Note – In these servers, the SP has software that emulates a BMC.

The IPMItool LAN interface communicates with the BMC over an Ethernet LAN connection using User Datagram Protocol (UDP) under IPv4. UDP datagrams are formatted to contain IPMI request/response messages with IPMI session headers and Remote Management Control Protocol (RMCP) headers.

Remote Management Control Protocol is a request-response protocol delivered using UDP datagrams to port 623. IPMI-over-LAN uses version 1 of the RMCP to support management both before installing the OS on the server, or if the server will not have an OS installed.

The LAN interface is an authenticated, multi-session connection; messages delivered to the BMC can (and should) be authenticated with a challenge/response protocol with either a straight password/key or an MD5 message-digest algorithm. IPMItool attempts to connect with administrator privilege level as this is required to perform chassis power functions.

With the `-I` option, you can direct IPMItool to use the LAN interface:

```
# ipmitool -I lan [option...] address password
```

To use the LAN interface with IPMItool, you must provide a host name on the command line.

The password field is optional; if you do not provide a password on the command line, IPMItool attempts to connect without authentication. If you specify a password, it uses MD5 authentication, if supported by the BMC; otherwise, it will use straight password/key.

Files

The file `/dev/ipmi0` is a character-device file used by the OpenIPMI kernel driver.

Examples

If you want to remotely control the power of an IPMI-over-LAN-enabled server, you can use the following commands:

```
# ipmitool -I lan -H spipaddr -P sppasswd chassis power on
```

The result returned is:

```
Chassis Power Control: Up/On
```

```
# ipmitool -I lan -H spipaddr -P sppasswd chassis power status
```

The result returned is:

```
Chassis Power is on
```

Viewing the IPMI System Event Log

To view the System Event Log (SEL), use IPMItool.

The out-of-band command is:

```
# ipmitool -I lan -H spipaddr -P ipmipasswd sel list
```

The in-band command (using OpenIPMI on a Linux-based server or LIPMI on a Solaris-based server) is:

```
# ipmitool -I open sel list
```

Note – To receive more verbose logging messages, you can run the following command:

```
# ssh -l spuser spipaddr sp get events
```

Clearing the IPMI System Event Log

You can use commands to clear the contents of the IPMI SEL.

Use one of the following commands, depending on your OS:

- For Linux: **`ipmitool -I open sel clear`**
- For Solaris: **`ipmitool -I lipmi sel clear`**

IPMI Troubleshooting

TABLE 2-3 describes some potential issues with IPMI and provides solutions.

TABLE 2-3 IPMI Troubleshooting

Issue	Solution
You cannot connect to the management controller using IPMItool over LAN.	Verify the network connection to the management controller and its IP address and verify the channel is enabled using the `ipmi get channels` command.
You cannot authenticate to the management controller using IPMItool over LAN.	Ensure that you are using the password assigned when you enabled IPMI LAN access from the management-controller shell prompt.
You have forgotten the password for IPMI access over LAN.	1. You can reset the IPMI setting, reset the SDRR and purge the SEL from the management-controller shell by running the command: # **`ssh`** *spipaddr* **`-l`** *spuser* **`ipmi reset -a`** 2. Now re-enable IPMI on LAN with the following commands: # **`ssh`** *spipaddr* **`-l`** *spuser* # **`ipmi enable channel lan`** # **`exit`**
IPMItool fails when using the “open” interface.	Ensure that the Linux kernel module `ipmi_kcs_drv` is loaded by running the `lsmod` command.

CHAPTER 3

SNMP Server Management

You can manage your server using the Simple Network Management Protocol (SNMP).

Simple Network Management Protocol

Simple Network Management Protocol (SNMP) is a network-management protocol used almost exclusively in TCP/IP networks. SNMP provides a means to monitor and control network devices, and to manage configurations, statistics collection, performance and security on a network.

SNMP-based management allows for third-party solutions to be used. This includes products such as HP OpenView and CA Unicenter.

The base component of an SNMP solution is the Management Information Base (MIB). The MIB is included on the Sun Fire V20z and Sun Fire V40z Servers Network Share Volume CD.

This configuration is beneficial when, for example, you have a cluster of machines serving Web content and the platform is connected to the Internet, but the SP is protected and only accessible on an internal network.

SNMP Integration

SNMP is an open network-management technology that enables the management of networks and entities connected to the network. Within the SNMP architecture is a collection of network-management stations and managed nodes.

Network-management stations execute management applications, which monitor and control managed nodes. Managed nodes are devices such as hosts, gateways and so on, which have management agents responsible for performing the management functions requested by the management stations.

SNMP is used to communicate management information between the management stations and the agents. In other words, SNMP is the protocol by which the agent and the management station communicate.

The monitoring of state through SNMP at any significant level of detail is accomplished primarily by polling for appropriate information on the part of the management station. Managed nodes may also provide unsolicited status information to management stations in the form of traps, which is likely to guide the polling at the management station.

Communication of information between management entities in a network is accomplished through the exchange of SNMP-protocol messages, both in the form of queries (get/set) by the management station and in the form of unsolicited messages (traps) indicated by the agent.

Your server includes SNMP agents that allow for health and status monitoring. The SNMP agent runs on the SP and therefore all SNMP-based management of the server should occur through the SP. The SNMP agent on these servers provides the following capabilities:

- Event management
- Inventory management
- Sensor and system state monitoring
- SP configuration monitoring

SNMP Management Information Base (MIB)

The Management Information Base (MIB) is a text file that describes SNMP data as managed objects. These servers provide SNMP MIBs so that you can manage and monitor your server using any SNMP-capable network management system, such as HP OpenView Network Node Manager (NNM), Tivoli, CA Unicenter, IBM Director and so on. The MIB data describes the information being managed, reflects current and recent server status, and provides server statistics.

Sun Fire V20z and Sun Fire V40z Servers MIB Tree

FIGURE 3-1 illustrates the MIB tree:

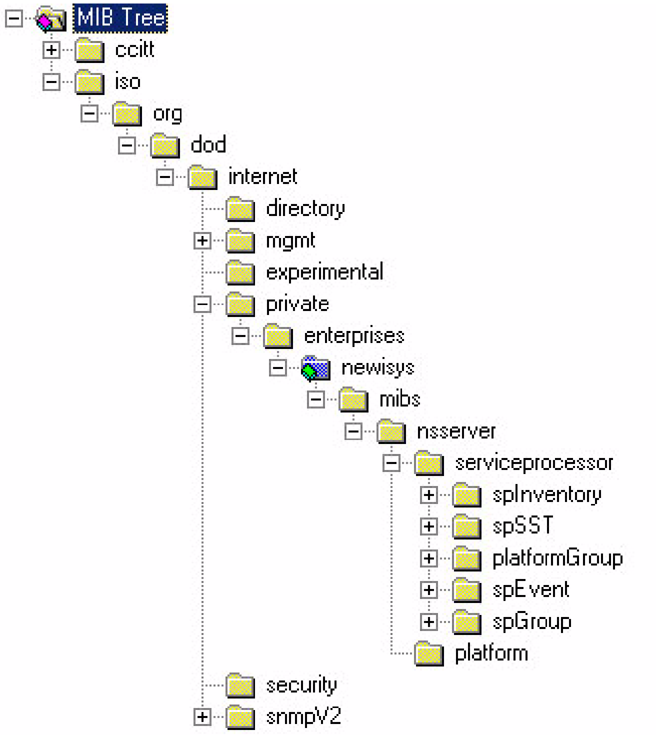

FIGURE 3-1 MIB Tree

Integrating MIBs with Third-Party Consoles

You use the server’s MIBs to integrate the management and monitoring of the server into SNMP management consoles. The MIB branch is a private enterprise MIB, located at object identifier (OID) 1.3.6.1.2.1.9237. The standard SNMP port 161 is used by the SNMP agent on the SP.

Configuring SNMP on Your Server

Note – There are several services that are supplied by the SNMP agent on the server. Depending on your business needs and the configuration of your current office network and management environment, you might want to take advantage of these services.

There are certain prerequisites and setup required on both the SP and the platform in order to enable and utilize each of these services:

- SNMP agent on the SP
- Proxy forwarder application/ProxyAgent [RFC 2271]
- Agent X [RFC 2741]

The following diagram illustrates the SNMP architecture and communication paths between the SP and the platform.

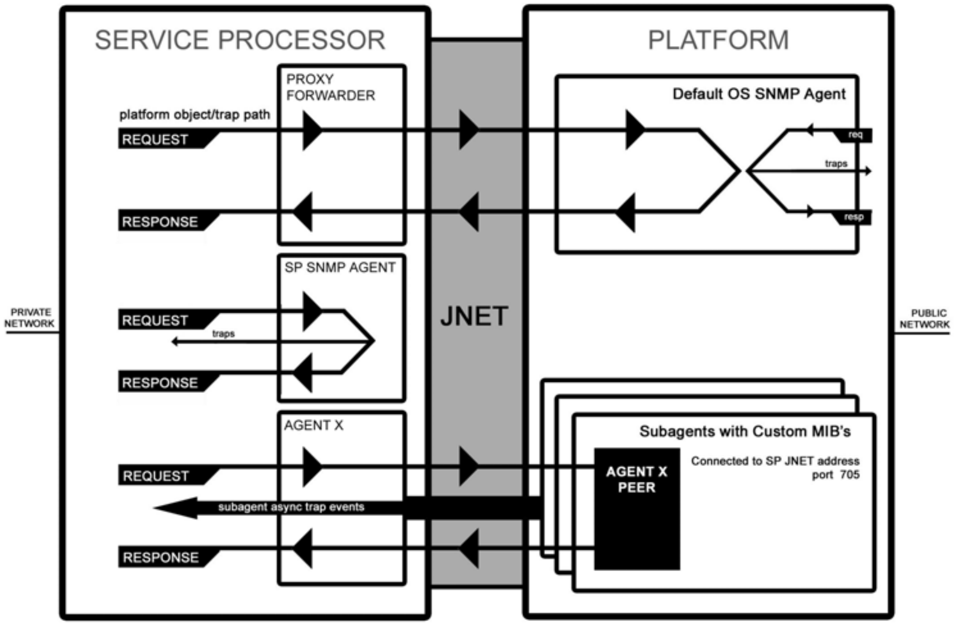

FIGURE 3-2 SNMP Architecture and Communications

SNMP Agent on the Service Processor

The SNMP agent running on the SP facilitates the management and monitoring of the server. The SNMP agent can be used to query various types of SP information. Refer to FIGURE 3-1 for a list of the MIBs; refer to TABLE 3-3 for a detailed description of the MIBs.

There is no configuration required to use this functionality other than integrating the server MIBs with your desired management station.

Refer to the procedure for using the SNMP agent on the SP, as explained in "Integrating MIBs with Third-Party Consoles" on page 43.

Note – The SNMP agent on these servers supports SNMP v1/v2c. For security reasons, there are no settable attributes in this agent.

Proxy Agent

The SP acts as an SNMP proxy-agent intermediary for the platform. Queries made from a management station to the SNMP agent on the SP are intercepted by the proxy agent on the SP and forwarded to the platform; the SP proxy agent contacts the platform to retrieve the requested information. The proxy agent then receives the data from the platform and sends the request back to the management station. The management station never knows that the request was proxied. The SP and the platform communicate over an internal private network.

To enable this facility, you must first run an SNMP agent on your platform operating system (see your operating system vendor to obtain this agent). This enables platform-level management transparently through the SP. Querying MIBs other than the server MIB (for example, the Host Resource MIB) and the MIBII System MIB on the SP obtains information from the platform by proxying the request to the platform SNMP agent.

Ensure that the SP can identify the read-only and read-write community names that are configured for your platform SNMP agent. Refer to "Setting the Community Name" on page 46.

Setting the Community Name

The SNMP agent on the SP acts as a proxy for the SNMP agent running on the platform. (Refer to “Configuring SNMP on Your Server” on page 44.) To properly proxy, you must use the community string. The community string needed to do so is the value defined when you configured the platform for SNMP.

If you find that your SNMP queries are not being proxied to the platform SNMP agents, validate that the community string on the SP matches that on the platform. The SP proxy community string can be changed to match the platform community string using the following command:

```
# sp set snmp proxy community
```

There are no restrictions on the length of the community strings; common names are *private* and *public*. The default name is *public*.

For more information, refer to “SP Set SNMP Proxy Community Subcommand” on page 220.

Agent X

A sub-agent using SNMP Agent X protocol on the platform can connect to the SNMP agent on the SP (through a special port) and forward query responses or unsolicited traps through the SP. This allows server-management traffic to be kept secure from the production network connected to the platform, if required.

To properly enable this facility, you must identify the IP address and port number pair associated with the SP (as seen from the platform). The Agent X port is fixed at 705 (TCP). However, the private-network IP address is configurable and, by default, this address is 169.254.101.2.

Refer to your application documentation for instructions on configuring the sub-agents.

Note – You can use the subcommand, `sp get jnet` on the SP to retrieve the JNET IP address of the SP. For more information, see Appendix H.

Using a Third-Party MIB Browser

The following example demonstrates integrating the server MIBs into an SNMP node manager.

1. **From the Manager Preferences menu, choose Load/Unload MIBS: SNMP.**

2. **Locate and select the SP-MasterAgent-MIB.mib.**

3. **Click Load.**

4. **Specify the directory in which the server MIBs are placed and click Open.**

5. **Repeat Step 2 through Step 4 to load other MIBS (for example, SP-SST-MIB.mib, SP-INVENTORY-MIB.mib, SP-EVENT-MIB.mib, SP-PLATFORM-MIB.mib, SP-GROUP-MIB.mib and so on).**

6. **Exit the Manager Preferences menu.**

7. **Open an SNMP MIB browswer.**

   The SNMP standard tree displays in the MIB Browser.

8. **Locate the Newisys branch located under private.enterprises.**

   Refer to FIGURE 3-1 for a sample view of the MIB tree.

Setting Logging Options

You can also easily integrate SP-generated traps and set logging options. The following example demonstrates the necessary steps when using HP OpenView NNM:

1. **Load the SP-EVENT-MIB.mib according to the previous procedure.**

2. **Choose Options>EventConfiguration**

3. **Select the spEvent module from the Enterprises list.**

4. **Double-click an event from the Events for Enterprise spEvent list.**

5. **Select the Event Message tab.**

6. **Select the Log and display in category radio dialog and choose a category from the corresponding dropdown list, or create your own event category.**

7. **Select the severity of the event from the Severity dropdown list.**

8. **Enter a message or $* to display all information in the Event Log Message field.**

9. **Click OK.**

SNMP Traps

SNMP traps are network-management notifications of an event occurring at a managed network node. These events can identify problems in the network, machines up or down, and so on. These servers use traps to signal conditions related to the server’s health, including critical conditions related to physical components, the return to a normal state for these components, and other situations related to the state of the software running on the SP (for example, network settings being reconfigured).

Traps are defined in the MIB files and are generated, received and processed by an SNMP management station. SNMP trap data is uniquely identified by the MIB. Each SNMP trap contains information identifying the server's name, IP address and other relevant data about the event.

Within the server event MIB, each trap has the following variables and event bindings; see TABLE 3-1.

TABLE 3-1 Server Event Traps

Event	Description
EventID	Uniquely identifies the event on the SP from where it came.
EventSource	Denotes the source module that generated the event.
EventComponent	Denotes the component ID about which the event refers.
EventDescription	The event message received from its source.
EventTimeStampInitial	The time at which this event ID was initially generated.
EventTimeStampLast	The most recent time at which this event ID was generated.

Configuring SNMP Trap Destinations

Although SNMP traps are generated for events that occur on the SP, you must configure where these traps are to be sent. There is no default destination for traps. You can use the server-management subcommands (see TABLE 3-2) on the SP to configure SNMP destinations.

For more information on these subcommands, refer to Appendix H.

TABLE 3-2 Subcommands for Configuring SNMP Destinations

Subcommand	Description
`sp get snmp-destinations`	Displays all the available SNMP destination IP addresses and host names to which the SP will send.
`sp add snmp-destination`	Adds a new SNMP destination one IP address or host name at a time.
`sp delete snmp-destination`	Removes an existing SNMP destination one IP address or host name at a time.

Configuring SNMP Destinations

Administration- and manager-level users can define SNMP destinations to which SNMP events (alerts) will be sent using this option. All users can view the current destinations (using read-only access).

The number of destinations you can create is limited due to memory constraints.

Server MIB Details

SNMP uses object identifiers (OIDs) to provide name variables by which objects are grouped together for easier reference. These servers provide agents for the MIBs shown in TABLE 3-3:

TABLE 3-3 SNMP MIBs

MIB	OID	Description
SP-MasterAgent-MIB .mib	.1.3.6.1.4.1.9237	Creates the main trunk of the server MIB tree. All other MIBs of the SP branch from this tree. To be loaded first while integrating with any third-party framework.
SP-INVENTORY-MIB .mib	.1.3.6.1.4.1.9237.2.1.1.1 .1.3.6.1.4.1.9237.2.1.1.1.2 .1.3.6.1.4.1.9237.2.1.1.1.3	Used for querying inventory information for all Sun Fire V20z and Sun Fire V40z servers hardware and software components. **Hardware Inventory Table**: Collects all hardware component inventory. **Software Inventory Table**: Collects all software component inventory.
SP-SST-MIB.mib	.1.3.6.1.4.1.9237.2.1.1.4	Defines objects for the System State Table in the SP. Contains all sensor readings, including the name of the sensor, its current value, maximum allowed value, measurement type, scale and scanning interval.
SP-PLATFORM-MIB.mib	.1.3.6.1.4.1.9237.2.1.1.5	Defines objects for the platform SNMP which includes osstate, platform state and platform IP table.
SP-EVENT-MIB.mib	.1.3.6.1.4.1.9237.2.1.1.6	Identifies the OIDs associated with all SNMP traps originated from the SP.
SP-GROUP-MIB.mib	.1.3.6.1.4.1.9237.2.1.1.7	Defines objects for the SP, including host name, DNS, a reboot node, a node to hold the last port 80 postcode, a clone tree and an IP table.

The events listed in TABLE 3-4 are sent to the SNMP destination by `SP-EVENT-MIB.mib`.

TABLE 3-4 SP Events *(1 of 2)*

Enterprise Trap ID	Event
1	spGenericEventInformational
2	spGenericEventWarning
3	spGenericEventCritical
4	spTemperatureEventInformational
5	spTemperatureEventWarning
6	spTemperatureEventCritical
7	spVoltageEventInformational
8	spVoltageEventWarning
9	spVoltageEventCritical
10	spFanEventInformational
11	spFanEventWarning
12	spFanEventCritical
13	spPlatformMachineCheckEventInformational
14	spPlatformMachineCheckEventWarning
15	spPlatformMachineCheckEventCritical
16	spPlatformStateChangeEventInformational
17	spPlatformStateChangeEventWarning
18	spPlatformStateChangeEventCritical
19	spPlatformBIOSEventInformational
20	spPlatformBIOSEventWarning
21	spPlatformBIOSEventCritical
22	spGenericEventInformational
23	spGenericEventWarning
24	spGenericEventCritical
25	spTemperatureEventInformational
26	spTemperatureEventWarning
27	spTemperatureEventCritical
28	spVoltageEventInformational

TABLE 3-4 SP Events *(2 of 2)*

Enterprise Trap ID	Event
29	spVoltageEventWarning
30	spVoltageEventCritical
31	spFanEventInformational
32	spFanEventWarning
33	spFanEventCritical
37	spPlatformStateChangeEventInformational
38	spPlatformStateChangeEventWarning
39	spPlatformStateChangeEventCritical
40	spPlatformBIOSEventInformational
41	spPlatformBIOSEventWarning
42	spPlatformBIOSEventCritical

SNMP Troubleshooting

TABLE 3-5 describes a potential issue with SNMP and provides a solution.

TABLE 3-5 SNMP Troubleshooting

Issue	Solution
SNMP queries to the SP time out.	The platform OS requires both the NPS driver suite RPM and an active SNMP daemon sharing the SP's community string.

CHAPTER 4

Further Management Information

Configuring Scripting Capabilities

A system administrator can log in to the Service Processor (SP) using secure shell (SSH) and issue commands, or more commonly, write a shell script that remotely invokes these operations.

Note – You must create a valid initial manager account before using SSH. The SP includes a setup account that can be used to set up an initial manager account. This initial manager user can create additional users.

The SP includes a suite of commands that enables management and monitoring of the server; this suite of commands is referred to as server management commands. From the command line, for instance, you can write data driven scripts that automate configuration of multiple machines.

The Sun Fire V20z and Sun Fire V40z Servers Network Share Volume CD contains sample scripts for getting started, which you can access after you extract the files on the CD. See “Network Share Volume (NSV) CD-ROM” on page 69 for more information about the script locations.

Using Shell Scripts

An administrator can make configuration changes for a single SP by using SSH to log in and run commands. For a multi-system environment in which configurations for all SPs must be synchronized, you can automate configuration changes.

As a Unix/Linux administrator, you can use SSH, trusted-host relationships or public-key authentication, and Unix/Linux shell scripting to automate tasks that need to be performed on multiple SPs.

1. Set up your system for scripting.

   Sun Fire V20z and Sun Fire V40z remote scripting solutions depend on SSH for authentication and data encryption. If you do not already have SSH, you can obtain a free implementation, OpenSSH, available at **www.openssh.org**. The SP allows the use of SSH v2 only. Refer to "Remote Scripting Using SSH" on page 56.

2. Create a trusted-host relationship or add your public key for SSH authentication.

   In order to use SSH in a scripted environment such that you are not prompted for a password upon the execution of each command, you can establish a trusted-host relationship between the machine from which the commands are sent and the SP on which the commands are executed. (This requires the prior creation of a manager-level user on the SP.) Refer to "Creating Trusted-Host Relationships" on page 58.

   You can also add a public key for SSH authentication, allowing you to log in via SSH and execute remote commands without being prompted for a password. Refer to "Adding Public Keys" on page 58.

3. Configure your client for scripting.

   You must configure the client machine on which you will be running scripts.

4. Create your scripts.

Remote Scripting Using SSH

Remote scripting to the SP is done by using a program called SSH. For example, as a user on the UNIX machine **client.company.com** with the SP name **sp.company.com**, you could execute a command on the SP from the UNIX client using the following format:

```
# ssh sp.company.com command
```

Because the SSH server must authenticate the remote user, the user must either enter a password, or a trusted-host relationship must exist, or the remote user's public key must be installed on the SP.

If using trusted-host relationships for passwordless access, the SP must have a local user of the same name as the remote user (or the remote user should be a member of a directory service group that is mapped to a local SP administrative group).

You can also add your public key file instead of creating a trusted-host relationship to be authenticated via SSH. Refer to “Adding Public Keys” on page 58.

When configured for passwordless access, the ssh daemon on the SP allows the remote user access to `sp.company.com` without a password, either for logging in, or for issuing remote ssh commands from the command line or from a script.

Configuring Multiple Systems for Scripting

There are two ways to configure multiple SPs for scripting:

- Execute the procedure to configure the client machine on which you will be running scripts for each SP.
- Set up the trust relationship or add your public key file on an initial machine and use the autoconfiguration feature to duplicate the configuration on each of the additional machines. Refer to “Creating Trusted-Host Relationships” on page 58 and “Adding Public Keys” on page 58.

Generating Host Keys

To establish a trusted-host relationship, you must set up a host key which is used to authenticate one host to another. The host’s SSH install should generate the host keys. If it does not, follow these steps to generate a host-key pair:

1. **Enter the following command:**

```
# ssh-keygen -q -t rsa -f rsa_key -C '' -N ''
```

2. **Move** `rsa_key` **to** `/etc/ssh/ssh_host_rsa_key`**.**

3. **Move** `rsa_key.pub` **to** `/etc/ssh/ssh_host_rsa_key.pub`**.**

4. **Ensure that only the root user has read or write permissions to** `/etc/ssh/ssh_host_rsa_key`**.**

   The `ssh_host_rsa_key.pub` file is the file you will transfer to the SP.

Note – Only protocol version 2 key types and 1024-bit key sizes (the default generated by ssh-keygen) are supported.

---

5. **Continue with "Creating Trusted-Host Relationships" on page 58 for instructions on creating public keys that can be used for passwordless access.**

Note – Use `scp` to copy the files to either `/tmp` or to your home directory. The sp commands will then install the file specified on the command line.

Creating Trusted-Host Relationships

Adding a trusted-host relationship is one way to allow for passwordless access and thus is a means for one-to-many scripting. Once a host-equivalence relationship has been created with a client, users on that client can remotely execute commands on the Service Processor without being prompted for a password, provided one of the following conditions is met:

- The user's login name on the client is the same as that of a local user on the SP.
- The user's login on the client belongs to a directory service group that is mapped to an SP administrative group. (In this case, the SSH command executes as a well-known auxiliary user on the SP: either rmonitor, radmin or rmanager.)

Note – Support is available for SSH protocol version 2 key types (RSA or DSA) only. If DNS is enabled on the SP, the client machine must be specified with its DNS name, not an IP address.

Manager-level users can create a trusted-host relationship for the specified host from the command line using the `access add trust` command:

```
# access add trust {-c | --client} HOST {-k | --keyfile} \PUBLIC
KEY FILE
```

Adding Public Keys

Adding a user's public key is another way to allow for passwordless access and thus provide one-to-many scripting. Once a public key for a specific user has been installed on the SP, that user can remotely execute commands on the SP without being prompted for a password, if that user has installed the associated private key on the client.

Note – Support is available for SSH protocol version 2 key types (RSA or DSA) only.

Only local users can add public keys. Users who obtain authorization from directory services group mappings are not able to add public keys.

Local admin-level or manager-level users can add public keys using the `access add public key` command:

```
# access add public key -l PUBLIC_KEY_FILE [-u user]
```

The public-key file is your RSA or DSA key. Up to 10 users can install public keys; only one key per user is allowed.

Admin-level users can only add their own public key. Manager-level users can add a public key for any local user. If the *user* is not specified in the command, the current user is the default.

Note – The maximum supported key length is 4096 bits.

Generating a Host-Key Pair

To establish a trusted-host relationship, you must set up a host key, which is used to authenticate one host to another. Follow these steps to generate a host-key pair by copying the public key to the SP to which you want passwordless access:

1. **Execute the following command:**

   ```
   # ssh-keygen -t rsa -N
   ```

2. **Accept the default values, installing to the following directory:**

   `$HOME/.ssh/id_rsa`

   The following files are created:

   ```
   $HOME/.ssh/id_rsa
   $HOME/.ssh/id_rsa.pub
   ```

Enabling SSH Access Using Trusted-Hosts

Follow these steps to add users to the `local /etc/password` file to attempt trusted-host access to the Service Processor:

1. **Set up your host keys by executing the following command:**

   ```
   # ssh-host-config
   ```

2. **Enable access for clients by launching a Bash shell.**

   - If you want all network accounts added, execute **`mkpasswd >> /etc/passwd`**.
   - If you want just local accounts added, execute **`mkpasswd -l >> /etc/passwd`**.

3. **Issue the following commands as a manager-level user on the client to establish a trusted-host relationship (***manager1* is used in the example in this step**):**

   a. **Copy the client key to** `/tmp` **on the SP.**

   ```
   # scp /etc/ssh_host_dsa_key.pub manager1@sp.test.com:/tmp
   ```

   b. **Authenticate yourself for the** `scp` **command by entering the password for your manager-level user.**

   c. **Add the client key to the set of trusted hosts for this SP.**

   ```
   # ssh sp.test.com access add trust -c client.test.com -k \
   /tmp/ssh_host_dsa_key.pub
   ```

   d. **Authenticate yourself for the ssh command.**

   From this point, any user with the same login on both `sp.test.com` and `client.test.com` has access without requiring a password to the like-named account on `sp.test.com`.

4. **Create or modify the file** `/etc/ssh_config` **to ensure it contains the following entry:**

   ```
   Host *
   HostbasedAuthentication yes
   ```

Enabling SSH Access Using Public Keys

Follow these steps to install public keys to enable SSH access.

1. **Set up your host keys. Refer to "Generating a Host-Key Pair" on page 59.**

2. **Install your public key using the** `access add public key` **command.**

3. **Run the following command:**

   ```
   # ssh-keygen -t rsa -N
   ```

   This command generates `~/.ssh/id_dsa` and `~/.ssh/id_dsa.pub`.

4. **Run the following command:**

   ```
   # scp ~/.ssh/id_rsa.pub SP_IP:/tmp
   ```

   Enter your password when prompted.

5. **Run the following command:**

   ```
   # ssh SP_IP access add public key -k /tmp/id_rsa.pub
   ```

   Enter your password when prompted.

6. **Run the following command:**

   ```
   # ssh SP_IP rm -f /tmp/id_rsa.pub
   ```

   From this point, you have access without requiring a password.

Guidelines for Writing Server Management Command Scripts

This section describes some basic guidelines for managing your systems by writing scripts for remote execution on one or more SPs.

- Shell Scripts: You should be familiar with standard shell scripting. Refer to "Using Shell Scripts" on page 56.
- SSH: You must currently use an SSH (Secure Shell) client to execute automated command scripts. Refer to "Remote Scripting Using SSH" on page 56.
- Authentication: To avoid being prompted each time you run a script on an SP, upload a public key or trusted-host key to each SP. Refer to "Creating Trusted-Host Relationships" on page 58 and "Adding Public Keys" on page 58.
- Authorization Levels: Access changes (such as adding users or uploading keys) typically requires manager-level access while most other management tasks can be performed by an administrator level user.
- Return Codes: Every subcommand returns a return code upon completion.
- Nowait Argument: Most commands complete their execution fairly quickly and are therefore performed synchronously. For some longer operations (such as rebooting the platform), a `--nowait` option is provided so that a script can initiate the operation without waiting for it to return.
- Quiet Argument: The delete and update operations (such as `access delete user`, `sp delete event`) accept multiple targets. To ensure a certain set of targets is deleted on a set of SPs, you can use the `--quiet` argument to suppress errors if one of the targets is not found, or to suppress interactive warning messages from the platform command.

Command Output

The following list defines common general output:

- Commands that complete successfully return `0` with no success return string. Some exceptions are commands that also return vital information.
- Table output, interactive warnings and any other non-error messages are directed to standard output.
- Commands that return errors display the return codes and a descriptive error string.

Following are common characteristics of table output from a `get` command:

- Heading columns are provided by default for output with more than one column.
- Single column output does not include a heading.
- To suppress headings, use the `-H` argument.

- Data for each column is left-aligned with at least one space between columns. Numeric data might be right-aligned.
- The `-D` argument allows you to specify a delimiter character when scripting. This is very useful in parsing fields with white space.
- If all lines have the same number and type of data values, each row is printed to a separate line so variable data can be parsed easily. For example, executing `access get users -g monitor` returns a list of monitor users each on a separate line.
- Commands that return multiple columns (such as `inventory get hardware`) may have a minimal default set of columns and a `--verbose` argument to display all columns. Some commands include arguments that allow you to select specific columns to output.

Other Tips For Best Results

- Externalize the set of SP IP addresses into a file to be shared across all of your scripts.
- Consider using a script to create the initial manager account and upload its public key on your SPs.
- Test the output and return codes of each command manually by using SSH to log in to the SP and run the commands individually.
- Test your scripts on a single staging machine before applying them to your remaining machines.
- To configure all of your SPs identically, consider configuring a single SP and then using the `sp load settings` command to synchronize that configuration on the remaining machines.

Note – If running the script from the SP, there is a limited number of commands (not a full bash environment).

Console Redirection Over Serial

Redirecting the console interaction over the serial port allows the user another method to monitor the server.

The BIOS redirects console output to serial by default (9600, 8N1, no handshake).

This section describes how to configure these options for both Linux- and Solaris-based servers.

Linux-based Server

Caution – Redirecting the console over serial is a procedure intended for advanced users of Linux only.

You can seriously disrupt the proper functioning of the server or render the server unbootable if you introduce a problem in the configuration files.

The goal of these configurations is to configure the bootloader to redirect its output, pass the kernel the proper parameters and configure a login session on the serial port.

The BIOS redirects console output to serial by default (9600, 8N1, no handshake) until a bootloader program is run from the hard disk drive. The bootloader must be configured to support the serial console in addition to the keyboard, video and mouse (KVM) console.

Two common bootloaders are `grub` and Linux Loader (LILO).

Caution – Do not edit the working-image section of your configuration files directly.

Copy the working-image section and paste it within the configuration file. Make your editing changes to this copied section.

grub

If you use grub, there are three steps to enable console redirection over serial; these steps all involve editing the grub configuration file:

- If you are using Red Hat Linux, the grub file is /etc/grub.conf.
- If you are using SUSE Linux, the grub file is /boot/grub/menu.1st.

Note – On Red Hat Linux systems, the file /etc/grub.conf might be a symbolic link to the file /boot/grub/grub.conf.

---

1. Pass the proper console parameters to the kernel.
2. Configure the grub menu system to redirect to the proper console.
3. Remove any splash images that would prevent the proper serial-console display.

For more information on the parameters, refer to the file kernel-parameters.txt in your kernel documentation.

For more information on grub, run the command info grub.

Note – If the arrow keys do not work through your remote serial concentrator, you can use the keystroke combinations of <CTRL+P> and <CTRL+N> to highlight the Previous and Next entry, respectively. Pressing Enter then boots that entry.

The parameter console=ttyS0 tells the system to send the data to the serial port first. The parameter console=tty0 tells the system to send the data to the KVM second.

A working-image section in your grub configuration file should have an entry for the kernel image to boot. The stock kernel entry looks like:

```
kernel /vmlinuz-kernel_revision ro root=/dev/sda5
```

where *kernel_revision* is simply the kernel version that you are using.

1. **Change the stock kernel entry of your image to include the console-kernel parameters, as follows:**

   ```
   kernel /vmlinuz-kernel_revision ro root=/dev/sda5
   console=ttyS0,9600 console=tty0
   ```

   Note – These options should all be on one line with no wrap to a second line.

   ---

2. **Add the following two lines to the top of your grub configuration file:**

   ```
   serial --unit=0 --speed=9600
   terminal serial console
   ```

   Adding these two lines at the beginning of the file sets up your serial port or your KVM as your grub console so that you can remotely or locally select a boot image from the grub menu.

3. **Comment out or remove the following line from your grub configuration file:**

   ```
   splashimage=(hd0,1)/boot/grub/splash.xpm.gz
   ```

   Removing the splashimage line allows for greater compatibility during your serial connection; with this line removed, the splash image does not prevent the proper grub menu from displaying.

LILO

Note – When you enable the BIOS option "Console Redirection After POST", and LILO is used as a bootloader, the system may hang with an "L" printed on the screen.

This issue arises because there is not enough lower memory available to load the second-stage boot file that LILO uses. If you turn off the option "Console Redirection After POST" in the BIOS, the system will boot normally. See "Enabling and Disabling BIOS Console Redirection" on page 68.

If you require the option "Console Redirection after POST", use grub or upgrade to a newer version of LILO. The current version of LILO is 22.5.9; to access the the LILO pages, visit http://lilo.go.dyndns.org/ and click on the link.

Before you upgrade, we recommend that you verify with your OS vendor that they support the updated version of LILO.

Passing proper parameters to kernel

LILO uses the append feature in an image section in order to pass to the kernel the proper parameters for using the serial console.

1. **In the file** /etc/lilo.conf **on your Sun Fire V20z or V40z server, enter the consoles in the** append **statement:**

   ```
   append="console=ttyS0,9600 console=tty0"
   ```

2. **After modifying the file** /etc/lilo.conf**, run** lilo **from the command line to activate the change.**

   For more information on LILO, run the commands `man lilo` or `man lilo.conf`.

getty

You can run a service called getty to enable login on the serial interface.

To enable getty, append the following line to the list of gettys in the /etc/inittab file:

```
7:12345:respawn:/sbin/agetty 9600 ttyS0
```

Note – It does not matter where you append this line in the list.

Note – Make certain that the first number is unique within the inittab file.

The list of gettys currently looks like the following:

```
# Run gettys in standard runlevels
1:2345:respawn:/sbin/mingetty tty1
2:2345:respawn:/sbin/mingetty tty2
3:2345:respawn:/sbin/mingetty tty3
4:2345:respawn:/sbin/mingetty tty4
5:2345:respawn:/sbin/mingetty tty5
6:2345:respawn:/sbin/mingetty tty6
```

securetty

To add the serial-console device `/dev/ttyS0 to the file /etc/securetty`, run the following command:

```
# echo ttyS0 >> /etc/securetty
```

Solaris-based Server

Caution – Redirecting the console over serial is a procedure intended for advanced users of Solaris only.

You can seriously disrupt the proper functioning of the server or render the server unbootable if you introduce a problem in the `bootenv.rc` file.

Note – The default setting for the output device is `screen` and for the input device is `keyboard`.

To change the settings

To enable Console Redirection over Serial on a Solaris-based server:

- **In a terminal window, run the `eeprom` command to change the settings for the output and input devices, as shown here.**

```
eeprom output-device=ttya
eeprom input-device=ttya
```

To verify the settings

To verify that the changes were made:

1. **In a terminal window, run the `eeprom` command with no arguments.**

   The contents of the `bootenv.rc` file display in the terminal window.

2. **Locate the following lines and verify that they display the correct values.**

```
output-device=ttya
input-device=ttya
```

To reset to the default settings

To reset the system to the default settings:

- **To reset the output and input devices to the default settings, run the `eeprom` command with the following arguments.**

```
eeprom output-device=screen
eeprom input-device=keyboard
```

Enabling and Disabling BIOS Console Redirection

Note – Console redirection is enabled by default in the BIOS.

If the default settings have been changed in the BIOS, the following procedure explains how to change the console-redirection settings.

1. **Boot or reboot the server.**
2. **When prompted, press <F2> to enter BIOS setup.**
3. **Select the Advanced menu from the category selections along the top.**
4. **Select Console Redirection.**

   Note – Make note of all settings in this menu, as they are required for configuring the remote-console access and the Serial-Over-LAN (SOL) feature.

   - To disable Console Redirection to Serial, select Disabled from the Com Port Address option.
   - To change the baud rate, select the desired bit rate from the Baud Rate option.
   - To disable Continue C.R. after POST, toggle the setting to OFF.

5. **Save the changes to the BIOS settings.**
6. **Press <F10> to exit the BIOS setup.**

   For the new settings to take effect, you must reboot the server.

Network Share Volume (NSV) CD-ROM

A network share volume (NSV) structure is included with the server on the Sun Fire V20z and Sun Fire V40z Servers Network Share Volume CD.

Although the SP functions normally without access to an external file system, a file system is required to enable several features, including event log files, software updates, diagnostics and the troubleshooting dump utility. You can configure the NSV to be shared among multiple SPs. Admin- and manager-level users can configure the external file system; regular users can only view the current configuration.

The following software components are included with the server:

- Platform BIOS
- SP base software
- SP value-add software
- Update file for downloading Java Runtime Environment (JRE) packages
- Network share volume software, which includes diagnostics
- Platform software
- Motherboard platform drivers

All of these software packages are packaged with the NSV and are installed on the file server when the external file system is installed and configured.

For instructions on extracting and installing the NSV software, refer to the *Sun Fire V20z and Sun Fire V40z Servers Installation Guide*.

Network Share Volume Structure

The following compressed packages are included with your server on the Sun Fire V20z and Sun Fire V40z Servers Network Share Volume CD:

TABLE 4-1 Network Share Volume Compressed Packages

File Name	File Contents
`nsv_V2.1.0.`*x*`.zip`	Service processor software
`nsv-redhat_V2.1.0.`*x*`.zip`	Drivers for Red Hat Linux OS
`nsv-solaris9_V2.1.0.`*x*`.zip`	Drivers for Solaris 9 OS
`nsv-suse_V2.1.0.`*x*`.zip`	Drivers for SUSE Linux OS

When extracted, the compressed packages in TABLE 4-1 populate the following files on the NSV:

```
/mnt/nsv/
diags
logs
scripts
snmp
spupdate
sw_images
```

`sw_images` (this folder appears after you extract one of the OS-specific Zip files)

TABLE 4-2 Extracted Files on the Network Share Volume

File Name	Description
`diags`	Offline location of the server diagnostics.
`logs`	Offline location of the log files for the SP.
`scripts`	Sample scripts that can be used for scripting commands.
`snmp`	SNMP MIBS. Refer to Chapter 3 for details.
`spupdate`	The server for updating the SP. Refer to Chapter 1 for details.
`sw_images`	Contains a directory hierarchy of platform and SP components, including subdirectories for each version.

Serial Over LAN

The Serial Over LAN (SOL) feature lets servers transparently redirect the serial character stream from the baseboard Universal Asynchronous Receiver/Transmitter (UART) to and from the remote-client system over LAN. Serial over LAN has the following benefits compared to a serial interface:

- Eliminates the need for a serial concentrator.
- Reduces the amount of cabling.
- Allows remote management of servers without video, mouse or keyboard (headless servers).

Serial over LAN requires a properly configured LAN connection and a console from which an `ssh` session can be established.

In a Linux environment, you can use a shell such as `csh` or `ksh` as your console. This console works well in a scripting environment in which you might want to monitor many servers.

Enabling or Disabling the SOL Feature on the Server

Note – When the SOL feature is enabled, you cannot access the server through the external DB9 serial port (COM A).

Note – The variable *spuser* is the user account created when securing the SP. The variable *spipaddr* is the IP address assigned to the SP.

For more information, see "Initial Setup of the Service Processor" on page 10.

You can enable or disable the SOL feature through the SP.

Enabling the SOL feature

To enable the feature, run the following command:

```
# ssh -l spuser spipaddr platform set console -s sp -e -S 9600
```

Note – Ensure that the baud rate value passed to the `-S` argument matches the speed that was specified for the serial-redirection feature of the BIOS and the speed used for your boot loader and OS configuration.

The default baud rate in the BIOS settings is 9600.

Disabling the SOL feature

To disable the feature, run the following command:

```
# ssh -l spuser spipaddr platform set console -s platform
```

Launching an SOL Session

To launch an SOL session, run the following command:

```
# ssh spipaddr -l spuser platform console
```

Terminating an SOL Session

To terminate an SOL session:

1. **Press Control-e.**

2. **Press the ‘c’ key.**

3. **Press the period key (.).**

You can also terminate an SOL session by terminating the `ssh` session:

1. **Press Enter.**

1. **Press the tilde key (~).**

2. **Press the period key (.).**

Escape Sequences for Remote Console Terminal

If you are accessing your server using a remote console terminal, you might need to use the escape sequences shown in TABLE 4-3. If a regular function key is not working properly, use the escape sequence listed next to it in the table.

You will most likely need to use the escape sequences if you are using a Linux or Solaris OS.

TABLE 4-3 Special Keys for Remote Console Terminal

Function Key	Escape Sequence
HOME	<ESC> h
END	<ESC> k
INSERT	<ESC> +
DELETE	<ESC> -
PAGE UP	<ESC> ?
PAGE DOWN	<ESC> /
ALT	<ESC>^A
CTRL	<ESC>^C
F1	<ESC> 1
F2	<ESC> 2
F3	<ESC> 3
F4	<ESC> 4
F5	<ESC> 5
F6	<ESC> 6
F7	<ESC> 7
F8	<ESC> 8
F9	<ESC> 9
F10	<ESC> 0
F11	<ESC> !
F12	<ESC> @

APPENDIX **A**

Server Management Commands Summary

The service processor (SP) includes a suite of commands that enables management and monitoring of the server; this suite of commands is referred to as the server-management commands.

Note – This appendix provides an overview of the server-management command types that are available on the SP. For a detailed description of the subcommands, arguments and return codes for each command type, refer to the appendices in this guide, as described in TABLE A-1.

Using the `ssh` Protocol

You must use `ssh` to execute these commands on the service processor (SP). There are two ways to do this:

- Use the interactive shell on the SP.
- Preface each command with a set piece of text.

Interactive Shell on the SP

To use the interactive shell:

- **Log into and authenticate on the interactive shell by running the command:**

```
# ssh -l spipaddr spuser
```

Preface Text

- **Preface each command with the following text:**

```
# ssh -l spipaddr spuser
```

Commands

The server-management commands take arguments, perform one or more actions, and display the result or text to the standard output device. Commands are grouped by similar function; each command has numerous subcommands supporting functions within that grouping.

Note – Every command (except `help`) returns a return code upon completion. See "Return Codes" on page 78 for a summary.

TABLE A-1 Server-Management Commands

Command	Description
access	Allows the authorized user to manage and monitor access control and security features of the SP, such as users, groups, SSL and so on. **See Appendix B, "Access Commands."**
diags	Manages diagnostics tests that are included with your server. **See Appendix C, "Diagnostics Commands."**
inventory	Allows the authorized user to monitor hardware and software inventory information. **See Appendix D, "Inventory Commands."**
ipmi	Manages IPMI functions. **See Appendix E, "IPMI Commands."**
platform	Allows the authorized user to manage and monitor platform activities, such as rebooting the platform operating system, gathering system status and so on. **See Appendix F, "Platform Commands."**

TABLE A-1 Server-Management Commands

Command	Description
sensor	Reports or sets the value of an environmental sensor or control. **See Appendix G, "Sensor Commands."**
sp	Allows the authorized user to manage and monitor the SP configurations, such as networking, external file system, SNMP, SMTP, SSL, event logs and so on. **See Appendix H, "Service Processor Commands."**
help	Returns the following text: `Available Commands: platform, access, sp, sensor, inventory, ipmi. Each of these commands includes a help option (--help).`

Return Codes

Every subcommand returns one or more of the following return codes upon completion. Refer to the following appendices in this user guide for each subcommand and the corresponding return codes for that subcommand.

TABLE A-2 Return Codes *(1 of 2)*

Return Code	ID	Description
NWSE_Success	0	Command successfully completed.
NWSE_InvalidUsage	1	Invalid usage: bad parameter usage, conflicting options specified.
NWSE_RPCTimeout	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
NWSE_RPCNotConnected	3	Unable to connect to the RPC server.
NWSE_InvalidArgument	4	One or more arguments were incorrect or invalid.
NWSE_NotFound	5	Entity (user, service, file, path, etc.) was not found.

TABLE A-2 Return Codes *(2 of 2)*

Return Code	ID	Description
NWSE_NoPermission	6	Not authorized to perform this operation.
NWSE_MissingArgument	7	Missing argument(s).
NWSE_NoMemory	8	Insufficient memory.
NWSE_Busy	9	Device or resource is busy.
NWSE_NotImplemented	10	Function not implemented.
NWSE_RPCConnected	11	RPC client already connected.
NWSE_RPCConnRefused	12	RPC connection refused.
NWSE_NoRouteToHost	13	No route to host (network down).
NWSE_HostDown	14	Host is down.
NWSE_UnknownError	15	Miscellaneous error not captured by other errors.
NWSE_GatewayOffNet	16	Gateway address is not on network.
NWSE_NetMaskIncorrect	17	An inappropriate netmask was specified.
NWSE_FileError	18	File open, file missing, or a read or write error occurred.
NWSE_Exist	19	Entity (user, service or other) already exists.
NWSE_NotRecognized	20	Request not understood.
NWSE_NotMounted	21	File system is not mounted.
NWSE_InvalidOpForState	22	Invalid operation for current state.
NWSE_TimedOut	23	Operation timed out.
NWSE_ServiceNotAvailable	24	Requested service is not available.
NWSE_DeviceError	25	Unable to read or write to the device.
NWSE_LimitExceeded	26	Limit has been exceeded.

APPENDIX **B**

Access Commands

The access command validates a user’s authority or controls authorization services. Using the access command, you can retrieve information about user groups, add a user to or delete a user from a group, and specify a mapping between site-defined administrative groups and the administrative groups that are used to authorize actions on the Service Processor.

TABLE B-1 lists the groups of access subcommands.

TABLE B-1 Access Subcommand Groups

Subcommand Group	Description
`access groups`	Returns the authorization group for a specific user or a list of defined groups.
`access map`	Maps, unmaps and returns a list of existing site-specified group names (the directory service group) mapped to one of the standard administrative groups.
`access public key`	Manages public keys and public key users.
`access services`	Enables, disables, or defines a directory services mechanism that determines a user's group memberships.
`access trust`	Creates a host-based trust relationship for the specified host.
`access user`	Manages local users or a group of users.

Note – Every command returns a return code upon completion.

Access Groups Subcommands

The subcommands in TABLE B-2 return the authorization group for a specific user or a list of defined groups.

TABLE B-2 `Access Group` Subcommands

Subcommand	Description
`access get group`	Returns the authorization group for the specified user.
`access get groups`	Returns a list of the groups defined, including the standard groups.

Access Get Group Subcommand

Description: Returns the authorization group for the specified user.

Format

Command format:

```
access get group
```

Return Codes

TABLE B-3 lists the return codes for this subcommand.

TABLE B-3 Return Codes for Subcommand `access get group`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.

Access Get Groups Subcommand

Description: Returns a list of the groups defined, including the standard groups.

Format

Command format:

```
access get groups
```

Return Codes

TABLE B-4 lists the return codes for this subcommand.

TABLE B-4 Return Codes for Subcommand `access get groups`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

Access Map Subcommands

The subcommands in TABLE B-5 manage mappings between existing site-specified groups and one of the standard administrative groups.

TABLE B-5 `Access Map` Subcommands

Subcommand	Description
`access get map`	Returns the names of all the site-specified groups mapped to a specific administrative group.
`access map`	Maps an existing site-specified group name (the directory-service group) to one of the standard administrative groups.
`access unmap`	Removes the directory-service group and administrative group mapping.

Access Get Map Subcommand

Description: Returns the names of all the site-specified groups mapped to a specific administrative group.

Format

Command format:

```
access get map [{-H | --noheader}]
[{-D | --delim <DELIMITER>}]
```

TABLE B-6 lists the arguments for this subcommand.

TABLE B-6 Arguments for Subcommand `access get map`

Argument	Description
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

To return mappings for all groups, omit the group name from the command line.

Return Codes

TABLE B-7 lists the return codes for this subcommand.

TABLE B-7 Return Codes for Subcommand `access get map`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.

Access Map Subcommand

Description: Maps an existing site-specified group name (the directory-services group) to one of the standard administrative groups.

Format

Command format:

```
access map {-d | --dsgroup} DIRECTORY-SERVICES-GROUP
{-g | --group} LOCAL-GROUP {-v / --verify}
```

TABLE B-8 lists the arguments for this subcommand.

TABLE B-8 Arguments for Subcommand `access map`

Argument	Description
`{-d | --dsgroup}`	The name of the directory-services group for which you wish to map to a standard administrative group.
`{-g | --group}`	The name of the standard administrative group to which you wish to map to the directory-services group.
`{-v / --verify}`	Verifies the group existence.

Return Codes

TABLE B-9 lists the return codes for this subcommand.

TABLE B-9 Return Codes for Subcommand `access map`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Access Unmap Subcommand

Description: Removes the directory service group and administrative group mapping.

Format

Command format:

```
access unmap [-a | --all] DIRECTORY-SERVICES-GROUP
```

TABLE B-10 lists the arguments for this subcommand.

TABLE B-10 Arguments for Subcommand `access unmap`

Argument	Description
`DIRECTORY-SERVICES-GROUP`	The name of the directory services group for which you wish to remove a mapping.
`[-a | --all]`	Removes mappings for all of the directory services groups.

Return Codes

TABLE B-11 lists the return codes for this subcommand.

TABLE B-11 Return Codes for Subcommand `access unmap`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access Directory Services Subcommands

Services defines a directory-services mechanism that determines the group memberships for a user. Remote users gain access to the Service Processor features only through these group mappings that relate a directory-services group to a local Service-Processor administrative group.

Therefore, using the command `access map`, the administrator must set up the appropriate directory-services configuration and create mappings from the directory-services groups to local Service-Processor administrative groups.

TABLE B-12 lists the `Access Directory Services` subcommands.

TABLE B-12 `Access Directory Services` Subcommands

Subcommand	Description
`access disable service`	Disables a directory service.
`access enable service`	Enables a directory service.
`access get services`	Defines a directory services mechanism that determines the group memberships for a user.

Access Disable Service Subcommand

Description: Disables a directory service (either NIS or ADS) from the name-service lookup system on the SP.

Format

Command format:

```
access disable service {nis | ads}
```

TABLE B-13 lists the argument for this subcommand.

TABLE B-13 Argument for Subcommand `access disable service`

Argument	Description
`{nis | ads }`	Specifies the service type: NIS or ADS.

Return Codes

TABLE B-14 lists the return codes for this subcommand.

TABLE B-14 Return Codes for Subcommand `access disable service`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Access Enable Service Subcommand

Description: Enables a directory service (either NIS or ADS) to name-service lookup system on the SP.

Format

Command format:

```
access enable service nis {-d | --domain} DOMAIN NAME {-s | --server }
SERVER
```

```
access enable service ads {-d | --domain} DOMAIN NAME {-s | --server }
SERVER {-k | --keytab} KEYTAB FILENAME {-o | --ou} ORGANIZATIONAL UNIT {-
l|--logon} LOGON
```

TABLE B-15 lists the arguments for this subcommand.

TABLE B-15 Arguments for Subcommand `access enable service`

Argument	Description
`{-d | --domain}`	Specifies the domain name.
`{-s | --server}`	Specifies the server.
`{-k | --keytab}`	For ADS only: Specifies the ADS keytab file name.
`{-o | --ou}`	For ADS only: Specifies the organizational unit under which the name-service library looks for group data.
`{-l | --logon}`	For ADS only: Specifies the logon ID for the active directory account.

To use ADS as a directory service on the SP, you must create an active directory account. The name-service library on the SP uses this account to authenticate itself to the LDAP interface of the active directory server. A Windows administrator can create the keytab for this account using the following command:

```
ktpass -princ <logon>@<domain> -pass <password> -mapuser <logon> -out
<output filename>
```

The keytab file must then be securely transferred to the SP using an encrypted file-transfer mechanism.

The clock on the SP must be accurate and DNS must be set up (meaning that the SP must have a DNS record).

If a directory service has been previously enabled, you can specify the following command and options; the saved settings are then used to re-enable the service.

```
access enable service -t <nis | ads>
```

Return Codes

TABLE B-16 lists the return codes for this subcommand.

TABLE B-16 Return Codes for Subcommand `access enable service`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.

TABLE B-16 Return Codes for Subcommand `access enable service`

Return Code	ID	Description
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Access Get Services Subcommand

Description: Returns a string containing the current naming services option (NIS or ADS).

Format

Command format:

```
access get services [ {-t | --type } nis
[{-d | --domain} | {-s | --server}]
[-H | --noheader]] [{-D | --delim <DELIMITER>}]

access get services [ {-t | --type } ads
[{ -d | --domain} | {-s | --server} |
{-l | --logonID} | {-o | --ou}]
[-H | --noheader]] [{-D | --delim <DELIMITER>}
```

TABLE B-17 lists the arguments for this subcommand.

TABLE B-17 Arguments for Subcommand `access get services`

Argument	Description
`{-t	--type }`	Returns information about the configuration of either the NIS or ADS service. You must specify `-t` to return a list of enabled services.
`{-d	--domain}`	Returns domain information. Only one of the parameters `-d` and `-s` are permitted at a time.
`{-s	--server}`	Returns server information. Only one of the parameters `-d` and `-s` are permitted at a time.

TABLE B-17 Arguments for Subcommand `access get services`

Argument	Description
`{-l | --ID}`	For ADS only: Returns the ADS logon ID. Only one of the parameters `-o` and `-l` are permitted at a time.
`{-o | --ou}`	For ADS only: Returns the organization unit information. Only one of the parameters `-o` and `-l` are permitted at a time.
`[-H | --noheader]`	Suppresses header output.
`{-D | --delim <DELIMITER>}`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE B-18 lists the return codes for this subcommand.

TABLE B-18 Return Codes for Subcommand `access get services`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.

Access Trust Subcommands

Adding host-based trusts provides many-to-one scripting solutions. Once a host equivalence relationship has been created with a client, users on that client can remotely execute commands on the SP without being prompted for a password.

TABLE B-19 lists the commands related to trusted-host relationships.

TABLE B-19 `Access Trust` Subcommands

Subcommand	Description
`access add trust`	Creates a host-based trust relationship for the specified host.
`access delete trust`	Removes a host-based trust relationship for the specified host.
`access get trusts`	Requests a list of hosts involved in trust relationships with the SP.

Access Add Trust Subcommand

Description: Creates a host-based trust relationship for the specified host. Adding host-based trusts provides many-to-one scripting solutions. Once a host equivalence relationship has been created with a client, users on that client can remotely execute commands on the SP without being prompted for a password, provided one of the following conditions is met:

- their login on the client has the same user name as a local user on the SP
- their login on the client is in a directory-service group that is mapped to an SP administrative group

Format

Command format:

```
access add trust {-c | --client} HOST {-k | --keyfile} PUBLIC KEY FILE
```

TABLE B-20 lists the arguments for this subcommand.

TABLE B-20 Arguments for Subcommand `access add trust`

Arguments	Description
`{-c | --client}`	Specifies the host for which to create the relationship.
`{-k | --keyfile}`	Specifies the public key file.

If the login is authorized through a mapping of a directory-service group, the `ssh` command is executed as the proxy user on the SP, either *rmonitor, radmin* or *rmanager*.

Support is available for SSH protocol version 2 key types (RSA or DSA) only.

If DNS is enabled on the SP, the client machine must be specified with its DNS name, (and not the IP address).

Generating Host Keys

The host's `ssh` install should generate the host keys. If it does not, follow these steps to manually generate the key pair:

1. **Enter the following command:**

   ```
   ssh-keygen -q -t rsa -f rsa_key -C '' -N ''
   ```

2. **Copy `rsa_key` to `/etc/ssh/ssh_host_rsa_key`.**

3. **Ensure that only *root* has read or write permission to this file. The `rsa_key.pub` file is the file you will transfer to the SP.**

Note – Only protocol version 2 key types and 1024 bit key sizes (the default generated by `ssh-keygen`) are supported.

---

4. **Copy the host's public key (the `rsa_key.pub` file) to the SP using `scp` (secure copy) or by copying the host key to an external file system that has been mounted on the SP.**

Note – Use `scp` to copy the files to either `/tmp` or to your home directory. The sp commands will then install the file specified on the command line to `/pstore`.

Note – If DNS is enabled on the SP, you must specify the client that is used in the trust commands with its DNS name (and not the IP address).

Return Codes

TABLE B-21 lists the return codes for this subcommand.

TABLE B-21 Return Codes for Subcommand `access add trust`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_FileError`	18	FileFile open, file missing, or a read or write error occurred.
`NWSE_Exist`	19	Entity (user, service or other) already exists.

Access Delete Trust Subcommand

Description: Removes a host-based trust relationship for the specified host.

Format

Command format:

```
access delete trust CLIENT HOSTNAME [-a | --all] [-q | --quiet]
```

TABLE B-22 lists the arguments for this subcommand.

TABLE B-22 Arguments for Subcommand `access delete trust`

Argument	Description
`CLIENT HOSTNAME`	Specifies the name of the client to remove.
`[-a | --all]`	Removes all trust relationships.
`[-q | --quiet]`	If the trust relationship to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE B-23 lists the return codes for this subcommand.

TABLE B-23 Return Codes for Subcommand `access delete trust`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_DeviceError`	25	Error deleting trusted host. Insufficient space in `/tmp`.

Access Get Trusts Subcommand

Description: Requests a list of hosts involved in trust relationships with the SP.

Format

Command format:

```
access get trusts
```

Return Codes

TABLE B-24 lists the return codes for this subcommand.

TABLE B-24 Return Codes for Subcommand `access get trusts`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access Public Key Subcommands

The subcommands listed in TABLE B-25 allow you to manage public keys and public-key users.

TABLE B-25 Access Public Key Subcommands

Subcommand	Description
`access add public key`	Installs a public key for SSH authentication.
`access get public key users`	Determines which users have public keys installed.
`access delete public key`	Removes a user's public key.

Access Add Public Key Subcommand

Description: Installs a public key for SSH authentication which enables SSH logins and remote command execution without being prompted for a password. You must first generate a key pair (RSA or DSA) which you can generate using the `ssh-keygen` command included with OpenSSH.

- Only local users can install public keys (not users who gain authorization through a mapping of a directory-services group)
- Managers can add keys for any local user.
- Up to 10 users can install public keys; one key per user.
- The maximum key length supported is 4096 bits.

Format

Command format:

```
access add public key {—k | --keyfile} PUBLIC_KEY_FILE [—u | --user] USER
```

TABLE B-26 lists the arguments for this subcommand.

TABLE B-26 Arguments for Subcommand `access add public key`

Arguments	Description
`{—k | --keyfile}`	Specifies the user's public RSA or DSA key.
`{-u / --user}`	Specifies the user for which this key will be installed. The default is the current user if no user is specified.

Return Codes

TABLE B-27 lists the return codes for this subcommand.

TABLE B-27 Return Codes for Subcommand `access add public key`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid. The group specified with -g is an invalid local SP administrative group or the length of the username or password exceeds the maximum length.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_Exist`	19	The user already exists.
`NWSE_LimitExceeded`	26	Limit has been exceeded.

Access Get Public Key Users Subcommand

Description: Determines which users have public keys installed.

Format

Command format:

```
access get public key users
```

Return Codes

TABLE B-28 lists the return codes for this subcommand.

TABLE B-28 Return Codes for Subcommand `access get public key users`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access Delete Public Key Subcommand

Description: All users can execute this command to remove their own public key. Manager-level users can execute this command to remove the public key for any user.

Format

Command format:

```
access delete public key [-u | --user] USER [-a | --all] [-q | --quiet]
```

TABLE B-29 lists the arguments for this subcommand.

TABLE B-29 Arguments for Subcommand `access delete public key`

Arguments	Description
`[-u | --user]`	The user whose public key will be removed. Defaults to the current user If USER is not specified. This argument is repeatable to remove multiple public keys at one time.
`[-a | --all]`	Removes all public keys.
`[-q | --quiet]`	If the user to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE B-30 lists the return codes for this subcommand.

TABLE B-30 Return Codes for Subcommand `access delete public key`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access User Subcommands

The subcommands listed in TABLE B-31 allow you to manage a single user or group of users.

TABLE B-31 `Access User` Subcommands

Subcommand	Description
`access add user`	Adds the specified local user to the specified group.
`access delete user`	Deletes the specified user.
`access get users`	Retrieves all the users in an administrative group or all users in all groups.
`access update password`	Updates the password of the specified user.
`access update user`	Updates the login information for the specified user.

Access Add User Subcommand

Description: Adds the specified local user to the specified group with the specified user ID and password.

Format

Command format:

```
access add user {-p | --password} PASSWORD {-g | --group} GROUP
{-u / --user} USERNAME
```

TABLE B-32 lists the arguments for this subcommand.

TABLE B-32 Arguments for Subcommand `access add user`

Arguments	Description
`{-p | --password}`	Specifies the password for the new user. The password is optional and if not specified, a prompt displays requesting confirmation.
`{-g | --group}`	Specifies the group to which the new user will belong.
`{-u / --user}`	Specifies the name of the new user to add. This argument is repeatable to add multiple users at one time.

Return Codes

TABLE B-33 lists the return codes for this subcommand.

TABLE B-33 Return Codes for Subcommand `access add user`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid. The group specified with `-g` is an invalid local SP administrative group or the length of the user name or password exceeds the maximum length.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_Exist`	19	The user already exists.

Access Delete User Subcommand

Description: Deletes a user:

Format

Command format:

```
access delete user USERNAME [-a | --all] [-q | --quiet]
```

TABLE B-34 lists the arguments for this subcommand.

TABLE B-34 Arguments for Subcommand `access delete user`

Argument	Description
`USERNAME`	Specifies the name of the user to remove. This argument is repeatable to remove multiple users at one time.
`[-a | --all]`	Removes all user accounts. The manager-level user executing the command is not removed.
`[-q | --quiet]`	If the user to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE B-35 lists the return codes for this subcommand.

TABLE B-35 Return Codes for Subcommand `access delete user`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Specified user was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access Get Users Subcommand

Description: Retrieves all the local users in an administrative group.

Format

Command format:

```
access get users {-g | --group} [{-H | noheader}][{-D | --delim
<DELIMITER>}]
```

TABLE B-36 lists the arguments for this subcommand.

TABLE B-36 Arguments for Subcommand `access get users`

Argument	Description
`{-g | --group}`	Specifies that group from which to retrieve all users.
`{ -H | --noheader }`	Specifies that column headings should be suppressed.
`{ -D | --delim }`	Specifies to delimit columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE B-37 lists the return codes for this subcommand.

TABLE B-37 Return Codes for Subcommand `access get users`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.

Access Update Password Subcommand

Note – This command is for managers to change other users' passwords; all users can change their own passwords.

Description: Changes the password of an existing user.

Format

Command format:

```
access update password {-p | --password} PASSWORD {u / --user} USER
```

TABLE B-38 lists the arguments for this subcommand.

TABLE B-38 Arguments for Subcommand `access update password`

Argument	Description
`{-u | --user}`	The name of the user whose password you wish to update. If a username is not specified, the current user is implied. You must have manager-level access to change another user's password. This argument is repeatable to update multiple user's passwords at one time.
`{-p | --password}`	The user's new password. If a password is not specified, a prompt appears to enter the password and again to confirm the password.

Return Codes

TABLE B-39 lists the return codes for this subcommand.

TABLE B-39 Return Codes for Subcommand `access update password`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

TABLE B-39 Return Codes for Subcommand `access update password`

Return Code	ID	Description
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Access Update User Subcommand

Description: Updates the login information (password or group) for the user.

Format

Command format:

```
access update user {-u | --user} USER {-p | --password} PASSWORD
{-g | --group} GROUP
```

TABLE B-40 lists the arguments for this subcommand.

Note – The `-p` and `-g` arguments are optional but you must specify at least one.

TABLE B-40 Arguments for Subcommand `access update user`

Argument	Description
`{-u	--user}`	The name of the user to update.
`{-p	--password}`	The user's new password. The `-p` and `-g` options are optional but you must specify at least one.
`{-g	--group}`	The new group to which to reassign to the user. The `-p` and `-g` options are optional but you must specify at least one.

Return Codes

TABLE B-41 lists the return codes for this subcommand.

TABLE B-41 Return Codes for Subcommand `access update user`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

APPENDIX C

Diagnostics Commands

The diags commands allow you to manage the diagnostics tests.

Known Issues with Diagnostics

Do Not Access the SP While Diagnostics Are Loaded

While running diagnostics on your server, do not interact with the Service Processor (SP) through the command-line interface or IPMI.

The sensor commands cannot be used reliably while the diagnostics are running. Issuing sensor commands, while diagnostics are loaded, may result in “false” or erroneous critical events being logged in the events log. The values returned by the sensors are not reliable in this case.

Benign Error Message

When the diagnostics are launched on the platform, the system tries to mount the floppy drive. The following error is returned:

```
mount : Mounting /dev/fd0 on /mnt/floppy failed. No such device.
```

You can safely ignore this error message.

TABLE C-1 lists the groups of `diags` subcommands.

Note – The diagnostics commands are also provided in the *Sun Fire V20z Server User Guide*, 817-5248.

TABLE C-1 Diagnostics Subcommand Groups

Subcommand	Description
`diags cancel tests`	Cancels one or more diagnostic tests, resulting in the deletion of the results data.
`diags get state`	Returns the state of the platform-diagnostics control server.
`diags get tests`	Returns data describing the diagnostic tests that are available and their requirements and parameters.
`diags run tests`	Submits one or more diagnostic tests for execution.
`diags start`	Starts the Service Processor (SP) and platform-diagnostics framework.
`diags terminate`	Terminates all diagnostics tests and terminates the diagnostics subsystem.

Note – Every command returns a return code upon completion.

Diags Cancel Tests Subcommand

Description: Cancels one or more diagnostic tests, resulting in the deletion of results data.

Format

Command format:

```
diags cancel tests [[{ -t | --test} TEST HANDLE] [{-a|--all}]
[{-H | --noheader}]]
```

TABLE C-2 lists the arguments for this subcommand.

Note – Specifying no arguments cancels all tests for each device in the server.

TABLE C-2 Arguments for Subcommand `diags cancel tests`

Arguments	Description
`{ -t | --test}`	Specifies the test to cancel. NOTE: The `TEST HANDLE` is the same `TEST HANDLE` that is output to the screen when you submit the test.
`{-a | --all}`	Cancels all tests.
`{-H | --noheader}`	Suppresses header output.

Return Codes

TABLE C-3 lists the return codes for this subcommand.

TABLE C-3 Return Codes for Subcommand `diags cancel tests`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform operation.
`NWSE_MissingArgument`	7	Missing argument(s).

Diags Get State Subcommand

Description: Returns the state of the platform-diagnostics control server.

Format

Command format:

```
diags get state
```

If the result returned from the command is that the platform is up and ready for diagnostics, then you can submit platform diagnostic tests for execution.

Success text message – The platform diagnostics are up and are available to receive test requests.

Platform diags not ready message – The platform diagnostics are loading, but not available yet.

Diagnostic system not loaded message – The diagnostics system is not loaded. Use the `diags start` command to load diagnostics.

Return Codes

TABLE C-4 lists the return codes for this subcommand.

TABLE C-4 Return Codes for Subcommand `diags get state`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_DeviceError`	25	Unable to read or write to the device.

Diags Get Tests Subcommand

Description: Returns data describing the diagnostic tests that are available. This data includes the specific test name and the module to which the test applies.

Format

Command format:

```
diags get tests [{ -a | --all}] [{-H | --noheader}]]
[{-D | --delim <DELIMITER>}]
```

TABLE C-5 lists the arguments for this subcommand.

TABLE C-5 Arguments for Subcommand `diags get tests`

Arguments	Description
`{-a | --all}`	Specifies to return information for all tests in the server. Specifying no arguments also returns all tests available for each device in the server.
`{-H | --noheader}`	Suppresses header output.
`{-D | --delim <DELIMITER>}`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE C-6 lists the return codes for this subcommand.

TABLE C-6 Return Codes for Subcommand `diags get tests`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

TABLE C-6 Return Codes for Subcommand `diags get tests`

Return Code	ID	Description
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_MissingArgument`	7	Missing argument(s).

Diags Run Tests Subcommand

Description: Submits one or more diagnostic tests for execution.

Format

Command format:

```
diags run tests [ [{ -n | --name} TEST NAME ] [{-a| --all}]
[-H | --noheader] [-P | --noprogress] [{-m | --module} MODULE NAME]
[-v | --verbose]
```

TABLE C-7 lists the arguments for this subcommand.

TABLE C-7 Arguments for Subcommand `diags run tests`

Arguments	Description
`{ -n | --name}`	Specifies the specific test(s) to execute. Run `diags get tests` for a list of individual test names.
`{-a | --all}`	Specifies that all tests are to be executed. Run `diags get tests` for a list of all available tests. Specifying no arguments also runs all tests for each device in the server.
`{-H |--noheader}`	Suppresses header output.
`{-P | --noprogress}`	Suppresses progress dots when waiting for test results.
`{-m | --module}`	Specifies that only tests for the specified module are to be executed. Run `diags get tests` for a list of modules.
`[-v | --verbose]`	If specified, the `Test Details` display following the test result line.

The following data displays after a test is run:

- Submitted Test Name
- Test Handle
- Test Result (for example: Passed, Failed)
- Details. If you specify the `-v` option, the Test Details are displayed, indicating detailed information about the test, such as high, low and nominal values, actual values and so on. Upon failure, the Failure Details are displayed with a text message indicating the cause of failure.

Return Codes

TABLE C-8 lists the return codes for this subcommand.

TABLE C-8 Return Codes for Subcommand `diags run tests`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_MissingArgument`	7	Missing argument(s).

Diags Start Subcommand

Description: Starts the SP and platform-diagnostics framework. You must execute this command before running any tests. After running this command, you can immediately run Service Processor tests or wait for the subcommand **`diags get state`** to return the result *platform diags available,* at which point you can run platform tests.

The platform state must be either off or OS Communicating. Refer to the subcommand **`platform get os state`** for details about these states.

Format

Command format:

```
diags start {--noplatform}
```

Arguments	Description
`{--noplatform}`	Lets you enable diagnostics when the platform power is on and the OS is running. **Note:** If you use this option, you cannot use any platform diagnostics, which include diagnostics for memory, NIC cards and storage.

Return Codes

TABLE C-9 lists the return codes for this subcommand.

TABLE C-9 Return Codes for Subcommand `diags start`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

TABLE C-9 Return Codes for Subcommand `diags start`

Return Code	ID	Description
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Diags Terminate Subcommand

Description: Terminates all diagnostics tests and the diagnostics session.

Format

Command format:

```
diags terminate
```

Return Codes

TABLE C-10 lists the return codes for this subcommand.

TABLE C-10 Return Codes for Subcommand `diags terminate`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_MissingArgument`	7	Missing argument(s).

APPENDIX **D**

Inventory Commands

The `inventory` command reports on the inventory of hardware and software for a Sun Fire V20z server.

TABLE D-1 lists the groups of `inventory` subcommands that you can use to retrieve specific information about hardware or software.

TABLE D-1 Inventory Subcommands Groups

Subcommand	Description
`inventory compare versions`	Returns a list of all installed software packages and the version differences with those listed in a release manifest.
`inventory get hardware`	Returns detailed information for all field-replaceable hardware components.
`inventory get software`	Returns inventory information for all installed or uninstalled software.
`inventory get all`	Returns detailed information for all hardware and software components.

Note – Every command returns a return code upon completion.

Inventory Compare Versions Subcommand

Description: Returns a list of all installed software packages and the version differences with those listed in a release manifest. You can use this command to verify that your installation is consistent with a supported release and to determine the packages that have been updated in a new release.

Format

Command format:

```
inventory compare versions {-f | ––file} RELEASE_MANIFEST_FILE {-v | ––
verbose}
```

TABLE D-2 lists the arguments for this subcommand.

TABLE D-2 Arguments for Subcommand `inventory compare versions`

Arguments	Description
`{-f | ––file}`	The file describing all of the packages and versions within a release of software. These files are at the root directory of an unzipped NSV file and are usually accessed via the share point at `/mnt`.
`{-v | ––verbose}`	Displays additional information, including the path to the matching package on the NSV, the installed package description and the matching manifest package description.

Return Codes

TABLE D-3 lists the return codes for this subcommand.

TABLE D-3 Return Codes for Subcommand `inventory compare versions`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

Inventory Get Hardware Subcommand

Description: Returns detailed information for all field-replaceable hardware components. By default, the name, type, OEM, manufacture date, hardware revision and part number display for each component.

Format

Command format:

```
inventory get hardware {-v | --verbose} [{-H | --noheader}]
[{-D | --delim <DELIMITER>}]
```

TABLE D-4 lists the arguments for this subcommand.

TABLE D-4 Arguments for Subcommand `inventory get hardware`

Arguments	Description
`{ -v | --verbose }`	Displays all columns.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

To obtain the board revision, product ID and PRS revision, you can run the following commands:

```
inventory get hardware -D '|' | awk -F '|' '/PRS/{print $8}'
```

```
inventory get hardware -D '|' | awk -F '|' '/PRS/{print $7}' |
awk '{print $4}'
```

```
inventory get hardware -D '|' | awk -F '|' '/PRS/{print $6}'
```

You can also obtain this information by running the `sensor get` command.

Return Codes

TABLE D-5 lists the return codes for this subcommand.

TABLE D-5 Return Codes for Subcommand `inventory get hardware`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

Inventory Get Software Subcommand

Description: Returns the inventory information for all installed or uninstalled software (located on the optional external file system).

Format

Command format:

```
inventory get software [{-a | --all}][{-H | --noheader}]
[{-D | --delim <DELIMITER>}]
```

TABLE D-6 lists the arguments for this subcommand.

TABLE D-6 Arguments for Subcommand `inventory get software`

Arguments	Description
`{-a | --all}`	Optional: Looks in the directory `/sw_images` on the Service Processor for software packages and uninstalled software.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE D-7 lists the return codes for this subcommand.

TABLE D-7 Return Codes for Subcommand `inventory get software`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

Inventory Get All Subcommand

Description: Returns detailed information for all field-replaceable hardware components and all installed or uninstalled software.

Format

Command format:

```
inventory get all {-a | --all} {-v | --verbose} [{-H | --noheader}]
[{-D | --delim <DELIMITER>}]
```

TABLE D-8 lists the arguments for this subcommand.

TABLE D-8 Arguments for Subcommand `inventory get all`

Arguments	Description
`{-a | --all}`	Optional: Looks in the directory `/sw_images` on the Service Processor for software packages and uninstalled software.
`{ -v | --verbose }`	Displays all columns.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE D-9 lists the return codes for this subcommand.

TABLE D-9 Return Codes for Subcommand `inventory get all`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

APPENDIX E

IPMI Commands

The ipmi command manages the Intelligent Platform Management Interface (IPMI) functions.

TABLE E-1 lists the groups of ipmi subcommands.

TABLE E-1 IPMI Subommands Groups

Subcommand	Description
ipmi disable channel	Disables one of two IPMI channels.
ipmi enable channel	Enables one of two IPMI channels.
ipmi get channels	Displays the list of IPMI channels and whether they are enabled or disabled.
ipmi enable pef	Enables platform-event filtering.
ipmi disable pef	Disables platform-event filtering.
ipmi get global enables	Displays the list of IPMI global enables and their current value.
ipmi set global enable	Enables one of two IPMI channels.
ipmi reset	Resets IPMI information back to default factory settings.

Note – Every command returns a return code upon completion.

IPMI Disable Channel Subcommand

Description: Allows you to disable one of two IPMI channels.

Format

Command format:

```
ipmi disable channel {sms | lan}
```

TABLE E-2 lists the arguments for this subcommand.

TABLE E-2 Arguments for Subcommand `ipmi disable channel`

Arguments	Description
`sms`	The ID of the channel to disable for the System Interface; not case-sensitive.
`lan`	The ID of the channel to disable for the LAN Interface; not case-sensitive.

Return Codes

TABLE E-3 lists the arguments for this subcommand.

TABLE E-3 Return Codes for Subcommand `ipmi disable channel`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

IPMI Enable Channel Subcommand

Description: Allows you to enable one of two IPMI channels.

Format

Command format:

```
ipmi enable channel {sms | lan}
```

TABLE E-4 lists the arguments for this subcommand.

TABLE E-4 Arguments for Subcommand `ipmi enable channel`

Arguments	Description
`sms`	The ID of the channel to enable for the System Interface; not case-sensitive.
`lan`	The ID of the channel to enable for the LAN Interface; not case-sensitive. If you are activating the LAN channel for the first time, you are prompted for a password to associate with the *null* user.

Return Codes

TABLE E-5 lists the return codes for this subcommand

TABLE E-5 Return Codes for Subcommand `ipmi enable channel`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

IPMI Get Channels Subcommand

Description: Displays the list of IPMI channels and whether they are enabled or disabled.

Format

Command format:

```
ipmi get channels
```

Return Codes

TABLE E-6 lists the return codes for this subcommand.

TABLE E-6 Return Codes for Subcommand `ipmi get channels`

Return Code	ID	Description
`NWSE_Success`	`0`	Command successfully completed.
`NWSE_InvalidUsage`	`1`	Invalid usage: bad parameter usage, conflicting options specified.

IPMI Disable PEF Subcommand

Description: Allows you to disable platform-event filtering (PEF).

Format

Command format:

```
ipmi disable pef
```

Return Codes

TABLE E-7 lists the return codes for this subcommand.

TABLE E-7 Return Codes for Subcommand `ipmi disable pef`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

IPMI Enable PEF Subcommand

Description: Allows you to enable platform-event filtering (PEF).

Format

Command format:

```
ipmi enable pef
```

Return Codes

TABLE E-8 lists the return codes for this subcommand.

TABLE E-8 Return Codes for Subcommand `ipmi enable pef`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

IPMI Get Global Enables Subcommand

Description: Displays the list of IPMI global enables and their current value.

Format

Command format:

```
ipmi get global enables
```

Return Codes

TABLE E-9 lists the return codes for this subcommand.

TABLE E-9 Return Codes for Subcommand `ipmi get global enables`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.

IPMI Set Global Enable Subcommand

Description: Allows you to set the value of several IPMI global-enable variables.

Format

Command format:

```
ipmi set global enable {-n |--name} GLOBAL_NAME {{-e|--enabled} |
{-d|--disabled}}
```

TABLE E-10 lists the arguments for this subcommand.

TABLE E-11 provides information about the aliases.

TABLE E-10 Arguments for Subcommand `ipmi set global enable`

Arguments	Description
`{-n |--name}`	The name of one of the IPMI global enable variables; see TABLE E-11. You can use either a quoted long string or an alias without quotes for the list of global enables.
`{-e|--enabled}`	Turns the channel on.
`{-d|--disabled}`	Turns the channel off.

TABLE E-11 Information about the aliases

Alias	Name String	Values	Default
`oem0`	OEM0 Enable	Enabled/ Disabled	Disabled
`oem1`	OEM1 Enable	Enabled/ Disabled	Disabled
`oem2`	OEM 2 Enable	Enabled/ Disabled	
`logging`	Enable System Event Logging	Enabled/ Disabled	Enabled

TABLE E-11 Information about the aliases

Alias	Name String	Values	Default
`msg_buf`	Enable Event Message Buffer	Enabled/ Disabled	
`msg_buf_interrupt`	Enable the Event Message Buffer Full	Enabled/ Disabled	
`msg_queue_interrupt`	Enable Receive Message Queue Interrupt	Enabled/ Disabled	Enabled

Return Codes

TABLE E-12 lists the return codes for this subcommand.

TABLE E-12 Return Codes for Subcommand `ipmi set global enable`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

IPMI Reset Subcommand

Description: Resets IPMI information back to default factory settings.

Format

Command format:

```
ipmi reset {-s | ––sdrr} {-c | ––config} {-p | ––password} {-a | ––all}
```

TABLE E-13 lists the arguments for this subcommand.

TABLE E-13 Arguments for Subcommand `ipmi reset`

Arguments	Description
`{-s	––sdrr}`	Copies the original database file to `pstore`.
`{-c	––config}`	Deletes the configuration file and global enables.
`{-p	––password}`	Deletes the password file.
`{-a	––all}`	Performs the functions of all the parameters.

Return Codes

TABLE E-14 lists the return codes for this subcommand.

TABLE E-14 Return Codes for Subcommand `ipmi reset`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

APPENDIX **F**

Platform Commands

The `platform` command reports or changes some aspect of the state of the platform.

TABLE F-1 lists the groups of `platform` subcommands.

TABLE F-1 Platform Subcommand Groups

Subcommand Group	Description
`platform console`	Manages access to the platform serial console.
`platform os state`	Manages the current state of the operating system (OS).
`platform power state`	Manages the state of the platform power.
`platform get hostname`	Displays the host name of the current primary platform.
`platform get product id`	Displays the product ID for the current system.

Note – Every command returns a return code upon completion.

Platform Console Subcommands

The subcommands listed in TABLE F-2 allow you to manage access to the platform serial console.

TABLE F-2 `Platform Console` Subcommands

Subcommand	Description
`platform console`	Provides access to the platform serial console.
`platform get console`	Retrieves the configuration of the Service Processor (SP) access to the platform serial console.
`platform set console`	Configures the SP access to the platform serial console.

Platform Console Subcommand

Description: For remote-management capability, this command provides access to the platform serial console. Used in conjunction with the subcommand `platform set console` and the appropriate BIOS/platform OS settings, this command enables you to view the platform serial console while logged in to the SP.

Format

Command format:

```
platform console
```

You must configure the BIOS settings using the BIOS Setup utility. To refresh the BIOS Setup screen, press **Control-R**. Choose the **Advanced** tab to set the configuration.

TABLE F-3 lists common COM1 values. TABLE F-4 lists common values for console redirection.

TABLE F-3 Common COM1 Values

I/O Device Configuration	
Serial port A	Enabled
Base I/O address	3F8
Interrupt	IRQ 4

TABLE F-4 Common Values for Console Redirection

Console Redirection	
Com Port Address	On-board COM A
Console connection	Direct
Baud Rate	19.2K
Flow Control	None
Console Type	ANSI

Note – You can change these values, as long as they are the same as serial-port values for the operating system (OS). If your operating system supports the COM2-4 values, you can set these for the BIOS settings.

The serial-console settings in the platform OS should be set to match the BIOS settings.

Enter the following while you are connected to the console:

`^Ec` *`character`*

where `^E` represents **Control-E** and *`character`* is one of the entries in TABLE F-5:

TABLE F-5 Serial-Console Values

Character	Function
`.`	Disconnects an attach read/write.
`b`	Sends a broadcast message.
`c`	Toggles flow control.
`d`	Takes down a console.
`e`	Changes the escape sequence.

TABLE F-5 Serial-Console Values

Character	Function
`f`	Forces an attach read/write.
`g`	Groups information.
`i`	Information dump.
`L`	Toggles logging on/off.
`l?`	Breaks the sequence list.
`l0`	Sends a break per configuration file.
`l1-9`	Sends a specific break sequence.
`o`	Re-opens the tty and log file.
`p`	Replays the last 60 lines.
`r`	Replays the last 20 lines.
`s`	Spy read only.
`u`	Shows the host status.
`v`	Shows the version information.
`w`	Shows who is logged on to this console.
`x`	Shows the console baud information.
`z`	Suspends the connection.
`<cr>`	Ignores/aborts the command.
`?`	Prints this message.
`^R`	Replays the last line.
`\ooo`	Sends the character by octal code.

Under certain circumstances, it might be necessary to send a serial-break sequence to the platform OS (for example, to simulate the SysRq key when CONFIG_MAGIC_SYSRQ is defined and enabled in a Linux kernel).

To perform this operation, use the following sequence:

```
^Ecl0
```

(**Control-E**, followed by the lowercase letter "C", the lowercase letter "L" and the digit "0".)

The `platform console` command responds by displaying the string `[halt sent]`, confirming that the break sequence has been generated.

In the event that console output becomes corrupted, `^Ecd ^Eco` usually restores proper operation; this problem is normally due to flow-control issues.

Example

The following example lists the steps you would perform to enable and run the platform console:

1. **Check or set the BIOS settings.**
2. **Run the command:**

   ```
   platform set console -s sp -S 19200 -e
   ```

3. **Run the command:**

   ```
   platform set console
   ```

Return Codes

TABLE F-6 lists the return codes for this subcommand.

TABLE F-6 Return Codes for Subcommand `platform console`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.

Platform Get Console Subcommand

Description: Retrieves the configuration information regarding the Service Processor (SP) access to the platform serial console.

Format

Command format:

```
platform get console [{-H|--noheader}] [{-D | --delim <DELIMITER>}]
```

TABLE F-7 lists the arguments for this subcommand.

TABLE F-7 Arguments for Subcommand `platform get console`

Arguments	Description
`{-H |--noheader}`	Suppresses column headers.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

The following output displays when successful:

```
Rear Panel        Enabled Speed    Pruning     Log Trigger
SP Console        Yes     115200   No          1024KB
```

or

```
Platform COMA     No      19200    Yes         64KB
```

One of the other lines of data displays, depending on whether the rear-panel serial port is connected to the platform or to the SP. See TABLE F-8.

TABLE F-8 Supplementary Output

Column	Description
`Enabled`	Displays `No` if the external serial port is connected to the platform. Otherwise, the external serial port is connected to the SP console; you can access the platform serial console through the SP command line by running the subcommand `platform console`.

TABLE F-8 Supplementary Output

Column	Description
`Speed`	Indicates the communications speed of the link.
`Prune`	Indicates whether ANSI escape code and duplicate information pruning is enabled.
`Log Trigger`	Indicates the approximate size at which log rotation occurs (for example, when the file `console.0` is removed, the current log is moved to `console.0` and a new log file is opened). Pruning of log-file contents happens only when rotation occurs. The minimum size for a log file is 64KB; the maximum size is 1024KB.

Return Codes

TABLE F-9 lists the return codes for this subcommand.

TABLE F-9 Return Codes for Subcommand `platform get console`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Platform Set Console

Description: Enables the configuration of SP access to the platform serial console, sets the speed of the connection and limits the size of the log files created.

Format

Command format:

```
platform set console {--serial|-s} platform
```

This option configures the external serial port so that it is connected to the platform serial console. This is the default setting.

```
platform set console {--serial|-s} sp
[{{--enable|-e}|{--disable|-d}}]
[{{--prune|-p}|{--noprune|-n}}] [{--speed|-S}
{1200|2400|4800|9600|19200|38400|115200}] [{--log|-l} size]
```

This option configures the external serial port so that it is connected to the SP serial console. You can then access the platform serial console through the SP command line by running the subcommand `platform console`.

TABLE F-10 lists the arguments for this subcommand.

Note – If `-s` is set to platform, none of the following arguments can be used.

TABLE F-10 Arguments for Subcommand `platform set console`

Arguments	Description
`{-S | --speed} {1200|2400|4800|9600|19200|38400|115200}`	Select the port speed for the platform console. BIOS, the platform OS and the console must all be configured for the same speed.
`{-d | --disable}`	Indicates that the platform console monitor is inactive. Cannot be used with: `-e`.
`{-e | --enable}`	Indicates that the platform console monitor is active. Cannot be used with: `-d`.
`{-l | --log} size`	Select the trigger size in KB for console log rotation. The acceptable values for log size are between 64 and 1024 inclusive.

TABLE F-10 Arguments for Subcommand `platform set console`

Arguments	Description
`{-n | --noprune}`	Indicates that the platform console log should be the raw console data. Cannot be used with: `-p`.
`{-p | --prune}`	Indicates that the platform console log is to be cleaned of ANSI sequences and pruned of duplicated information. Cannot be used with: `-n`.

`{-s | --serial}`
`{sp|platform}`
	Specify whether the serial port is connected to the platform COMA port, or the SP serial console.
	Cannot be used with: `-e` [platform] `-d` [platform] `-p` [platform] `-n` [platform] `-S` [platform] `-l` [platform].

Return Codes

TABLE F-11 lists the return codes for this subcommand.

TABLE F-11 Return Codes for Subcommand `platform set console`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_DeviceError`	25	Unable to read or write to the device.

Platform OS State Subcommands

The subcommands listed in TABLE F-12 allow you to manage the operating system (OS).

TABLE F-12 `Platform OS State` Subcommands

Subcommand	Description
`platform get os state`	Retrieves the current state of the platform OS (for example, running, booting, off and so on).
`platform set os state`	Reboots the platform into the default OS, BIOS setup or BIOS update, or shuts down the platform.
`platform set os state boot`	Serves as an alias for the subcommand `platform set os state reboot` and only functions when the platform power state is off.

The subcommand `platform set os state reboot` causes the platform to turn on and boot the OS if the platform is off, but reboots the OS if the platform is on.

The subcommand `platform set os state` waits for the platform to boot; the subcommand `platform set power state` only waits for the power to come on.

Platform Get OS State Subcommand

Description: Retrieves the current state of the platform OS.

Format

Command format:

```
platform get os state
```

The values for the current state include:

- Off
- On
- Communicating
- Diagnostics
- Sleeping
- BIOS booting

- BIOS setup
- OS booting
- OS shutting down

When the platform is in the *Communicating* state (in which the OS is communicating with the SP), if the platform drivers are uninstalled, the SP remains in the *Communicating* state even though it can no longer communicate with the platform.

Refer to “Platform Set OS State Subcommand” on page 148 for more information about setting the state.

Return Codes

TABLE F-13 lists the return codes for this subcommand.

TABLE F-13 Return codes for Subcommand `platform get os state`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Platform Set OS State Subcommand

Description: Provides the ability to reboot the platform into the default OS, BIOS setup or BIOS update, or to shut down the platform. Rebooting to BIOS setup allows you to configure the BIOS parameters while BIOS update allows you to reflash the BIOS image.

Format

Command format:

```
platform set os state reboot [{-W | --nowait}] [{-b | --bios}]
[{-f|--forced}] [-q | --quiet]
```

```
platform set os state reboot-to-diags [{-f |--forced}] [START | STOP]
```

```
platform set os state shutdown[{-W | --nowait}] [{-f |--forced}]
[-q | --quiet]
```

```
platform set os state update-bios [-q | --quiet]
[{-W | --nowait}] BIOS_IMAGE
```

TABLE F-14 lists the arguments for this subcommand.

TABLE F-14 Arguments for Subcommand `platform set os state`

Arguments	Description
`[-W	--nowait]`	If specified, the subcommand returns immediately instead of waiting for the operation to complete.
`{-f	--forced}`	Results in a hard power off.
`[-b	--bios]`	Only applicable to the subcommand `platform set os state reboot`. Takes you to the BIOS Setup utility.
`[-q	--quiet]`	Suppresses interative warning messages. No error messages are blocked.
`update-bios`	Command option for flash updating the BIOS image.
`BIOS_IMAGE`	Only applicable to the subcommand `set os state update-bios`. Indicates the name of the file containing the new BIOS image to use when updating the BIOS.

The subcommand `platform set os state` waits for the platform to boot; the subcommand `platform set power state` only waits for the power to come on.

The subcommand `platform set os state reboot` causes the platform to turn on and boot the OS if the platform is off, but reboots the OS if the platform is on.

When the platform is in the *Communicating* state (in which the OS is communicating with the SP), if the platform drivers are uninstalled, the SP remains in the *Communicating* state even though it can no longer communicate with the platform.

Refer to "Platform Get OS State Subcommand" on page 146 for a list of possible states.

Return Codes

TABLE F-15 lists the return codes for this subcommand.

TABLE F-15 Return Codes for Subcommand `platform set os state`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Platform Set OS State Boot Subcommand

Description: This command serves as an alias for the subcommand **`platform set os state reboot`** and only functions when the platform power state is off.

Format

Command format:

```
platform set os state boot [{-f |--forced}] {[-b | --bios]}
[-q | --quiet] [-W | --nowait]
```

TABLE F-16 lists the arguments for this subcommand.

TABLE F-16 Arguments for Subcommand `platform set os state boot`

Arguments	Description
`{-f | --forced}`	Results in a hard power off. This option is ignored.
`[-b | --bios]`	Allows you to reflash the BIOS image.
`[-q | --quiet]`	Suppresses interative warning messages. No error messages are blocked.
`[-W | --nowait]`	If specified, the command returns immediately instead of waiting for the operation to complete.

Return Codes

TABLE F-17 lists the return codes for this subcommand.

TABLE F-17 Return Codes for Subcommand `platform set os state boot`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

Platform Power State Subcommands

The subcommands listed in TABLE F-18 allow you to manage the platform power.

TABLE F-18 `Platform Power State` Subcommands

Subcommand	Description
`platform get power state`	Provides the ability to determine the platform power state (for example, whether it is on or off).
`platform set power state`	Provides the ability to turn the platform power on or off.

The subcommand `platform set power state` does not affect the platform if the platform is already on; if the platform is off, it powers on and boots the OS. In other words, the subcommand `platform set power state` ensures that the platform is on, but does not reboot it if it is not on.

The subcommand `platform set os state` waits for the platform to boot; the subcommand `platform set power state` only waits for the power to come on.

Platform Get Power State Subcommand

Description: Provides the ability to determine the platform power state from within a script (whether the platform is on or off).

Format

Command format:

```
platform get power state
```

Return Codes

TABLE F-19 lists the return codes for this subcommand.

TABLE F-19 Return codes for Subcommand `platform get power state`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Platform Set Power State Subcommand

Description: Provides the ability to turn the platform power on or off from within a script. However, there are equivalent, less-destructive commands available. This command does not notify the platform OS of the request through the supplied channels.

Format

Command format:

```
platform set power state [{-W|--nowait}] [{-f|--forced}]
[{-t|--timeout} TIME] {off|on|cycle}
```

TABLE F-20 lists the arguments for this subcommand.

TABLE F-20 Arguments for Subcommand `platform set power state`

Arguments	Description
`{-W |--nowait}`	If specified, the command returns immediately instead of waiting for the operation to complete.
`{-f | --forced}`	Results in a hard power off.
`{-t |--timeout}`	Specifies the maximum time to wait for the operation to complete (in seconds).
`{off | on | cycle}`	Specifies whether to turn the platform power on or off or to cycle. Specifying the cycle argument causes platform power to be turned off, then on.

The subcommand `platform set power state` does not affect the platform if the platform is already on; if the platform is off, it powers on and boots the OS. In other words, the subcommand `platform set power state` ensures that the platform is on, but does not reboot it if it is not on.

The subcommand `platform set os state` waits for the platform to boot; the subcommand `platform set power state` only waits for the power to come on.

Return Codes

TABLE F-21 lists the return codes for this subcommand.

TABLE F-21 Return Codes for Subcommand `platform set power state`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_MissingArgument`	7	Missing argument(s).
`NWSE_TimedOut`	23	Operation timed out.

Platform Get Hostname Subcommand

Description: Displays the host name of the current primary platform. The data is refreshed only when the platform is rebooted.

Format

Command format:

```
platform get hostname [{-H|--noheader}]
```

TABLE F-22 lists the argument for this subcommand.

TABLE F-22 Argument for Subcommand `platform get hostname`

Arguments	Description
`{-H | --noheader}`	Suppresses column headers.

Return Codes

TABLE F-23 lists the return codes for this subcommand.

TABLE F-23 Return Codes for Subcommand `platform get hostname`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.

Platform Get Product ID Subcommand

Description: Displays the product ID for the current system.

Format

Command format:

```
platform get product-id
```

Note – You can also retrieve the product ID, board revision number and PRS revision number by running the subcommands `sensor get` and `inventory get hardware`.

Return Codes

TABLE F-24 lists the return codes for this subcommand.

TABLE F-24 Return Codes for Subcommand `platform get product-id`

Return Code	ID	Description
NWSE_Success	0	Command successfully completed.
NWSE_InvalidUsage	1	Invalid usage: bad parameter usage, conflicting options specified.
NWSE_NotFound	5	Entity (user, service, file, path or other) was not found.
NWSE_FileError	18	File open, file missing, or a read or write error occurred.
NWSE_ServiceNotAvailable	24	Requested service is not available.

APPENDIX **G**

Sensor Commands

The `sensor` command reports or sets the value of an environmental sensor or control.

TABLE G-1 lists the groups of `sensor` subcommands.

TABLE G-1 Sensor Subcommand Groups

Subcommand Group	Description
`sensor get`	Returns all data associated with a sensor.
`sensor set`	Sets some of the data associated with a specific sensor or a class of sensors.

Note – Every command returns a return code upon completion.

Note – There are some sensors whose value does not change, some that are there to provide information in the event of a problem, and others to facilitate the proper operation of the software.

Many of these sensors do not have a related component (parent) associated with them. For example, the die-temperature sensor for a CPU has the CPU as its parent component and a fan-speed sensor has the fan as its parent component; the product-id sensor, however, only reports a static value and has no parent relationship.

This relationship establishes the component(s) which is affected by changes in the value of the sensor. You cannot modify the thresholds for sensors without a parent relationship since an event will never occur for these threshold crossings.

Sensor Get Subcommand

Description: Returns all data associated with a sensor.

By default, only the sensor ID and its current value are displayed. You can specify on the command line the order of the data output.

Note – The *identifier* field is always displayed first, unless you suppress it with the `-I` option.

Format

Command format:

```
sensor get [{{-i | --id} ID | {-t | --type} TYPE_ID}]
[{-v | --value}] [{-n | --nominal}]
[{-C | --crithigh}] [{-c | --critlow}]
[{-W | -warnhigh}] [{-w | --warnlow}]
[{-N | --name}] [{-d | --description}]
[{-S | --sensor-type}] [{-p | --parent-comp}]
[{-s | --severity}] | [{--verbose}]
[{-I | --noid}] [{-H | noheader}]
[{-D | --delim <DELIMITER>}]
```

TABLE G-2 lists the arguments for this subcommand.

TABLE G-2 Arguments for Subcommand `sensor get`

Arguments	Description
`{-i | --id}`	SENSOR_ID, PRODUCT-ID, BOARD-REVISION, PRS-REVISION Specifies the sensor for which the data is desired. You can specify this argument multiple times, in which case the sensor data is reported in the order specified. You can also retrieve the product ID, board-revision number and PRS revision number using this flag. Specify `[-vIH]` following the ID to convert the output to the appropriate product ID. For example, product ID 255 indicates the 2100 server and product ID 239 indicates the 4300 server. You can also obtain this information using the `inventory get hardware` command.
`{-t | --type}`	Specifies the sensor class for which the data is desired. You can specify this argument multiple times, in which case the sensor output is grouped by type in the order specified. Current sensor classes are voltage, fan, temperature, current, power and switch.
`{-v | --value}`	Displays the current value. of the sensor.
`{-n | --nominal}`	Displays the nominal value of the sensor.
`{-C | --crithigh}`	Displays the *critical high* threshold value for the sensor. Thresholds configured to a value other than the factory value display with a trailing asterisk (`*`) character.
`{-c | --critlow}`	Displays the *critical low* threshold value for the sensor.
`{-W | -warnhigh}`	Displays the *warning high* threshold value for the sensor.
`{-w | --warnlow}`	Displays the *warning low* threshold value for the sensor.
`{-N | --name}`	Displays the name of the sensor.
`{-d | --description}`	Displays a description of the sensor.
`{-S | --sensor-type}`	Displays the type of sensor (for use with `--type`).
`{-p | --parent-comp}`	Displays the parent component list for the sensor. These are the components that are affected by changes in the value of a sensor (for example, the components that change severity as the sensor changes severity).
`{-s | --severity}`	Displays the current severity lever of the sensor (nominal, warning or critical).
`{--verbose}`	Displays all columns; you cannot use this argument with any of the other column addition options.

TABLE G-2 Arguments for Subcommand `sensor get`

Arguments	Description
`{-I | --noid}`	Suppresses the display of the sensor ID column. By default, this column always displays when more than one sensor is selected.
`[-H | --noheader]`	Suppresses the column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE G-3 lists the arguments for this subcommand.

TABLE G-3 Return Codes for Subcommand `sensor get`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Note – There are some sensors whose value does not change, some that are there to provide information in the event of a problem, and others to facilitate the proper operation of the software.

Many of these sensors do not have a related component (parent) associated with them. For example, the die-temperature sensor for a CPU has the CPU as its parent component and a fan-speed sensor has the fan as its parent component; the product-id sensor, however, only reports a static value and has no parent relationship.

This relationship establishes the component(s) which is affected by changes in the value of the sensor. You cannot modify the thresholds for sensors without a parent relationship since an event will never occur for these threshold crossings.

Sensor Set Subcommand

Description: Allows you to set some of the data associated with a specific sensor or a class of sensors.

Format

Command format:

```
sensor set [{-i | --id} SENSOR_ID [{-i | --id} SENSOR_ID] …]
{[[{-C | --crithigh} VALUE] [[{-c | --critlow} VALUE]
[{-W | --warnhigh] VALUE] [{-w | --warnlow} VALUE] [{-v | --value}
{on|off}] | {-r | --reset}}
```

```
sensor set [{-t | --type] TYPE_ID] {[[{-C | --crithigh} VALUE]
[[{-c| --critlow} VALUE] [{-W | --warnhigh] VALUE] [{-w | --warnlow} VALUE]
[{-v | --value} {on|off}] | {-r | --reset}}
```

```
sensor set [{-R | --resetall}]
```

TABLE G-4 lists the arguments for this subcommand.

TABLE G-4 Arguments for Subcommand `sensor set`

Arguments	Description
`{-i | --id}`	Specifies the specific sensor on which to operate. You can specify multiple sensors by repeating `--id`.
`{-t | --type}`	Specifies the specific sensor class on which to operate (for example, fan, voltage and so on).
`{-C | --crithigh}`	Specifies the *critical high* threshold value for the sensor. • Setting the string to `clear` disables the threshold. • Setting the string to `reset` sets the value to the original factory-specified value. • If the value specified ends in a percent sign (%), the threshold is set to that percentage of the nominal value for the sensor. • Any other value is interpreted as the actual value to which to set the threshold.
`{-c | --critlow}`	Specifies the *critical low* threshold value for the sensor. Setting the string to `clear` disables the threshold.
`{-W | -warnhigh}`	Specifies the *warning high* threshold value for the sensor. Setting the string to `clear` disables the threshold.
`{-w | --warnlow}`	Specifies the *warning low* threshold value for the sensor. Setting the string to `clear` disables the threshold.

TABLE G-4 Arguments for Subcommand `sensor set`

Arguments	Description
`{-v | --value}`	Sets the value of the sensor.
`{-r | --reset}`	Resets all thresholds for the specified sensor(s) to the factory defaults.
`{-R | --resetall}`	Resets all thresholds for all sensors to the factory defaults.

Return Codes

TABLE G-5 lists the arguments for this subcommand.

TABLE G-5 Return Codes for Subcommand `sensor set`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

Note – There are some sensors whose value does not change, some that are there to provide information in the event of a problem, and others to facilitate the proper operation of the software.

Many of these sensors do not have a related component (parent) associated with them. For example, the die-temperature sensor for a CPU has the CPU as its parent component and a fan-speed sensor has the fan as its parent component; the product-id sensor, however, only reports a static value and has no parent relationship.

This relationship establishes the component(s) which is affected by changes in the value of the sensor. You cannot modify the thresholds for sensors without a parent relationship since an event will never occur for these threshold crossings.

APPENDIX **H**

Service Processor Commands

The sp command gets or sets the configuration values for the Service Processor (SP), generates or manages events and notices; or adds or modifies subscribers, event routes and email-notification groups for the SP event manager.

TABLE H-1 lists the groups of sp subcommands.

Note – Every command returns a return code upon completion.

TABLE H-1 Service Processor Subcommand Groups

Subcommand	Description
`Date`	Sets or retrieves the date and time on the SP RTC.
`DNS`	Displays or configures the DNS client configuration on the SP.
`Events`	Returns detailed information or clears an event.
`Hostname`	Displays or resets the host name or domain name of the SP.
`IP`	Sets, modifies or retrieves the SP network configuration.
`JNET Address`	Sets or retrieves the jnet address.
`Locate Light`	Sets the state or reads the value of the locatelight switch.
`Logfile`	Retrieves or configures the event log file.
`Miscellaneous`	Reads status for a component, retrieves the last port 80 postcode, restores settings to defaults, stores data in tar zipped format, or captures debug data.
`Mount`	Displays, creates, resets or deletes a mount point.
`SMTP`	Manages information about SMTP email delivery.

TABLE H-1 Service Processor Subcommand Groups

Subcommand	Description
`SNMP`	Manages SNMP functions.
`SSL`	Manages SSL capabilities.
`Update Flash`	Sets the update flag to start the full flash update or copies the Value-Add file to the Value-Add component of the SP flash.

SP Date Subcommands

The subcommands in TABLE H-2 manage the date and time on the SP.

TABLE H-2 `SP Date` Subcommands

Subcommand	Description
`sp get date`	Retrieves the date and time from the SP RTC.
`sp set date`	Sets the date and time on the SP RTC.

SP Get Date Subcommand

Description: Retrieves the date and time from the SP RTC.

Format

Command format:

```
sp get date
```

Return Codes

TABLE H-3 lists the return codes for this subcommand.

TABLE H-3 Return Codes for Subcommand `sp get date`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.

SP Set Date Subcommand

Description: Sets the date and time on the SP RTC.

Format

Command format:

```
sp set date DATE_STRING
```

TABLE H-4 lists the argument for this subcommand.

TABLE H-4 Argument for Subcommand `sp set date`

Arguments	Description
`DATE STRING`	Specifies the date and time on the Service Processor RTC. The date string is a UTC date of the form YYYY-MM-DD HH:MM:SS.

You can use this command to initially set the platform RTC after the platform has lost CMOS backup power. If the platform is in the state in which the operating system (OS) is communicating with the SP, the platform time will override the SP time, which allows the platform and sp event times to be in sync in the event log.

Return Codes

TABLE H-5 lists the return codes for this command.

TABLE H-5 Return Codes for Subcommand `sp set date`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing or a read or write error occurred.

SP DNS Subcommands

The subcommands in TABLE H-6 manage the DNS configuration on the SP.

TABLE H-6 `SP DNS` Subcommands

Subcommand	Description
`sp disable dns`	Disables the DNS configuration on the SP.
`sp enable dns`	Configures the DNS configuration on the SP.
`sp get dns`	Displays the current DNS configuration on the SP.

SP Disable DNS Subcommand

Description: Disables the DNS configuration on the SP.

```
sp disable dns
```

When the SP is configured to use Dynamic Host Control Protocol (DHCP), DHCP automatically configures DNS settings. Changes to the DNS settings in this configuration can be replaced with the DHCP client.

Return Codes

TABLE H-7 lists the return codes for this command:

TABLE H-7 Return Codes for Subcommand `sp disable dns`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP Enable DNS Subcommand

Description: Configures the DNS configuration on the SP.

Because applications do not see updated DNS resolver configurations (in `/etc/resolv.conf`) until they are restarted, this command restarts server processes that depend on DNS. This currently includes the `sshd` daemon and the Security Manager.

Format

Command format:

```
sp enable dns { -n | --nameserver} NAMESERVER IP…
{-s | --searchdomain } SEARCH DOMAIN…
```

TABLE H-8 lists the arguments for this subcommand.

TABLE H-8 Arguments for Subcommand `sp enable dns`

Argument	Description
`{ -n	--nameserver}`	Displays the nameserver IP-addresses. If there is more than one, the addresses print on separate lines.
`{-s	--searchdomain }`	Displays the search domain(s). If there is more than one, the search domains print on separate lines.

Return Codes

TABLE H-9 lists the return codes for this subcommand.

TABLE H-9 Return Codes for Subcommand `sp enable dns`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP Get DNS Subcommand

Description: Displays the current DNS configuration on the SP.

Format

Command format:

```
sp get dns [{-n | --nameserver } | -s | --searchdomain } |
{-H | --noheader }] [{-D | --delim <DELIMITER>}]
```

TABLE H-10 lists the arguments for this subcommand.

TABLE H-10 Arguments for Subcommand `sp get dns`

Argument	Description
`{ -n | --nameserver }`	Displays the name server(s). If there is more than one nameserver, they print on separate lines.
`{ -s | --searchdomain }`	Displays the searchdomain(s). If there is more than one searchdomain, they print on separate lines.
`{ -H | --noheader }`	Suppresses column headings.
`[{-D | --delim <DELIMITER>}]`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-11 lists the return codes for this subcommand.

TABLE H-11 Return Codes for Subcommand `sp get dns`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.

SP Events Subcommands

The subcommands in TABLE H-12 manage events on the SP.

TABLE H-12 `SP Events` Subcommands

Subcommand	Description
`sp delete event`	Clears an existing event using the event ID.
`sp get events`	Returns detailed information about all active SP events.

SP Delete Event Subcommand

Description: Clears an existing event using the event ID.

Format

Command format:

```
sp delete event { EVENT ID | {-a | --all}} [-q | --quiet]
```

TABLE H-13 lists the arguments for this subcommand.

TABLE H-13 Arguments for Subcommand `sp delete event`

Argument	Description
`EVENT ID`	Specifies the existing event to clear. This argument is repeatable to clear multiple events at one time.
`[-a | --all]`	Removes all events.
`[-q | --quiet]`	If the event to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE H-14 lists the return codes for this subcommand.

TABLE H-14 Return Codes for Subcommand `sp delete event`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_InvalidOpForState`	22	Invalid operation for current state.

SP Get Events Subcommand

Description: Returns detailed information about all active SP events. By default, event ID, last update, component, severity and a message are displayed.

Administrators can view detailed information about all the currently active system events and perform various actions related to each event.

You can view this information in the System Events table, which contains a row for each unique active system event, or using this command. For a list of all possible events, refer to the TABLE 3-4 in Chapter 3.

Format

Command format:

```
sp get events [ {-i | --id} <EVENT ID> ] [{-d | --detail} ]
[{ -v | --verbose}] [{-H | noheader}][{-D | --delim <DELIMITER>}]
```

TABLE H-15 lists the arguments for this subcommand.

TABLE H-15 Arguments for Subcommand `sp get events`

Argument	Description
`{-i | --id}`	Specifies to display only information about this event; otherwise information for all existing events returns.
`{-d | --detail}`	Specifies to display the history of either one or all events.
`{ -v| --verbose}`	Specifies to display all columns.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Specifies to delimit columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-16 lists the return codes for this subcommand.

TABLE H-16 Return Codes for Subcommand `sp get events`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) not found.
`NWSE_NoMemory`	8	Insufficient memory.

SP Hostname Subcommands

The subcommands in TABLE H-17 manage the SP host and domain.

TABLE H-17 `SP Hostname` Subcommands

Subcommand	Description
`sp get hostname`	Displays the current host name and optionally the domain name of the SP.
`sp set hostname`	Resets the host name or domain name of the SP to the specified name.

SP Get Hostname Subcommand

Description: Displays the current host name and optionally the domain name of the SP. This name is used by many of the networking programs to identify the machine. It is also used to identify a logging subdirectory for event logs.

Format

Command format:

```
sp get hostname [-f | --fqdn]
```

TABLE H-18 lists the argument for this subcommand.

TABLE H-18 Argument for Subcommand `sp get hostname`

Argument	Description
`[-f | --fqdn]`	Causes the fully qualified hostname to display.

Return Codes

TABLE H-19 lists the return codes for this subcommand.

TABLE H-19 Return Codes for Subcommand `sp get hostname`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.

SP Set Hostname Subcommand

Description: Resets the host name or domain name of the SP to the specified name. This name is used by many of the networking programs to identify the machine.

Format

Command format:

```
sp set hostname HOSTNAME
```

TABLE H-20 lists the argument for this subcommand.

TABLE H-20 Argument for Subcommand `sp set hostname`

Argument	Description
`HOSTNAME`	Specifies the name of the host to set.

Return Codes

TABLE H-21 lists the return codes for this subcommand.

TABLE H-21 Return Codes for Subcommand `sp set hostname`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.

SP IP Subcommands

The subcommands in TABLE H-22 manage the SP network configuration.

TABLE H-22 `SP IP` Subcommands

Subcommand	Description
`sp get ip`	Retrieves the ethernet-based network configuration information for the SP.
`sp set ip`	Sets or modifies the SP network configuration.

SP Get IP Subcommand

Description: Retrieves the ethernet-based network-configuration information for the SP, including IP address, network mask and gateway. In addition, it indicates whether the SP is configured to use DHCP or a static IP address.

Format

Command format:

```
sp get ip [-H | noheader] [{-D | --delim <DELIMITER>}]
```

TABLE H-23 lists the arguments for this subcommand.

TABLE H-23 Arguments for Subcommand `sp get ip`

Argument	Description
`{ -H	--noheader }`	Suppresses column headings.
`{ -D	--delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-24 lists the arguments for this subcommand.

TABLE H-24 Return Codes for Subcommand `sp get ip`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.

SP Set IP Subcommand

Description: Sets or modifies the SP network configuration.

Format

Command format:

```
sp set ip dhcp [--nowait]
```

```
sp set ip static {-i | --ipaddress} IP_ADDRESS
[{-n | --netmask} NETMASK] [{-g | --gateway} GATEWAY}] [-w | --nowait]
```

TABLE H-25 lists the arguments for this subcommand.

TABLE H-25 Arguments for Subcommand `sp set ip`

Argument	Description
`{-i | --ipaddress}`	Specifies the IP address you wish to set.
`{-n | --netmask}`	Specifies the netmask; the default value is 255.255.255.0.
`{-g | --gateway}`	Specifies the gateway; the default value is the existing gateway.
`{-w | --nowait}`	If you specify the `-nowait` option, loss of connectivity will occur some time after the command returns. If you do not specify the `-nowait` option, your connections to the SP will be lost before the command returns.

Return Codes

TABLE H-26 lists the return codes for this subcommand.

TABLE H-26 Return Codes for Subcommand `sp set ip`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_GatewayOffNet`	16	Gateway address is not on network.
`NWSE_NetMaskIncorrect`	17	An inappropriate netmask was specified.

SP JNET Address Subcommands

The JNET address is used for communications between the SP and the platform. The subcommands in TABLE H-27 manage the SP JNET address.

TABLE H-27 `SP JNET` Subcommands

Subcommand	Description
`sp get jnet`	Retrieves the JNET address.
`sp set jnet`	Sets the JNET address.

SP Get JNET Subcommand

Description: Retrieves the IP address of the platform JNET driver.

Format

Command format:

```
sp get jnet [{-H | --noheader}] [{-D | --delim <DELIMITER>}]
```

TABLE H-28 lists the arguments for this subcommand.

TABLE H-28 Arguments for Subcommand `sp get jnet`

Argument	Description
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-29 lists the return codes for this subcommand.

TABLE H-29 Return Codes for Subcommand `sp get jnet`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_HostDown`	14	Host is down.

SP Set JNET Subcommand

Description: Sets or modifies the SP and platform network addresses for JNET. Because of the firewall between these drivers, you must specify both addresses at the same time.

Both the SP and Platform JNET addresses must be on the same Class C subnet.

Format

Command format:

```
sp set jnet {-p | --platform} IP ADDRESS {-s | --sp} IP ADDRESS
```

TABLE H-30 lists the arguments for this subcommand.

TABLE H-30 Arguments for Subcommand `sp set jnet`

Argument	Description
`{-p | --platform}`	Specifies the IP address for the platform.
`{-s | --sp}`	Specifies the IP address for the SP.

Note – If you change the default addresses of JNET using this command and then re-install the platform operating system or reset the SP through the subcommand `sp reset to default-settings`, you must re-issue the subcommand `sp set jnet` to re-establish the JNET connection.

Otherwise, the connection will be out-of-sync (one address will be modified and one will be re-set to the default address.)

Return Codes

TABLE H-31 lists the return codes for this subcommand.

TABLE H-31 Return Codes for Subcommand `sp set jnet`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_HostDown`	14	Host is down.

SP Locate Light Subcommands

The subcommands in TABLE H-32 manage the locatelight switch.

TABLE H-32 `SP Locatelight` Subcommands

Subcommand	Description
`sp get locatelight`	Reads the value of the locatelight switch (which represents the state of the front and rear panel identification lights).
`sp set locatelight`	Sets the state of the locatelight switch.

SP Get Locatelight Subcommand

Description: Reads the value of the locatelight switch (which represents the state of the front and rear panel identification lights). The possible states are blinking or off.

Format

Command format:

```
sp get locatelight
```

Return Codes

TABLE H-33 lists the return codes for this subcommand.

TABLE H-33 Return Codes for Subcommand `sp get locatelight`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP Set Locatelight Subcommand

Description: Sets the state of the locatelight switch (which describes the state of the front and rear panel identification lights).

Format

Command format:

```
sp set locatelight {blink | off}
```

Return Codes

TABLE H-34 lists the return codes for this subcommand.

TABLE H-34 Return Codes for Subcommand `sp set locatelight`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP Logfile Subcommands

The subcommands in TABLE H-35 manage the SP log files.

TABLE H-35 `SP Logfile` Subcommands

Subcommand	Description
`sp get logfile`	Retrieves the event-log file configuration.
`sp set logfile`	Configures the event log file that is the destination of all Event Manager events and notices.

SP Get Logfile Subcommand

Description: Retrieves the event log file configuration.

Format

Command format:

```
sp get logfile [-H | --noheader] [{-D | --delim <DELIMITER>}]
```

TABLE H-36 lists the arguments for this subcommand.

TABLE H-36 Arguments for Subcommand `sp get logfile`

Argument	Description
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-37 lists the return codes for this subcommand.

TABLE H-37 Return Codes for Subcommand `sp get logfile`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoMemory`	8	Insufficient memory.

SP Set Logfile Subcommand

Description: Configures the event log file that is the destination of all Event Manager events and notices.

Format

Command format:

```
sp set logfile [ {-f | --file} FILENAME] [ {-s | --size} SIZE]
```

You must specify the name of the file to which the Event Manager sends logs. When setting the log file using this command, specify only the name of the log file without the path. File names cannot contain the forward slash character (/), backward relative-path reference (..) or the less-than symbol (<).

TABLE H-38 lists the arguments for this subcommand.

TABLE H-38 Arguments for Subcommand `sp set logfile`

Argument	Description
`{-f | --file}`	Specifies the name of the file within the directory to which the Event Manager sends logs.
`{-s | --size}`	Specifies the size of the file in megabytes. A minimum size of 0.01 MB is required for this log file.

Return Codes

TABLE H-39 lists the arguments for this subcommand.

TABLE H-39 Return Codes for Subcommand `sp set logfile`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.

SP Miscellaneous Subcommands

The subcommands in TABLE H-40 manage miscellaneous SP functions.

TABLE H-40 Miscellaneous SP Subcommands

Subcommand	Description
`sp create test events`	Tests and validates different types of configurations you may be considering for the SP.
`sp get port 80`	Retrieves the last port 80 postcode from the PRS Port80 register.
`sp get status`	Returns the status of the overall system.
`sp get tdulog`	Captures data and stores it on the SP in compressed format.
`sp load settings`	Configures an SP with the same configuration as that of another Service Processor.
`sp reboot`	Restarts the SP.
`sp reset`	Restores selected settings of the SP to the default factory configuration.

SP Create Test Events Subcommand

Description: This command helps you test and validate different types of configurations that you might be considering for the SP (for example, configurations involving event forwarding, such SNMP, SMTP, log files or directory services).

Typically, you would have to wait for an event to be generated on the SP in order to validate that these configurations are working properly. However, using this command, you can generate test events that will be routed appropriately according to these configurations.

Format

Command format:

```
sp create test events
```

Return Codes

TABLE H-41 lists the return codes for this command.

TABLE H-41 Return Codes for Subcommand `sp create test events`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_NoMemory`	8	Insufficient memory.

SP Get Port 80 Subcommand

Description: Retrieves the last Port 80 post code from the PRS Port80 register. The register is written by platform BIOS during platform boot. This command is used to debug platform boot problems.

Format

Command format:

```
sp get port80 {-m | --monitor}
```

TABLE H-42 lists the arguments for this subcommand.

TABLE H-42 Arguments for Subcommand `sp get port80`

Argument	Description
`{-m | --monitor}`	Allows for continuous monitoring of the port 80 traffic.

You can also retrieve the last ten Port 80 post codes using the operator panel.

For more details about using the operator-panel menus, refer to the *Sun Fire V20z Server User Guide* (817-5248-xx).

See TABLE H-44 for a list of the Power On Self Test (POST) codes for the Phoenix BIOS.

See TABLE H-45 for a list of the boot block codes on Flash ROM.

Return Codes

TABLE H-43 lists the return codes for this subcommand.

TABLE H-43 Return Codes for Subcommand `sp get port80`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

BIOS POST Codes

TABLE H-44 lists the POST codes for the Phoenix BIOS.

TABLE H-44 BIOS POST Codes

POST Code	Description
02	Verify real mode
03	Disable non-maskable interrupt (NMI)
04	Get CPU type
06	Initialize system hardware
07	Disable shadow and execute code from the ROM
08	Initialize chipset with initial POST values
09	Set IN POST flag
0A	Initialize CPU registers
0B	Enable CPU cache
0C	Initialize caches to initial POST values
0E	Initialize I/O component
0F	Initialize the local bus IDE
10	Initialize power management
11	Load alternate registers with initial POST values
12	Restore CPU control word during warm boot

TABLE H-44 BIOS POST Codes

POST Code	Description
13	Initialize PCI bus mastering devices
14	Initialize keyboard controller
16	BIOS ROM checksum
17	Initialize cache before memory autosize
18	8254 programmable interrupt timer initialization
1A	8237 DMA controller initialization
1C	Reset programmable interrupt controller
20	Test DRAM refresh
22	Test 8742 keyboard controller
24	Set ES segment register to 4GB
26	Enable gate A20 line
28	Autosize DRAM
29	Initialize POST memory manager
2A	Clear 512KB base RAM
2C	RAM failure on address line xxxx
2E	RAM failure on data bits xxxx of low byte of memory bus
2F	Enable cache before system BIOS shadow
30	RAM failure on data bits xxxx of high byte of memory bus
32	Test CPU bus clock frequency
33	Initialize Phoenix Dispatch Manager
36	Warm start shut down
38	Shadow system BIOS ROM
3A	Autosize cache
3C	Advanced configuration of chipset registers
3D	Load alternate registers with CMOS values
41	Initialize extended memory for RomPilot
42	Initialize interrupt vectors
45	POST device initialization
46	Check ROM copyright notice
47	Initialize I20 support

TABLE H-44 BIOS POST Codes

POST Code	Description
48	Check video configuration against CMOS
49	Initialize PCI bus and devices
4A	Initialize all video adapters in system
4B	QuietBoot start (optional)
4C	Shadow video BIOS ROM
4E	Display BIOS copyright notice
4F	Initialize MultiBoot
50	Display CPU type and speed
51	Initialize EISA board
52	Test keyboard
54	Set key click if enabled
55	Enable USB devices
58	Test for unexpected interrupts
59	Initialize POST display service
5A	Display prompt “Press F2 to enter SETUP”
5B	Disable CPU cache
5C	Test RAM between 512KB and 640KB
60	Test extended memory
62	Test extended memory address lines
64	Jump to UserPatch1
66	Configure advanced cache registers
67	Initialize Multi Processor APIC
68	Enable external and CPU caches
69	Setup system management mode (SMM) area
6A	Display external L2 cache size
6B	Load custom defaults (optional)
6C	Display shadow area message
6E	Display possible high address for UMB recovery
70	Display error messages
72	Check for configuration errors

TABLE H-44 BIOS POST Codes

POST Code	Description
76	Check for keyboard errors
7C	Set up hardware interrupt vectors
7D	Initialize Intelligent System Monitoring
7E	Initialize coprocessor if present
80	Disable onboard super I/O ports and IRQs
81	Late POST device initialization
82	Detect and install external RS232 ports
83	Configure non-MCD IDE controllers
84	Detect and install external parallel ports
85	Initialize PC compatible PnP ISA devices
86	Reinitialize onboard I/O ports
87	Configure motherboard configurable devices (optional)
88	Initialize BIOS data area
89	Enable non-maskable interrupts (NMIs)
8A	Initialize extended BIOS data area
8B	Test and initialize PS/2 mouse
8C	Initialize floppy controller
8F	Determine number of ATA drives (optional)
90	Initialize hard disk controllers
91	Initialize local bus hard disk controllers
92	Jump to UserPatch2
93	Build MPTABLE for multi processor boards
95	Install CD ROM for boot
96	Clear huge ES segment register
97	Fixup multi processor table
98	Search for option ROMs
99	Check for SMART drive (optional)
9A	Shadow option ROMs
9C	Set up power management
9D	Initialize security engine (optional)

TABLE H-44 BIOS POST Codes

POST Code	Description
9E	Enable hardware interrupts
9F	Determine number of ATA and SCSI drives
A0	Set time of day
A2	Check key lock
A4	Initialize typematic rate
A8	Erase F2 prompt
AA	Scan for F2 key stroke
AC	Enter setup
AE	Clear boot flag
B0	Check for errors
B1	Inform RomPilot about the end of POST
B2	POST done - prepare to boot operating system
B4	One short beep
B5	Terminate QuietBoot (optional)
B6	Check password
B7	Initialize ACPI BIOS
B9	Prepare boot
BA	Initialize DMI parameters
BB	Initialize PnP option ROMs
BC	Clear parity checkers
BD	Display multiboot menu
BE	Clear screen
BF	Check virus and backup reminders
C0	Try to boot with interrupt 19
C1	Initialize POST Error Manager (PEM)
C2	Initialize error logging
C3	Initialize error display function
C4	Initialize system error handler
C5	PnP dual CMOS (optional)
C6	Initialize notebook docking (optional)

TABLE H-44 BIOS POST Codes

POST Code	Description
C7	Initialize notebook docking late
C8	Force check (optional)
C9	Extended checksum (optional)
CA	Redirect Int 15h to enable remote keyboard
CB	Redirect Int 13 to Memory Technologies Devices such as ROM, RAM, PCMCIA and serial disk
CC	Redirect Int 10h to enable remote serial video
CD	Re-map I/O and memory for PCMCIA
CE	Initialize digitizer and display message
D2	Unknown interrupt

Boot Block Codes for Flash ROM

TABLE H-45 lists the boot block codes in Flash ROM.

TABLE H-45 Boot Block Codes in Flash ROM

Post Code	Description
E0	Initialize the chipset
E1	Initialize the bridge
E2	Initialize the CPU
E3	Initialize the system timer
E4	Initialize system I/O
E5	Check force recovery boot
E6	Checksum BIOS ROM
E7	Go to BIOS
E8	Set Huge Segment
E9	Initialize Multi Processor
EA	Initialize OEM special code
EB	Initialize PIC and DMA
EC	Initialize Memory type
ED	Initialize Memory size

TABLE H-45 Boot Block Codes in Flash ROM

Post Code	Description
EE	Shadow Boot Block
EF	System memory test
F0	Initialize interrupt vectors
F1	Initialize Run Time Clock
F2	Initialize video
F3	Initialize System Management Manager
F4	Output one beep
F5	Clear Huge Segment
F6	Boot to mini DOS
F7	Boot to Full DOS

SP Load Settings Subcommand

Description: Configures an SP with the same configuration as that of another SP.

You can also perform autoconfiguration from the operator panel to perform this same function. For more information, see “Autoconfiguring the SP (Optional Method)” on page 24.

Format

Command format:

```
sp load settings {{ -s | --sp } SP_IP_OR_HOST [-H |--noheader]
```

TABLE H-46 lists the arguments for this subcommand.

TABLE H-46 Arguments for Subcommand `sp load settings`

Argument	Description
`{ -s | --sp }`	The IP address of the machine from which to copy the configuration information.
`[-H |--noheader]`	Suppresses header output.

Return Codes

TABLE H-47 lists the return codes for this subcommand.

TABLE H-47 Return Codes for Subcommand `sp load settings`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_HostDown`	14	Host is down.

SP Get Status Subcommand

Description: Returns the status of the overall system.

Format

Command format:

```
sp get status
```

TABLE H-48 lists the arguments for this subcommand.

TABLE H-48 Arguments for Subcommand `sp get status`

Argument	Description
`Nominal`	All components are operating within normal parameters.
`Warning`	One or more components are operating at warning levels.
`Critical`	One or more components are operating out of specification or have failed.

Return Codes

TABLE H-49 lists the return codes for this subcommand.

TABLE H-49 Return Codes for Subcommand `sp get status`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP Get TDULog Subcommand

Description: The Troubleshooting Dump Utility (TDU) captures debug data. When you execute this command, this data is gathered and stored on the Service Processor in a compressed tar file.

Format

Command format:

```
sp get tdulog [{-f | --filename} FILENAME or STDOUT ]
[{-c | --cpuregs} CPU REGISTERS]
[{-p | --pciregs} PCI REGISTERS]
[{-r | --reset} RESET PLATFORM]
```

TABLE H-50 lists the arguments for this subcommand.

TABLE H-50 Arguments for Subcommand `sp get tdulog`

Argument	**Description**
`{-f | --filename}`	*Optional*. The name of the output file to which the log files are copied, or the fully qualified path name. File names cannot contain the backward relative path reference (..) or the less than symbol (<). The following log files are created by default: `envLog`: contains the environment variables `vpdLog`: contains raw VPD data Additional log files are created for CPU2 and CPU3 registers. The TDU data can also be redirected to `stdout`. If the file name is `stdout`, the output is sent to `stdout` and the log files are not created. An NFS-mounted file share must be used to store the output file. If you do not provide a file name, it creates a file named `tdulog.tar` in `/logs/<`*hostname*`>`, where the <*hostname*> is the host name of the SP. If the host name is `localhost`, then the MAC address is used instead.
`{-c | --cpuregs}`	Reads the K-8 registers (GPRs, MSRs, TCB and machine check) from up to four CPUs.
`{-p | --pciregs}`	Reads all PCI registers on the system.
`{-r | --reset}`	Resets the platform if unable to access HDT mode.

The register name, address and data are logged to a file. For example, the information for CPU0 is shown in TABLE H-51.

TABLE H-51 Sample Information for Subcommand `sp get tdulog` on CPU0

Reg Name	**Reg Addr**	**Reg Data**
`MSR_MCG_CAP_MSR`	`0xc0020179`	`0x0000000000000105`
`MSR_MCG_STAT_MSR`	`0xc002017a`	`0x0000000000000000`
`MSR_MCG_CTL_MSR`	`0xc002017b`	`0x000000000000001F`
`MSR_MC0_CTL`	`0xc0020400`	`0x000000000000007F`

Return Codes

TABLE H-52 lists the return codes for this subcommand.

TABLE H-52 Return Codes for Subcommand `sp get tdulog`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_MissingArgument`	7	Missing argument(s).
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_NotMounted`	21	File system is not mounted.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Reboot Subcommand

Description: Restarts the SP. This command is useful in emergency situations in which you may not have physical access to a machine.

Format

Command format:

```
sp reboot [ {-f | --forced} ]
```

TABLE H-53 lists the argument for this command.

TABLE H-53 Argument for Subcommand `sp reboot`

Argument	Description
{-f | --forced}	Results in a hard power off.

Return Codes

TABLE H-54 lists the return codes for this command.

TABLE H-54 Return Codes for Subcommand `sp reboot`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoMemory`	8	Insufficient memory.

SP Reset Subcommand

Description: Restores selected settings of the SP to the default factory configuration.

The SP configuration files are stored in the directory `/pstore`. When you boot the system, the SP copies these configuration files from `/pstore` to `/etc` whenever the files are missing from `/etc`. Resetting the SP to its default configuration is accomplished by deleting the configuration files in the directory `/pstore`. A reboot of the SP is necessary for the SP reset to take effect.

By default, the SP reboots 60 seconds after the subcommand `sp reset to default-settings` executes, unless you specify the `--nowait` option, in which case the reboot occurs immediately.

Format

Command format:

```
sp reset to default-settings {-a | --all} {-c | --config}
{-n | --network} {-s | --ssh} {-u | --users} {-W | --nowait}
```

TABLE H-55 lists the arguments for this command.

TABLE H-55 Arguments for Subcommand `sp reset`

Argument	Description
`{-a | --all}`	Resets all SP settings to their default configuration. When the SP reboots, the settings are reset to their default values.
`{-c | --config}`	Resets other system configuration settings to their default configuration. When the SP reboots, the system settings are reset to their default values.
`{-n | --network}`	Resets network settings to their default configuration. When the SP reboots, it has no network capabilities or host name. Its NFS mounts fail and you cannot log into the SP remotely through `ssh`. Set up the network configuration for the SP through the operator panel to restore network functions. Set the host name for the SP in order to refer to the SP by name and set up the file `resolv.conf` in order to refer to other systems by name instead of by dot-quad IP addresses. This option deletes all the network files in the directory `/pstore`.
`{-s | --ssh}`	Resets SSH settings to their default configuration. When the SP reboots, new `ssh` public and private keys are generated. Using `ssh` to access the SP from a remote system that had previously logged into the SP causes a failure with a message about the “Remote Host Identification” changing because the `ssh` key on the SP has changed. The remote system must delete its `ssh` public key entry for the SP in order to `ssh` into the SP successfully. This option deletes all the files in the directory `/pstore/ssh/`.
`{-u | --users}`	Resets user authentication settings to their default configuration. When the SP reboots, all user accounts will have been deleted and you cannot log into the SP remotely through `ssh`.
`[-W | --nowait]`	Reboots the SP immediately.

Note – If you change the default addresses of JNET using this command and then re-install the platform operating system or reset the SP by running the subcommand `sp reset to default-settings`, you must re-issue the subcommand `sp set jnet` to re-establish the JNET connection.

Otherwise, the connection will be out-of-sync (one address will be modified and one will be re-set to the default address).

Return Codes

TABLE H-56 list the return codes for this command.

TABLE H-56 Return Codes for Subcommand `sp reset`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.

SP Mount Subcommands

The subcommands in TABLE H-57 manage the SP mount points.

TABLE H-57 `SP Mount` Subcommands

Subcommand	Description
`sp add mount`	Creates or resets a mount point.
`sp delete mount`	Deletes the specified mount point.
`sp get mounts`	Displays the current mount points on the SP.

SP Add Mount Subcommand

Description: Creates or resets a mount point.

Format

Command format:

```
sp add mount [{-l | --local} MOUNTPOINT ] {-r | --remote} SERVER:FILESYSTEM
[{-u|--user} USER] [{-p|--password} PASSWORD]
```

TABLE H-58 lists the arguments for this subcommand.

TABLE H-58 Arguments for Subcommand `sp add mount`

Argument	Description
`{-l | --local}`	*Optional*; Specifies the local mount point. The only mount point supported is `/mnt`.
`{-r | --remote}`	Specifies the remote server and file system. If *SERVER* specifies a host name, DNS must be properly configured.
`{-u | --user}`	Specifies the user name for the mount. Only required for SMB.
`{-p | --password}`	Specifies the password for the mount user. Only required for SMB.

Note – Several error messages may appear when executing an `smb` mount while mounting windows partitions. Check that the mount succeeded after the call by running the subcommand `sp get mount`.

The required formats for remote NFS and SMB mounts are as follows:

- For NFS: *server_name*:*server_exported_mountpoint*
- For SMB: //*server_name*/*windows_share_name*

Return Codes

TABLE H-59 lists the return codes for this subcommand.

TABLE H-59 Return Codes for Subcommand `sp add mount`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.

SP Delete Mount

Description: Deletes a mount point.

Format

Command format:

```
sp delete mount LOCAL_MOUNT_POINT [-q | --quiet]
```

TABLE H-60 lists the arguments for this subcommand.

TABLE H-60 Arguments for Subcommand `sp delete mount`

Argument	Description
`LOCAL MOUNT POINT`	Specifies the mount point to remove. If you do not specify the local mount point, `/mnt` is implicit as the default value.
`[-q | --quiet]`	If the mount point to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE H-61 lists the return codes for this subcommand.

TABLE H-61 Return Codes for Subcommand `sp delete mount`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.

SP Get Mount Subcommand

Description: Displays the current mount points on the SP.

Format

Command format:

```
sp get mounts [{-l | --local} MOUNTPOINT] [-H | --noheader]
[{-D | --delim <DELIMITER>}]
```

TABLE H-62 lists the arguments for this subcommand.

TABLE H-62 Arguments for Subcommand `sp get mount`

Arguments	Description
`{-l | --local}`	Specifies the local mount point. If you do not specify `-l`, `/mnt` is implicit as the local mount point.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-63 lists the return codes for this subcommand.

TABLE H-63 Return Codes for Subcommand `sp get mount`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) not found.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_Busy`	9	Device or resource is busy.
`NWSE_RPCConnected`	11	RPC client already connected.
`NWSE_RPCConnRefused`	12	RPC connection refused.

TABLE H-63 Return Codes for Subcommand `sp get mount`

Return Code	ID	Description
`NWSE_NoRouteToHost`	13	No route to host (network down).
`NWSE_HostDown`	14	Host is down.
`NWSE_NotMounted`	21	File system is not mounted.

SP SMTP Subcommands

The subcommands in TABLE H-64 manage SMTP communications.

TABLE H-64 `SP SMTP` Subcommands

Subcommand	Description
`sp get smtp server`	Retrieves the SMTP server information.
`sp get smtp subscribers`	Returns detailed information about one or all SMTP subscribers.
`sp set smtp server`	Configures the SP SMTP client with the address for the remote SMTP server.
`sp update smtp subscriber`	Updates the information for an existing SMTP subscriber.

SP Get SMTP Server Subcommand

Description: Retrieves the SMTP server information, including the from address.

Format

Command format:

```
sp get smtp server [-H | --noheader] [{-D | --delim <DELIMITER>}]
```

TABLE H-65 lists the arguments for this subcommand.

TABLE H-65 Arguments for Subcommand `sp get smtp server`

Argument	Description
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-66 lists the return codes for this subcommand.

TABLE H-66 Return Codes for Subcommand `sp get smtp server`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

SP Set SMTP Server Subcommand

Description: Configures the SP SMTP client with the information for the remote SMTP server, including the address and optional port number.

Format

Command format:

```
sp set smtp server [{-f | --from} FROM FIELD ] IP OR HOSTNAME OF SMTP SERVER
```

TABLE H-67 lists the arguments for this subcommand.

TABLE H-67 Arguments for Subcommand `sp set smtp server`

Arguments	Description
`{-f | --from}`	Specifies the from field for the SMTP server.
`IP OR HOSTNAME OF SMTP SERVER`	Specifies the IP address or the host name of the SMTP server.

The value you supply is prepended onto `@hostname | ip_address`. The default value is `system`.

For example, if you enter *admin* for `sp_22`, email messages are sent from `admin@sp_22`.

If the host name is not set, the IP will be used as follows: `admin@10.10.30.55`.

Return Codes

TABLE H-68 lists the return codes for this subcommand.

TABLE H-68 Return Codes for Subcommand `sp set smtp server`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

SP Get SMTP Subscribers Subcommand

Description: Returns detailed information about one or all SMTP subscribers.

Format

Command format:

```
sp get smtp subscribers [{-n | --name} <NAME>] [-H | noheader]
[{-D | --delim <DELIMITER>}]
```

TABLE H-69 lists the arguments for this subcommand.

TABLE H-70 lists the default SMTP subscribers.

TABLE H-69 Arguments for Subcommand `sp get smtp sebscribers`

Arguments	Description
`{ -n | --namserver }`	Specifies the name of the SMTP subscriber for which to retrieve information. If you do not specify this option, the command returns information for all subscribers.
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

TABLE H-70 Default SMTP Subscribers

Subscriber	Description
`SMTP_Info_Short`	Short email message, informational severity
`SMTP_Info_Long`	Long email message, informational severity
`SMTP_Warning_Short`	Short email message, warning severity
`SMTP_Warning_Long`	Long email message, warning severity
`SMTP_Critical_Short`	Short email message, critical severity
`SMTP_Critical_Long`	Long email message, critical severity

Long email messages contain full event details in the message body, while short email messages contain no message body and a descriptive subject line (the same subject as used for long messages).

The short-email format is intended to be used with pagers and other wireless access devices with which message-size constraints may prevent reception of the long-format message.

Return Codes

TABLE H-71 lists the return codes for this subcommand.

TABLE H-71 Return Codes for Subcommand `sp get smtp subscribers`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.

SP Update SMTP Subscriber Subcommand

Description: Updates the information for an existing SMTP subscriber.

Format

Command format:

```
sp update smtp subscriber
{-n | --name} NAME {-r | --recipients} ADDRESS LIST
```

TABLE H-72 lists the arguments for this subcommand.

TABLE H-73 lists the default SMTP subscribers.

Note – All options replace the existing values with the new value. Unspecified options leave existing settings as they are. For example, if you only specify the `-r` option for an existing subscriber, the existing email address list is replaced with the new list specified in the command.

TABLE H-72 Arguments for Subcommand `sp update smtp subscriber`

Arguments	Description
`{-n | --name}`	Specifies the name of the SMTP subscriber to update. This argument is repeatable to update multiple SMTP subscribers at one time.
`{-r | --recipients}`	Specifies the address list of recipients for the SMTP subscriber.

TABLE H-73 Default SMTP Subscribers

Subscriber	Description
`SMTP_Info_Short`	Short email message, informational severity
`SMTP_Info_Long`	Long email message, informational severity
`SMTP_Warning_Short`	Short email message, warning severity
`SMTP_Warning_Long`	Long email message, warning severity
`SMTP_Critical_Short`	Short email message, critical severity
`SMTP_Critical_Long`	Long email message, critical severity

Long email messages contain full event details in the message body, while short email messages contain no message body and a descriptive subject line (the same subject as used for long messages).

The short-email format is intended to be used with pagers and other wireless access devices with which message-size constraints may prevent reception of the long-format message.

Return Codes

TABLE H-74 lists the return codes for this command.

TABLE H-74 Return Codes for Subcommand `sp update smtp subscriber`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

SP SNMP Subcommands

The subcommands in TABLE H-75 manage SNMP communications.

TABLE H-75 SP SNMP Subcommands

Subcommand	Description
sp add snmp destination	Adds an SNMP destination.
sp delete snmp destination	Deletes the SNMP destination.
sp get snmp destinations	Displays the available SNMP destinations (IP address or hostname) to which the Service Processor is configured to send.
sp get snmp proxy community	Returns the community name currently being used by the Service Processor SNMPD to proxy the platform SNMP agent.
sp set snmp proxy community	Sets the proxy entries that specify the OID to be referred, the IP to which they are referred and the community string to use while proxying.

SP Add SNMP Destination Subcommand

Description: Adds a single SNMP destination (either IP address or host name).

Format

Command format:

```
sp add snmp-destination IP ADDRESS/HOSTNAME
```

TABLE H-76 lists the argument for this subcommand.

TABLE H-76 Argument for Subcommand `sp add snmp-destination`

Arguments	Description
`IP ADDRESS/HOSTNAME`	Specifies the IP address or name of the host for the destination you wish to add. This argument is repeatable to add multiple destinations at one time; however, the number of destinations you can create is limited due to memory constraints.

Return Codes

TABLE H-77 lists the return codes for this subcommand.

TABLE H-77 Return Codes for Subcommand `sp add snmp-destination`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_Exist`	19	Entity (user, service or other) already exists.

SP Delete SNMP Destination Subcommand

Description: Deletes a single SNMP destination (either IP address or host name).

Format

Command format:

```
sp delete snmp-destination { IP_ADDRESS/HOSTNAME | {-a | --all}
[-q | --quiet]
```

TABLE H-78 lists the arguments for this subcommand.

TABLE H-78 Arguments for Subcommand `sp delete snmp-destination`

Arguments	Description
`IP ADDRESS/HOSTNAME`	Specifies the IP address or hostname of the destination to remove. This argument is repeatable to remove multiple destinations at one time.
`[-a | --all]`	Removes all SNMP destinations.
`[-q | --quiet]`	If the SNMP destination to delete is not found, this argument specifies that no error be returned.

Return Codes

TABLE H-79 lists the return codes for this subcommand.

TABLE H-79 Return Codes for Subcommand `sp delete snmp-destination`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.

TABLE H-79 Return Codes for Subcommand `sp delete snmp-destination`

Return Code	ID	Description
`NWSE_RPCConnRefused`	12	RPC connection refused.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.

SP Get SNMP Destinations Subcommand

Description: Displays the available SNMP destinations (IP address or host name) to which the SP is configured to send. Many networking programs use this information to identify the machine.

Format

Command format:

```
sp get snmp-destinations
```

Return Codes

TABLE H-80 lists the return codes for this subcommand.

TABLE H-80 Return Codes for Subcommand `sp get snmp-destinations`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.

SP Get SNMP Proxy Community Subcommand

Description: Returns the community name the SP is currently using to proxy the platform SNMP agent.

Format

Command format:

```
sp get snmp proxy community
```

Return Codes

TABLE H-81 lists the return codes for this subcommand.

TABLE H-81 Return Codes for Subcommand `sp get snmp proxy community`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

SP Set SNMP Proxy Community Subcommand

Description: The SNMP agent on the SP acts as a proxy for the master SNMP agent running on the platform. These proxy entries specify the OID to be referred, the IP address to which they are referred and the community string to use while proxying. The community string is the value configured on the platform-side SNMP configuration.

Format

Command format:

```
sp set snmp proxy community COMMUNITY STRING
```

TABLE H-82 lists the argument for this subcommand.

TABLE H-82 Argument for Subcommand `sp set snmp proxy community`

Argument	Description
`COMMUNITY STRING`	Specifies the name of the community to configure.

There are no restrictions on the length of the community strings; common names are *private* and *public*. The default name of the community string is *private*.

If you run the subcommand `sp get snmp proxy community` without setting it, the return value is `private`. Otherwise, you can set it to any string.

Return Codes

TABLE H-83 lists the return codes for this subcommand.

TABLE H-83 Return Codes for Subcommand `sp set snmp proxy community`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_RPCTimeout`	2	Request was issued, but was not serviced by the server. RPC procedure timed out and the request may or may not have been serviced by the server.
`NWSE_RPCNotConnected`	3	Unable to connect to the RPC server.

SP SSL Subcommands

The subcommands in TABLE H-84 manage SSL capabilities.

TABLE H-84 `SP SSL` Subcommands

Subcommand	Description
`sp disable ssl-required`	Disables forced use of the secure HTTP (https) protocol.
`sp enable ssl-required`	Enables forced use of the secure HTTP (https) protocol.
`sp get ssl`	Determines if the Apache Web server is using factory-supplied files or user-supplied files.
`sp set ssl`	Allows you to use site SSL certificates in the SP environment.

SP Disable SSL-Required Subcommand

Description: Disables automatic redirect to secure HTTP URLs. With SSL disabled, HTTP requests are serviced directly without redirecting to HTTPS. HTTPS requests continue to be secure.

Format

Command format:

```
sp disable ssl-required
```

Return Codes

TABLE H-85 lists the return codes for this command.

TABLE H-85 Return Codes for Subcommand `sp disable ssl-required`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.

TABLE H-85 Return Codes for Subcommand `sp disable ssl-required`

Return Code	ID	Description
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Enable SSL-Required Subcommand

Description: Enables automatic redirect to secure HTTP URLs. With SSL enabled, HTTP requests are automatically redirected to equivalent HTTPS requests to maintain site security.

SSL version 0.9.6j is supported.

Format

Command format:

```
sp enable ssl-required
```

Return Codes

TABLE H-86 lists the return codes for this command.

TABLE H-86 Return Codes for Subcommand `sp enable ssl-required`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Get SSL Subcommand

Description: Determines if automatic redirect to secure HTTP is required or optional, and whether Apache Web Server is using factory or user-supplied SSL certificate files.

Format

Command format:

```
sp get ssl [{-H | noheader}] [{-D | --delim <DELIMITER>}]
```

TABLE H-87 lists the arguments for this subcommand.

TABLE H-87 Arguments for Subcommand `sp get ssl`

Arguments	Description
`{ -H | --noheader }`	Suppresses column headings.
`{ -D | --delim }`	Delimits columns with the specified delimiter. Headings are also delimited unless suppressed. The delimiter can be any character or string.

Return Codes

TABLE H-88 lists the return codes for this subcommand.

TABLE H-88 Return Codes for Subcommand `sp get ssl`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Set SSL Subcommand

Description: Allows you to use site SSL certificates in the Service Processor environment. This command allows you to replace the Server Certificate in the SP Value-Add image with your own internally-generated certificate and to restore the factory settings.

Format

Command format:

```
sp set ssl [-f]
```

```
sp set ssl {-c | --certfile} <FULL PATH OF THE SERVER CERTIFICATE FILE>
{-k | --keyfile} <FULL PATH OF PRIVATE KEY FILE>
```

TABLE H-89 lists the arguments for this subcommand.

TABLE H-89 Arguments for Subcommand `sp set ssl`

Argument	Description
`[-f]`	Restores factory settings.
`{-c | --certfile}`	Flags the names of the files to be installed.
`{-k | --keyfile}`	Flags the names of the files to be installed.

Return Codes

TABLE H-90 lists the return codes for this subcommand.

TABLE H-90 Return Codes for Subcommand `sp set ssl`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Update Subcommands

The subcommands in TABLE H-91 manage the SP flash.

TABLE H-91 `SP Flash` Subcommands

Subcommand	Description
`sp update flash all`	Sets the update flag to start the full flash update on the next reset of the SP.
`sp update flash applications`	Copies the file `Value-Add` to the Value-Add component of the SP flash.
`sp update flash pic`	Updates the PIC firmware to a newer version.
`sp update diags`	Updates the diagnostics to a newer version.

SP Update Flash All Subcommand

Note – Before using this command you must start the Java Update Server. For instructions on starting Java Update Server, see "Updating the Service Processor Base Component" on page 23.

Description: Updates the entire SP flash image (kernel, base file system and value add) as part of a major SP software update. This command requires the use of the Java Update Server and verifies that it is available before beginning the update process. Once verified, the SP is rebooted and the update process is initiated. When completed, the SP is automatically rebooted using the updated image.

Format

Command format:

```
sp update flash all {-i | --ipaddress}
<IP ADDRESS xxx.xxx.xxx.xxx> [{-p | --port}] <PORT#>
```

TABLE H-92 lists the arguments for this subcommand.

TABLE H-92 Arguments for Subcommand `sp update flash all`

Argument	Description
`{-i | --serverip}`	The IP address of the remote server on which the Java spUpdate program is running.
`{-p | --port}`	*Optional*: The port number on the remote server on which the Java spUpdate program is listening for SP flash update requests. If the port number is not provided, the command tries to connect to the default port. The default port number is 52708.

Return Codes

TABLE H-93 lists the return codes for this subcommand.

TABLE H-93 Return Codes for Subcommand `sp update flash all`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.
`NWSE_ServiceNotAvailable`	24	Requested service is not available.

SP Update Flash Applications Subcommand

Description: The SP file system is divided into two componenets: Base and Value-Add. The Base component includes the repository and the Value-Add component includes the application software.

This command copies the file `Value-Add` to the Value-Add component of the SP flash. The new Value-Add image takes effect after you reset the SP.

Format

Command format:

```
sp update flash applications {-f | --filename}
```

TABLE H-94 lists the arguments for this subcommand.

TABLE H-94 Arguments for Subcommand `sp update flash applications`

Argument	Description
`{-f | --filename}`	Specifies the full path of the file.

Return Codes

TABLE H-95 lists the return codes for this subcommand.

TABLE H-95 Return Codes for Subcommand `sp update flash applications`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NotFound`	5	Entity (user, service, file, path or other) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_DeviceError`	25	Unable to read or write to the device.

SP Update Flash PIC Subcommand

Description: This command updates the PIC firmware to a newer version. An input PIC-update image file is provided.

Format

Command format:

```
sp update flash pic {-f | --filename} FULL PATH OF THE FILE | [{-v | --
version}]
```

TABLE H-96 lists the arguments for this subcommand.

TABLE H-96 Arguments for Subcommand `sp update flash pic`

Argument	Description
`{-f | --filename}`	Specifies the full path of the file.
`{-v | --version}`	Outputs the current version of the firmware.

Return Codes

TABLE H-97 lists the return codes for this subcommand.

TABLE H-97 Return Codes for Subcommand `sp update flash pic`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_NotFound`	5	Entity (user, service, file, path, etc.) was not found.
`NWSE_NoPermission`	6	Not authorized to perform this operation.
`NWSE_NoMemory`	8	Insufficient memory.
`NWSE_FileError`	18	File open, file missing, or a read or write error occurred.
`NWSE_DeviceError`	25	Unable to read or write to the device.

SP Update Diags Subcommand

Description: Updates the current version of diagnostics available.

While the SP functions normally without access to an external file system, a file system is required to enable several features, including diagnostics. The SP software uses a default version of diagnostics. However, if a new version is released and stored on the Network Share Volume, you must explicitly point to that new version to use it.

Format

Command format:

```
sp update diags {-p | --path} <PATH_TO_DIAGS_FOLDER>
```

TABLE H-98 lists the argument for this subcommand.

TABLE H-98 Argument for Subcommand `sp update diags`

Argument	Description
{-p | --path}	Points to the location of the new diagnostics.

Return Codes

TABLE H-99 lists the return codes for this subcommand.

TABLE H-99 Return Codes for Subcommand `sp update diags`

Return Code	ID	Description
`NWSE_Success`	0	Command successfully completed.
`NWSE_InvalidUsage`	1	Invalid usage: bad parameter usage, conflicting options specified.
`NWSE_InvalidArgument`	4	One or more arguments were incorrect or invalid.
`NWSE_UnknownError`	15	Miscellaneous error not captured by other errors.

Index