System Management Services (SMS) 1.6 Administrator Guide

for Sun Fire™ High-End Systems

Sun Microsystems, Inc.
www.sun.com

Part No. 819-4660-10
May 2006, Revision A

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

Contents

Figures

Tables

Code Samples

Preface

The *System Management Services (SMS) 1.6 Administrator Guide* describes how to perform various administration and monitoring tasks associated with the SMS software.

Before You Read This Book

This manual is intended for the Sun Fire™ system administrator, who has a working knowledge of UNIX® systems, particularly those based on the Solaris™ Operating System (Solaris OS). If you do not have such knowledge, read the Solaris User and System Administrator documentation provided with your system, and consider UNIX system administration training.

All members of the next-generation Sun Fire server family can be configured as loosely coupled clusters. However, it is outside of the scope of this document to address system management for Sun Fire high-end system cluster configurations.

How This Book Is Organized

This guide contains the following chapters:

Chapter 1 introduces the System Management Services software and describes its command-line interface.

Chapter 2 introduces security on the domains and system controllers.

Chapter 3 introduces administrative privileges.

Chapter 4 describes SMS domain internals and explains their use.

Chapter 5 describes domain configuration, options, and procedures.

Chapter 6 describes the automatic diagnosis and domain recovery features.

Chapter 7 describes Capacity on Demand (COD).

Chapter 8 describes the control functions.

Chapter 9 describes network services available and explains their use.

Chapter 10 describes status monitoring.

Chapter 11 describes event monitoring.

Chapter 12 describes system controller (SC) failover.

Chapter 13 describes SMS utilities for creating and restoring backups, configuring networks and user groups, and upgrading SMS software.

Appendix A provides a list of SMS man pages.

Appendix B describes SMS error messages.

Using UNIX Commands

This document might not contain information on basic UNIX commands and procedures such as shutting down the system, booting the system, and configuring devices. See the following for this information:

- Software documentation that you received with your system
- Solaris Operating System (OS) documentation, which is at:

  `http://docs.sun.com`

Typographic Conventions

Typeface or Symbol	Meaning	Examples
`AaBbCc123`	The names of commands, files, and directories; on-screen computer output	Edit your `.login` file. Use `ls -a` to list all files. `% You have mail.`

`AaBbCc123`	What you type, when contrasted with on-screen computer output	`%` **`su`**
		`Password:`

AaBbCc123	Book titles, new words or terms, words to be emphasized. Replace command-line variables with real names or values.	Read Chapter 6 in the *User's Guide*. These are called *class* options. To delete a file, type **`rm`** *filename*.

Shell Prompts

Shell	Prompt
C shell	*sc_name*`:`*sms-user*`:> or` *domain_id*`:`*sms-user*`:>`
C shell superuser	*sc_name:*`# or` *domain_id*`:#`
Bourne shell and Korn shell	`>`
Bourne shell and Korn shell superuser	`#`

Related Documentation

The SMS documents are available at:

```
http://www.sun.com/products-n-
solutions/hardware/docs/Servers/High-
End_Servers/Sun_Fire_e25K-
e20K/SW_FW_Documentation/SMS/index.html
```

The other documents can be found by typing in the name of the document in Search at:

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

Application	Title	Part Number	Format	Location
Software Overview	*Sun Fire High-End Systems Software Overview Guide*	819-4658-10	PDF HTML	Online
Installation	*System Management Services (SMS) 1.6 Installation Guide*	819-4659-10	PDF HTML	Online
Reference (man pages)	*System Management Services (SMS) 1.6 Reference Manual*	819-4662-10	PDF HTML	Online
Release Notes	*System Management Services (SMS) 1.6 Release Notes*	819-4663-10	PDF HTML	Online
Dynamic Reconfiguration	*Sun Fire High-End and Midrange Systems Dynamic Reconfiguration User Guide*	819-1501-10	PDF HTML	Online
OpenBoot	*OpenBoot™ 4.x Command Reference Manual*	816-1177-10	PDF HTML	Online
Site Planning	*Sun Fire 15K/12K System Site Planning Guide*	806-3510-12	PDF HTML	Online
Security	*Solaris Security Toolkit 4.2 Administration Guide*	819-1402-10	PDF HTML	Online
Security	*Solaris Security Toolkit 4.2 Reference Manual*	819-1503-10	PDF HTML	Online
Security	*Solaris Security Toolkit 4.2 Release Notes*	819-1504-10	PDF HTML	Online
Security	*Solaris Security Toolkit 4.2 Man Page Guide*	819-1505-10	PDF HTML	Online
Solaris 10 OS IP Services	*System Administration Guide: IP Services*	816-4554	PDF HTML	Online

Documentation, Support, and Training

Sun Function	URL
Documentation	http://www.sun.com/documentation/
Support	http://www.sun.com/support/
Training	http://www.sun.com/training/

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.

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:

System Management Services (SMS) 1.6 Administrator Guide, part number 819-4660-10

CHAPTER 1

Introduction to System Management Services

This manual describes the System Management Services (SMS) 1.6 software that is available with the Sun Fire high-end server system.

This chapter includes the following sections:

Sun Fire High-End Systems

The system controller (SC) in Sun Fire high-end systems is a multifunction, CP1500- or CP2140-based printed circuit board (PCB) that provides critical services and resources required for the operation and control of the Sun Fire system.

A Sun Fire high-end system is often referred to as the *platform*. System boards within the platform can be logically grouped together into separately bootable systems called *dynamic system domains*, or simply *domains*.

Up to 18 domains can exist simultaneously on a single Sun Fire E25K/15K, and up to 9 domains on the Sun Fire E20K/12K. (Domains are introduced in this chapter, and are described in more detail in Chapter 5). The SMS software lets you control and monitor domains, as well as the platform itself.

The SC provides the following services for the Sun Fire system:

- Manages the overall system configuration.
- Acts as a boot initiator for system domains.

- Serves as the syslog (system log) host for system domains. Note that an SC can still be a syslog client of a LAN-wide syslog host.
- Provides a synchronized hardware clock source.
- Sets up and configures dynamic domains.
- Monitors system environmental information, such as power supply, fan, and temperature status.
- Hosts field-replaceable unit (FRU) logging data.
- Provides redundancy and automated SC failover in dual-SC configurations.
- Provides a default name service for the domains based on virtual host IDs, and provides MAC addresses for the domains.
- Provides administrative roles for platform management.

Redundant SCs

There are two SCs within a Sun Fire platform. The SC that controls the platform is referred to as the *main SC*, while the other SC acts as a backup and is called the *spare SC*. The software running on the main SC monitors both SCs to determine when an automatic failover should be performed.

Configure the two SCs with the same configuration. This duplication includes the Solaris Operating System (OS), SMS software, security modifications, patch installations, and all other system configurations.

Note – For failover to be supported, both SCs must be configured with identical versions of the Solaris OS and SMS software.

The failover functionality between the SCs is controlled by daemons running on the main and spare SCs. These daemons communicate across private communication paths built into the Sun Fire platform. Other than the communication between these daemons, there is no special trust relationship between the two SCs.

SMS software packages are installed on the SC. In addition, SMS communicates with the Sun Fire high-end system over an Ethernet connection. See "Management Network Services" on page 184.

Note – SMS 1.6 cannot communicate with SMS 1.4.1 across the I2 network. If one of the SCs is running SMS 1.4.1 and the other is running SMS 1.6, the I2 network tests will fail, and the SCs will communicate instead through high-availability SRAM (HASRAM) For information about the I2 network, see "I2 Network" on page 182.

SMS Features

SMS 1.6 supports Sun Fire high-end domains running the Solaris 8 2/04, Solaris 9 4/04, Solaris 10 3/05, Solaris 10 1/06, and Solaris 10 6/06 OSs. SMS 1.6 supports the Solaris 10 1/06, Solaris 10 6/06, Solaris 9 4/04, Solaris 9 9/04, and Solaris 9 9/05 OSs on the system controllers. The commands provided with the SMS software can be used remotely.

Note – The supported firmware version for SMS 1.6 is 5.2.0.

Note – Graphical user interfaces for many of the commands in SMS are provided by the Sun™ Management Center. For more information, see "Sun Management Center" on page 14.

SMS enables the *platform* administrator to perform the following tasks:

- Administer domains by logically grouping *domain configurable units* (DCUs) together. DCUs are system boards such as CPU and I/O boards. Domains are able to run their own OSs and handle their own workloads. See Chapter 5.
- Dynamically reconfigure a domain so that currently installed system boards can be *logically* attached to or detached from the OS while the domain continues running in multiuser mode. This feature is known as *dynamic reconfiguration* and is described in the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*. (A system board can be *physically* swapped in and out when it is not attached to a domain, while the system continues running in multiuser mode).
- Perform automatic dynamic reconfiguration of domains using a script. Refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.
- Monitor and display the temperatures, currents, and voltage levels of one or more system boards or domains.
- Monitor and control power to the components within a platform.
- Execute diagnostic programs such as power-on self-test (POST).

In addition, SMS:

- Warns platform administrators of impending problems, such as high temperatures or malfunctioning power supplies.
- Notifies platform administrators when a software error or failure has occurred.
- Monitors a dual-SC configuration for single points of failure and performs an automatic failover from the main SC to the spare depending on the failure condition detected.

- Automatically reboots a domain after a system software failure (such as a panic).
- Keeps logs of interactions between the SC environment and the domains.
- Provides support for the Sun Fire high-end system dual-grid power option.

SMS enables the *domain* administrator to perform the following tasks:

- Administer domains by logically grouping *domain configurable units* (DCUs) together. DCUs are system boards such as CPU and I/O boards. Domains are able to run their own OSs and handle their own workloads. See Chapter 5.
- Boot domains for which the administrator has privileges.
- Dynamically reconfigure a domain for which the administrator has privileges, so that currently installed system boards can be *logically* attached to or detached from the OS while the domain continues running in multiuser mode. This feature is known as *dynamic reconfiguration* and is described in the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*. (A system board can be *physically* swapped in and out when it is not attached to a domain, while the system continues running in multiuser mode.)
- Perform automatic dynamic reconfiguration of domains using a script for which the administrator has privileges. Refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.
- Monitor and display the temperatures, currents, and voltage levels of one or more system boards or domains for which the administrator has privileges.
- Execute diagnostic programs such as power-on self-test (POST) for which the administrator has privileges.

Features Provided in Previous Releases of SMS

Previous SMS releases provided the following:

- Dynamic system domain (DSD) configuration
- Configured domain services
- Domain control capabilities
- Automatic diagnosis and domain recovery
- Capacity on Demand (COD)
- Domain status reporting
- Hardware control capabilities
- Hardware status monitoring, reporting, and handling
- Hardware error monitoring, reporting, and handling
- System controller (SC) failover
- Configurable administrative privileges
- Dynamic FRUID

New Features Provided in SMS 1.6 Release

SMS 1.6 provides the following new features:

- Support for Solaris 10 OS or higher on domains
- Support for Solaris 10 1/06 and 6/06 OS on the system controllers
- Support for UltraSPARC® IV 1.65-GHz processor
- Readiness for UltraSPARC IV+ 1.8-GHz processor
- Voltage core monitoring (VCMON)
- 2 GB DIMMs
- Improved memory refresh rate
- Secure by default for system controllers
- Support for Solaris Security Toolkit 4.2
- Support for Availability (AVL) 2.0 FS-2 software (Solaris 10 6/06 required)
    - UltraSPARC IV+ Processor Diagnosis Enhancements
    - Anchored Page Retire
    - Datapath Diagnosis Coordination (Domain FMA and SC)
    - Supported Platforms: UltraSPARC III Enterprise Server, Sun Fire V1280 and Netra 1280, and Sun Fire 15K families

VCMON

A voltage core monitoring parameter (VCMON) was added to the SMS software. When VCMON is enabled, it monitors any voltage changes or drifts on the processors. If VCMON detects an upward change in voltage (which usually indicates a socket attach issue), it notifies the user with an FMA event and marks the component health status (CHS) of that processor as faulty.

System Architecture

SMS uses a distributed client-server architecture. `init`(1M) starts, and restarts as necessary, one process: `ssd`(1M). `ssd` is responsible for monitoring all other SMS processes and restarting them as necessary. See FIGURE 4-1.

The Sun Fire high-end systems platform, the SC, and other workstations communicate over Ethernet. You perform SMS operations by entering commands on the SC console after remotely logging in to the SC from another workstation on the local area network (LAN). You must log in as a user with the appropriate platform or domain privileges if you want to perform SMS operations, such as monitoring and controlling the platform.

Note – If SMS is stopped on the main SC and the spare SC is powered off, the domains shut down gracefully and the platform is powered down. If the spare SC is simply powered off without a shutdown of SMS, SMS will not have time to power off the platform and the domains will crash.

Dual-system controllers are supported within the Sun Fire high-end systems platform. One SC is designated as the primary or main system controller, and the other is designated as the spare system controller. If the main SC fails, the failover capability automatically switches to the spare SC as described in Chapter 12.

Most domain-configurable units are active components. This means that you must check the system state before powering off any DCU.

Caution – Circuit breakers must be on whenever a board is present, including expander boards, whether or not the board is powered on.

For details, see "Power Control" on page 173.

SMS Administration Environment

Administration tasks on the Sun Fire high-end system are secured by group privilege requirements. SMS installs the following 39 UNIX groups to the `/etc/group` file.

- `platadmn` – Platform administrator
- `platoper` – Platform operator
- `platsvc` – Platform service
- `dmn`[*A*...*R*]`admn` – domain [*domain-id* | *domain-tag*] administrator (18)
- `dmn`[*A*...*R*]`rcfg` – domain [*domain-id* | *domain-tag*] configurator (18)

The `smsconfig`(1M) command enables an administrator to add, remove, and list members of platform and domain groups, as well as set platform and domain directory privileges using the `-a`, `-r`, and `-l` options.

`smsconfig` also can configure SMS to use alternate group names, including NIS (Network Information Service) managed groups using the `-g` option. Group information entries can come from any of the sources for groups specified in the`/etc/nsswitch.conf` file (refer to `nsswitch.conf`(4)). For instance, if domain A was known by its domain tag as the Production Domain, an administrator could create an NIS group with the same name and configure SMS to use this group as the domain A administrator group instead of using the default, `dmnaadmn`. For more information, see Chapter 3, and refer to the `smsconfig` man page.

Network Connections for Administrators

The nature of the Sun Fire high-end systems physical architecture, with an embedded system controller, as well as the supported administrative model (with multiple administrative privileges, and thus multiple administrators) dictates that an administrator use a remote network connection from a workstation to access SMS command interfaces to manage the Sun Fire high-end system.

Caution – Shutting down a remote workstation while a `tip` session is active into a Sun Fire high-end system SC will bring both SCs down to the OpenBoot™ `ok` prompt. This will not affect the domains, and after powering the remote system back on you can restore the SCs by typing `go` at the `ok` prompt. However, you should end all `tip` sessions before shutting down a remote workstation.

Since the administrators provide information to verify their identity (passwords) and might need to display sensitive data, it is important that the remote network connection be secure. Physical separation of the administrative networks provides some security on the Sun Fire high-end system. Multiple external physical network connections are available on each SC. SMS software supports up to two external network communities.

For more information on Sun Fire high-end system networks, see "Management Network Services" on page 184. For more information on securing the Sun Fire high-end system, see Chapter 2, "Using Solaris Security Toolkit to Secure the System Controller" on page 29.

SMS Operating System

SMS provides a command-line interface (CLI) to the various functions and features the program contains. You can interact with the SC and the domains on a system by using the CLI commands.

For the examples in this guide, the *sc-name* is `sc0` and *sms-user* is the *user-name* of the administrator, operator, configurator, or service personnel logged in to the system.

The privileges allotted to the user are determined by the platform or domain groups to which the user belongs. In these examples, the *sms-user* is assumed to have both platform and domain administrator privileges, unless otherwise noted.

For more information on the function and creation of SMS user groups, see Chapter 3 and refer to the *System Management Services (SMS) 1.6 Installation Guide*.

▼ To Begin Using the SC

1. **Boot the SC.**

Note – This procedure assumes that `smsconfig -m` has already been run. If `smsconfig -m` has not been run, you will receive the following error when SMS attempts to start and SMS will exit.

```
sms: smsconfig(1M) has not been run. Unable to start sms services.
```

2. **Log in to the SC and verify that SMS software startup has completed. Type:**

```
sc0:sms-user:> showplatform
```

Output similar to the following is displayed if you have platform privileges.

```
sc0:sms-user:>  showplatform

PLATFORM:
========
Platform Type: Sun Fire 15000

CSN:
====
Chassis Serial Number: 353A00053

COD:
====
Chassis HostID : 5014936C37048
PROC RTUs installed : 8
PROC Headroom Quantity : 0
PROC RTUs reserved for domain A : 4
PROC RTUs reserved for domain B : 0
PROC RTUs reserved for domain C : 0
PROC RTUs reserved for domain D : 0
PROC RTUs reserved for domain E : 0
PROC RTUs reserved for domain F : 0
PROC RTUs reserved for domain G : 0
PROC RTUs reserved for domain H : 0
PROC RTUs reserved for domain I : 0
PROC RTUs reserved for domain J : 0
PROC RTUs reserved for domain K : 0
PROC RTUs reserved for domain L : 0
PROC RTUs reserved for domain M : 0
PROC RTUs reserved for domain N : 0
PROC RTUs reserved for domain O : 0
PROC RTUs reserved for domain P : 0
PROC RTUs reserved for domain Q : 0
PROC RTUs reserved for domain R : 0


Available Component List for Domains:
=====================================
Available for domain newA:
          SB0 SB1 SB2 SB7
          IO1 IO3 IO6
Available for domain engB:
          No System boards
          No IO boards
Available  for domain domainC:
          No System boards
          IO0 IO1 IO2 IO3 IO4
Available  for domain eng1:
          No System boards
          No IO boards
Available  for domain E:
          No System boards
          No IO boards
Available  for domain domainF:
          No System boards
          No IO boards
Available  for domain dmnG:
          No System boards
          No IO boards
Available  for domain domain H:
          No System boards
          No IO boards
Available  for domain I:
          No System boards
          No IO boards
Available  for domain dmnJ:
          No System boards
          No IO boards
Available  for domain K:
          No System boards
          No IO boards
Available  for domain L:
          No System boards
          No IO boards
Available  for domain M:
          No System boards
          No IO boards
Available  for domain N:
          No System boards
          No IO boards
Available  for domain O:
          No System boards
          No IO boards
Available  for domain P:
          No System boards
          No IO boards
Available  for domain Q:
          No System boards
          No IO boards
Available  for domain dmnR:
          No System boards
          No IO boards


Domain Ethernet Addresses:
=============================
Domain ID   Domain Tag        Ethernet Address
A           newA              8:0:20:b8:79:e4
B           engB              8:0:20:b4:30:8c
C           domainC           8:0:20:b7:30:b0
D               -             8:0:20:b8:2d:b0
E           eng1              8:0:20:f1:b7:0
F           domainF           8:0:20:be:f8:a4
G           dmnG              8:0:20:b8:29:c8
H               -             8:0:20:f3:5f:14
I               -             8:0:20:be:f5:d0
J           dmnJ              UNKNOWN
K               -             8:0:20:f1:ae:88
L               -             8:0:20:b7:5d:30
M               -             8:0:20:f1:b8:8
N               -             8:0:20:f3:5f:74
O               -             8:0:20:f1:b8:8
```

```
P                -             8:0:20:b8:58:64
Q                -             8:0:20:f1:b7:ec
R           dmnR               8:0:20:f1:b7:10


Domain Configurations:
======================
DomainID    Domain Tag     Solaris Nodename     Domain Status
A           newA           -                    Powered Off
B           engB           sun15-b              Keyswitch Standby
C           domainC        sun15-c              Running OBP
D           -              sun15-d              Running Solaris
E           eng1           sun15-e              Running Solaris
F           domainF        sun15-f              Running Solaris
G           dmnG           sun15-g              Running Solaris
H           -              sun15-g              Solaris Quiesced
I           -              -                    Powered Off
J           dmnJ           -                    Powered Off
K           -              sun15-k              Booting Solaris
L           -              -                    Powered Off
M           -              -                    Powered Off
N           -              sun15-n              Keyswitch Standby
O           -              -                    Powered Off
P           -              sun15-p              Running Solaris
Q           -              sun15-q              Running Solaris
R           dnmR           sun15-r              Running Solaris
```

At this point, you can begin using SMS programs.

SMS Console Window

An SMS console window provides a command-line interface from the SC to the Solaris OS on the domains.

▼ To Display a Console Window Locally

1. **Log in to the SC, if you have not already done so.**

Note – You must have domain privileges for the domain on which you want to run `console`.

---

2. Type:

```
sc0:sms-user:> console -d domain-indicator option
```

where:

`-d` Specifies the domain using a *domain-indicator*:

domain-id – ID for a domain. Valid *domain-id*s are 'A'...'R' and are case insensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-f` Force
Opens a domain console window with locked write permission, terminates all other open sessions, and prevents new ones from being opened. This constitutes an exclusive session. Use it only when you need exclusive use of the console (for example, for private debugging). To restore multiple-session mode, either release the lock (~^) or terminate the console session (~.).

`-g` Grab
Opens a console window with unlocked write permission. If another session has unlocked write permission, the new console window takes it away. If another session has locked permission, this request is denied and a read-only session is started.

`-l` Lock
Opens a console window with locked write permission. If another session has unlocked write permission, the new console window takes it away. If another session has locked permission, the request is denied and a read-only session is started.

`-r` Read Only
Opens a console window in read-only mode.

The `console` command creates a remote connection to the domain's virtual `console` driver, making the window in which the command is executed a console window for the specified domain (*domain-id* or *domain-tag*).

If `console` is invoked without any options when no other console windows are running for that domain, it comes up in an exclusive locked write mode session.

If `console` is invoked without any options when one or more nonexclusive console windows are running for that domain, it will appear in read-only mode.

Locked write permission is more secure. It can only be removed if another console is opened using `console -f` or if `~*` (tilde-asterisk) is entered from another running console window. In both cases, the new console session is an exclusive session, and all other sessions are forcibly detached from the domain virtual console.

The console command can use either Input Output Static Random Access Memory (IOSRAM) or the internal management network for domain console communication. You can manually toggle the communication path by using the ~= (tilde-equal sign) command. Doing so is useful if the network becomes inoperable, in which case the console session appears to be hung.

Many console sessions can be attached simultaneously to a domain, but only one console will have write permissions; all others will have read-only permissions. Write permissions are in either locked or unlocked mode.

Tilde Escape Sequences

In a domain console window, a tilde (~) that appears as the first character of a line is interpreted as an escape signal that directs console to perform some special action, as shown in the following table:

TABLE 1-1 Tilde Usage

Character	Description
~?	Status message.
~.	Disconnects console session.
~#	Breaks to OpenBoot PROM or kadb.
~@	Acquires unlocked write permission. See option -g.
~^	Releases write permission.
~=	Toggles the communication path between the network and IOSRAM interfaces. You can use ~= only in private mode (see ~*).
~&	Acquires locked write permission; see option -l . You can issue this signal during a read-only or unlocked write session.
~*	Acquires locked write permission, terminates all other open sessions, and prevents new sessions from being opened; see option -f . To restore multiple-session mode, either release the lock or terminate this session.

The rlogin command also processes tilde-escape sequences whenever a tilde is seen at the beginning of a new line. If you must send a tilde sequence at the beginning of a line and you are connected using rlogin, use two tildes (the first escapes the second for rlogin). Alternatively, do not enter a tilde at the beginning of a line when running inside of an rlogin window.

If you use a kill -9 command to terminate a console session, the window or terminal in which the console command was executed goes into raw mode, and appears hung. Press CTRL-J, then type stty sane, then press CTRL-J to escape this condition.

In the domain console window, vi(1) runs properly and the escape sequences (tilde commands) work as intended only if the environment variable TERM has the same setting as that of the console window.

For example:

```
sc0:sms-user:> setenv TERM xterm
```

To resize the window, type:

```
sc0:sms-user:> stty rows 20 cols 80
```

For more information on the domain console, see Chapter 9 and refer to the console man page.

Remote Console Session

In the event that a system controller hangs and that console cannot be reached directly, SMS provides the smsconnectsc command to remotely connect to the hung SC. This command works from either the main or spare SC. For more information and examples, refer to the smsconnectsc man page.

You may also connect to the hung SC using an external console connection, but you cannot run smsconnectsc and use an external console at the same time.

Sun Management Center

Sun Management Center for Sun Fire high-end systems is an extensible monitoring and management tool that integrates standard Simple Network Management Protocol (SNMP)-based management structures with new intelligent and autonomous agent and management technology based on the client-server paradigm.

Sun Management Center is used as the graphical user interface (GUI) and SNMP manager-agent infrastructure for the Sun Fire system. The features and functions of Sun Management Center are not covered in this manual. For more information, refer to the latest Sun Management Center documentation available at: www.docs.sun.com

CHAPTER 2

SMS 1.6 Security

This chapter provides an overview of security as it pertains to SMS 1.6 and the Sun Fire high-end (E20K/12K and E25K/15K) systems. Security options consist of securing the domains (optional suggestion) and system controllers (strongly suggested) of a given system, as well as overall system hardening. Hardening is the modification of Solaris OS configurations to improve the security of a system.

These suggestions apply to environments where security is a concern, particularly environments where the uptime requirements of the system controllers or the information on the Sun Fire server is critical to the organization.

The system controllers control the hardware components that make up a Sun Fire high-end system. Because they are a central control point for the entire frame, the SCs represent an attack point for intruders. To improve reliability, availability, serviceability, and security (RASS), the system controllers must be secured against malicious misuse and attack. Overviews of domain and system controller security issues follow.

This chapter contains the following sections:

Domain Security Overview

The Sun Fire high-end system platform hardware can be partitioned into one or more environments capable of running separate images of the Solaris OS. These environments are called *dynamic system domains* (DSDs) or *domains*.

A domain is logically equivalent to a physically separate server. The Sun Fire high-end system hardware enforces strict separation of the domain environments. This means that, except for errors in hardware shared by multiple domains, no hardware error in one domain affects another. For domains to act like separate servers, Sun Fire software was designed and implemented to enforce strict domain separation.

SMS provides services to all domains. In providing those services, no data obtained from one client domain is leaked into data observable by another. This is particularly true for sensitive data such as buffers of console characters (including administrator passwords) or potentially sensitive data such as I/O buffers containing client domain-owned data.

SMS limits administrator privilege. This enables you to control the extent of damage that can occur due to administrator error, as well as to limit the exposure to damage caused by an external attack on a system password. See Chapter 3.

System Controller Security Overview

Securing the system controllers is the first priority in configuring Sun Fire high-end systems to be resistant to unauthorized access and to function properly in hostile environments. Before securing the system controllers, it is important to understand the services and daemons that are running on the system. This section describes the software, services, and daemons specific to the system controllers. The functionality is described at a high level, with references to other Sun documentation for more detailed information. This section provides administrators with a baseline of functionality required for the system controllers to perform properly.

The system controllers (SCs) are multifunction system boards within the Sun Fire frame. These SCs are dedicated to running the SMS software. The SMS software is used to configure dynamic domains, provide console access to each domain, control whether a domain is powered on or off, and provide other functions critical to operating and monitoring Sun Fire high-end systems.

The following list is an overview of the many services the system controllers provide for the Sun Fire high-end systems:

- Manages the overall system configuration.
- Acts as a boot initiator for its domains.
- Serves as the syslog host for its domains; note that an SC can still be a syslog client of a LAN-wide syslog host.
- Provides a synchronized hardware clock source.
- Sets up and configures dynamic domains.
- Monitors system environmental information, such as power supply, fan, and temperature status.
- Hosts field-replaceable unit (FRU) logging data.
- Provides redundancy and automated SC failover.
- Provides a default name service for the domains based on virtual host IDs, and MAC addresses for the domains.
- Provides administrative roles for frame management.

Redundant System Controllers

Sun Fire frames have two system controllers. Our security suggestions are the same for both system controllers. The SC that controls the platform is referred to as the *main* SC, while the other SC acts as a backup and is called the *spare* SC. The software running on the SC monitors the system controllers to determine when to perform an automatic failover.

Note – For our sample configuration, the main SC is sc0 and the spare SC is sc1.

We suggest that the two system controllers have the same configuration. This duplication includes the Solaris OS, security modifications, patch installations, and all other system configurations, as well as the same version of SMS software.

The failover functionality between the system controllers is controlled by daemons running on the main and spare system controllers. These daemons communicate across private communication paths built into the Sun Fire frames. Other than the communication between these daemons, there is no special trust relationship between the two system controllers.

SC Network Interfaces

Several network interfaces are used on an SC to communicate with the platform, domains, and other system controllers. Most of these interfaces are defined as regular Ethernet network connections through /etc/hostname.* entries.

Main SC Network Interfaces

A typical main SC (sc0 in our sample) has two files in the /etc directory with contents similar to the following:

```
# more /etc/hostname.scman0
192.168.103.1 netmask + broadcast + private up
# more /etc/hostname.scman1
192.168.103.33 netmask + private up
```

In addition, a typical main SC has corresponding entries in /etc/netmasks:

```
10.1.72.0 255.255.248.0
192.168.103.0   255.255.255.224
192.168.103.32  255.255.255.252
```

Note – Non-routed (RFC 1918) internet protocol (IP) addresses are used in all SC examples. We suggest that you use these types of IP addresses when deploying Sun Fire system controllers. The SMS software defines internal SC network connections to be private and not advertised.

Domain-to-SC Communication (scman0) Interface

The /etc/hostname.scman0 entry sets up the I1 or domain-to-SC SMS Management Network (MAN). The first IP address in our example, 192.168.103.1, is controlled by the SMS software to be always available only on the main SC.

From a security perspective, misuse of or attacks on the I1 MAN network between the domains and the system controllers might adversely impact domain separation. The hardware implementation of the I1 network within a Sun Fire high-end chassis addresses these concerns by permitting only SC-to-domain and domain-to-SC communication. The I1 MAN network is implemented as separate point-to-point physical network connections between the system controllers and each of the 9 domains supported by a Sun Fire E20K/12K server or 18 domains supported by a Sun Fire E25K/15K server. Each of these connections terminates at separate I/O boards on each domain and SC.

On the system controllers, these multiple separate networks are consolidated into one meta-interface to simplify administration and management. The I1 MAN driver software performs this consolidation and enforces domain separation and failovers to redundant communication paths.

Direct communication between domains over the I1 network is not permitted by the hardware implementation of the I1 network. By implementing the network in this manner, each SC-to-domain network connection is physically isolated from other connections.

Note – Although the scman0 network supports regular IP-based network traffic, it should be used only by Sun Fire management traffic. Any other use of this internal network might affect the reliability, availability, serviceability, and security of the entire platform. Refer to the scman (7D) and dman (7D) man pages for more information.

SC-to-SC Communication (scman1) Interface

The /etc/hostname.scman1 entry is used to configure the I2 or SC-to-SC MAN. This network connection, on which both system controllers have an IP address, is for the heartbeat connections between the two system controllers.

Both of the I1 and I2 MAN network connections are implemented internally in the Sun Fire high-end chassis. No external wiring is used.

Spare SC Network Interfaces

The spare SC has the same physical network interfaces as the main SC. The scman0 network interface is plumbed by the Solaris OS through the /etc/hostname.scman0 file on the spare SC in the same manner and with the same information as on the main SC. The difference between the main and spare system controllers is that the interface is inactive on the spare. The spare system controller's scman0 port on the I/O hubs is disabled and mand does not provide path information to scman0 on the spare.

The scman1 interface, which is for SC-to-SC communication, has the following configuration information for this interface:

```
# more /etc/hostname.scman1
192.168.103.34 netmask + broadcast + private up
```

In addition, the spare SC has the following corresponding /etc/netmasks information:

```
10.1.72.0 255.255.248.0
192.168.103.0   255.255.255.224
192.168.103.32  255.255.255.252
```

Main and Spare Network Interface Sample Configurations

Use the following command to verify the status of the main SC:

```
# showfailover -r
MAIN
```

Our network configuration sample appears as follows on the main SC (sc0):

```
# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232
index 1 inet 127.0.0.1 netmask ff000000

hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500
index 2 inet 10.1.72.80 netmask fffff800 broadcast 10.1.79.255
ether 8:0:20:a8:db:2e

scman0:flags=
1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3 inet 192.168.103.1 netmask ffffffe0 broadcast
192.168.103.31 ether 8:0:20:a8:db:2e

scman1:flags=
1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 4 inet 192.168.103.33 netmask fffffffc broadcast
192.168.103.35 ether 8:0:20:a8:db:2e
```

Note – Although the scman0 network supports regular IP-based network traffic, it should be used only by Sun Fire management traffic. Any other use of this internal network might affect the reliability, availability, and serviceability, and security of the entire platform. Refer to the scman (7D) and dman (7D) man pages for more information.

Our sample network configuration appears as follows on the spare SC (sc1):

```
# ifconfig -a
lo0: flags=1000849<UP,LOOPBACK,RUNNING,MULTICAST,IPv4> mtu 8232
index 1
        inet 127.0.0.1 netmask ff000000

hme0: flags=1000843<UP,BROADCAST,RUNNING,MULTICAST,IPv4> mtu 1500
index 2
inet 10.1.72.81 netmask ffffff00 broadcast 10.1.72.255 ether
8:0:20:a8:ba:c7

scman0:flags=
1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 3 inet 192.168.103.1 netmask ffffffe0 broadcast
192.168.103.31 ether 8:0:20:a8:ba:c7

scman1: flags=
1008843<UP,BROADCAST,RUNNING,MULTICAST,PRIVATE,IPv4> mtu 1500
index 4
inet 192.168.103.34 netmask fffffffc broadcast 192.168.103.35
ether 8:0:20:a8:ba:c7
```

What Has Changed in SMS 1.6

Solaris Security Toolkit 4.2 software works with either Solaris 9 OS or Solaris 10 OS, and provides an automated, extensible, and scalable mechanism to build and maintain secure Solaris OS systems. Using the Solaris Security Toolkit software, you can harden and audit the security of systems.

Security options for a system using SMS 1.6 depends on whether the software is to be installed fresh or as an upgrade.

Secure By Default (Fresh Installation)

If the SMS version is a fresh installation, the `smsinstall` command is used and then automatic hardening is accomplished as a function of the installation (secure by default). That is, the system is hardened as the system controllers are made secure. In this instance the domains can also be made secure manually with Solaris Security Toolkit (SST) 4.2.0 software, which is downloaded as a function of the installation. If you are going to install SMS 1.6 fresh, proceed to "Initial or Fresh SMS Installation Using `smsinstall` Command (Secure by Default)" on page 27.

Note – The minimum supported version of SST on Solaris 10 OS is 4.2.0. The minimum supported version of SST on Solaris 8 and 9 OS is 4.1.1.

Secure By Choice (Upgrade)

If the installation is an upgrade, automatic system hardening does not occur. In this instance, the `smsupgrade` command is used, Solaris Security Toolkit software is installed as a function of the upgrade and can then be used to harden, undo hardening, and audit the security posture of a system (secure by choice). This includes the system controllers as well as domains. For an upgrade to SMS 1.6, as well as post-SMS hardening procedures proceed to "SMS Upgrade Installation Using `smsupgrade` Command (Secure by Choice)" on page 28.

Installation Changes

A list of major changes that have occurred for installing SMS 1.6, regardless of which installation method is used, follows:

- SMS automatically checks for the presence of Solaris Security Toolkit Version 4.2. If an earlier version is present, the installation process is temporarily halted and the user is prompted to remove the incompatible version before continuing. Once the incompatible version is removed, the installation process is restarted and Solaris Security Toolkit version 4.2 is automatically installed.
- FixModes and MD5 software are now automatically installed as a function of installing SMS 1.6.
- Due to improved filtering, do not disable ARP traffic on the I1 MAN network.

Assumptions and Limitations

The suggestions herein are based on several assumptions and limitations as to what can be done to secure Sun Fire system controllers, resulting in a supported configuration.

Note – The suggestions in this document are for System Management Services (SMS) 1.6 software, and differences between SMS 1.6 and previous releases are not discussed. It is suggested that all customers upgrade their software to SMS 1.6 when possible.

Solaris OS hardening can be interpreted in many ways. For purposes of developing a hardened SC configuration, we address hardening all possible Solaris OS options. That is, anything that can be hardened is hardened. When there are good reasons for leaving services and daemons as they are, we do not harden or modify them.

Note – Hardening Solaris OS configurations to the level described in this article might not be appropriate for your environment. For some environments, you might want to perform fewer hardening operations than suggested here. The configuration remains supported in these cases; however, additional hardening beyond what is suggested in this document is not supported.

You can customize a copy of the Sun Fire high-end servers SC module of the Solaris Security Toolkit to disable certain hardening scripts. It is strongly suggested that any modifications to the default modules be made in copies of those files, which will simplify upgrades to newer Solaris Security Toolkit versions.

Note – Standard security rules apply to the hardening of system controllers: *That which is not specifically permitted is denied*.

Additional software that you can install on the system controllers, such as Sun Remote Services Event Monitoring, Sun Remote Services Net Connect, and Sun Management Center software has been omitted from this document. We suggest that you carefully consider the security implications implicit with the installation of these types of software.

Obtaining Support

The SC configuration for Sun Fire high-end systems implemented by the Solaris Security Toolkit software (`sunfire_15k_sc-secure.driver`) is a Sun supported configuration. A hardened SC is supported *only* if the security modifications are performed using the Solaris Security Toolkit.

Initial or Fresh SMS Installation Using `smsinstall` Command (Secure by Default)

In this instance, the `smsinstall` command is used to install SMS 1.6 software. Automatic secure by default will occur wherein the system controllers of a system are automatically hardened and made secure as a function of the installation process.

The Sun Fire 15K and 12K SC module `sunfire_15k_sc-secure.driver` performs hardening tasks. This Solaris Security Toolkit driver is implemented by default and disables all those services which can be disabled without adversely affecting SMS. A user can enable as many services as required, but cannot disable more services than were disabled by the SMS installation software.

Customizing the Solaris Security Toolkit

You might determine that your system requires some of the services and daemons disabled by the Solaris Security Toolkit. To customize the Solaris Security Toolkit software to meet your particular requirements, see "Customizing the Solaris Security Toolkit Driver" on page 30.

Optionally Securing Domains

An option also exists to further harden a system by securing the system domains as indicated in the following Sun BluePrints™ Online articles available at:

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

- *Securing the Sun Fire high-end Domains*
- *Solaris Operating System Security – Updated for Solaris 8 (2/04) Operating System*
- *Solaris Operating System Security – Updated for Solaris 9 (4/04) Operating System*

SMS Upgrade Installation Using `smsupgrade` Command (Secure by Choice)

In this instance, the `smsupgrade` command is used to install SMS 1.6 software. Automatic hardening by default is not accomplished. However, Solaris Security Toolkit software is installed as a function of the upgrade and can be used to manually harden, undo hardening and audit the security posture of a system

The following security options are available:

Strongly suggested:

- Use Solaris Security Toolkit to secure the system controllers.

Optional:

- Secure domains.
- Disable all IP traffic between the SC and a domain by excluding that domain from the SC's MAN driver.

Optionally Securing Domains

For systems where domain separation is critical, we suggest disabling IP connectivity between the SC and specific domains that require separation.

To implement securing the system controllers, refer to "Using Solaris Security Toolkit to Secure the System Controller" on page 29. To implement the optional securing of domains refer to the following Sun BluePrints Online articles available at:

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

- *Securing the Sun Fire high-end Domains*
- *Solaris Operating System Security – Updated for Solaris 8 (2/04) Operating System*
- *Solaris Operating System Security – Updated for Solaris 9 (4/04) Operating System*

Using Solaris Security Toolkit to Secure the System Controller

To effectively secure system controllers, changes are required to both the Solaris OS software running on the system controllers and the configuration of the Sun Fire high-end platform. Customized modules added to Solaris Security Toolkit software simplify the Solaris OS installation and deployment of these suggestions. These modules automate the implementation of the security suggestions.

Solaris Security Toolkit software is always being updated. Solaris Security Toolkit version 4.2 is downloaded as a function of the `smsupgrade` command. However, to ensure you have the latest version of Solaris Security Toolkit when you are installing SMS, see the following web site:

`http://www.sun.com/security/jass`

If you download a later version, install it to the `Bundled_Products` directory of the SMS zip file, replacing the old package with the same name. You must decompress the Solaris Security Toolkit packages after downloading them.

Note – For instructions on installing the Solaris Security Toolkit packages manually, refer to the *Solaris Security Toolkit Installation Guide*.

Note – Disable failover before hardening either of the system controllers. Re-enable failover only after both system controllers are hardened *and* tested.

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

Solaris Security Toolkit Software

Version 4.2 of the Solaris Security Toolkit software is included as a part of the SMS zip file as a function of the `smsupgrade` command and installed on the system controllers. Informational messages show the progress of the installation of Solaris Security Toolkit, and advise users to use the Solaris Security Toolkit software to automate installing other security software and implementing the Solaris OS modifications for hardening the system controllers.

If the SC already has a version of Solaris Security Toolkit installed, smsupgrade will abort before installing SMS packages and ask users to save any Solaris Security Toolkit customizations, if any, and remove the old Solaris Security Toolkit package before reinvoking smsupgrade.

Customizing the Solaris Security Toolkit Driver

You might determine that your system requires some of the services and daemons disabled by the Solaris Security Toolkit, or you might want to enable any of the inactive scripts available in the Solaris Security Toolkit.

To enable various other services on the SC to customize the hardening, refer to Chapter 7 of the *Solaris Security Toolkit Administrative Manual*. If there are some services that must remain enabled, and the Solaris Security Toolkit automatically disables them, you can override the defaults.

To prevent the toolkit from disabling a service, comment out the call to the appropriate finish script in the driver. For example, if your environment requires Network File System (NFS)-based services, you can leave them enabled. Comment out the disable-nfs-server.fin and disable-rpc.fin scripts by appending a # sign before them in the copy of the sunfire_15k_domain-hardening.driver script.

For more information about Solaris Security Toolkit editing and creating driver scripts, refer to the Solaris Security Toolkit documentation.

Note – During the installation and modifications implemented in this section, all nonencrypted access mechanisms to the SC–such as Telnet and FTP–are disabled. The hardening steps do not disable console serial access over SC serial ports.

Implementing any modifications to the system controllers requires modifying the files included with the Solaris Security Toolkit. The following procedures provide instructions for using some of these options.

To Disable I1 Traffic (Domain Exclusion)

Domain exclusion requires that you unconfigure domain network interfaces to be excluded from the I1 network configuration and then restart the `mand` daemon.

Note – Earlier SMS versions could use the SST software to exclude domains from communicating with the system controller (disabling the I1 network between a domain and the SC). This functionality is not supported in the latest SST version and must now be performed manually as indicated in this procedure.

- **As user, specify NONE as the MAN hostname for the domain to be excluded.**

  For example, for domain A:

```
#smsconfig -m I1 A

 Enter the MAN hostname for DA-I1 [ DA-I1 ]: NONE

 Network: I1 DA-I1
 Hostname: NONE  IP Address: NONE

 Do you want to accept these settings? [y,n]y


 #pkill -HUP mand
```

To Enable FTP or Telnet

Note – The Solaris Security Toolkit `user.init` file should be edited to contain any user-defined variables such as the following.

- To enable FTP, set Solaris Security Toolkit `user.init` file as follows:
  JASS_SVCS_ENABLE = ftp
- To enable Telnet, set Solaris Security Toolkit `user.init` file as follows:
  JASS_SVCS_ENABLE = telnet

For more information, refer to “Customizing the Hardening Configuration” in Chapter 7 of the *Solaris Security Toolkit Administration Guide*.

▼ To View the Contents of the Driver File

- **To view the contents of the driver file and obtain information about the Solaris OS modifications, refer to the Solaris Security Toolkit documentation available either in the `/opt/SUNWjass/Documentation` directory or through the web at:**

`http:/www.sun.com/security/jass`

▼ To Undo a Solaris Security Toolkit Run

Each Solaris Security Toolkit run creates a run directory in `/var/opt/SUNWjass/run`. The names of these directories are based on the date and time the run is initiated. In addition to displaying the output to the console, the Solaris Security Toolkit software creates a log file in the `/var/opt/SUNWjass/run` directory.

Caution – Do not modify the contents of the `/var/opt/SUNWjass/run` directories under any circumstances. Modifying the files can corrupt the contents and cause unexpected errors when you use Solaris Security Toolkit software features such as `undo`.

The files stored in the `/var/opt/SUNWjass/run` directory track modifications performed on the system and enable the `jass-execute` undo feature.

Note – By default, the Solaris Security Toolkit overwrites any files backed up while earlier runs were being undone. In some cases, this action overwrites changes made to files since the run was performed. If you have concerns about overwriting changes, use the `-n` (no force) option to prevent modified files from being overwritten. Refer to the Solaris Security Toolkit documentation for more details about this option.

- **To undo a single run or a series of runs, use the `jass-execute -u` command.**

  For example, on a system where two separate Solaris Security Toolkit runs are performed, you could undo the second run, as shown in the following example:

```
# pwd
/opt/SUNWjass
# ./jass-execute -u
Please select a JASS run to restore through:
1. September 25, 2005 at 06:28:12
(/var/opt/SUNWjass/run/20050925062812)
2. December 10, 2005 at 19:04:36
(/var/opt/SUNWjass/run/20051210190436)
3. Restore from all of them
Choice{‘q‘ to exit)? 2
./jass-execute: NOTICE: Restoring to previous run
//var/opt/SUNWjass/run/20021210190436

============================================================
undo.driver: Driver started.
============================================================
[...]
```

  Refer to the Solaris Security Toolkit documentation for details on the capabilities and options available in the `jass-execute` command.

CHAPTER 3

SMS Administrative Privileges

This chapter provides a brief overview of administrative privileges as they pertain to SMS 1.6 and the Sun Fire high-end server system. This chapter contains the following sections:

Administrative Privileges Overview

SMS splits domain and platform administrative privileges. It is possible to assign separate administrative privileges for system management over each domain and for system management over the entire platform. There is also a subset of privileges available for platform operator and domain configurator-class users. Administrative privileges are granted so that audits can identify the individual who initiated any action.

SMS uses site-established Solaris user accounts and grants administrative privileges to those accounts through the use of Solaris *group* memberships. This allows a site considerable flexibility with respect to creating and consolidating default privileges. For example, by assigning the same Solaris group to represent the administrator privilege for more than one domain, groups of domains can be administered by one set of domain administrators.

SMS also allows the site considerable flexibility in assigning multiple administrative roles to individual administrators. For example, you can set up a single user account with group membership in the union of all configured administrative privilege groups.

- The *platform administrator* has control over the platform hardware. Limitations have been established with respect to controlling the hardware used by a running domain, but ultimately the platform administrator can shut down a running domain by powering off server hardware.
- Each *domain administrator* has access to the Solaris console for that domain and the privilege to exert control over the software that runs in the domain or over the hardware assigned to the domain.
- Levels of each type of administrative privilege provide a subset of status and monitoring privileges to a *platform operator* or *domain configurator.*

SMS provides an administrative privilege that grants access to functions provided exclusively for servicing the product in the field.

Administrative privilege configuration can be changed at will, by the superuser, using `smsconfig -g` without the need to stop or restart SMS.

SMS implements Solaris access control list (ACL) software to configure directory access for SMS groups using the `-a` and `-r` options of the `smsconfig` command. ACLs restrict access to platform and domain directories providing file system security. For information on ACLs, refer to the *Solaris 9 System Administration Guide: Security Services*.

Platform Administrator Group

The group identified as the platform administrator (`platadmn`) group provides configuration control, a means to obtain environmental status, the ability to assign boards to domains, power control, and other generic service processor functions. In short, the platform administrator group has all platform privileges excluding domain control and access to installation and service commands (FIGURE 3-1).

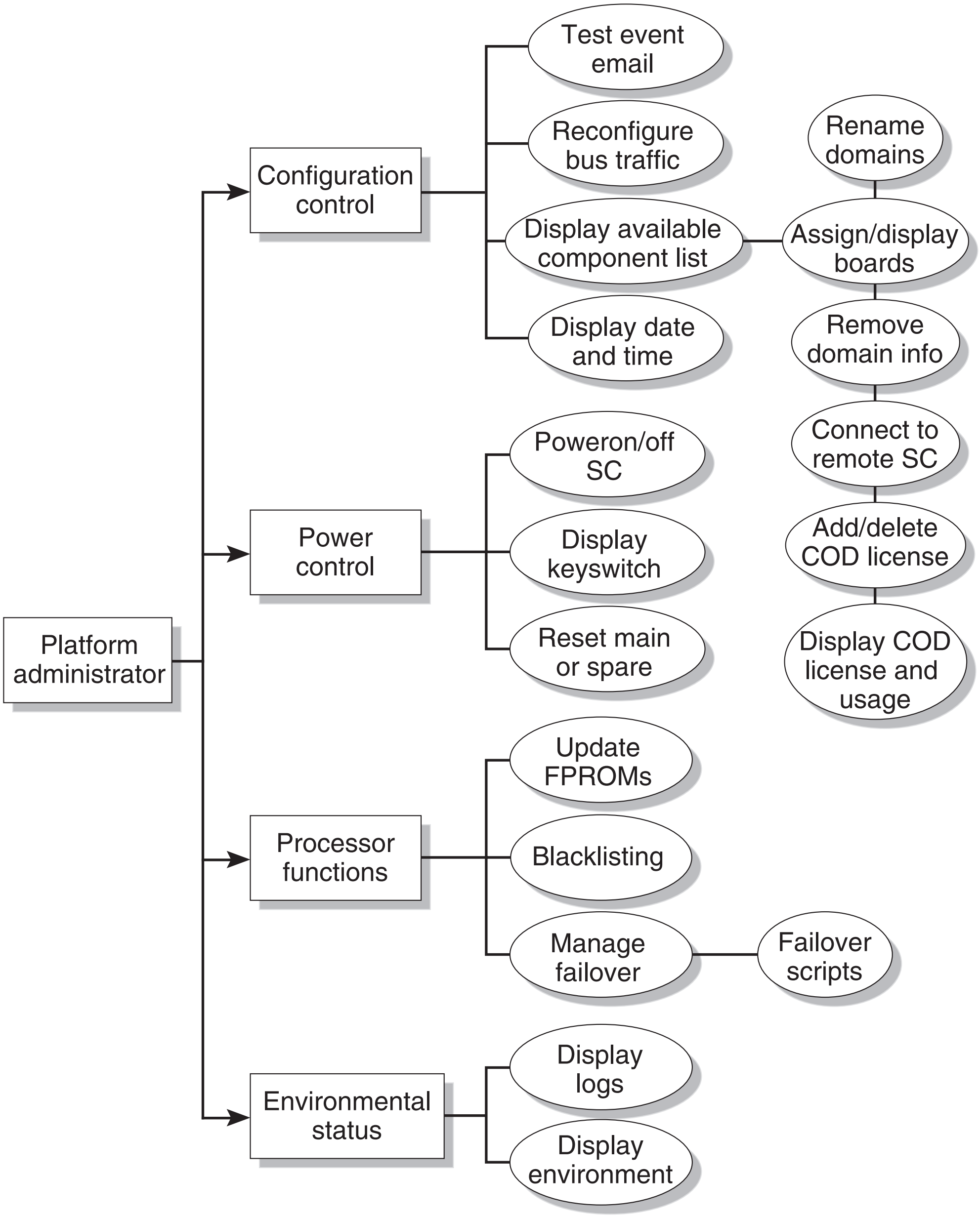


FIGURE 3-1 Platform Administrator Privileges

Platform Operator Group

The platform operator (platoper) group has a subset of platform privileges. This group has no platform control other than being able to perform power control. Therefore, this group is limited to platform power and status privileges (FIGURE 3-2).

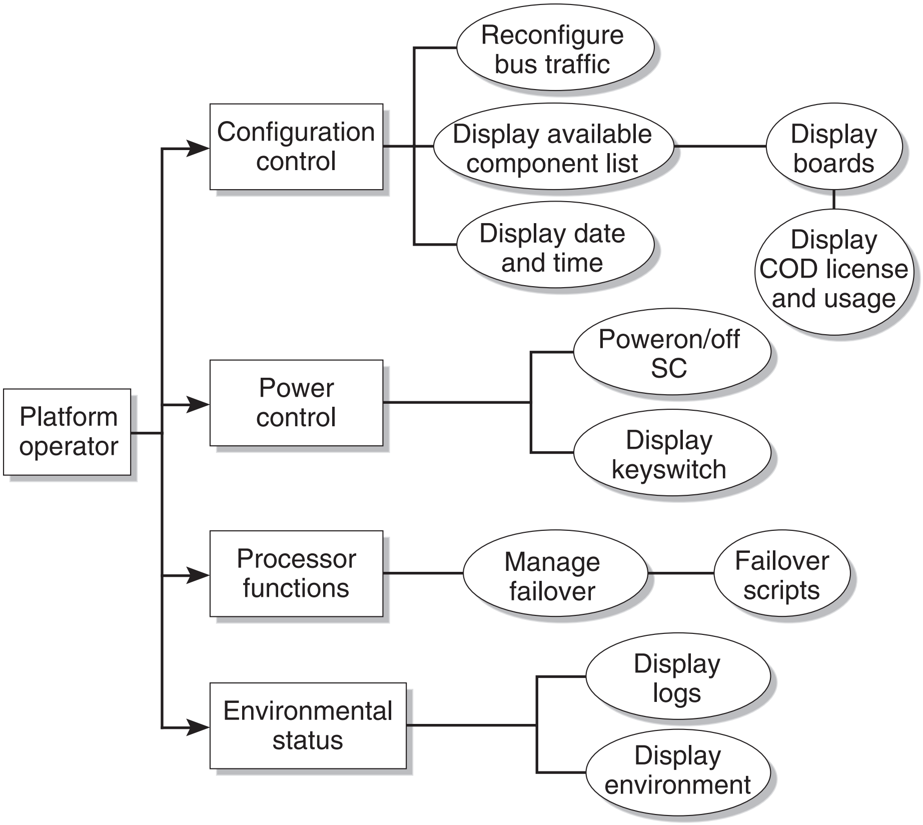


FIGURE 3-2 Platform Operator Privileges

Platform Service Group

The platform service (platsvc) group possesses platform service command privileges in addition to limited platform control and platform configuration status privileges (FIGURE 3-2).

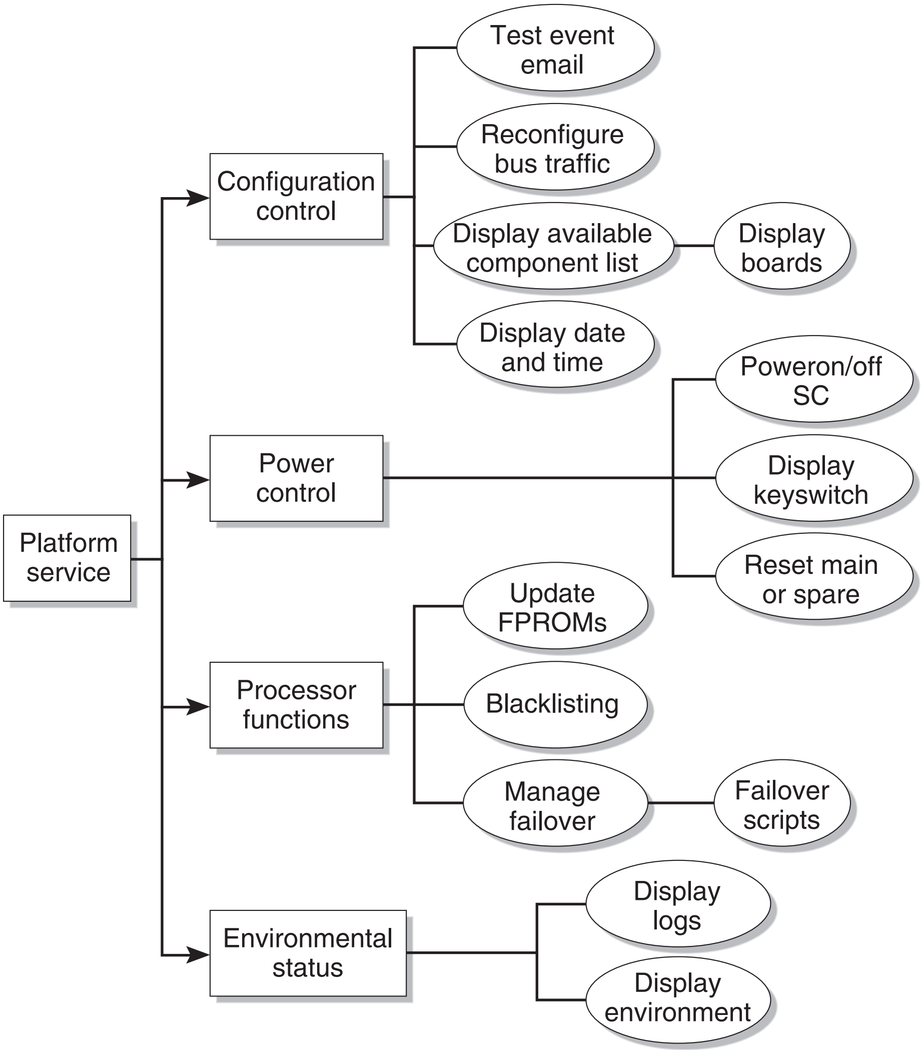


FIGURE 3-3 Platform Service Privileges

Domain Administrator Group

The domain administrator (`dmn`*[domain-id]*`admn`) group provides the ability to access the console of its respective domain as well as perform other operations that affect, directly or indirectly, the respective domain. Therefore, the domain administrator group can perform domain control, domain status, and console access, but cannot perform platform-wide control or platform resource allocation (FIGURE 3-4).

There are 18 possible Sun Fire domains, A-R, identified by *domain-id*. Therefore, there are 18 domain administrator groups, each providing strict access over their respective domains.

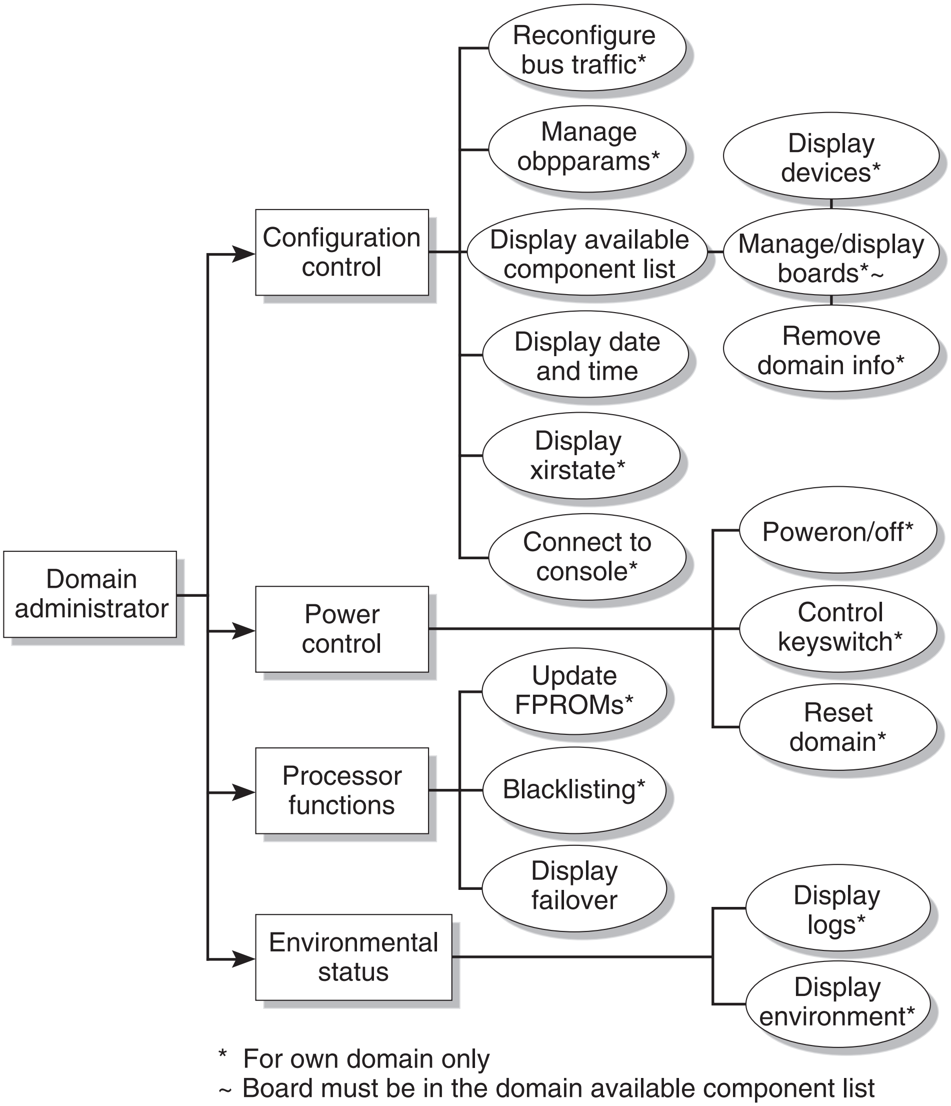


FIGURE 3-4 Domain Administrator Privileges

Domain Configuration Group

The domain configuration (dmn*[domain-id]*rcfg) group has a subset of domain administration group privileges. This group has no domain control other than being able to power control boards in its domain or (re)configure boards into or from its domain (FIGURE 3-5).

There are 18 possible Sun Fire domains, identified by *domain-id*s. Therefore, there are 18 domain configuration groups, each allowing strict access over their respective domains.

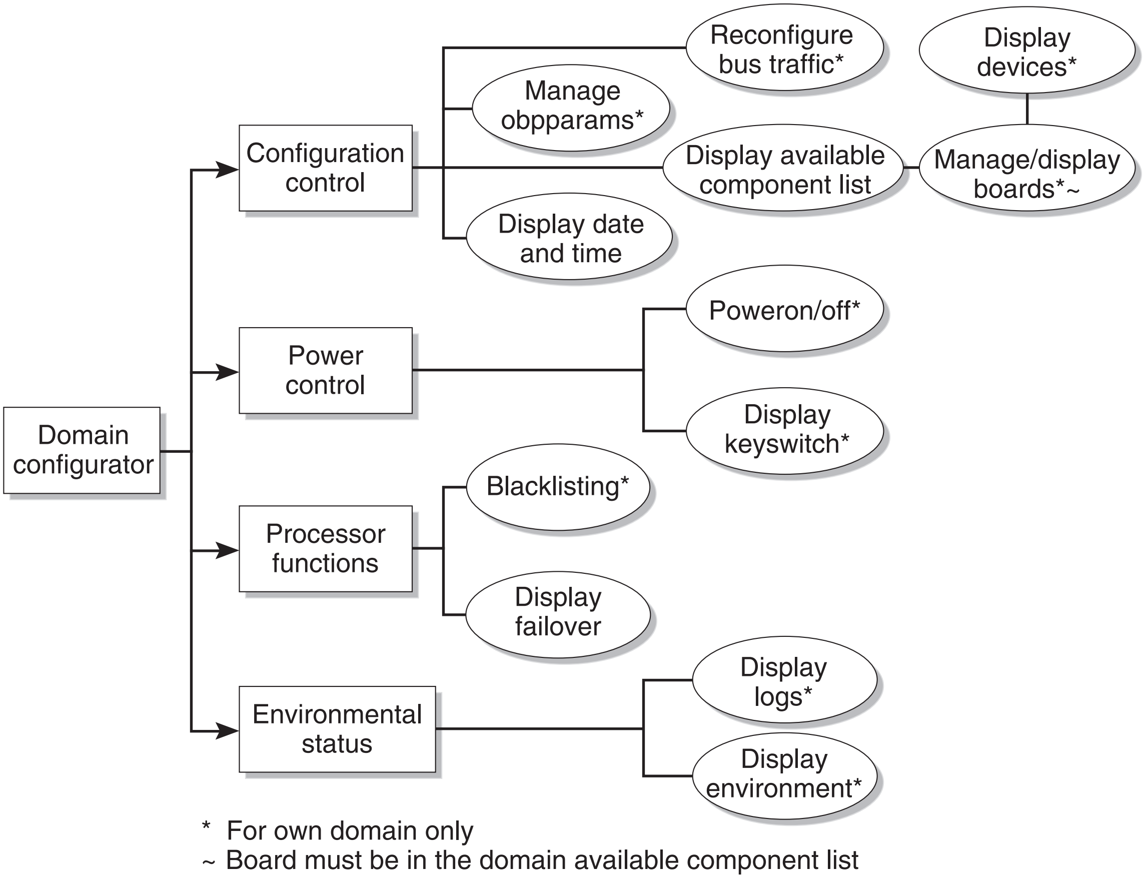


FIGURE 3-5 Domain Configurator Privileges

Superuser Privileges

The superuser privileges are limited to installation, help, and status privileges (FIGURE 3-6).

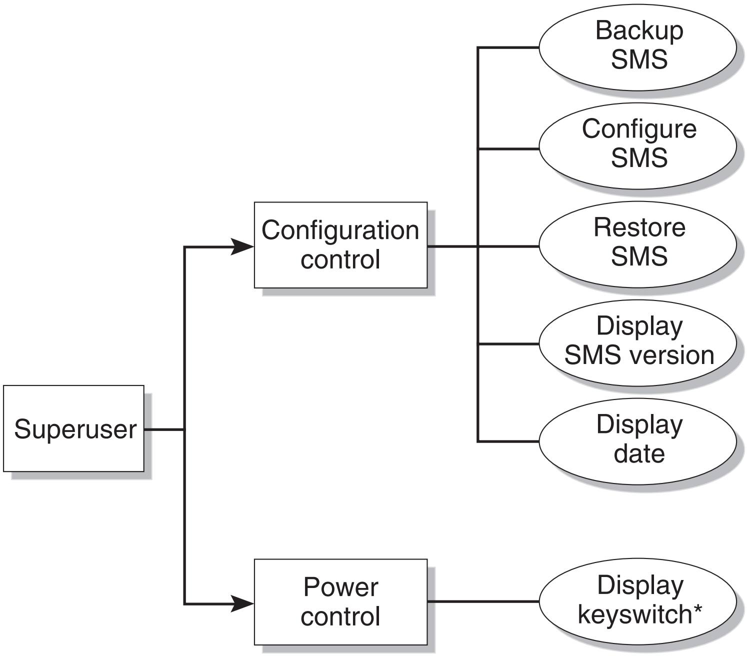


FIGURE 3-6 Superuser Privileges

All Privileges

TABLE 3-1 lists all group privileges.

TABLE 3-1 All Group Privileges

Command	Group Privileges					
	Platform Administrator	**Platform Operator**	**Domain Administrator**	**Domain Configurator**	**Platform Service**	**Superuser**
`addboard`	A user with only platform administrator privileges can perform only the `-c` *assign*.	No	Users with only domain *X* administrator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	Users with only domain *X* configurator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	No	No
`addcodlicense`	Yes	No	No	No	No	No
`addtag`	Yes	No	No	No	No	No
`cancelcmdsync`	Yes	Yes	Yes	Yes	Yes	No
`console`	No	No	Yes (for own domain)	No	No	No

TABLE 3-1 All Group Privileges *(Continued)*

Command	Group Privileges					
	Platform Administrator	**Platform Operator**	**Domain Administrator**	**Domain Configurator**	**Platform Service**	**Superuser**
`deleteboard`	A user with only platform administrator privileges can perform `-c` *unassign* only if the boards are in the *assign*ed state and not active in a running domain.	No	Users with only domain *X* administrator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	Users with only domain *X* configurator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	No	No
`deletecodlicense`	Yes	No	No	No	No	No
`deletetag`	Yes	No	No	No	No	No
`disablecomponent`	Yes (platform only)	No	Yes (for own domain)	Yes (for own domain)	No	No
`enablecomponent`	Yes (platform only)	No	Yes (for own domain)	Yes (for own domain)	No	No
`flashupdate`	Yes	No	Yes (for own domain)	No	No	No
`help`	Yes	Yes	Yes	Yes	Yes	Yes
`initcmdsync`	Yes	Yes	Yes	Yes	Yes	No

TABLE 3-1 All Group Privileges *(Continued)*

Command	Group Privileges					
	Platform Administrator	**Platform Operator**	**Domain Administrator**	**Domain Configurator**	**Platform Service**	**Superuser**
`moveboard`	A user with only platform administrator privileges can perform the `-c` *assign* only if the board is in the *assign*ed state and not active in the domain the board is being removed from.	No	Users must belong to both domains affected. If the boards are not already assigned to the domain the boards are being moved into, the boards must be in the available component list of that domain.	Users must belong to both domains affected. If the boards are not already assigned to the domain the boards is being moved into, the boards must be in the available component list of that domain.	No	No
`poweron`	Yes	No	Yes (for own domain)	Yes (for own domain)	No	No
`poweroff`	Yes	No	Yes (for own domain)	Yes (for own domain)	No	No
`rcfgadm`	A user with only platform administrator privileges can perform`-x` *assign*. The user can execute `-x` *unassign* only if the boards are in the *assign*ed state and not active in a running domain.	No	Users with only domain *X* administrator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	Users with only domain *X* configurator privileges can execute this command on their respective domain. If the boards are not already assigned to the domain, the boards must be in the available component list of that domain.	No	No
`reset`	No	No	Yes (for own domain)	No	No	No

TABLE 3-1 All Group Privileges *(Continued)*

Command	Group Privileges					
	Platform Administrator	**Platform Operator**	**Domain Administrator**	**Domain Configurator**	**Platform Service**	**Superuser**
`resetsc`	Yes	No	No	No	No	No
`runcmdsync`	Yes	Yes	Yes	Yes	Yes	No
`savecmdsync`	Yes	Yes	Yes	Yes	Yes	No
`setbus`	Yes	No	Yes (for own domain)	Yes (for own domain)	No	No
`setcsn`	Yes	No	No	No	Yes	No
`setdatasync`	Yes	Yes	Yes	Yes	Yes	No
`setdate`	Yes	No	Yes (for own domain)	No	No	No
`setdefaults`	Yes	No	Yes (for own domain)	No	No	No
`setfailover`	Yes	No	No	No	No	No
`setkeyswitch`	No	No	Yes (for own domain)	No	No	No
`setobpparams`	No	No	Yes (for own domain)	Yes (for own domain)	No	No
`setupplatform`	Yes	No	No	No	No	No
`showboards`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showbus`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showcmdsync`	Yes	Yes	Yes	Yes	Yes	No
`showcodlicense`	Yes	Yes	No	No	No	No
`showcodusage`	Yes	Yes	No	No	No	No
`showcomponent`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showdatasync`	Yes	Yes	Yes	Yes	Yes	No
`showdate`	Yes (platform only)	Yes (platform only)	Yes (for own domain)	Yes (for own domain)	Yes (platform only)	No
`showdevices`	No	No	Yes (for own domain)	Yes (for own domain)	No	No

TABLE 3-1 All Group Privileges *(Continued)*

Command	Group Privileges					
	Platform Administrator	**Platform Operator**	**Domain Administrator**	**Domain Configurator**	**Platform Service**	**Superuser**
`showenvironment`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showfailover`	Yes	Yes	No	No	Yes	No
`showkeyswitch`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showlogs`	Yes (platform only)	Yes (platform only)	Yes (for own domain)	Yes (for own domain)	Yes (platform only)	No
`showobpparams`	No	No	Yes (for own domain)	Yes (for own domain)	No	No
`showplatform`	Yes	Yes	Yes (for own domain)	Yes (for own domain)	Yes	No
`showxirstate`	No	No	Yes (for own domain)	No	No	No
`smsbackup`	No	No	No	No	No	Yes
`smsconfig`	No	No	No	No	No	Yes
`smsconnectsc`	Yes	No	No	No	No	No
`smsrestore`	No	No	No	No	No	Yes
`smsversion`	No	No	No	No	No	Yes
`testemail`	Yes	No	No	No	Yes	No

CHAPTER 4

SMS Internals

SMS operations are generally performed by a set of daemons and commands. This chapter provides an overview of how SMS works and describes the SMS daemons, processes, commands, and system files. For more information, refer to the *System Management Services (SMS) 1.6 Reference Manual*.

Caution – Changes made to files in /opt/SUNWSMS can cause serious damage to the system. Only very experienced system administrators should risk changing the files described in this chapter.

This chapter contains the following sections:

- "Startup Flow" on page 49
- "SMS Daemons" on page 50

Startup Flow

The following events take place when the SMS boots:

1. User powers on the Sun Fire high-end (CPU/disk and DVD-ROM) platform. The Solaris OS on the SC boots automatically.

2. During the boot process, the /etc/init.d/sms script is called. This script, for security reasons, disables forwarding, broadcast, and multicasting over the MAN network. The script then starts the SMS software by invoking a background process, which starts and monitors ssd. ssd is the SMS startup daemon responsible for starting and monitoring all the SMS daemons and servers.

3. ssd(1M) in turn invokes the following daemons and processes: mld, pcd, hwad, tmd, dsmd, esmd, mand, osd, dca, efe,codd, efhd, elad, erd, smnptd, picld, and wcapp.

For more information about the SMS daemons, see "SMS Daemons" on page 50. For more information about efe, refer to the latest Sun Management Center documentation available at: http://docs.sun.com

4. Once the daemons are running, you can use SMS commands such as console.

   SMS startup can take a few minutes during which time any commands run will return an error message indicating that SMS has not completed startup. The message "SMS software start-up complete" is posted to the platform log when startup is complete, and can be viewed using the showlogs(1M) command.

SMS Daemons

The SMS 1.6 daemons play a central role on Sun Fire high-end systems. Daemons are persistent processes that provide SMS services to clients using an API.

Note – SMS daemons are started by ssd and should *not* be started manually from the command line. Issuing a kill command against any daemon will seriously affect the robustness of SMS software and should not be done unless specifically requested by Sun service personnel.

Daemons are always running, initiated at system startup and restarted whenever necessary.

Each daemon is fully described in its corresponding man page with the exception of efe, which is referenced separately in the Sun Management Center documentation.

This section looks at the SMS daemons, their relationship to one another, and which CLIs access them.

FIGURE 4-1 illustrates the Sun Fire high-end system software components and their high-level interaction.

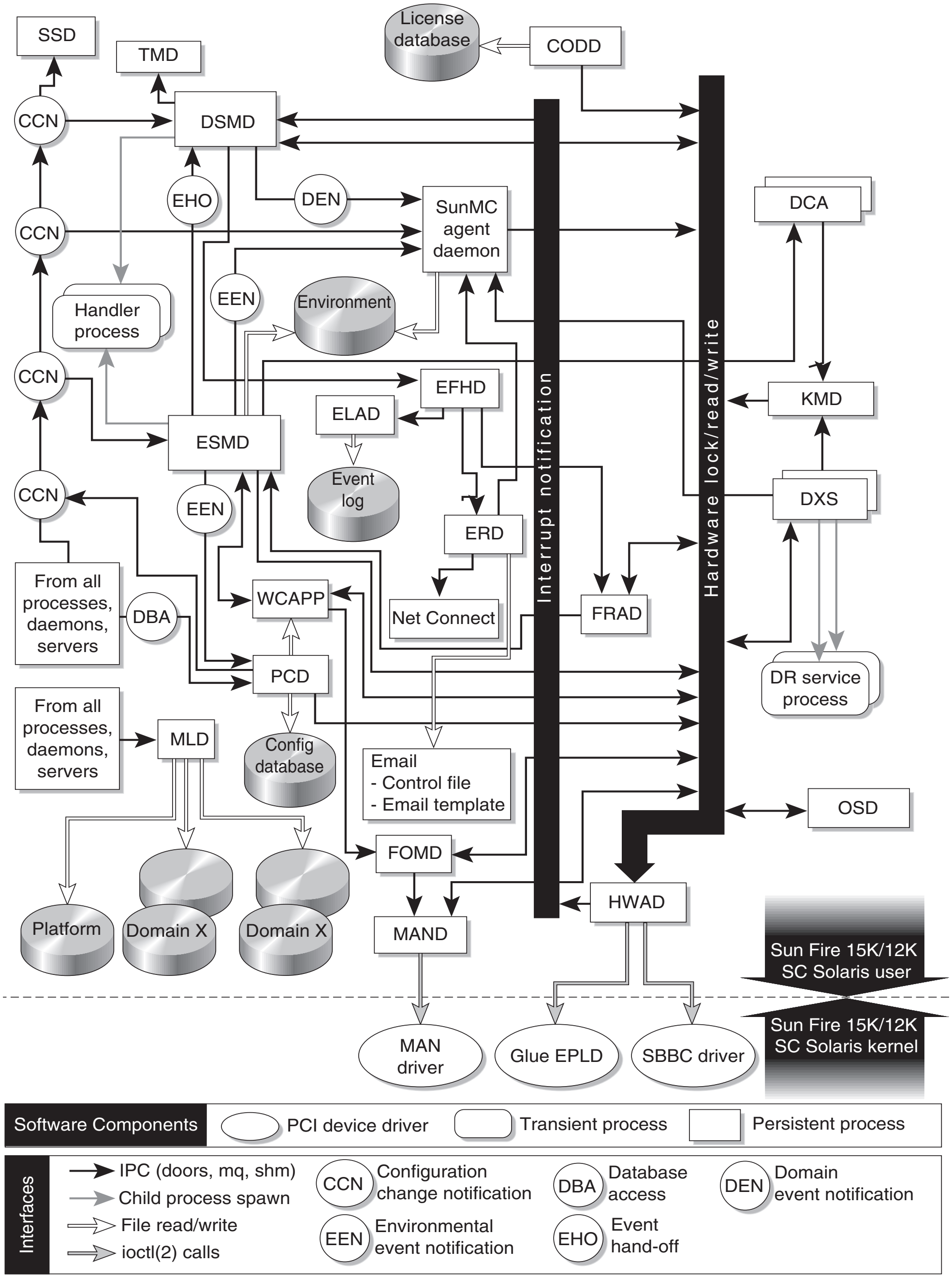


FIGURE 4-1 Sun Fire High-End System Software Components

Note – The domain *X* server (dxs) and domain configuration agent (dca), while not daemons, are essential server processes and included in the following table and section. Each domain runs an instance of dxs and dca. The maximum number of instances (at one instance of each daemon per domain) is 18 on the Sun Fire 15K/E25K and 9 on the Sun Fire 12K/E20K.

TABLE 4-1 Daemons and Processes

Daemon Name	Description
codd	The capacity on demand daemon monitors the COD resources being used and verifies that the resources used are in agreement with the licenses in the COD license database file. This daemon is started automatically by the SMS startup daemon.
dca	The domain configuration agent provides a communication mechanism between the dca on the system controller and the domain configuration server (dcs) on the specified domain. There is a separate instance of the dca daemon for every domain, up to a maximum of 18 domains. This daemon is started automatically by the SMS startup daemon.
dsmd	The domain status monitoring daemon monitors domain status, CPU reset conditions, and the Solaris OS heartbeat for up to 18 domains on the Sun Fire 15K/E25K and up to 9 on the Sun Fire 12K/E20K. This daemon is started automatically by the SMS startup daemon.
dxs	The domain *X* server provides software support for a domain including dynamic reconfiguration (DR), hot-pluggable PCI I/O board support, domain driver requests and events, and virtual console support. There is a separate instance of the dxs daemon for each domain up to 18 domains on the Sun Fire 15K/E25K, and up to 9 instances on the Sun Fire 12K/E20K. This daemon is started automatically by the SMS startup daemon.
efe	The event front end daemon is part of Sun Management Center and acts as an intermediary between the Sun Management Center agent and SMS. For more information about efe, refer to the *Sun Management Center 3.5 Supplement for Sun Fire 15K/12K Systems.*
efhd	The error and fault-handling daemon performs automatic error diagnosis and updates the component health status of components associated with a fault. This daemon is started automatically by the SMS startup daemon.
elad	The event log access daemon controls access to the SMS event log, which records fault and error events identified by the automatic diagnosis (AD) engine. This daemon also starts a new event log file whenever the current event log reaches its size limit and deletes the oldest archive file.

TABLE 4-1 Daemons and Processes *(Continued)*

Daemon Name	Description
`erd`	The event reporting daemon reports fault event messages to platform and domain logs, provides fault information to Sun Management Center and Sun Remote Services Net Connect, and delivers email reports that contain fault event messages. This daemon is started automatically by the SMS startup daemon.
`esmd`	The environmental status monitoring daemon monitors system cabinet environmental conditions, such as fan trays, power supplies, and temperatures. This daemon is started automatically by the SMS startup daemon.
`fomd`	The failover monitoring daemon detects faults on the local and remote SCs and takes appropriate action (initiating a failover.) This daemon is started automatically by the SMS startup daemon.
`frad`	The FRU access daemon provides the mechanism by which SMS daemons can access any field-replaceable unit (FRU) serial electrically erasable programmable read-only memory (SEEPROM) on a Sun Fire high-end system. This daemon is started automatically by the SMS startup daemon.
`hwad`	The hardware access daemon provides hardware access to SMS daemons and a mechanism for all daemons to exclusively access, control, monitor, and configure the hardware. This daemon is started automatically by the SMS startup daemon.
`kmd`	The key management daemon manages the IPSec security associations (SAs) needed to secure the communication between the SCs, and servers running on a domain. This daemon is started automatically by the SMS startup daemon.
`mand`	The management network daemon supports the MAN drivers, providing required network configuration. The role played by `mand` is specified by the `fomd`. This daemon is started automatically by the SMS startup daemon.
`mld`	The messages logging daemon provides message logging support for the platform and domains. This daemon is started automatically by the SMS startup daemon.
`osd`	The OpenBoot PROM server daemon provides software support for the OpenBoot PROM process running on a domain through the mailbox that resides on the domain. When the domain OpenBoot PROM writes requests to the mailbox, the `osd` daemon executes those requests. On the main SC it is responsible for booting domains. This daemon is started automatically by the SMS startup daemon.
`pcd`	The platform configuration database daemon provides and manages controlled access to platform, domain, and system board configuration data. This daemon is started automatically by the SMS startup daemon.

TABLE 4-1 Daemons and Processes *(Continued)*

Daemon Name	Description
`ssd`	The SMS startup daemon starts, stops, and monitors all the key SMS daemons and servers.
`tmd`	The task management daemon provides task management services, such as scheduling for SMS. `setkeyswitch` and other commands use `tmd` to schedule hardware power-on self-test invocations. This daemon is started automatically by the SMS startup daemon.
`wcapp`	The optional wPCI application daemon implements Sun Fire Link clustering functionality and provides information to the external Sun Fire Link fabric manager server. For more information about `wcapp`, refer to the *Sun Fire Link Fabric Administrator's Guide.*

Capacity on Demand Daemon

The capacity on demand daemon, `codd` (1M), is a process that runs on the main system controller (SC).

This process does the following:

- Monitors the COD resources being used and verifies that the resources used are in agreement with the licenses in the COD license database
- Provides information on installed licenses, resource use, and board status
- Handles the requests to add or delete COD license keys
- Configures headroom quantities and domain right-to-use (RTU) license reservations

FIGURE 4-2 illustrates the `CODD` client-server relationships to the SMS daemons and CLI commands.

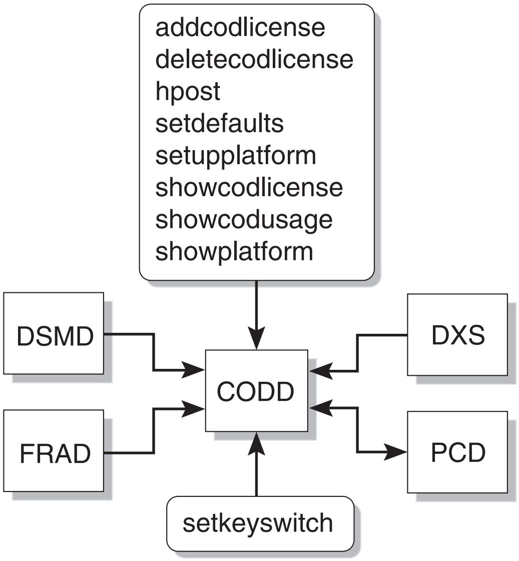


FIGURE 4-2 CODD Client-Server relationships

Domain Configuration Agent

The domain configuration agent daemon, dca(1M), supports remote dynamic reconfiguration (DR) by enabling communication between applications and the domain configuration server (dcs) running on a Solaris 8, 9, or 10 domain. One dca per domain runs on the SC. Each dca communicates with its dcs over the Management Network (MAN).

ssd(1M) starts dca when the domain is brought up. ssd restarts dca if it is terminated while the domain is still running. dca is terminated when the domain is shut down.

dca is an SMS application that waits for dynamic reconfiguration requests. When a DR request arrives, dca creates a dcs session. Once a session is established, dca forwards the request to dcs. dcs attempts to honor the DR request and sends the results of the operation to the dca. Once the results have been sent, the session is ended. The remote DR operation is complete when dca returns the results of the DR operation.

FIGURE 4-3 illustrates the DCA client-server relationships to the SMS daemons and CLIs.

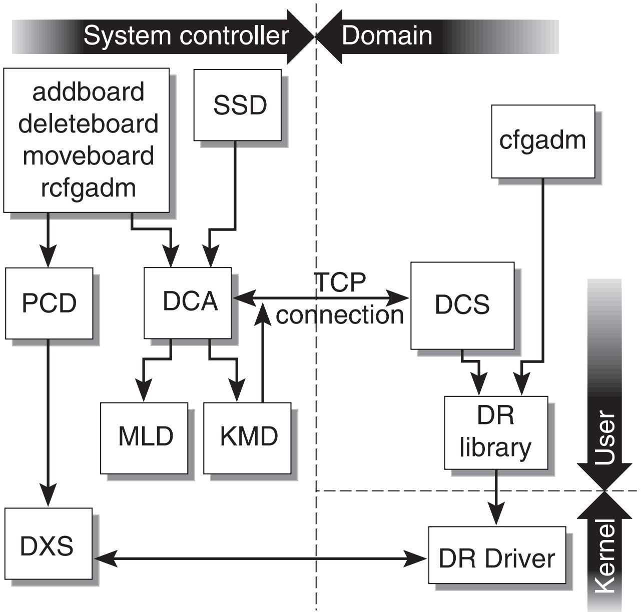


FIGURE 4-3 `DCA` Client-Server Relationships

Domain Status Monitoring Daemon

The domain status monitoring daemon, `dsmd`(1M), monitors domain state signatures, CPU reset conditions, and Solaris heartbeat for up to 18 domains on a Sun Fire 15K and up to 9 on a Sun Fire 12K system. This daemon also handles domain stop events related to hardware failure.

`dsmd` detects timeouts that can occur in reboot transition flow and panic transition flow, and handles various domain hung conditions.

`dsmd` notifies the domain *X* server (`dxs`(1M)) and Sun Management Center of all domain state changes, and automatically recovers the domain based on the domain state signature, domain stop events, and automatic system recovery (ASR) policy. ASR policy consists of those procedures that restore the system to running all properly configured domains after one or more domains have been rendered inactive. This inactivity can be due to software or hardware failures or to unacceptable environmental conditions. For more information, see "Automatic System Recovery (ASR)" on page 165 and "Domain Stop Events" on page 214.

`dsmd` also passes automatic diagnosis (AD) information related to the domain stop to `efhd`.

FIGURE 4-4 illustrates DSMD client-server relationships to the SMS daemons and CLIs.

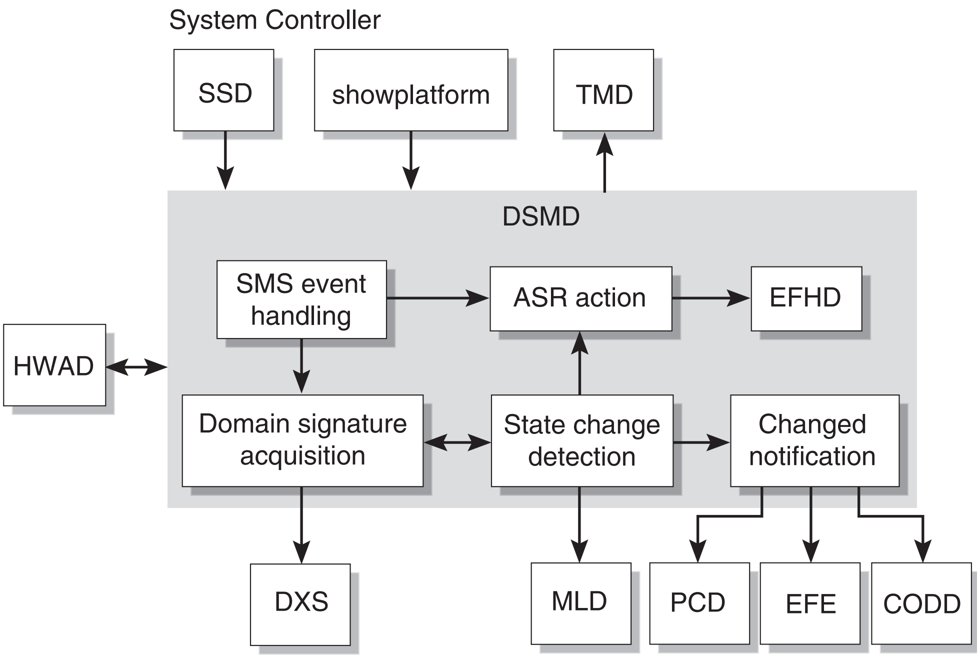


FIGURE 4-4 DSMD Client-Server Relationships

Domain X Server

The domain X server, dxs(1M), provides software support for a running domain. This support includes virtual console functionality, dynamic reconfiguration support, and HPCI support. dxs handles domain driver requests and events. dxs provides an interface for getting and setting HPCI slot status. The slot status includes cassette presence, power, frequency, and health of the cassette. This interface makes it possible to power control HPCI cassettes for hot-plug operations.

The virtual console functionality enables one or more users running the console program to access the domain's virtual console. dxs acts as a link between SMS console applications and the domain virtual console drivers.

A Sun Fire 15K system can support up to 18 different domains. A Sun Fire 12K system can support up to 9 domains. Each domain might require software support from the SC, and dxs provides that support. The following domain-related projects require dxs support:

- DR
- HPCI
- Virtual console

There is one domain *X* server for each Sun Fire high-end system domain. dxs is started by ssd for every active domain, that is, a domain running OS software, and terminated when the domain is shut down.

FIGURE 4-5 illustrates DXS client-server relationships to the SMS daemons.

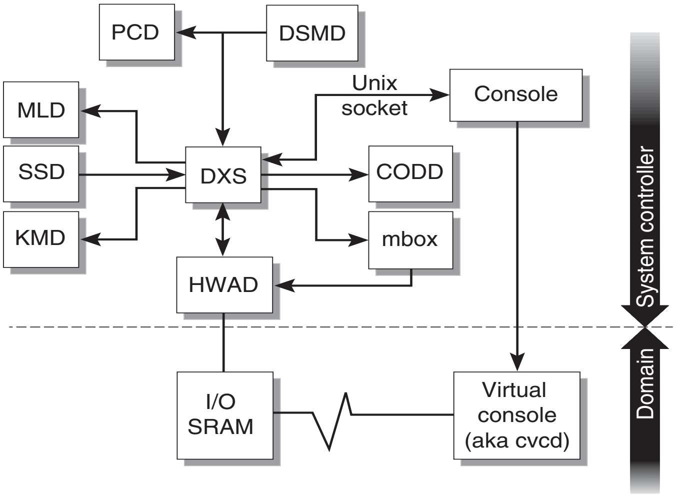


FIGURE 4-5 DXS Client-Server Relationships

Error and Fault Handling Daemon

The error and fault handling daemon, efhd(1M), does the following:

- Performs automatic error diagnosis based on the domain stop information passed by dsmd(1M)
- Updates the component health status for those components that have been associated with a fault, as determined by the diagnosis engine (SMS or the Solaris OS) or by POST
- Passes the fault event to erd(1M) for error reporting

FIGURE 4-5 illustrates EFHD client-server relationships to the SMS daemons.

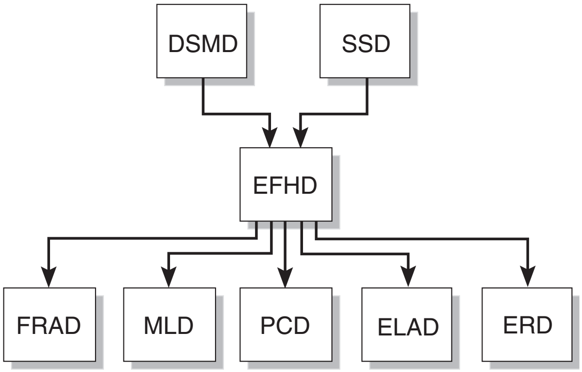


FIGURE 4-6 EFHD Client-Server Relationships

Event Log Access Daemon

The event log access daemon, elad(1M), controls access to the SMS event log, which records fault and error events identified by the automatic diagnosis (AD) or POST diagnosis engines on a Sun Fire high-end system. elad also archives events when the event log fills.

FIGURE 4-7 illustrates the ELAD client-server relationships to the SMS daemons and CLI commands.

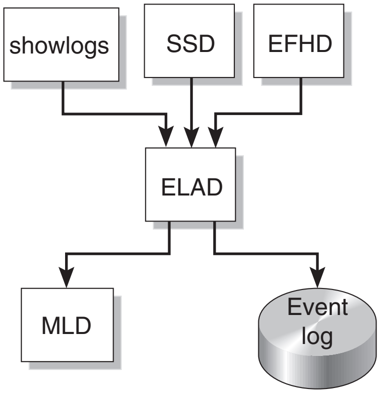


FIGURE 4-7 ELAD Client-Server Relationships

Event Reporting Daemon

The event reporting daemon, erd(1M), provides reporting services that deliver fault event text messages to the platform and domain logs, fault event information to Sun Management Center and Sun Remote Services (SRS) Net Connect, and email that contains fault event messages.

erd reads the email control file and the email template file each time email event notification occurs.

FIGURE 4-8 illustrates the ERD client-server relationships to the SMS daemons.

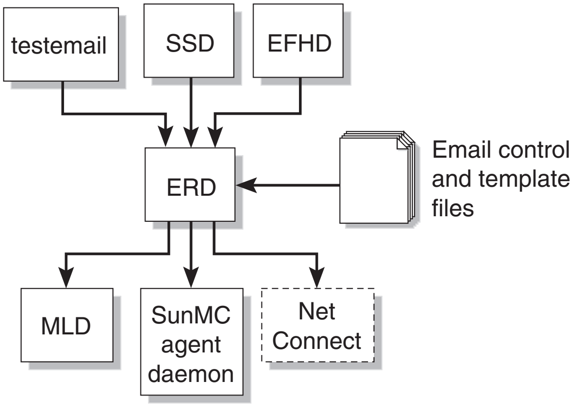


FIGURE 4-8 ERD Client-Server Relationships

Environmental Status Monitoring Daemon

The environmental status monitoring daemon, esmd(1M), monitors system cabinet environmental conditions, for example, voltage, temperature, fan tray, power supply and clock phasing. esmd logs abnormal conditions and takes action to protect the hardware, if necessary.

See "Environmental Events" on page 210 for more information about esmd.

FIGURE 4-9 illustrates ESMD client-server relationships to the SMS daemons.

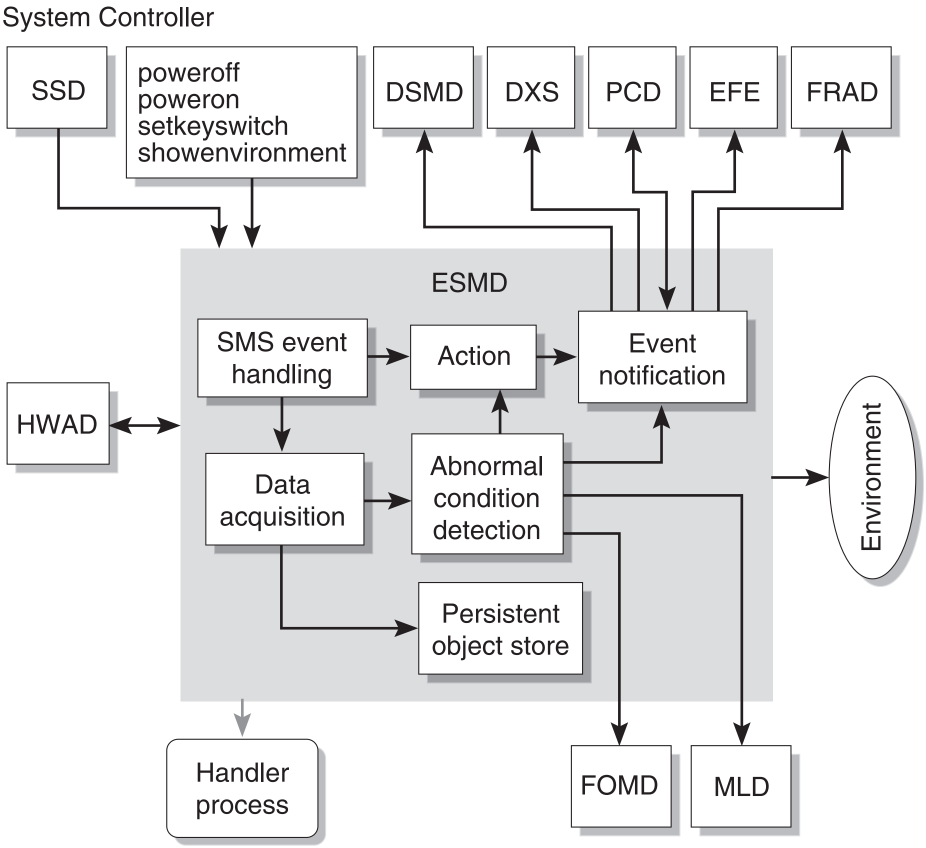


FIGURE 4-9 `ESMD` Client-Server Relationships

Failover Management Daemon

The failover management daemon, `fomd`(1M), is the core of the SC failover mechanism. `fomd` detects faults on the local and remote SCs and takes the appropriate action (initiating a failover or takeover). `fomd` tests and ensures that important configuration data is kept synchronized between both SCs. `fomd` runs on both the main and spare SCs.

For more information on `fomd,` see Chapter 12.

FIGURE 4-10 illustrates `FOMD` client-server relationships to the SMS daemons.

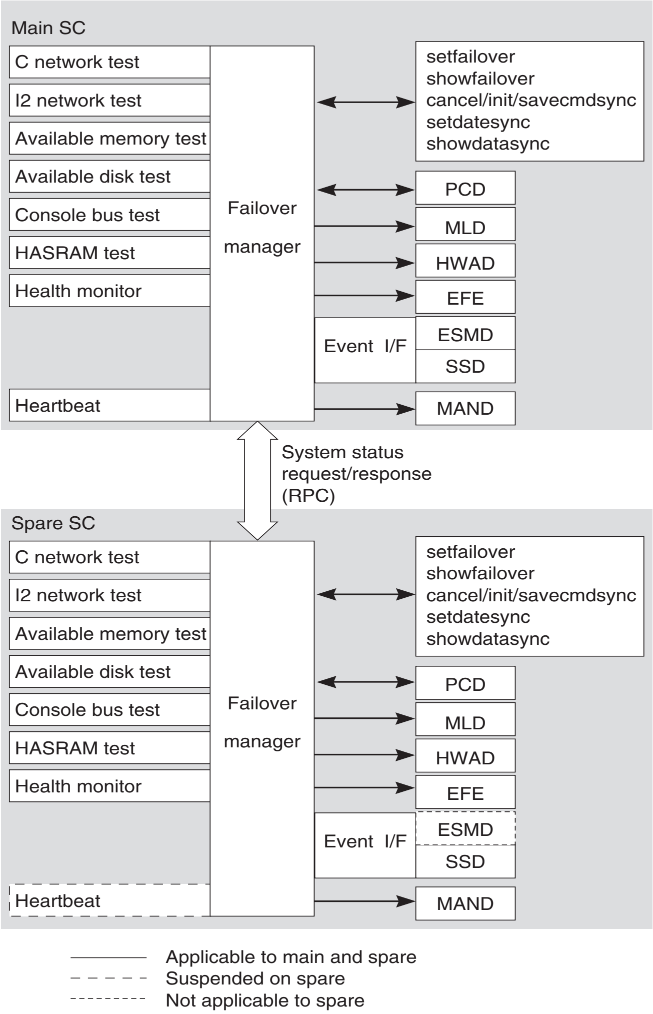


FIGURE 4-10 `FOMD` Client-Server Relationships

FRU Access Daemon

The FRU access daemon, `frad`(1M), is the field-replaceable unit (FRU) access daemon for SMS. `frad` provides controlled access to any SEEPROM within the Sun Fire high-end platform that is accessible by the SC. `frad` supports dynamic FRUID, which provides improved FRU data access using the Solaris platform information and control library daemon (PICLD). FRU identification is for Sun Service use only and transparent to the user.

frad is started by ssd.

FIGURE 4-11 illustrates FRAD client-server relationships to the SMS daemons.

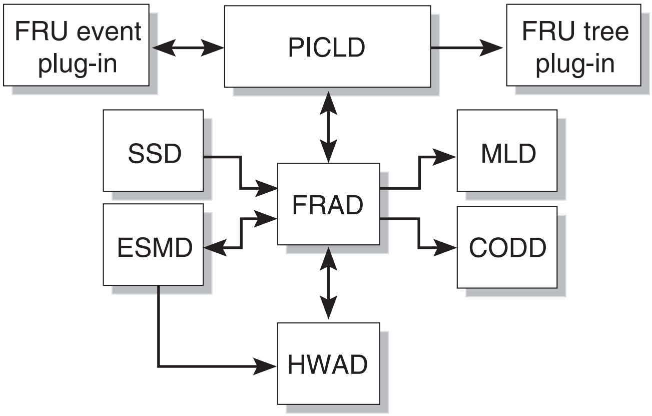


FIGURE 4-11 FRAD Client-Server Relationships

Hardware Access Daemon

The hardware access daemon, hwad(1M), provides hardware access to SMS daemons and a mechanism for all daemons exclusively to access, control, monitor, and configure the hardware.

hwad runs in either main or spare mode when it comes up. The failover daemon (fomd(1M)) determines which role hwad plays.

On both the main and spare, hwad does the following:

- Opens all the drivers (sbbc, echip, gchip, and consbus) and uses ioctl(2) calls to interface with them.
- Configures the local system clock and sets the clock source for each board present in the system.
- Disables SC-to-SC interrupt.
- Disables DARB interrupts by clearing SBBC system interrupt enable register.
- Creates an echip interface, which waits for any interrupt coming from the echip driver. At startup, this is the SC heartbeat interrupt.

On the main SC, hwad does the following:

- Reads the contents of the device presence register to identify the boards present in the system and makes them accessible to the clients.
- Takes control of I^{2}C steering and initializes all board objects present in the machine.
- Checks that clocks are phase locked. If they are, hwad checks that all clock sources are pointing to the main SC. If the clocks are not phase locked, hwad does not change any clock sources and disables automatic clock switch.
- Initializes the DARB interrupt, enables DARB interrupt, and enables PCI interrupt generation. Disables clock failure interrupt in gchip, disables console bus error interrupt in Echip, disables power supply failure interrupt in echip.
- Initializes the interrupt handler for events and creates threads to service events for mand, dsmd, and each osd.
- Creates the IOSRAM interfaces for 18 domains. This enables communication between the SC and the domain.

On the Spare SC, hwad performs these tasks:

- Sets the spare SC clock to the main SC clock.
- Sets the reference select to 0.
- Initializes SC to SC interrupt.

hwad directs communication to the IOSRAM (tunnel switch) for dynamic reconfiguration (DR).

hwad notifies dsmd(1M) if there is a dstop or rstop. It also notifies related SMS daemons, depending on the type of the Mbox interrupt that occurs.

hwad detects and logs console bus and JTAG errors.

Hardware access to a Sun Fire high-end system on the SC is done either by going through the PCI bus or console bus. Through the PCI bus you can access:

- SC boot bus controller (BBC) internal registers
- SC local JTAG interface
- Global I^{2}C devices for clock and power control/status

Through the console bus you can access:

- Various application specific integrated circuits (ASICs)
- Read/write chips
- Local I^{2}C devices on various boards for temperature and chip level power control/status

FIGURE 4-12 illustrates HWAD client-server relationships to the SMS daemons and CLIs.

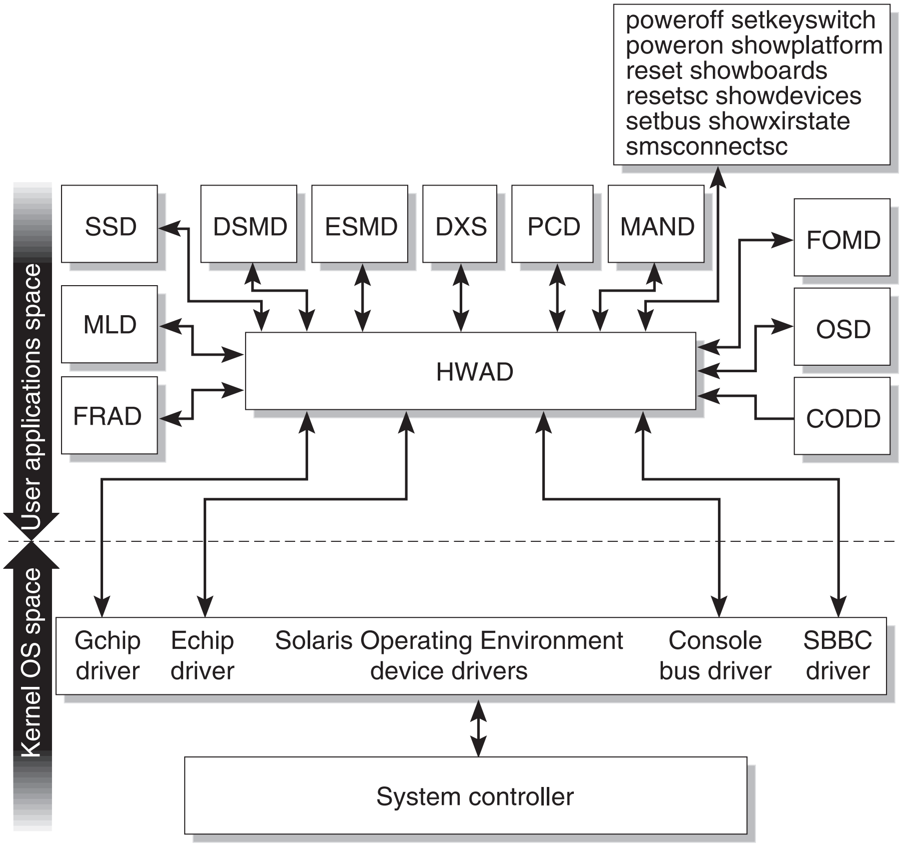


FIGURE 4-12 HWAD Client-Server Relationships

Key Management Daemon

The key management daemon, kmd(1M), provides a mechanism for managing security for socket communications between the SC and the domains.

The current default configuration includes authentication policies for the dca(1M) and dxs(1M) clients on the SC, which connect to the dcs(1M) and cvcd(1M) servers on a domain.

kmd manages the IPSec security associations (SAs) needed to secure the communication between the SC and servers running on a domain.

kmd manages per-socket policies for connections initiated by clients on the SC to servers on a domain.

At system startup, kmd creates a domain interface for each domain that is active. An active domain has a valid IOSRAM and is running the Solaris OS. Domain change events can trigger creation or removal of a domain kmd interface.

kmd manages shared policies for connections initiated by clients on the domain to servers on the SC. The kmd policy manager reads a configuration file and stores policies used to manage security associations. A request received by kmd is compared to the current set of policies to ensure that it is valid and to set various parameters for the request.

Static global policies are configured using ipsecconf(1M) and its associated data file (/etc/inet/ipsecinit.conf). Global policies are used for connections initiated from the domains to the SC. Corresponding entries are made in the kmd configuration file. Shared security associations for domain-to-SC connections are created by kmd when the domain becomes active.

Note – To work properly, policies created by ipsecconf and kmd must match.

The kmd configuration file is used for both SC-to-domain and domain-to-SC initiated connections. The kmd configuration file resides in /etc/opt/SUNWSMS/config/kmd_policy.conf.

The format of the kmd configuration files is as follows:

```
dir:d_port:protocol:sa_type:aut_alg:encr_alg:domain:login
```

where:

dir
Identified using the sctodom or domtosc strings.

d_port
The destination port.

protocol
Identified using the tcp or udp strings.

sa_type
The security association type. Valid choices are the ah or esp strings.

auth_alg
The authentication algorithm. The authentication algorithm is identified using the none or hmac-md5 strings, or by leaving the field blank.

`encr_alg`

The encryption algorithm. The encryption algorithm is identified using the `none` or `des` strings, or by leaving the field blank.

`domain`

The *domain-id* associated with the domain. Valid *domain-id*s are integers 0–17, space. Using a space in the *domain-id* field defines a policy that applies to all domains. A policy for a specific domain overrides a policy applied to all domains.

`login_name`

The login name of the user affected by the policy. Currently this includes `sms-dxs`, `sms-dca`, and `sms-mld`.

For example:

```
# Copyright (c) 2004 by Sun Microsystems, Inc.
# All rights reserved.
#
# This is the policy configuration file for the SMS Key Management Daemon.
# The policies defined in this file control the desired security for socket
# communications between the system controller and domains.
#
# The policies defined in this file must match the policies defined on the
# corresponding domains. See /etc/inet/ipsecinit.conf on the Sun Fire high-end
# system domain.
# See also the ipsec(7P), ipsecconf(1M) and sckmd(1M) man pages.
#
# The fields in the policies are a tuple of eight fields separated by the pipe
’|’ # character.
#
#<dir>|<d_port>|<protocol>|<sa_type>|<auth_alg>|<encr_alg>|<domain>|<login>|
#
# <dir>         --- direction to connect from. Values: sctodom, domtosc
# <d_port>      --- destination port
# <protocol>    --- protocol for the socket. Values: tcp, udp
# <sa_type>     --- security association type. Values: ah, esp
# <auth_alg>    --- authentication algorithm. Values: none, md5, sha1
# <encr_alg>    --- encryption algorithm. Values: none, des, 3des
# <domain>      --- domain id. Values: integers 0 - 17, space
#                   A space for the domain id defines a policy which applies
#                   to all domains. A policy for a specific domain overrides
#                   a policy which applied to all domains.
# <login>       --- login name. Values: Any valid login name
#
# ----------------------------------------------------------------------------
sctodom|665|tcp|ah|md5|none| |sms-dca|
sctodom|442|tcp|ah|md5|none| |sms-dxs|
```

FIGURE 4-13 illustrates KMD client-server relationships to the SMS daemons.

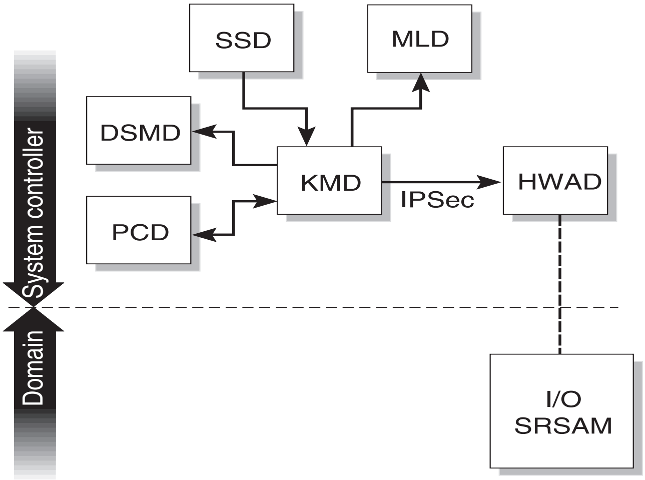


FIGURE 4-13 KMD Client-Server Relationships

Management Network Daemon

The management network daemon, mand(1M), supports the Management Network (MAN). (For more information about the MAN network, see "Management Network Services" on page 184.) By default, mand comes up in spare mode and switches to main when told to do so by the failover daemon (fomd(1M)). fomd determines which role mand plays.

At system startup, mand comes up in the role of spare and configures the SC-to-SC private network. This information is obtained from the file /etc/opt/SUNWSMS/config/MAN.cf, which is created by the smsconfig(1M) command. The failover daemon (fomd(1M)) directs mand to assume the role of main.

In the main role, mand does the following:

- Registers for domain change events from platform configuration database (pcd) to track changes in the domain active board list.
- Creates the mapping between domain_tag and IP address in the pcd.
- Initializes the scman(7d) driver with the current domain configuration.
- Registers for events from hwad to track active Ethernet information from the dman(7d) driver.
- Updates the scman driver and pcd, as appropriate.

- Registers for domain keyswitch events to communicate system startup MAN information to each domain when the domain is powered on (`setkeyswitch on`). This information includes Ethernet and MAN IP addressing, and active board list information used during the initial software installation on the domain.

FIGURE 4-14 illustrates `MAND` client-server relationships to the SMS daemons.

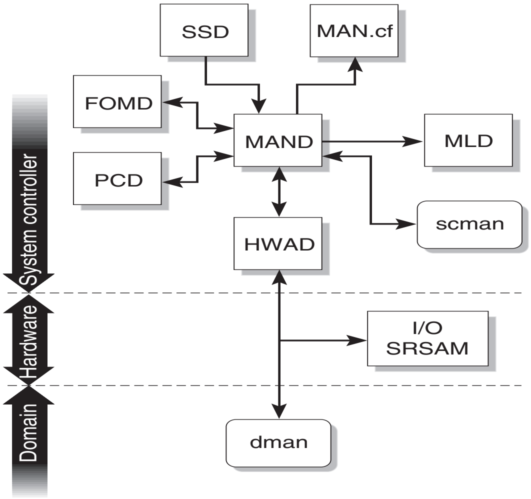


FIGURE 4-14 `MAND` Client-Server Relationships

Message Logging Daemon

The message logging daemon, `mld`(1M), captures the output of all other SMS daemons and processes. `mld` supports three configuration directives: File, Level, and Mode, in the `/var/opt/SUNWSMS/adm/.logger` file.

- File – Specifies the default output locations for the message files. The default is `msgdaemon` and should *not* be changed.
  - Platform messages are stored on the SC in `/var/opt/SUNWSMS/adm/platform/messages`
  - Domain messages are stored on the SC in `/var/opt/SUNWSMS/adm/`*domain-id*`/messages`
  - Domain `console` messages are stored on the SC in `/var/opt/SUNWSMS/adm/`*domain-id*`/console`
  - Domain `syslog` messages are stored on the SC in `/var/opt/SUNWSMS/adm/`*domain-id*`/syslog`.

- Level – Specifies the minimum level necessary for a message to be logged. The supported levels are NOTICE, WARNING, ERR, CRIT, ALERT, and EMERG. The default level is NOTICE.
- Mode – Specifies the verbosity of the messages. Two modes are available: verbose and terse. The default is verbose.

mld monitors the size of each of the message log files. For each message log type, mld keeps up to ten message files at a time, *x*.0 though *x*.9. For more information on log messages, see "Message Logging" on page 199.

FIGURE 4-15 illustrates MLD client-server relationships to the SMS daemons and CLIs.

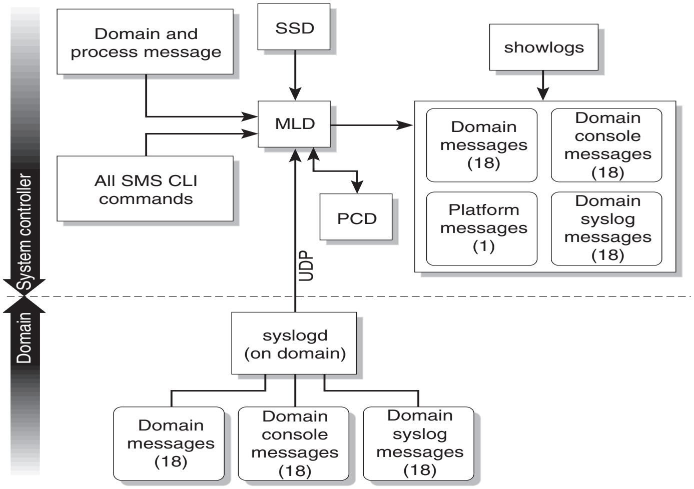


FIGURE 4-15 MLD Client-Server Relationships

OpenBoot PROM Support Daemon

The OpenBoot PROM support daemon, osd(1M), provides support to the OpenBoot PROM process running on a domain. osd and OpenBoot PROM communication is through a mailbox that resides on the domain. The osd daemon monitors the OpenBoot PROM mailbox. When the OpenBoot PROM writes requests to the mailbox, osd executes the requests accordingly.

osd runs at all times on the SC, even if there are no domains configured. osd provides virtual time of day (TOD) service, virtual nonvolatile random access memory (NVRAM), and virtual REBOOTINFO for OpenBoot PROM, and an interface to dsmd(1M) to facilitate auto-domain recovery. osd also provides an interface for the following commands: setobpparams(1M), showobpparams(1M), setdate(1M), and showdate(1M). See also Chapter 5.

osd is a trusted daemon in that it will not export any interface to other SMS processes. It exclusively reads and writes from and to all OpenBoot PROM mailboxes. There is one OpenBoot PROM mailbox for each domain.

osd has two main tasks: to maintain its current state of the domain configuration, and to monitor the OpenBoot PROM mailbox.

FIGURE 4-16 illustrates OSD client-server relationships to the SMS daemons and CLIs.

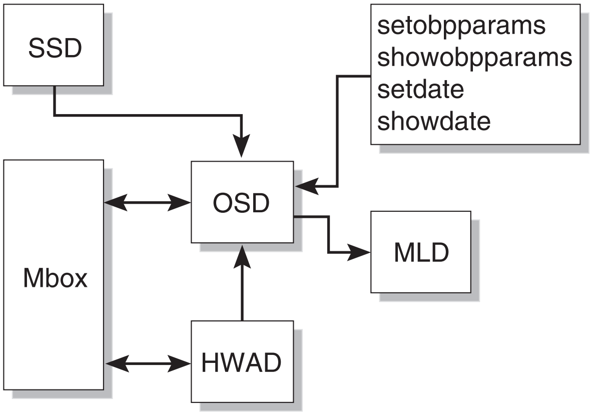


FIGURE 4-16 OSD Client-Server Relationships

Platform Configuration Database Daemon

The platform configuration daemon, pcd(1M), is a Sun Fire high-end system management daemon that runs on the SC with primary responsibility for managing and providing controlled access to platform and domain configuration data.

pcd manages an array of information that describes the Sun Fire system configuration. In its physical form, the database information is a collection of flat files, each file appropriately identifiable by the information contained within it. All SMS applications must go through pcd to access the database information.

In addition to managing platform configuration data, pcd is responsible for platform configuration change notifications. When pertinent platform configuration changes occur within the system, the pcd sends out notification of the changes to clients who have registered to receive the notification.

FIGURE 4-17 illustrates PCD client-server relationships to the SMS daemons and CLIs.

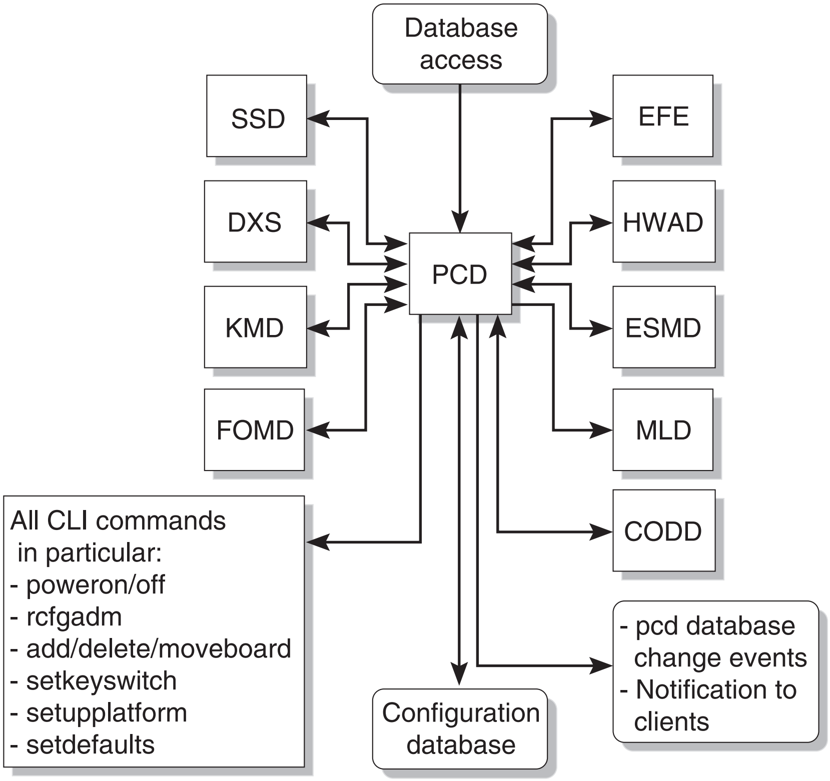


FIGURE 4-17 `PCD` Client-Server Relationships

Platform Configuration

The following information uniquely identifies the platform:

- Platform type
- Platform name
- Chassis HostID

  The Chassis HostID is used only by the COD feature to identify the platform for COD licensing purposes. The Chassis HostID is the centerplane serial number and is recorded internally within the system. To view the Chassis HostID, run the `showplatform -p cod` command.

- Chassis serial number

  The chassis serial number identifies a Sun Fire high-end system and is used to identify the platform in messages and events. It is also used by service providers to correlate events and service actions to the correct system. The chassis serial number is printed on a label located on the front of the system chassis, near the bottom center. Starting with the SMS 1.4 release, the chassis serial number is automatically recorded by Sun manufacturing on systems that ship with SMS installed. To view the chassis serial number, run the `showplatform -p csn` command.

If you are upgrading to SMS 1.6 or later from an earlier SMS version, use the `setcsn`(1M) command to record the chassis serial number. For details on the `setcsn` command, refer to the command description in the *System Management Services (SMS) 1.6 Reference Manual*.

- Cacheable address slice map
- System clock frequency
- System clock type
- SC IP address
- SC0-to-SC1 IP address
- SC1-to-SC0 IP address
- SC-to-SC IP netmask
- COD instant access CPUs (headroom)

Domain Configuration

The following information is domain-related:

- *domain-id*
- *domain-tag*
- OS version (currently not used)
- OS type (currently not used)
- Available component list
- Assigned board list
- Active board list
- Golden IOSRAM I/O board
- Virtual keyswitch setting for a domain
- Active Ethernet I/O board
- Domain creation time
- Domain dump state
- Domain bringup priority
- IP host address
- Host name
- Host netmask

- Host broadcast address
- Virtual OpenBoot PROM address
- Physical OpenBoot PROM address
- COD RTU license reservation

System Board Configuration

The following information is related to system boards:

- Expander position
- Slot position
- Board type
- Board state
- Domain Identifier assigned to board
- Available component list state
- Board test status
- Board test level
- Board memory clear state
- COD enabled flag

SMS Startup Daemon

The SMS startup daemon, `ssd`(1M), is responsible for starting and maintaining all SMS daemons and domain *X* servers.

`ssd` checks the environment for availability of certain files and the availability of the Sun Fire high-end system, sets environment variables, and then starts `esmd`(1M) on the main SC. `esmd` monitors environmental changes by polling the related hardware components. When an abnormal condition is detected, `esmd` handles it or generates an event so that the correspondent handlers take appropriate action and/or update their current status. Some of those handlers are `dsmd`, `pcd`, and Sun Management Center (if installed). The main objective of `ssd` is to ensure that the SMS daemons and servers are always up and running.

FIGURE 4-18 illustrates `SSD` client-server relationships to the SMS daemons.

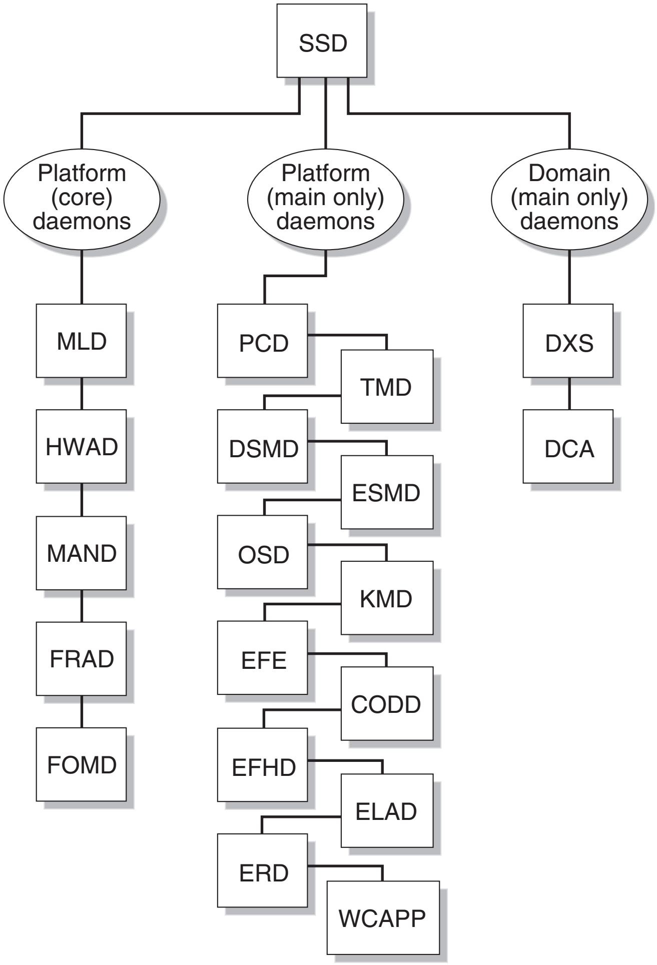


FIGURE 4-18 SSD Client-Server Relationships

Scripts

ssd uses a configuration file, ssd_start, to determine which SMS components to start, and in which order to start them. This configuration file is located in the /etc/opt/SUNWSMS/startup directory.

Caution – This is a system configuration file. Mistakes in editing this file can render the system inoperable. `args` is the only field that should ever be edited in this script. Refer to the daemon man pages for specific options, and pay particular attention to syntax.

`ssd_start` consists of entries in the following format:

name:args:nice:role:type:trigger:startup-timeout:shut down-timeout:uid:start-order:stop-order

where:

name	The name of the program.
args	The valid program options or arguments. Refer to the daemon man pages for more information.
nice	Specifies a process priority tuning value. Do *not* adjust.
role	Specifies whether the daemon is platform or domain specific.
type	Specifies whether the program is a daemon or a server.
trigger	Specifies whether the program should be started automatically or upon event reception.
startup-timeout	The time in seconds `ssd` will wait for the program to start up.
stop-timeout	The time in seconds `ssd` will wait for the program to shut down.
uid	The *user-id* the associated program will run under.
start-order	The order in which `ssd` will start up the daemons. Do *not* adjust. Changing the default values can result in the SMS daemons not working properly.
stop-order	The order in which `ssd` will shut down the daemons. Do *not* adjust. Changing the default values can result in the SMS daemons not working properly.

Spare Mode

Each time ssd starts, it comes up in spare mode. Once ssd has started the platform core daemons running, it queries fomd(1M) for its role. If the fomd query returns with spare, ssd stays in this mode. If the fomd returns with main, then ssd transitions to main mode.

After this initial query phase, ssd only switches between modes through events received from the fomd.

When in spare mode, ssd starts and monitors all of the core platform role, auto trigger programs in the ssd_start file. Currently, this list is made up of the following programs:

- mld
- hwad
- mand
- frad
- fomd

If, while in main mode, ssd receives a spare event, then ssd shuts down all programs except the core platform role and auto trigger programs found in the ssd_start file.

Main Mode

ssd stays in spare mode until it receives a main event. At that time, ssd starts and monitors (in addition to the daemons that are already running) all of the main platform role event trigger programs in the ssd_start file. This list is made up of the following programs:

- pcd
- tmd
- dsmd
- esmd
- osd
- kmd
- efe
- codd
- efhd
- elad
- erd
- wcapp

Finally, after starting all the platform role, event trigger programs, ssd queries the pcd to determine which domains are active. For each of these domains, ssd starts all the domain role, event trigger programs found in the ssd_start file.

Domain-Specific Process Startup

ssd uses domain start and stop events from pcd as instructions for starting and stopping domain-specific servers.

Upon reception, ssd either starts or stops all of the domain role, event trigger programs (for the domain identified) found in the ssd_start file.

Monitoring and Restarts

Once ssd has started a process, it monitors the process and restarts it in the event the process fails.

SMS Shut Down

In certain instances, such as SMS software upgrades, the SMS software must be shut down. ssd provides a mechanism to shut down itself and all SMS daemons and servers under its control.

ssd notifies all SMS software components under its control to shut down. After all the SMS software components have been shut down, ssd shuts itself down.

Task Management Daemon

The task management daemon, tmd(1M), provides task management services such as scheduling for SMS. This reduces the number of conflicts that can arise during concurrent invocations of the hardware tests and configuration software.

Currently, the only service exported by tmd is the hpost(1M) scheduling service. In a Sun Fire high-end system, hpost is scheduled based on the following two factors.

- Restriction of hpost. When the platform first comes up and no domains have been configured, a single instance of hpost takes exclusive control of all expanders and configures the centerplane ASICs. All subsequent hpost invocations wait until this is complete before proceeding.

  Only a single hpost invocation can act on any one expander at a time. For a Sun Fire high-end system configured without split expanders, this restriction does not prevent multiple hpost invocations from running. This restriction does come into play, however, when the machine is configured with split expanders.

- System-wide hpost throttle limit. There is a limit to the number of concurrent hpost invocations that can run at a single time without saturating the system. The ability to throttle hpost invocations is available using the -t option in ssd_startup.

Caution – Changing the default value can adversely affect system functionality. Do not adjust this parameter unless instructed by a Sun service representative to do so.

FIGURE 4-19 illustrates TMD client-server relationships to the SMS daemons.

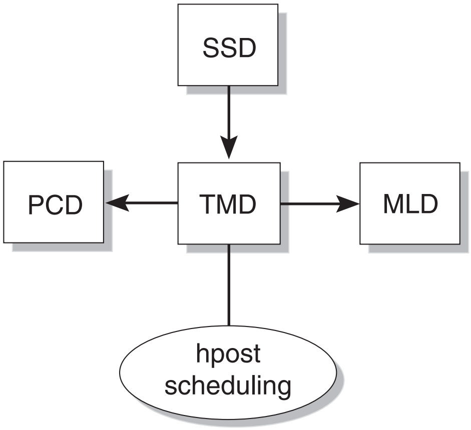


FIGURE 4-19 TMD Client-Server Relationships

Environment Variables

Basic SMS environment defaults *must* be set in your configuration files to run SMS commands.

- PATH to include /opt/SUNWSMS/bin
- LD_LIBRARY_PATH to include/opt/SUNWSMS/lib
- MANPATH to include /opt/SUNWSMS/man

Setting other environment variables when you log in can save time. TABLE 4-2 suggests some useful SMS environment variables.

TABLE 4-2 Example Environment Variables

Variable	Description
SMSETC	The path to the /etc/opt/SUNWSMS directory containing miscellaneous SMS-related files.
SMSLOGGER	The path to the /var/opt/SUNWSMS/adm directory containing the configuration file for message logging, .logger.
SMSOPT	The path to the /opt/SUNWSMS directory containing the SMS package binaries, libraries, and object files; configuration and startup files.
SMSVAR	The path to the /var/opt/SUNWSMS directory containing platform and domain message and data files.

CHAPTER 5

SMS Domain Configuration

A *dynamic system domain* (DSD) is an independent environment, a subset of a server, that is capable of running a unique version of firmware and a unique version of the Solaris OS. Each domain is insulated from the other domains. Continued operation of a domain is not affected by any software failures in other domains or by most hardware failures in any other domain.

The system controller (SC) supports commands that enable you to logically group system boards into *dynamic system domains*, or simply *domains*, which are able to run their own OS and handle their own workload. Domains can be created and deleted without interrupting the operation of other domains. You can use domains for many purposes. For example, you can test a new OS version or set up a development and testing environment in a domain. In this way, if problems occur, the rest of your system is not affected.

You can also configure several domains to support different departments, with one domain per department. You can temporarily reconfigure the system into one domain to run a large job over the weekend.

The Sun Fire 15K system allows up to 18 domains to be configured. The Sun Fire 12K system allows up to 9 domains to be configured.

Domain configuration establishes mappings between the domains and the server's hardware components. Also included in domain configuration is the establishment of various system management parameters and policies for each domain. This chapter discusses all aspects of domain configuration functionality that the Sun Fire high-end system provides.

This chapter contains the following sections:

Domain Configuration Units

A domain configuration unit (DCU) is a unit of hardware that can be assigned to a single domain. DCUs are the hardware components from which domains are constructed. DCUs that are not assigned to any domain are said to be in *no-domain*.

All DCUs are system boards and all system boards are DCUs. The Sun Fire high-end system DCUs are:

- System board
- Sun Fire HsPCI I/O board (HPCI)
- Sun Fire HsPCI+ I/O board (HPCI+)
- Sun Fire MaxCPU board (MCPU)
- Sun Fire Link wPCI board (WPCI)

Sun Fire high-end system hardware requires the presence of at least one regular system board, plus at least one of the I/O board types in each configured domain. csb, exb boards, and the SC are *not* DCUs.

Note – MaxCPU boards do not contain memory. To set up a domain, at least one regular CPU board is required.

Domain Configuration Requirements

You can create a domain out of any group of system boards, provided the following conditions are met:

- The boards are present and not in use in another domain.
- At least one board has a CPU and memory.
- At least one board is an I/O board.
- At least one board has a network interface.
- The boards have sufficient memory to support an autonomous domain.
- The name you give the new domain is unique (as specified in the addtag(1M) command).
- You have an idprom.image file for the domain that was shipped to you by the factory. If your idprom.image file has been accidentally deleted or corrupted and you do not have a backup, contact your Sun field support representative.

- At least one boot disk must be connected to one of the boards that will be grouped together into a domain. Alternatively, if a domain does not have its own disk, there must be at least one network interface so that you can boot the domain from the network.

DCU Assignment

The assignment of DCUs to a domain is the result of one of three logical operations acting on a DCU (system board):

- Adding the board (from *no-domain*) to a domain
- Removing the board from a domain (leaving the board in *no-domain*)
- Moving the board from one domain to another

Static Versus Dynamic Domain Configuration

Although there are logically three DCU assignment operations, the underlying implementation is based upon four domain configuration operations:

- Adding a board to an inactive domain
- Removing a board from an inactive domain
- Adding a board to an active domain
- Removing a board from an active domain

The first two domain configuration operations apply to inactive domains; that is, to domains that are not running OS software. These operations are called *static domain configuration* operations. The latter two domain configuration operations apply to active domains, that is, those running OS software, and are called *dynamic domain configuration* operations.

Dynamic domain configuration requires interaction with the domain's Solaris software to introduce or remove the DCU-resident resources such as CPUs, memory, or I/O devices from Solaris OS control. Sun Fire high-end system dynamic reconfiguration (DR) provides a capability called remote DR for an external agent, such as the SC, to request dynamic configuration services from a domain's Solaris environment.

The SC command user interfaces utilize remote DR as necessary to accomplish the requested tasks. Local automatic DR allows applications running on the domain to be aware of impending DR operations and to take action, as appropriate, to adjust to resource changes. This improves the likelihood of success of DR operations, particularly those which require active resources to be removed from domain use. For more information on DR, refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.

When a domain is configured for local automatic DR, remote DR operations initiated from the SC benefit from the automation of DR operations for that domain. With local automatic DR capabilities available in Sun Fire domains, simple scripts can be constructed and placed in a `crontab`(1) file, allowing simple platform reconfigurations to take place on a time schedule.

SMS allows you to add boards to or remove boards from an active (running) domain. Initiation of a remote DR operation on a domain requires administrative privilege for that domain. SMS grants the ability to initiate remote DR on a domain to individual administrators on a per-domain basis.

The remote DR interface is secure. Since invocation of DR operations on the domain itself requires superuser privilege, remote DR services are provided only to known, authenticated remote agents.

The user command interfaces that initiate DCU assignment operations are the same whether the affected domains have local automatic DR capabilities or not.

SMS provides for the addition or removal of a board from an active domain, such as static domain configuration using `addboard`, `deleteboard`, and `moveboard`. For more information, refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.

Global Automatic Dynamic Reconfiguration

Remote DR and local automatic DR functions are building blocks for a feature called global automatic DR. Global automatic DR introduces a framework that can be used to automatically redistribute the system board resources on a Sun Fire system. This redistribution can be based upon factors such as production schedule, domain resource utilizations, domain functional priorities, and so on. Global automatic DR accepts input from customers describing their Sun Fire resource utilization policies and then uses those policies to automatically marshal the Sun Fire high-end system resources to produce the most effective utilization. For more information on DR, refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.

Configuration for Platform Administrators

This section briefly describes the configuration services available to the platform administrator.

Available Component List

Each domain (A-R) defaults to having a 0-board list of boards that are available to an administrator or configurator to assign to their respective domains. Boards can be added to the available component list of a domain by a platform administrator using the `setupplatform`(1M) command. Updating an available component list requires `pcd` to perform the following tasks:

- Update the domain configuration available component list
- Update the available component list state for each board to show the domain to which it is now `available`
- Notify `dxs` of boards added to their respective domain's available component list

After `pcd` notifies `dxs` about any added boards, `dxs` in turn notifies the running domain of the arrival of an `available` board.

▼ To Set Up the Available Component List

`setupplatform` sets up the available component list for domains. If a *domain-id* or *domain-tag* is specified, a list of boards must be specified. If no value is specified for a parameter, it will retain its current value.

1. **In an SC window, log in as a platform administrator.**

2. **Type the following command:**

```
sc0:sms-user:> setupplatform -d domain-indicator -a location
```

where:

`-a`	Adds the slot to the available component list for the specified domain.
`-d` *domain-indicator*	Specifies the domain using: *domain-id* – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive. *domain-tag* – Name assigned to a domain using `addtag`(1M).
location	The board (DCU) location.

The following *location* forms are accepted:

Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

The following is an example of making boards at SB0, IO1, and IO2 available to domain A:

```
sc0:sms-user:> setupplatform -d A -a SB0 IO1 IO2
```

The platform administrator can now assign the board to domain A using the `addboard`(1M) command or leave that up to the domain administrator.

A platform administrator has privileges for only the `-c assign` option of the `addboard` command. All other board configuration requires domain privileges. For more information, refer to the `addboard` man page.

Configuring Domains

To Name or Change Domain Names From the Command Line

You do not need to create domains on the Sun Fire high-end system. Eighteen domains have already been established (domains A–R, case insensitive). These domain designations are customizable. This section describes how to uniquely name domains.

Note – Before proceeding, see “Domain Configuration Requirements” on page 82. If the system configuration must be changed to meet any of these requirements, call your service provider.

---

1. **Log in to the SC.**

2. **Type the following command:**

```
sc0:sms-user:> addtag -d domain-indicator  new-tag
```

where:

-d *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using addtag(1M).

new-tag The new name you want to give to the domain. It must be unique among all domains controlled by the SC.

Naming a domain is optional.

The following is an example of naming Domain A to dmnA:

```
sc0:sms-user:> addtag -d A dmnA
```

▼ To Add Boards to a Domain From the Command Line

1. **Log in to the SC.**

Note – Platform administrators are restricted to using the `-c assign` option. The option can be used only for boards classified as `available`, not for boards classified as `active`.

The system board must be in the `available` state to the domain to which it is being added. Use the `showboards` (1M) command to determine a board's state.

2. Type the following command:

```
sc0:sms-user:> addboard -d domain-indicator -c assign location...
```

where:

`-d` *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-c assign` Specifies the transition of the board from the current configuration state to the `assign`ed state.

location The board (DCU) location. Multiple locations are permitted.

The following *location* forms are accepted:

Valid form for Sun Fire 15K/E25K	Valid form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

For example:

```
sc0:sms-user:> addboard -d C -c assign SB0 I01 SB1 I02
```

SB0, IO1, SB1, and IO2 have now changed from a state of being `available` to domain C to being assigned to that domain.

`addboard` performs tasks synchronously and does not return control to the user until the command is complete. If the command fails, the board does not return to its original state. A `dxs` or `dca` error is logged to the domain and `pcd` reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain in order to use that board.

▼ To Delete Boards From a Domain From the Command Line

Note – Platform administrators are restricted to using the `-c unassign` option. The option can be used only for boards with a status of assigned, not boards with a status of `active`.

1. **Log in to the SC.**

   The system board must be in the assigned state to the domain from which it is being deleted. Use the `showboards` (1M) command to determine a board's state.

2. **Type the following command:**

```
sc0:sms-user:> deleteboard -c unassign location...
```

where:

`-c unassign` Specifies the transition of the board from the current configuration state to a new `unassigned` state.

location The board (DCU) location. Multiple locations are permitted.

The following *location* forms are accepted:

Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

For example:

```
sc0:sms-user:> deleteboard -c unassign SB0
```

SB0 has now changed from being assigned to the domain to being available to it.

If `deleteboard` fails, the board does not return to its original state. A `dxs` or `dca` error is logged to the domain and `pcd` reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain in order to use that board.

▼ To Move Boards Between Domains From the Command Line

Note – Platform administrators are restricted to the `-c assign` option. The option can only be used for boards with a status of `assigned`. It cannot be used for `active` boards.

---

1. **Log in to the SC.**

   The system board must be in the `assigned` state to the domain from which it is being deleted. Use the `showboards` (1M) command to determine a board's state.

2. **Type the following command:**

```
sc0:sms-user:> moveboard -d domain-indicator  -c assign location
```

where:

-d *domain-indicator* — Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using addtag(1M).

-c assign — Specifies the transition of the board from the current configuration state to an assigned state.

location — The board (DCU) location.

The following *location* forms are accepted:

Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

moveboard performs tasks synchronously and does not return control to the user until the command is complete. You can only specify one *location* when using moveboard.

For example:

```
sc0:sms-user:> moveboard -d C -c assign SB0
```

SB0 has been moved from its previous domain and assigned to domain C.

If moveboard fails, the board does not return to its original state. A dxs or dca error is logged to the domain and pcd reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain the board was in when the error occurred, in order to use that board.

▼ To Set Domain Defaults

The SMS setdefaults(1M) command removes all instances of a previously active domain.

1. **Log in to the SC.**

   Platform administrators can set domain defaults for all domains, but only one domain at a time. The domain must *not* be active and `setkeyswitch` must be set to `off`.

   `setdefaults` removes *all* `pcd` entries except network information and log files. This includes removing the NVRAM and boot parameter data.

   By default, you are asked whether you want to remove the NVRAM and boot parameter data. If you respond no, the data is preserved. If you use the `-p` option you are not prompted and the data is automatically preserved.

2. **Type the following command:**

```
sc0:sms-user:> setdefaults -d domain-indicator [-p]
```

where:

`-d` *domain-indicator* — Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-p` — Preserves the NVRAM and boot parameter data without a prompt.

For more information on `setdefaults`, refer to the `setdefaults` man page or the *System Management Services (SMS) 1.6 Reference Manual*.

To Obtain Board Status

1. **Log in to the SC.**

   Platform administrators can obtain board status for all domains.

2. **Type:**

```
sc0:sms-user:> showboards [-d domain-id|-d domain-tag]
```

The board status is displayed.

The following partial example for the Sun Fire 15K system shows the board information for a user with platform administrator privileges. All domains are visible. On a Sun Fire 12K system, nine domains would be shown.

```
sc0:sms-user:> showboards

Location  Pwr  Type         Board Status  Test Status   Domain
----      ---  ----         ------------  -----------   ------
SB0        On   CPU          Active        Passed        domainC
SB1        On   CPU          Active        Passed        A
SB2        On   CPU          Active        Passed        A
SB3        On   CPU          Active        Passed        engB
SB4        On   CPU          Active        Passed        engB
SB5        On   CPU          Active        Passed        engB
SB6        On   CPU          Active        Passed        A
SB7        On   CPU          Active        Passed        domainC
SB8       Off  CPU          Available     Unknown       Isolated
SB9        On   CPU          Active        Passed        dmnJ
SB10      Off  CPU          Available     Unknown       Isolated
SB11      Off  CPU          Available     Unknown       Isolated
SB12       Off  CPU          Assigned      Unknown       engB
SB13        -  Empty Slot   Available       -           Isolated
SB14       Off  CPU          Assigned      Failed        domainC
SB15       On   CPU          Active        Passed        P
SB16       On   CPU          Active        Passed        domainC
SB17        -   Empty Slot   Assigned        -           dmnR
IO0         -  Empty Slot   Available       -           Isolated
IO1        On   HPCI         Active        Passed        A
IO2        On   MCPU         Active        Passed        engB
IO3        On   MCPU         Active        Passed        domainC
IO4        On   HPCI+        Available     Degraded      domainC
IO5        Off  HPCI+        Assigned      Unknown       engB
IO6        On   HPCI         Active        Passed        A
IO7        On   HPCI         Active        Passed        dmnJ
IO8        On   WPCI         Active        Passed        Q
IO9        On   HPCI+        Assigned      iPOST         dmnJ
IO10       Off  HPCI         Assigned      Unknown       engB
IO11       Off  HPCI         Assigned      Failed        engB
IO12       Off  HPCI         Assigned      Unknown        engB
IO13        -  Empty Slot   Available       -           Isolated
IO14      Off  HPCI+        Available     Unknown       Isolated
IO15       On   HPCI         Active        Passed        P
IO16       On   HPCI         Active        Passed        Q
IO17        -   Empty Slot   Assigned        -           dmnR
```

▼ To Obtain Domain Status

1. **Log in to the SC.**

   Platform administrators can obtain domain status for all domains.

2. **Type the following command:**

```
sc0:sms-user:> showplatform -d domain-indicator
```

where:

`-d` *domain-indicator*   Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

The status listing is displayed.

The following partial example for the Sun Fire 15K system shows the domain information for a user with platform administrator privileges. All domains are visible. On a Sun Fire 12K system, nine domains would be shown.

```
sc0:sms-user:> showplatform
...
Domain configurations:
======================
Domain ID Domain Tag    Solaris Nodename   Domain Status
A         newA          sun15-b0           Powered Off
B         engB          sun15-b1           Keyswitch Standby
C         domainC       sun15-b2           Running OBP
D         eng1          sun15-b3           Loading Solaris
E         -             sun15-b4           Running Solaris
F         domainF       sun15-b5           Running Solaris
G         dmnG          sun15-b6           Running Solaris
H         -             sun15-b7           Solaris Quiesced
I         -             sun15-b8           Powered Off
J         dmnJ          sun15-b9           Powered Off
K         -             sun15-b10          Booting Solaris
L         -             sun15-b11          Powered Off
M         -             sun15-b12          Powered Off
N         -             sun15-b13          Keyswitch Standby
O         -             sun15-b14          Powered Off
P         -             sun15-b15          Running Solaris
Q         -             sun15-b16          Running Solaris
R         dmnR          sun15-b17          Running Solaris
```

Virtual Time of Day

The Solaris environment uses the functions provided by a hardware time of day (TOD) chip to support Solaris system date and time. Typically, Solaris software reads the current system date and time at boot using a get TOD service. From that point forward, Solaris software either uses a high-resolution hardware timer to represent current date and time or, if configured, uses Network Time Protocol (NTP) to synchronize current system date and time to a (presumably more accurate) time source.

The SC is the only computer on the platform that has a real-time clock. The virtual TOD for domains is stored as an offset from that real-time clock value. Each domain can be configured to use NTP services instead of `setdate` (1M) to manage the running system date and time. For more information on NTP, see "Configuring NTP" on page 98 or refer to the `xntpd`(1M) man page in the *man Pages(1M): System Administration Commands* section of the Solaris 9 Reference Manual Collection.

Note – NTP is a separate package that must be installed and configured on the domain in order to function as described. Use `setdate` on the domain prior to installing NTP.

However system date and time is managed while Solaris software is running, an attempt is made to keep the boot-time TOD value accurate by setting the TOD when variance is detected between the current TOD value and the current system date and time.

Since the Sun Fire high-end system hardware provides no physical TOD chip for Sun Fire domains, SMS provides the time-of-day services required by the Solaris environment for each domain. Each domain is supplied with a TOD service that is logically separate from that provided to any other domain. This difference allows system date and time management on a Sun Fire high-end system domain to be as flexible as that provided by standalone servers. In the unlikely event that a domain needs to be set up to run at a time other than real-world time, the Sun Fire high-end system TOD service allows that domain to be configured without affecting the TOD values supplied to other domains running real world time.

Time settings are implemented using `setdate`(1M). You must have platform administrator privileges to run `setdate`. See "All Privileges" on page 43 for more information.

Setting the Date and Time

setdate (1M) allows the SC platform administrator to set the system controller date and time values. After setting the date and time, setdate(1M) displays the current date and time for the user.

▼ To Set the Date on the SC

1. **Log in to the SC.**
2. **Type the following command:**

```
sc0:sms-user:> setdate 021210302000.00
System Controller: Tue Feb 12 10:30 2002 US/Pacific
```

Optionally, setdate(1M) can set a domain TOD. The domain's keyswitch must be in the off or standby position. You must have platform administrator privileges to run this command on the domain.

▼ To Set the Date for Domain eng2

1. **Log in to the SC.**
2. **Type the following command:**

```
sc0:sms-user:> setdate -d eng2 021210302000.00
Domain eng2: Tue Feb 12 10:30 2002 US/Pacific
```

showdate(1M) displays the current SC date and time.

▼ To Display the Date on the SC

1. **Log in to the SC.**
2. **Type the following command:**

```
sc0:sms-user:> showdate
System Controller: Tue Feb 12 10:30 2002 US/Pacific
```

Optionally, showdate(1M) can display the date and time for a specified domain. Superuser or any member of a platform or domain group can run showdate.

▼ To Display the Date on Domain `eng2`

1. **Log in to the SC.**
2. **Type the following command:**

```
sc0:sms-user:> showdate -d eng2
Domain eng2: Tue Feb 12 10:30 2002 US/Pacific
```

Configuring NTP

The NTP daemon, called `xntpd(1M)` for the Solaris OS, provides a mechanism for keeping the time settings synchronized between the SC and the domains. The OpenBoot PROM obtains the time from the SC when the domain is booted, and NTP keeps the time synchronized on the domain from that point on.

NTP configuration is based on information provided by the system administrator.

The NTP packages are compiled with support for a local reference clock. This means that your system can poll itself for the time instead of polling another system or network clock. The poll is done through the network loopback interface. The numbers in the IP address are `127.127.1.0.` This section describes how to set the time on the SC using `setdate`, and then to set up the SC to use its own internal time-of-day clock as the reference clock in the `ntp.conf` file.

NTP can also keep track of the drift (difference) between the SC clock and the domain clock. NTP corrects the domain clock if it loses contact with the SC clock, provided that you have a drift file declaration in the `ntp.conf` file. The drift file declaration specifies to the NTP daemon the name of the file that stores the error in the clock frequency computed by the daemon. See the following procedure for an example of the drift file declaration in an `ntp.conf` file.

If the `ntp.conf` file does not exist, create it as described in the following procedure. You must have an `ntp.conf` file on both the SC and the domains.

▼ To Create the `ntp.conf` File

1. **Log in to the main SC as superuser.**
2. **Change to the** `/etc/inet` **directory and copy the NTP *server* file to the NTP configuration file:**

```
sc0:# cd /etc/inet
sc0:# cp ntp.server ntp.conf
```

3. **Using a text editor, edit the** `/etc/inet/ntp.conf` **file created in the previous step.**

   The `ntp.conf` file for the Solaris 9 OS is located in `/etc/inet`.

   The following is an example of server lines in the `ntp.conf` file on the main SC, to synchronize clocks.

```
server 127.127.1.0
fudge 127.127.1.0 stratum 13
driftfile /var/ntp/ntp.drift
statsdir /var/ntp/ntpstats/
filegen peerstats file peerstats type day enable
filegen loopstats file loopstats type day enable
filegen clockstats file clockstats type day enable
```

4. **Save the file and exit.**

5. **Stop and restart the NTP daemon:**

```
sc0:# /etc/init.d/xntpd stop
sc0:# /etc/init.d/xntpd start
```

6. **Log in to the spare SC as superuser.**

7. **Change to the** `/etc/inet` **directory and copy the NTP *server* file to the NTP configuration file:**

```
sc1:# cd /etc/inet
sc1:# cp ntp.server ntp.conf
```

8. **Using a text editor, edit the** `/etc/inet/ntp.conf` **file created in the previous step.**

   The `ntp.conf` file for the Solaris 9 OS is located in `/etc/inet`.

   The following is an example of server lines in the `ntp.conf` file on the spare SC, to synchronize clocks.

```
server 127.127.1.0
fudge 127.127.1.0 stratum 13
driftfile /var/ntp/ntp.drift
statsdir /var/ntp/ntpstats/
filegen peerstats file peerstats type day enable
filegen loopstats file loopstats type day enable
filegen clockstats file clockstats type day enable
```

9. **Stop and restart the NTP daemon:**

```
sc1:# /etc/init.d/xntpd stop
sc1:# /etc/init.d/xntpd start
```

10. **Log in to each domain as superuser.**

11. **Change to the** `/etc/inet` **directory and copy the NTP *client* file to the NTP configuration file:**

```
domain-id:# cd /etc/inet
domain-id:# cp ntp.client ntp.conf
```

12. **Using a text editor, edit the** `/etc/inet/ntp.conf` **file created in the previous step.**

    The `ntp.conf` file for the Solaris 9 OS is located in `/etc/inet`.

    For the Solaris 9 OS, you can add lines similar to the following to the `/etc/inet/ntp.conf` file on the domains:

```
server main-sc-hostname prefer
server spare-sc-hostname
```

13. **Save the file and exit.**

14. **Change to the initialization directory, and stop and restart the NTP daemon on the domain:**

```
domain-id:# /etc/init.d/xntpd stop
domain-id:# /etc/init.d/xntpd start
```

NTP is now installed and running on your domain. Repeat Step 10 through Step 14 for each domain.

For more information on the NTP daemon, refer to the `xntpd`(1M) man page in the *man Pages(1M): System Administration Commands* section of the Solaris 9 Reference Manual Collection.

Virtual ID PROM

Each configurable domain has a virtual ID PROM that contains identifying information about the domain, such as hostID and domain Ethernet address. The hostID is unique among all domains on the same platform. The Ethernet address is world unique.

Sun Fire high-end system management software provides a virtual ID PROM for each configurable domain containing identifying information that can be read, but not written, from the domain. The information provided meets the requirements of the Solaris environment.

The `flashupdate` Command

SMS provides the `flashupdate`(1M) command to update the Flash PROM in the system controller (SC), and the Flash PROMs in a domain's CPU and MaxCPU boards after SMS software upgrades or applicable patch installation. `flashupdate` displays both the current Flash PROM and the flash image file information prior to any updates.

Note – Once you have updated the SC FPROMs you must reset the SC using the `reset-all` command at the OpenBoot PROM (`ok`) prompt. No CLIs should be executed on a system board while `flashupdate` is running on that board. Wait until `flashupdate` completes before running any SMS commands involving that system board.

Note – After running a `flashupdate` command, new firmware is not active on system boards until the system power-on self test (POST) control application, `hpost`, is performed per board with a dynamic reconfiguration operation. For single boards, use the `deleteboard`(1M) or `addboard`(1M) commands to perform an `hpost`. For all boards in a domain, use the `setkeyswitch`(1M) command.

For more information and examples, refer to the `flashupdate` man page.

Configuration for Domain Administrators

This section briefly describes the configuration services available to the domain administrator.

Configuring Domains

The addboard, deleteboard, and moveboard commands offer more functionality to the domain administrator than to the platform administrator.

▼ To Add Boards to a Domain From the Command Line

1. **Log in to the SC as a domain administrator for that domain.**

Note – For the domain administrator to add a board to a domain, that board must appear in the domain's available component list.

The system board must be in the available or assigned state to the domain to which it is being added. Use the showboards (1M) command to determine a board's state.

2. Type the following command:

```
sc0:sms-user:> addboard -d domain-indicator -c function location
```

where:

`-d` *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-c` *function* Specifies the transition of the board from the current configuration state to a new configuration state.

location The board (DCU) location.

Configuration states are as follows:

`assign` Assigns the board to the logical domain. The board belongs to the domain but is not active.

`connect` Transitions an assigned board to the connected/unconfigured state. This is an intermediate state and has no standalone implementation.

`configure` Transitions an assigned board to the connected/configured state. The hardware resources on the board can be used by Solaris software.

If the `-c` *function* option is not specified, the default expected configuration state is `configure`. For more detailed information on the configuration states, refer to the `addboard`(1M) man page.

Multiple locations are accepted.

The following *location* forms are accepted:

Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

For example:

```
sc0:sms-user:> addboard -d C -c assign SB0 I01 SB1 I02
```

In this example, SB0, IO1, SB1, and IO2 have changed from being available to domain C to being assigned to it.

`addboard` performs tasks synchronously and does not return control to the user until the command is complete. If the board is not powered on or tested, specify the `-c connect|configure` option, then the command will power on the board and test it.

If `addboard` fails, the board does not return to its original state. A `dxs` or `dca` error is logged to the domain and `pcd` reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain in order to use that board.

▼ To Delete Boards From a Domain From the Command Line

1. **Log in to the SC as a domain administrator for that domain.**

   The system board must be in the `assigned` or `active` state to the domain from which it is being deleted. Use the `showboards` (1M) command to determine a board's state.

2. Type the following command:

```
sc0:sms-user:> deleteboard -c function location
```

where:

`-c` *function* Specifies the transition of the board from the current configuration state to a new configuration state.

location The board (DCU) location.

Configuration states are:

`unconfigure` Transitions an assigned board to the connected or unconfigured state. The hardware resources on the board can no longer be used by Solaris software.

`disconnect` Transitions an assigned board to the disconnected or unconfigured state.

`unassign` Unassigns the board from the logical domain. The board no longer belongs to the domain, and its state is changed to `available`.

If the `-c` *function* option is not specified, the default expected configuration state is `unassign`. For more detailed information on the configuration states, refer to the `deleteboard`(1M) man page.

Multiple locations are accepted.

The following *location* forms are accepted:

Valid form for Sun Fire 15K/E25K	Valid form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

For example:

```
sc0:sms-user:> deleteboard -c unassign SB0
```

In this example, SB0 has changed from being assigned to the domain to being available to it.

Note – A domain administrator can unconfigure and disconnect a board but is not allowed to delete a board from a domain unless the deleteboard [*location*] field appears in the domain's available component list.

If deleteboard fails, the board does not return to its original state. A dxs or dca error is logged to the domain and pcd reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain in order to use that board.

▼ To Move Boards Between Domains From the Command Line

Note – You must have domain administrator privileges for both domains involved.

1. **Log in to the SC as a domain administrator for that domain.**

   The system board must be in the assigned or active state to the domain from which it is being deleted. Use the showboards (1M) command to determine a board's state.

2. Type the following command:

```
sc0:sms-user:> moveboard -d domain-indicator  -c function location
```

where:

`-d` *domain-indicator* — This is the domain to which the board is being moved. Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-c` *function* — Specifies the transition of the board from the current configuration state to an new configuration state.

location — The board (DCU) location.

Configuration states are:

`assign` — Unconfigures the board from the current logical domain. Moves the board out of the logical domain by changing its state to `available`. Assigns the board to the new logical domain. The board belongs to the new domain but is not active.

`connect` — Transitions an assigned board to the connected or unconfigured state. This is an intermediate state and has no standalone implementation.

`configure` — Transitions an assigned board to the connected or configured state. The hardware resources on the board can be used by Solaris software.

If the `-c` option is not specified, the default expected configuration state is `configure`. For more detailed information on the configuration states, refer to the `moveboard`(1M) man page.

The following *location* forms are accepted:

Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K
SB(0...17)	SB(0...8)
IO(0...17)	IO(0...8)

moveboard performs tasks synchronously and does not return control to the user until the command is complete. If the board is not powered on or tested, specify -c connect|configure; then, the command will power on the board and test it. You can only specify one *location* when using moveboard.

If moveboard fails, the board does not return to its original state. A dxs or dca error is logged to the domain and pcd reports an error to the platform log file. If the error is recoverable, you can retry the command. If it is *unrecoverable*, you must reboot the domain the board was in when the error occurred, in order to use that board.

▼ To Set Domain Defaults

The SMS setdefaults(1M) command removes all instances of a previously active domain.

1. **Log in to the SC.**

   Domain administrators can set domain defaults for all domains, but only one domain at a time. The domain must *not* be active and setkeyswitch must be set to off. setdefaults removes all pcd entries except network information, log files and, optionally, NVRAM and boot parameter data.

2. **Type the following command:**

```
sc0:sms-user:> setdefaults -d domain-indicator
```

where:

-d *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using addtag(1M).

For more information on setdefaults, refer to the setdefaults man page or the *System Management Services (SMS) 1.6 Reference Manual*.

▼ To Obtain Board Status

1. **Log in to the SC.**

   Domain administrators can obtain board status only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showboards [-d domain-id|domain-tag]
```

The board status is displayed.

The following partial example shows the board information for a user with domain administrator privileges for domain A.

```
sc0:sms-user:> showboards -d A

Location     Pwr    Type  Board Status  Test Status   Domain
-------     -----   ----  ------------  -----------   ------
SB1          On     CPU    Active        Passed        A
SB2          On     CPU    Active        Passed        A
IO1          On     HPCI   Active        Passed        A
```

▼ To Obtain Domain Status

1. **Log in to the SC.**

   Domain administrators can obtain domain status only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showplatform -d domain-indicator
```

where:

-d *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using addtag(1M).

The status listing is displayed.

The following partial example shows the domain information for a user with domain administrator privileges for domains newA, engB, and domainC.

```
sc0:sms-user:> showplatform
...
Domain configurations:
======================
Domain ID Domain Tag    Solaris Nodename   Domain Status
A         newA          sun15-b0           Powered Off
B         engB          sun15-b1           Keyswitch Standby
C         domainC       sun15-b2           Running OBP
```

▼ To Obtain Device Status

1. **Log in to the SC.**

Domain administrators can obtain board status only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showdevices [-d domain-id|domain-tag]
```

The device status is displayed.

The following partial example shows the device information for a user with domain administrator privileges for domain A.

```
sc0:sms-user:> showdevices IO1

IO Devices
----------
domain  location device  resource          usage
A       IO1      sd3     /dev/dsk/c0t3d0s0  mounted filesystem "/"
A       IO1      sd3     /dev/dsk/c0t3s0s1  dump device (swap)
A       IO1      sd3     /dev/dsk/c0t3s0s1  swap area
A       IO1      sd3     /dev/dsk/c0t3d0s3  mounted filesystem "/var"
A       IO1      sd3     /var/run           mounted filesystem "/var/run"
```

Virtual Keyswitch

Each Sun Fire high-end system domain has a virtual keyswitch. Like the Sun Enterprise server's physical keyswitch, the Sun Fire high-end system domain virtual keyswitch controls whether the domain is powered on or off, whether increased diagnostics are run at boot, and whether certain operations (for example, flash PROM updates and domain reset commands) are permitted.

Only domains configured with their virtual keyswitch powered on are booted, monitored, and subject to automatic recovery actions, should they fail.

Virtual keyswitch settings are implemented using setkeyswitch(1M). You must have domain administrator privileges for the specified domain in order to run setkeyswitch. See "All Privileges" on page 43 for more information.

The setkeyswitch Command

setkeyswitch (1M) changes the position of the virtual key switch to the specified value. pcd (1M) maintains the state of each virtual key switch between power cycles of the SC or physical power cycling of the power supplies.

`setkeyswitch`(1M) is responsible for loading the bootbus SRAM of all the configured processors. All the processors are started, with one processor designated as the boot processor. `setkeyswitch`(1M) loads OpenBoot PROM into the memory of the Sun Fire high-end system domain and starts OpenBoot PROM on the boot processor.

The primary task of OpenBoot PROM is to boot and configure the OS from either a mass storage device or from a network. OpenBoot PROM also provides extensive features for testing hardware and software interactively.

The `setkeyswitch`(1M) command syntax follows:

```
sc0:sms-user:> setkeyswitch -d domain-indicator [-q -y|-n]
on|standby|off|diag|secure -l level
```

where:

`-d` *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

`-q` Quiet. Suppresses all messages to `stdout` including prompts. When used alone `-q` defaults to the `-n` option for all prompts. When used with either the `-y` or the `-n` option, `-q` suppresses all user prompts, and automatically answers with either Y or N based on the option chosen.

`-n` Automatically answers no to all prompts. Prompts are displayed unless used with `-q` option.

`-y` Automatically answers yes to all prompts. Prompts are displayed unless used with `-q` option.

`-l` *level* Specifies the `hpost` level to be used at system startup.

The following operands are supported:

- `on`
  - From the `off` or `standby` position, `on` powers on all boards assigned to the domain (if not already powered on), then the domain is brought up.
  - From the `diag` position, `on` is a position change and does not affect a running domain.
  - From the `secure` position, `on` restores write permission to the domain.

- standby
  - From the on, diag, or secure position, standby optionally displays a confirmation prompt. If you answer 'yes' then it determines if the domain is in a suitable state to be reset and deconfigured (for example, the OS is not running).
  - If the domain is in a suitable state to be reset and deconfigured, then setkeyswitch resets and deconfigures all boards assigned to the domain.
  - If the domain is not in a suitable state, then before the reset and deconfiguration occur, setkeyswitch gracefully shuts down the domain.
  - From the off position, standby powers on all boards assigned to the domain (if not already powered on).
- off
  - From the on, diag, or secure position, off optionally displays a confirmation prompt. If you answer 'yes' it then determines if the domain is in a suitable state to be powered off (for example, the OS is not running).
  - If the domain is in a suitable state to be powered off, then setkeyswitch powers off all boards assigned to the domain. If it is not, then setkeyswitch aborts and logs a message to the domain log.
  - From the standby position, off powers off all the boards in the domain.
- diag
  - From the off or standby position, diag powers on all boards assigned to the domain (if not already powered on). Then the domain is brought up just as in the on position, except that POST is invoked with verbosity and diag levels set to their defaults (at minimum).
  - From the on position, diag is nothing more than a position change, but upon automatic system recovery (ASR) of the domain, POST is invoked with verbosity and the diag levels set to their defaults (at minimum).
  - From the secure position, diag restores write permission to the domain and upon ASR, POST is invoked with verbosity and the diag levels set to their defaults.

  For more information on ASR, see "Automatic System Recovery (ASR)" on page 165.
- secure
  - From the off or standby position, secure powers on all boards assigned to the domain (if not already powered on). Then the domain is brought up just as in the on position, except that the secure position removes write permission to the domain. For example, flashupdate and reset will not work.
  - From the on position, secure removes write permission to the domain. From the diag position, secure removes write permission to the domain (as described in the diag example).

▼ To Set the Virtual Keyswitch On in Domain A

1. **Log in to the SC.**

   Domain administrators can set the virtual keyswitch only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> setkeyswitch -d A on
```

`showkeyswitch` (1M) displays the position of the virtual keyswitch of the specified domain. The state of each virtual keyswitch is maintained between power cycles of the SC or physical power cycling of the power supplies by the `pcd` (1M). Superuser or any member of a platform or domain group can run `showkeyswitch`.

▼ To Display the Virtual Keyswitch Setting in Domain A

1. **Log in to the SC.**

   Domain administrators can obtain keyswitch status only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showkeyswitch -d A
Virtual keyswitch position: ON
```

Virtual NVRAM

Each domain has a virtual NVRAM containing OpenBoot PROM data, such as the OpenBoot PROM variables. OpenBoot PROM is a binary image stored on the SC in `/opt/SUNWSMS/hostobjs` which `setkeyswitch` downloads into domain memory at boot time. There is only one version of OpenBoot PROM for all domains.

SMS software provides a virtual NVRAM for each domain and allows OpenBoot PROM full read/write access to this data.

The only interface available to read from or write to most NVRAM variables is OpenBoot PROM. The exceptions are those OpenBoot PROM variables which must be altered in order to bring OpenBoot PROM up in a known working state, or to diagnose problems that hinder OpenBoot PROM from coming up. These variables are not a replacement for the OpenBoot PROM interface.

These limited number of OpenBoot PROM variable values in the domain NVRAM are readable and writable from SMS using setobpparams(1M). You must have domain administrator privileges to run set/showobpparams. If you change variables for a running domain, you must reboot the domain in order for the changes to take effect.

Note – Only experienced system administrators who are familiar with OpenBoot PROM commands and their dependencies should attempt to use setobpparams in any manner other than that described.

Setting the OpenBoot PROM Variables

setobpparams(1M) sets and gets a subset of a domain's virtual NVRAM variables and REBOOTINFO data using the following syntax.

```
sc0:sms-user:> setobpparams -d domain-indicator param=value...
```

where:

-d *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using addtag(1M).

param=value is one of the following variables and its corresponding value:

Variables	=	Default Value	Comment
diag-switch?	=	false	When set to false, the default boot device is specified by boot-device and the default boot file by boot-file. If set to true, OpenBoot PROM runs in diagnostic mode, and you must set either diag-device or diag-file to specify the correct default boot device or file. These default boot device and file settings cannot be set using setobpparams. Use setenv(1) in OpenBoot PROM.
auto-boot?	=	false	When set to true, the domain boots automatically after poweron or reset-all. The boot device and boot file used are based on the settings for diag-switch (see above). Neither boot-device nor boot file can be set using setobpparams. If the ok prompt is unavailable during such as a repeated panic, use setobpparams to set auto-boot? to false. When the auto-boot? variable is set to false using setobpparams, the reboot variables are invalidated. In addition, the system will not boot automatically and will stop at OpenBoot PROM. At that point, you can set new NVRAM variables. See "To Recover From a Repeated Domain Panic" on page 117.
security-mode	=	none	Firmware security level. Valid variable values for security-mode are: • none – No password required (default). • command – All commands except for boot(1M) and go require the password. • full – All commands except for go require the password.
use-nvramrc?	=	false	When set to true, this variable executes commands in NVRAMRC during system startup.
fcode-debug?	=	false	When set to true, this variable includes name fields for plug-in device FCodes.

The following is an example of how setobpparams can be useful.

To Recover From a Repeated Domain Panic

In the following example, domain A encounters repeated panics caused by a corrupted default boot disk.

1. **Log in to the SC with domain administrator privileges.**
2. **Stop automatic reboot:**

```
sc0:sms-user:> setkeyswitch -d A standby
sc0:sms-user:> setobpparams -d A ’auto-boot?=false’
```

Note – Most, but not all, shells require using single quotes around the variable values to prevent the question mark from being treated as a special character.

---

3. **Repost the domain:**

```
sc0:sms-user:> setkeyswitch -d A off
sc0:sms-user:> setkeyswitch -d A on
```

4. **Once the domain has come up to the OK prompt, set NVRAM variables to a new uncorrupted boot-device.**

```
ok setenv boot-device bootdisk-alias
```

where:

bootdisk-alias A user-defined alias you created. The boot device must correspond to the bootable disk on which you have installed the OS.

5. **Now that you have set up a new alias for your boot device, boot the disk by typing:**

```
ok boot
```

For more information on OpenBoot variables, refer to the *OpenBoot 4.x Command Reference Manual*.

▼ To Set the OpenBoot PROM Security Mode Variable in Domain A

1. **Log in to the SC.**

   Domain administrators can set the OpenBoot PROM variables only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> setobpparams -d A security-mode=full
```

`security-mode` has been set to full. All commands except `go` require a password on domain A. You must reboot a running domain in order for the change to take effect.

▼ To See the OpenBoot PROM Variables

1. **Log in to the SC.**

   Domain administrators can set the OpenBoot PROM variables only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showobpparams -d domain-indicator
```

where:

`-d` *domain-indicator* Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

SMS NVRAM updates are supplied to OpenBoot PROM at OpenBoot PROM initiation (or domain reboot time). For more information refer to the *OpenBoot PROM 4.x Command Reference Manual.*

Degraded Configuration Preferences

In most situations, hardware failures that cause a domain crash are detected and eliminated from the domain configuration either by POST or OpenBoot PROM during the subsequent automatic recovery boot of the domain. However, there can be situations where failures are intermittent or the boot-time tests are inadequate to detect failures that cause repeated domain failures and reboots. In those situations, Sun Fire high-end system management software uses configurations or configuration policies supplied by the domain administrator to eliminate hardware from the domain configuration in an attempt to get a stable domain environment running.

The following commands can be run by either platform or domain administrators. Domain administrators are restricted to the domains for which they have privileges.

The `setbus` Command

`setbus`(1M) dynamically reconfigures bus traffic on active expanders in a domain to use either one centerplane support board (CSB) or both. Using both CSBs is considered *normal* mode. Using one CSB is considered *degraded* mode.

`setbus` resets any boards that are powered on but not active. Any attach-ready state is lost. For more information on attach-ready states, refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.

You must have platform administrator privileges or domain privileges for the specified domain in order to run `setbus`.

This feature allows you to swap out a CSB without having to power off the system. Valid buses are:

- `a` – configures the address bus
- `d` – configures the data bus
- `r` – configures the response bus

▼ To Set All Buses on All Active Domains to Use Both CSBs

1. **Log in to the SC.**

   Domain administrators can set the bus only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> setbus -c CS0,CS1
```

For more information on reconfiguring bus traffic, refer to the `setbus`(1M) man page.

The `showbus` Command

`showbus`(1M) displays the bus configuration of expanders in active domains. This information defaults to displaying configuration by slot order. Any member of a platform or domain group can run `showbus`.

▼ To Show All Buses on All Active Domains

1. **Log in to the SC.**
2. **Type the following command:**

```
sc0:sms-user:> showbus
```

For more information on reconfiguring bus traffic, refer to the `showbus`(1M) man page.

CHAPTER 6

Automatic Diagnosis and Recovery

This chapter describes the automatic error diagnosis and domain recovery features. This chapter contains the following sections:

Automatic Diagnosis and Recovery Overview

When certain hardware errors occur in a Sun Fire high-end system, the system controller performs specific diagnosis and domain recovery steps. The following automatic diagnosis engines (DEs) identify and diagnose hardware errors that affect the availability of the system and its domains:

- SMS diagnosis engine

  The SMS DE diagnoses hardware errors associated with domain stops (dstops).

- Solaris OS diagnoses engine

  The Solaris OS DE (also referred to as the Solaris DE) identifies nonfatal domain hardware errors and reports them to the system controller.

- POST diagnosis engine

  The POST DE identifies any hardware test failures that occur when the power-on self-test is run.

The following sections describe the diagnosis and recovery steps that occur for the hardware errors identified by the different diagnosis engines.

Hardware Errors Associated With Domain Stops

FIGURE 6-1 shows the basic diagnosis and domain recovery steps performed when hardware errors associated with a dstop are identified by the SMS diagnosis engine.

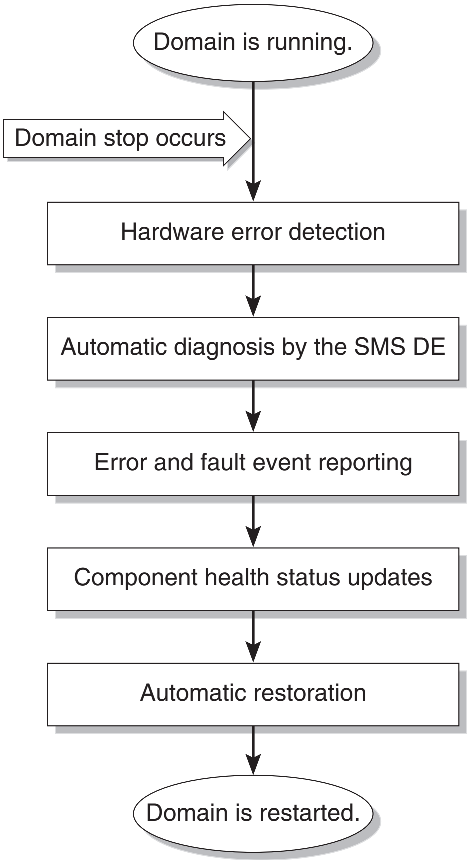


FIGURE 6-1 Automatic Diagnosis and Recovery Process for Hardware Errors Associated With a Stopped Domain

The following summary describes the process shown in FIGURE 6-1.

- **Hardware error detection.** The system controller provides information on hardware errors involving CPU boards, processors, I/O controllers, and memory banks.

A dump file is generated whenever a dstop occurs. This file (`/var/opt/SUNWSMS/`*sms-version*`/adm/`*domain-id*`/dump/dsmd.dstop.`*yymmdd.hhmm.ss*) captures the domain hardware errors associated with the dstop.

- **Automatic diagnosis.** The SMS DE determines a failure based on the hardware errors captured in the dstop dump file. The DE might identify one or more FRUs that are responsible for the error. Depending on the hardware error, the DE might identify one faulty FRU or one or more suspect FRUs.

  In situations where multiple FRUs are identified by the DE, further analysis by your service provider might be required to determine the faulty FRU.

- **Error and fault event reporting**. The DE reports diagnosis information through the following:
  - Auto-diagnosis fault messages that appear in the domain and platform log files.

    CODE EXAMPLE 6-1 shows the information displayed for a domain stop and the auto-diagnosis message that describes a fault event on domain D. The event message begins with the `[AD]` indicator. See "Reviewing Diagnosis Events" on page 138 for a description of the event message contents.

CODE EXAMPLE 6-1 Example of a Dstop and Auto-Diagnosis Event Message in the Platform Log File

```
Jul 30 14:23:26 2005 smshostname dsmd[14838]-D(): [2516 589424843782403 ERR
EventHandler.cc 136] Domain stop has been detected in domain D
Jul 30 14:23:27 2005 smshostname dsmd[14838]-D(): [2525 589425136691417 NOTICE
SysControl.cc 2360] Taking hardware configuration dump. Dump
file: -D/var/opt/SUNWSMS/SMS1.6/adm/D/dump/dsmd.dstop.030730.1423.27
Jul 30 14:24:37 2005 smshostname erd[14864]-D(): [11900 589495236849691 CRIT Mes
sageReportingService.cc 381] [AD]  Event: SF15000-8000-GK  CSN: 352A00005
DomainID: D  ADInfo: 1.SMS-DE.1.6 Time: Wed Jul 30 14:23:27 PDT 2005
Recommended-Action: Service action required
```

  - Email notification of fault events. For details, see "Enabling Email Event Notification" on page 127.
  - Fault event notification if you are using Sun Management Center. For details, refer to the *Sun Management Center Supplement for Sun Fire High-End Systems*.
  - Notification of fault events if you are using Sun Remote Services (SRS) Net Connect and have configured Net Connect accordingly.

For general information on SRS Net Connect, refer to

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

For SRS Net Connect product documentation, refer to

```
https://srsnetconnect3.sun.com
```

and

```
http://docs.sun.com
```

- Event log output from the `showlogs` (1M) command if you have platform administrator privileges

  The `showlogs` event output supplements the diagnosis information presented in the platform and domain message logs or the event email. The `showlogs` event output can be used for additional troubleshooting purposes by your service provider. For details on the event information displayed, see "Obtaining Diagnosis and Recovery Information" on page 138.

Note – Contact your service provider when you see these event messages or when you are notified of these events. Your service provider will review the auto-diagnosis information and initiate the appropriate service action.

- **Component health status updates**. The SMS DE records the diagnosis information for each affected component and maintains this health history as part of the component health status (CHS).
- **Automatic restoration.** As part of the domain restoration process, POST reviews the updated component health status of the affected components and uses the CHS information to determine which components to deconfigure from the system. The appropriate components are then deconfigured, and the domain is restarted.

Nonfatal Domain Hardware Errors

FIGURE 6-2 shows the basic steps involved in the diagnosis of nonfatal domain hardware errors. These errors do not cause a domain to stop.

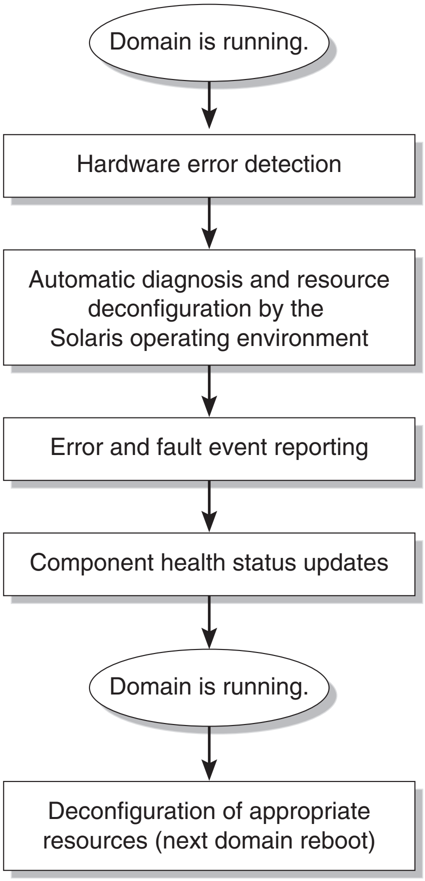

Domain is running.
Hardware error detection
Automatic diagnosis and resource deconfiguration by the Solaris operating environment
Error and fault event reporting
Component health status updates
Domain is running.
Deconfiguration of appropriate resources (next domain reboot)

FIGURE 6-2 Automatic Diagnosis Process for Nonfatal Domain Hardware Errors

The steps shown in FIGURE 6-2 are similar to the steps discussed in the section "Hardware Errors Associated With Domain Stops" on page 122, except for the following differences:

- **Hardware error detection.** The Solaris OS determines when a nonfatal domain hardware error has occurred and reports the error to the system controller. The affected domain is not stopped.
- **Automatic diagnosis and resource deconfiguration.** The Solaris OS identifies the failure and the resources that caused the failure. If appropriate, the Solaris OS may also deconfigure the affected resources. For example, a CPU module might be taken offline because of nonfatal errors that occur within the module, or a virtual memory page might be retired due to errors contained in the page.

- **Error and fault event reporting**. The Solaris OS provides diagnosis information through the same channels as the SMS DE: event messages that appear in the domain and platform logs, fault event notification if using Sun Management Center, or email event notification within SMS or through SRS Net Connect if you configured those features, and `showlogs`(1M) event output.

  CODE EXAMPLE 6-2 shows the diagnosis of a nonfatal hardware error and the event message information displayed. The event message begins with the `[DOM]` indicator. See "Reviewing Diagnosis Events" on page 138 for a description of the event message contents.

CODE EXAMPLE 6-2 Example of a Nonfatal Domain Hardware Error Identified by Solaris and the Domain Event Message

```
Sep 12 14:47:24 2005 smshostname dsmd[7839]: [0 876197473671508 ERR
SoftErrorHandler.cc 577] E$ Slot 3 SubSlot 5
Sep 12 14:47:25 2005 smshostname dsmd[7839]: [2552 876198449525014 ERR
SoftErrorHandler.cc 592] Soft Error: Comp ID : 0x62 Error Code: 3 Error Type: 1
Error Bit/Pin: 104
Sep 12 14:47:58 2005 smshostname erd[17227]: [11900 876231607099583 CRIT
MessageReportingService.cc 243] [DOM]  Event: SF15000-8000-FF  CSN: 352A00006
DomainID: D  ADInfo: 1.SF-SOLARIS-DE.5-9-cs3:4791004-on81:08/18/2005  Time: Fri
Sep 12 14:47:38 PDT 2005  Recommended-Action: Service action required
```

Note – Contact your service provider when you see these event messages or when you are notified of these events. Your service provider will review the auto-diagnosis information and initiate the appropriate service action.

- **Component health status updates.** SMS updates the component health status of the affected hardware resources, using the information supplied by the Solaris OS.
- **Deconfiguration of appropriate resources.** In cases where the Solaris OS could not previously deconfigure faulty domain resources, those resources are deconfigured from the system at the next domain reboot.

POST-Detected Hardware Failures

Whenever POST is run to test and configure system board components, any components that fail the self-test are automatically unconfigured from the system. POST updates the component health status of the affected components accordingly.

CODE EXAMPLE 6-3 shows an auto-diagnosis event message reported by the POST DE for Domain B. See "Reviewing Diagnosis Events" on page 138 for a description of the event message contents.

CODE EXAMPLE 6-3 Example of a POST Auto-Diagnosis Event Message

```
Sep  8 13:31:16 2005 smshostname erd[11987]: [11900 240509936296585 C
RIT
MessageReportingService.cc 243] [AD]  Event: SF15000-8000-4L  CSN: 352A00005
DomainID: B  ADInfo: 1.POST-DE.1.4.1  Time: Mon Sep  8 13:30:47 PDT 2005
Recommended-Action: Service action required
```

When you see these messages or when you are notified of these events, contact your service provider to initiate the appropriate service action.

Enabling Email Event Notification

Email event notification is an optional feature that automatically generates an email notice informing designated recipients of domain fault events when they occur. You can receive immediate notice of critical fault events without manually monitoring the platform or domain message logs.

CODE EXAMPLE 6-4 shows an example email that reports a fault event in which two components are indicted (suspected of causing a fault). The following sections explain how to control email content and notification.

CODE EXAMPLE 6-4 Example Event Email

```
Date: Tue, 19 Aug 2005 10:45:28 -0600 (MDT)
Subject: FAULT: SF15000, csn: 352A00007, main fault class: list.suspects
From: smshostname@xyz.com
To: undisclosed-recipients:;

FAULT: platform: SF15000, csn: 352A00007, main fault class: list.suspects
EVENT CODE: SF15000-8000-GK
EMBEDDED FAULT(S): fault.board.sb.1112
fault.board.ex.1112

Fault event in domain(s) R at Fri Jun 27 00:08:05 PDT 2005.
Fault severity = SMIEVENT_SEV_FATAL <7>
Indictment Count: 2
Indictment list:
sb11
ex11
```

The following files work together to generate event email:

- Email template

  This template identifies the event information to be reported in the email. This information includes the email subject line and specific event items (tags) to be reported in the email.

- Email control file (`event_email.cf`)

  This file (`/etc/opt/SUNWSMS/SMS/config/event_email.cf`) uses certain event information, namely the event class and the domain affected by the event, to assign the specified email recipients and email templates that control the event information to be reported.

Note – The event email feature uses the standard `sendmail` utility to send email to designated email recipients.

▼ To Enable Email Event Notification

1. **In the email template file, identify the event tags to be reported in email.**

   Copy the sample email template (`sample_email`) provided with SMS and edit the copied file. For details on modifying the email template, see "Configuring an Email Template" on page 129.

2. **In the email control file, set the parameters that determine who receives the email and the email templates to be used.**

   Edit the email control file (`event_email.cf`) included with SMS and assign the email notification parameters.

   For details on modifying the control file, see "Configuring the Email Control File" on page 132.

Note – If you use the email notification feature, review the email destination addresses to ensure that the recipients receive notifications for events pertaining only to the domains that they have authorization to see. Implement and enforce a process for maintaining appropriate security separation whenever people change responsibilities, and gain or lose authorization.

Configuring an Email Template

A sample email template file called `sample_email` (`/etc/opt/SUNWSMS/SMS/config/templates`) is provided with SMS. CODE EXAMPLE 6-5 shows the default template. The text in angle brackets identifies the event information to be displayed in the body of the event email.

CODE EXAMPLE 6-5 Default Sample Email Template

```
# Sample Email Template File - This sample is intended to convey
# a terse fault event notification to a pager.
#
# The following is the subject line for the email with the event
# descriptor from the event and the platform model and serial
# number inserted.
#
FAULT: <PLATFORM_MODEL>, serial# <PLATFORM_SERIAL_NUMBER>, code <EVENT_CODE>
#
# The following lines are the body of the email notification.
#
Fault event in domain(s) <EVENT_DOMAINS_AFFECTED> at <EVENT_TIMESTAMP>.
Fault severity = <EVENT_SEVERITY>
```

CODE EXAMPLE 6-5 Default Sample Email Template *(Continued)*

```
Indictment Count: <EVENT_INDICTMENT_COUNT>
Indictment list:
<EVENT_INDICTMENT_LIST>

Member fault list:
<EVENT_FAULT_MEMBERS>
# End of email template.
```

You can use the sample template file as is, or you can copy the sample template file to a new file, which can then be edited to identify additional or different event tags to be contained in the email. You must have superuser privileges to copy and rename the sample template file. The name of the file can be any text string that you choose.

When you edit the file, specify the event tags to be reported in the email subject line and email body. Specify these tags on new, uncommented lines in the file (lines that do not begin with a # sign). For a list of the tags that can be specified in the email template, see TABLE 6-1.

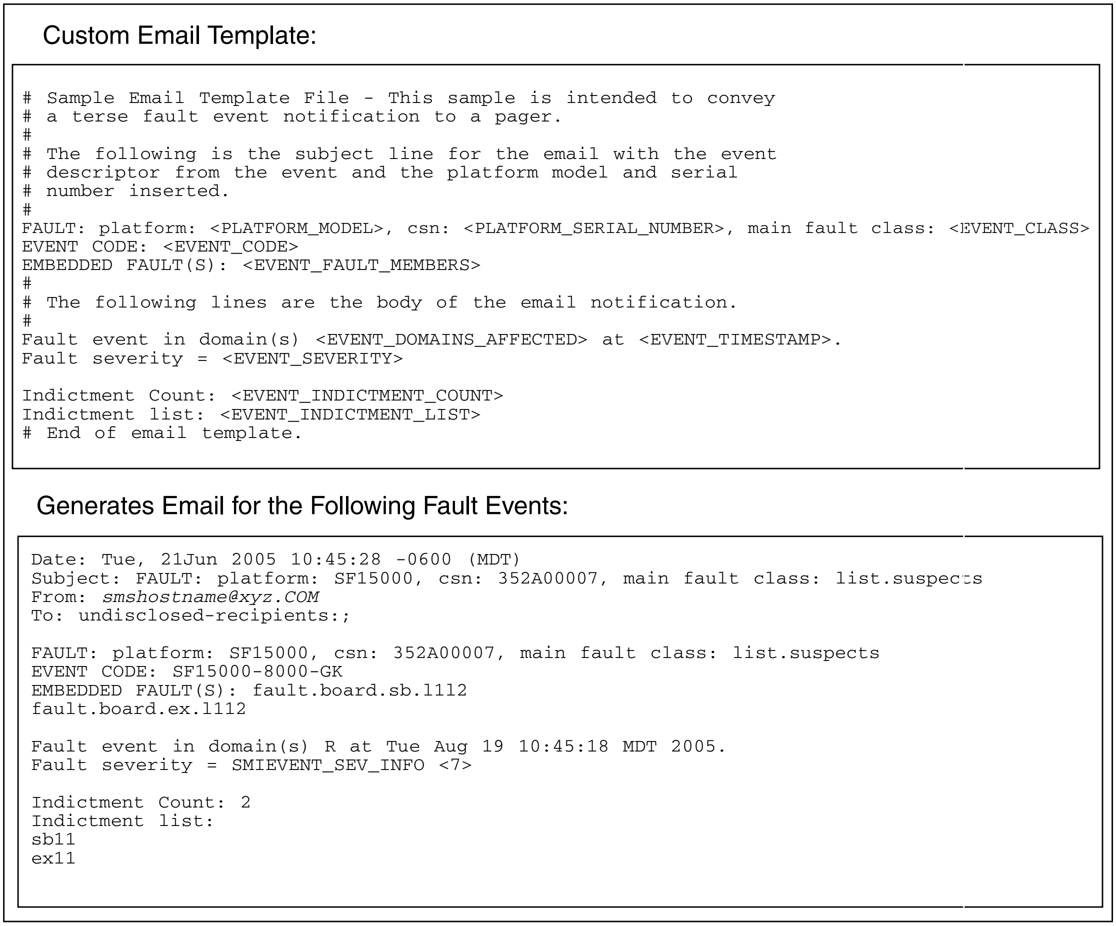
TABLE 6-1 Event Tags in the Email Template File

Event Tag	Information Displayed
`<EVENT_CLASS>`	A dot-separated alphanumeric text string that describes the event category (error report, fault event, or a list of suspected faults). For example: `list.suspects`
`<EVENT_CODE>`	A dash-separated alphanumeric text string that uniquely identifies an event type, for example: `SF15000-8000-GK`. The event code summarizes the fault classes involved in the event and is used by your service provider to obtain further information about the event.
`<EVENT_DE_NAME>`	Name of the diagnosis engine (DE) used to determine the fault event: `SMS-DE`, `SF-SOLARIS-DE`, or `POST-DE`.
`<EVENT_DE_VERSION>`	Version of the diagnosis engine used to determine the event.
`<EVENT_DOMAINS_AFFECTED>`	A comma-separated list of domains affected by the event.
`<EVENT_FAULT_MEMBERS>`	List of fault event classes associated with the fault event. For example: `fault.board.sb.l112`
`<EVENT_INDICTMENT_COUNT>`	Number of components indicted or suspected of causing the fault event.
`<EVENT_INDICTMENT_LIST>`	The indicted components. Each component is listed on a separate line.

TABLE 6-1 Event Tags in the Email Template File *(Continued)*

Event Tag	Information Displayed
`<EVENT_SEVERITY>`	The severity of the event, ranging from 0 to 7. For example, test event messages have a severity level 2 and fault events that cause a domain stop have a severity level 7 (`SMIEVENT_SEV_FATAL`).
`<EVENT_TIMESTAMP>`	The day and time of the event.
`<PLATFORM_SERIAL_NUMBER>`	The chassis serial number that identifies the Sun Fire high-end system.
`<PLATFORM_MODEL>`	The number of the product model `(SF15000, SFE25000, SF12000 or SFE20000)` affected by the event.

FIGURE 6-3 shows the email template used to generate the email example shown in CODE EXAMPLE 6-4.



Custom Email Template:

```
# Sample Email Template File - This sample is intended to convey
# a terse fault event notification to a pager.
#
# The following is the subject line for the email with the event
# descriptor from the event and the platform model and serial
# number inserted.
#
FAULT: platform: <PLATFORM_MODEL>, csn: <PLATFORM_SERIAL_NUMBER>, main fault class: <EVENT_CLASS>
EVENT CODE: <EVENT_CODE>
EMBEDDED FAULT(S): <EVENT_FAULT_MEMBERS>
#
# The following lines are the body of the email notification.
#
Fault event in domain(s) <EVENT_DOMAINS_AFFECTED> at <EVENT_TIMESTAMP>.
Fault severity = <EVENT_SEVERITY>

Indictment Count: <EVENT_INDICTMENT_COUNT>
Indictment list: <EVENT_INDICTMENT_LIST>
# End of email template.
```

Generates Email for the Following Fault Events:

```
Date: Tue, 21Jun 2005 10:45:28 -0600 (MDT)
Subject: FAULT: platform: SF15000, csn: 352A00007, main fault class: list.suspects
From: smshostname@xyz.COM
To: undisclosed-recipients:;

FAULT: platform: SF15000, csn: 352A00007, main fault class: list.suspects
EVENT CODE: SF15000-8000-GK
EMBEDDED FAULT(S): fault.board.sb.l112
fault.board.ex.l112

Fault event in domain(s) R at Tue Aug 19 10:45:18 MDT 2005.
Fault severity = SMIEVENT_SEV_INFO <7>

Indictment Count: 2
Indictment list:
sb11
ex11
```

FIGURE 6-3 Example Email Template and Generated Email

Configuring the Email Control File

The email control file contains the email notification parameters that do the following:

- Identify the email recipients based on the event class and the domain in which the event occurred
- Identify the email templates to be used
- Indicate whether the event message structure is to be sent as an attachment with the event email

You specify these notification parameters in the email control file supplied with SMS (/etc/opt/SUNWSMS/SMS/config/event_email.cf). This file, shown in CODE EXAMPLE 6-6, contains comment lines that begin with a pound (#) sign. These comment lines explain how to update the file.

CODE EXAMPLE 6-6 Email Control File (event_email.cf)

```
#
# Copyright (c) 2004 by Sun Microsystems, Inc.
# All rights reserved.
#
# Email Control File
#
# ident "@(#)event_email.cf 1.6     03/08/19 SMI"
#
# The following fields are required to receive email notification of fault
# events
# Event_Class  Domains  Template  From  Include-event?  Recipients Script
# Event_Class and Domains are regular expressions filtering for specific event
# types and affected domains. Domains are required to be upper case.
# The following example, uncommented, generates an email for any List Event
# containing a Fault Event, affecting any domain, and sends it to
# two recipients.
# The Packed Event List is included as an attachment to the email.
#
# Event_Class   Domains   Template   From   Include-event?   Recipients   Script
#^fault[.] [A-R] sample_email FMA@xyz.com Y adm@xyz.com,adm2xyz.com sendmail.sh
#
#
# The following example, uncommented, generates an email for any Event
# that contains a Fault Event and affects domains A through C.  The Packed
# Event List is not sent as an attachment. The user would be required to add his
# custom fault_email template to the directory
# /etc/opt/SUNWSMS/config/templates, and for tag
# replacement to work should refer to the documentation, or look at the
# sample_email template in that directory.
#^fault[.] [A-C] fault_email FMA@xyz.com N admin.manager@xyz.com sendmail.sh
```

Use a text editor to edit the file and add the notification parameters in new, uncommented lines. You must have superuser privileges to edit the email control file and add the required email parameters. Separate each parameter with spaces or tabs. You can enter multiple notification lines that control how different event email messages are to be distributed, perhaps by domain, event class, or email template. The notification parameters that you configure are described in TABLE 6-2.

You can use regular expressions to specify ranges or specific matches for the *Event_Class* and *Domains* parameters. The email control file supports extended regular expressions as explained in the regexp(5) man page. Some examples of valid regular expressions include:

- . (period) – Matches any single character.
- ^ (circumflex) – Forces a match to start at the beginning of the string. For example, ^fault matches any string that starts with fault .
- [BDG] – Matches any single character, B or D or G.
- [B-F] – Matches any single character ranging between B and F, such as B or C or D or E or F.

TABLE 6-2 Email Control File Parameters

Email control parameter	Description
Event_Class	The fault event class to be used as a filter. Specify the event class as a regular expression, so that this parameter can apply to a wide range of event classes. For example, the default format fault.* causes all fault events that match the string fault to be reported in the event email.
Domains	The domains to be used as filters. The default format [A-R] causes the fault events from domains A through R to be identified in the email. The domains must be specified in uppercase letters.
Template	The name of the email template file to be used to generate the email contents.
From	The email alias from which the email is generated.
Include-event?	One of the following states: • Y – Yes, include the binary file of the event message structure as an email attachment. This file can be used by your service provider for troubleshooting purposes. • N – No, do not include the binary file of the event message structure as an email attachment.
Recipients	The email aliases of the individuals to receive the event email. Separate each alias with a comma.
Script	The shell script used to send the email to the designated recipients. The sendmail.sh script in /etc/opt/SUNWSMS/config/scripts is the standard script and is used by default, but you can replace this with your own custom script in the same directory.

CODE EXAMPLE 6-7 shows an updated email control file in which notification parameters have been added to the bottom of the file. The sendmail.sh script will be used to send event email to the two specified recipients. An event email will be generated for all fault events that occurred in domains A through C and will be formatted based on the template file called sample_email. The event message structure will be sent as a binary file attachment that accompanies the email.

CODE EXAMPLE 6-7 Sample Email Control File

```
#
# Copyright (c) 2004 by Sun Microsystems, Inc.
# All rights reserved.
# Email Control File
#
# ident "@(#)event_email.cf 1.1     03/03/12 SMI"
#
# The following fields are required to receive email notification of fault
# events
# Event_Class  Domains  Template  From  Include-event?  Recipients-Script
# Event_Class and Domains are regular expressions filtering for specific event
# types and affected domains. Domains are required to be upper case.
# The following example, uncommented, generates an email for any List Event
# containing a Fault Event, affecting any domain, and sends it to
# two recipients. Recipients are email addresses separated by commas if there
# are more than 1. Embedded blanks are not permitted in the Recipients list.
# The Packed Event List is included as an attachment to the email.
#
# Event_Class  Domains  Template  From  Include-event?  Recipients Script
#^fault[.] [A-R] sample_email FMA@xyz.com Y adm1@xyz.com,adm2@xyz.com sendmail.sh
#
#
# The following example, uncommented, generates an email for any Event
# that contains a Fault Event and affects domains A through C.  The Packed
# Event List is not sent as an attachment. The user would be required to add his
# custom fault_email template to the directory
# /etc/opt/SUNWSMS/config/templates, and for tag
# replacement to work should refer to the documentation, or look at the
# sample_email template in that directory.
#
#^fault[.] [A-C] sample_email FMA@xyz.com Y adm1@xyz.com,adm2@xyz.com sendmail.sh
^fault[.] [A-C] sample_email FMA@xyz.com Y adm1@xyz.com,adm2@xyz.com sendmail.sh
```

Testing Email Event Notification

Use the `testemail`(1M) command to verify email event notification. This command also enables you to track events and check any changes to the email control file.

▼ To Test Email Event Notification

1. **Set up the email event templates and the email control file as described in "Enabling Email Event Notification" on page 127.**

2. **In an SC window, log in as platform administrator or platform service and type:**

```
sc0:sms-user:> /opt/SUNWSMS/SMS/lib/smsadmin/testemail -c event-class-
list -d domain-id [-i resource-indictment-list]
```

where:

event-class-list is a list of one or more fault event classes to be tracked

domain-id specifies a single domain, `A-R`

resource-indictment-list is an optional list of one or more components that map to each event class specified. For a list of the valid component values, refer to the `testemail`(1M) man page.

For example, the following command generates an event type `fault.test.email` originating on domain A.

```
sc0:sms-user:> /opt/SUNWSMS/SMS/lib/smsadmin/testemail -c
fault.test.email -d A
```

3. **Verify that the test event was recorded in the platform or domain message logs.**

For example, a message similar to the following is displayed in the platform message log:

```
Aug 19 10:45:28 2005smshostname [6696:1]: [11917 682823530704603 ERR teste
mailApp.cc 345] Test fault with code SF15000-8000-Y1 generated by user root
using testEmailReporting - please ignore
```

4. **If the test event was successfully recorded in the message logs, verify that the designated recipients received the test email.**

   For example, the test email might resemble the following:

```
Date: Tue, 19 Aug 2005 10:45:28 -0600 (MDT)
Subject: FAULT: SF15000, serial# 352A0008, code SF15000-8000-Y1
From: smshostname@xyz.com
To: undisclosed-recipients:;

FAULT: SF15000, serial# 352A0008, code SF15000-8000-Y1
Fault event in domain(s) A at Tue Aug 19 10:45:18 MDT 2005.
Fault severity = SMIEVENT_SEV_INFO <2>
Indictment Count: 0
Indictment list:

Member fault list:
fault.test.email
```

   If the test email was not generated, review the next section for troubleshooting suggestions.

What To Do If Test Email Fails

If you did not receive test email notification, do the following;

1. Review your email event templates and the email control file to verify that the files have been set up correctly.
2. Check the domain and platform message logs to verify that the test events were recorded.
3. Verify that the `sendmail` daemon is running. For example:

```
sc0:sms-user:> ps -ef | grep sendmail
    root  256     1  0   Aug 06 ?        0:05 /usr/lib/sendmail -bd -q15m
sms-user  525 28546  0 21:23:15 pts/27   0:00 grep sendmail
```

   If the `sendmail` daemon is not running, you might have a problem with your installation setup that requires correction. Proceed to Step 4.

4. Manually start sendmail, which will run until the next reboot, by logging on as superuser and restarting the sendmail daemon:

```
sc0:# /usr/lib/sendmail -bd -q15m &
```

5. Check /var/log/syslog on the SC to see if email was sent by the Mail Transfer Agent (MTA), sendmail.

   If sendmail is not configured or was configured incorrectly, error messages would appear in this log file.

6. Verify that the domain and nameserver IP entries (to route the email messages outside of the system controller) exist in the /etc/resolv.conf file.

7. Restart sendmail.sh:

```
sc0:#:/etc/inet.d/sendmail stop
sc0:#:/etc/inet.d/sendmail start
```

Obtaining Diagnosis and Recovery Information

This section describes the various ways to monitor diagnostic errors and obtain additional information about fault and error events.

Reviewing Diagnosis Events

Automatic diagnosis [AD] and domain [DOM] event messages are displayed in the platform message logs and on the domain console or in the syslog host, if a loghost server was configured. The [AD] or [DOM] event messages (see CODE EXAMPLE 6-1, CODE EXAMPLE 6-2, and CODE EXAMPLE 6-3) include the following information:

- [AD] or [DOM] – Beginning of the message. AD indicates that the SMS or POST automatic diagnosis engine generated the event message. DOM indicates that the Solaris OS on the affected domain generated the automatic diagnosis event message.
- Event – The event code, a dash-separated alphanumeric text string that uniquely identifies an event type. This code is used by your service provider to obtain further information about the event and the platform involved.

- CSN – Chassis serial number, which identifies your Sun Fire high-end system.
- DomainID – The domain affected by the hardware error. Valid domains are A through R.
- ADInfo – The version of the auto-diagnosis message, the name of the diagnosis engine (SMS-DE, SF-SOLARIS-DE, or POST-DE), and the diagnosis engine version (the SMS version or the version of Solaris OS in use).
- Time – The day of the week, month, time (hours, minutes, and seconds), time zone, and year of the auto-diagnosis.
- Recommended-Action: Service action required – Instructs the platform or domain administrator to contact their service provider for further service action. Also indicates the end of the auto-diagnosis message.

Reviewing the Event Log

If you have platform administrator or platform service privileges, you can use the showlogs command to view the contents of the event log, to obtain more detailed information about a particular type of event. The information displayed can also be used by your service provider for troubleshooting purposes.

You can obtain information on the following types (classes) of events recorded in the event log:

- Ereports – Error reports provide data on unexpected component behavior or conditions.
- List events – List events provide a list of fault events or suspected faults associated with a hardware error.

TABLE 6-3 describes some of the various ways to view event information through the showlogs command.

TABLE 6-3 showlogs(1M) Command Options for Displaying Error and Fault Event Information

Command Options	Description
`showlogs -E -p e`	Displays the last event in the event log in a condensed format.
`showlogs -E -p e` *number*	Displays the event data for the last *number* of events in a condensed format. For example, `showlogs -E -p e 3` displays condensed event information for the last three events in the event log,
`showlogs -p e list`	Displays the last list event in the event log.

TABLE 6-3 `showlogs`(1M) Command Options for Displaying Error and Fault Event Information

Command Options	Description
`showlogs -p e ereport`	Displays the last ereport (error report) in the event log. An error report contains specific information about the hardware entity, such as an unexpected condition or behavior.
`showlogs -d` *domain-ID* `-p e` *number*	Displays the last *number* of events in the specified domain.
`showlogs -E -p e` *event-code*	Displays condensed event log information for the specified event code.

For details on the `showlogs` command options and examples of event output, refer to the `showlogs`(1M) command description in the *System Management Services (SMS) 1.6 Reference Manual*.

CHAPTER 7

Capacity on Demand

Product Names are configured with processors (CPUs) on system boards. These boards are purchased as part of your initial system configuration or as add-on components. The right to use the CPUs on these boards is included with the initial purchase price.

The Capacity on Demand (COD) option provides additional processing resources that you pay for when you use them. Through the COD option, you purchase and install unlicensed COD system boards in your system. Each COD system board contains four CPUs, which are considered as available processing resources. However, you do not have the right to use these COD CPUs until you also purchase the right-to-use (RTU) licenses for them. The purchase of a COD RTU license entitles you to receive a license key, which enables the appropriate number of COD processors.

You use COD commands included with the SMS software to allocate, activate, and monitor your COD resources.

This chapter contains the following sections:

COD Overview

The COD option provides additional CPU resources on COD system boards that are installed in your system. Although your Product Name comes configured with a minimum number of standard (active) system boards, your system can have a mix of both standard and COD system boards installed, up to the maximum capacity allowed for the system. At least one active CPU is required for each domain in the system.

If you want the COD option, and your system is not currently configured with COD system boards, contact your Sun sales representative or authorized Sun reseller to purchase COD system boards. Your salesperson will work with your service provider to install the COD system boards in your system.

The following sections describe the main elements of the COD option:

- "COD Licensing Process" on page 142
- "COD RTU License Allocation" on page 142
- "Instant Access CPUs" on page 143
- "Resource Monitoring" on page 144

COD Licensing Process

COD RTU licenses are required to enable COD CPU resources. COD licensing involves the following tasks:

1. Obtaining COD RTU license certificates and COD RTU license keys for COD resources to be enabled.

   You can purchase COD RTU licenses at any time from your Sun sales representative or reseller. You can then obtain a license key (for the COD resources purchased) from the Sun License Center.

2. Entering the COD RTU license keys in the COD license database.

   The COD license database stores the license keys for the COD resources that you enable. You record this license information in the COD license database by using the `addcodlicense`(1M) command. The COD RTU licenses can be used for any COD CPU resource installed in the system.

For details on completing the licensing tasks, see "To Obtain and Add a COD RTU License Key to the COD License Database" on page 145.

COD RTU License Allocation

With the COD option, your system is configured to have a certain number of COD CPUs available, as determined by the number of COD system boards and COD RTU licenses that you purchase. The COD RTU licenses that you obtain are handled as a pool of available licenses.

When you activate a domain containing a COD system board or when a COD system board is connected to a domain through a dynamic reconfiguration (DR) operation, the following occurs automatically:

- The system checks the current installed COD RTU licenses.
- The system obtains a COD RTU license (from the license pool) for each CPU on the COD board.

The COD RTU licenses are allocated to the CPUs on a "first come, first serve" basis. However, you can allocate a specific quantity of RTU licenses to a particular domain by using the `setupplatform`(1M) command. For details, see "To Enable Instant Access CPUs and Reserve Domain RTU Licenses" on page 150.

If there is an insufficient number of COD RTU licenses and a license cannot be allocated to a COD CPU, the COD CPU is not configured into the domain and is considered as unlicensed. A COD CPU is considered to be unused when it is assigned to a domain but the CPU is not active.

If a COD system board does not have sufficient COD RTU licenses for its COD CPUs, the system will disable the unlicensed CPUs and configure the board into the domain. If none of the CPUs have COD RTU licenses, then the system will fail the entire board, and will not configure that board into the domain. For additional details and examples, see "Deconfigured and Unlicensed COD CPUs" on page 158.

When you remove a COD system board from a domain through a DR operation or when a domain containing a COD system board is shut down normally, the COD RTU licenses for the CPUs on those boards are released and added to the pool of available licenses.

You can use the `showcodusage` command to review COD usage and COD RTU license states. For details on `showcodusage` and other commands that provide COD information, see "COD Resource Usage" on page 153.

Note – You can move COD boards between Sun Fire high-end systems (Sun Fire 25K/E15K, 20K/E12K, 6800, 4810, 4800, and 3800 servers), but the associated license keys are tied to the original platform for which they were purchased and are non-transferable.

Instant Access CPUs

If you require COD CPU resources before you complete the COD RTU license purchasing process, you can temporarily enable a limited number of resources called *instant access CPUs* (also referred to as *headroom*). The maximum number of instant access resources available on Product Names is eight CPUs.

Instant access CPUs are disabled by default on Sun Fire high-end systems. To use these resources, activate them by using the `setupplatform`(1M) command. Warning messages are logged on the platform console, informing you that the number of instant access CPUs (headroom) used exceeds the number of COD licenses available. Once you obtain and add the COD RTU license keys for instant access CPUs to the COD license database, these warning messages will stop.

For details on activating instant access CPUs, see, "To Obtain and Add a COD RTU License Key to the COD License Database" on page 145.

Instant Access CPUs as Hot Spares

You can temporarily enable an available, instant access CPU to replace a failed non-COD CPU. In this case, the instant access CPU is considered as a *hot spare* (a spare CPU that can be used immediately to replace a failed non-COD CPU). However, once the failed non-COD CPU has been replaced, you must deactivate the instant access CPU (see "To Enable Instant Access CPUs and Reserve Domain RTU Licenses" on page 150). Contact your Sun sales representative or reseller to purchase a COD RTU license for the instant access CPU in use if you want to continue using it.

Resource Monitoring

Information about COD events, such as the activation of instant access CPUs (headroom) or license violations, is recorded in the platform log and can be viewed by using the `showlogs` command.

Other commands, such as the `showcodusage`(1M) command, provide information on COD components and COD configuration. For details on obtaining COD information and status, see "Monitoring COD Resources" on page 152.

Getting Started With COD

Before you can use COD on Product Names, you must complete certain prerequisites. These tasks include:

- Installing the same version of the SMS software on both the main and spare system controller (SC).

  For details on upgrading the software, refer to *the System Management Services (SMS) 1.6 Installation Guide*.

Note – SMS software versions before SMS 1.3 will not recognize COD system boards.

- Contacting your Sun sales representative or reseller and doing the following:
  - Signing the COD contract addendum, in addition to the standard purchasing agreement contract for your Product Name.
  - Purchasing COD system boards and arranging for their installation.
- Performing the COD RTU licensing process as described in "To Obtain and Add a COD RTU License Key to the COD License Database" on page 145.

Managing COD RTU Licenses

COD RTU license management involves the acquisition and addition of COD RTU licenses keys to the COD license database. You can also remove COD RTU licenses from the license database if needed.

▼ To Obtain and Add a COD RTU License Key to the COD License Database

1. **Contact your Sun sales representative or authorized Sun reseller to purchase a COD RTU license for each COD CPU to be enabled.**

   Sun will send you a COD RTU License Certificate for each CPU license that you purchase. The COD RTU license sticker on the License Certificate contains a right-to-use serial number used to obtain a COD RTU license key.

2. **Contact the Sun License Center and provide the following information to obtain a COD RTU license key:**

   - The COD RTU serial number from the license sticker on the COD RTU License Certificate.
   - Chassis HostID, which uniquely identifies the platform.

You can obtain the Chassis HostID by running the command `showplatform -p cod` as platform administrator.

For instructions on contacting the Sun License Center, refer to the COD RTU License Certificate that you received or check the Sun License Center web site:

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

The Sun License Center will send you an email message containing the RTU license key for the COD resources that you purchased.

3. **Add the license key to the COD license database by using the** `addcodlicense` **(1M) command.**

   In an SC window, log in as a platform administrator and type:

   ```
   sc0:sms-user:> addcodlicense license-signature
   ```

   where *license-signature* is the complete COD RTU license key assigned by the Sun License Center. You can copy the license key string that you receive from the Sun License Center.

4. **Verify that the specified license key was added to the COD license database by running the** `showcodlicense -r` **command (see** "To Review COD License Information" on page 147**).**

   The COD RTU license key that you added should be listed in the `showcodlicense`(1M) command output.

▼ To Delete a COD License Key From the COD License Database

1. **In an SC window, log in as a platform administrator and type:**

   ```
   sc0:sms-user:> deletecodlicense license-signature
   ```

   where :

   license-signature is the complete COD RTU license key to be removed from the COD license database.

   The system verifies that the license removal will not cause a COD RTU license violation, which occurs when there is an insufficient number of COD licenses for the number of COD resources in use. If the deletion will cause a COD RTU license violation, the SC will not delete the license key.

Note – You can force the removal of the license key by specifying the `-f` option with the `deletecodlicense`(1M) command. However, be aware that the license key removal could cause a license violation or an overcommitment of RTU license reservations. An RTU license overcommitment occurs when there are more RTU domain reservations than RTU licenses installed in the system. For additional details, refer to the `deletecodlicense`(1M) command description in the *System Management Services (SMS) 1.6 Reference Manual*.

2. **Verify that the license key was deleted from the COD license database by running the `showcodlicense -r` command, described in the next procedure.**

   The deleted license key should not be listed in the `showcodlicense` output.

▼ To Review COD License Information

1. **In an SC window, log in as a platform administrator and type one of the following to display COD license information:**

   - To view license data in an interpreted format, type:

```
sc0:sms-user:> showcodlicense
```

For example:

```
sc0:sms-user:> showcodlicense

              Lic                                          Tier
 Description  Ver   Expiration  Count   Status  Cls  Num  Req
 -----------  ---  -----------  -----  -------  ---  ---  ---
 PROC          01         NONE     16     GOOD    1    1    0
```

TABLE 7-1 describes the COD license information in the `showcodlicense` output.

TABLE 7-1 COD License Information

Item	Description
Description	Type of resource (processor)
Lic Ver	Version number of the license
Expiration	None. Not supported (no expiration date)
Count	Number of RTU licenses granted for the given resource

TABLE 7-1 COD License Information *(Continued)*

Item	Description
Status	One of the following states: • GOOD – Indicates the resource license is valid • EXPIRED – Indicates the resource license is no longer valid
Cls	Not applicable
Tier Num	Not applicable
Req	Not applicable

- To view license data in raw license key format, type:

```
sc0:sms-user:> showcodlicense -r
```

The license key signatures for COD resources are displayed. For example:

```
sc0:sms-user:> showcodlicense -r
01:5014936C37048:45135285:0201000000:8:00000000:0000000000000000000000
```

Note – The COD RTU license key listed above is provided as an example and is not a valid license key.

For details on the showcodlicense(1M) command, refer to the command description in the *System Management Services (SMS) 1.6 Reference Manual*.

Activating COD Resources

To activate instant access CPUs and allocate COD RTU licenses to specific domains, use the setupplatform command. TABLE 7-2 describes the various setupplatform command options that can be used to configure COD resources.

TABLE 7-2 `setupplatform` Command Options for COD Resource Configuration

`setupplatform` **Command Options**	**Description**
`setupplatform -p cod`	Enable or disable instant access CPUs (headroom) and allocate domain COD RTU licenses
`setupplatform -p cod` *headroom-number*	Enable or disable instant access CPUs (headroom)
`setupplatform -p cod -d` *domainid RTU-number*	Reserve a specific quantity of COD RTU licenses for a particular domain

For details on the `setupplatform` command options, refer to the command description in the *System Management Services (SMS) 1.6 Reference Manual*.

▼ To Enable Instant Access CPUs and Reserve Domain RTU Licenses

1. **In an SC window, log in as a platform administrator and type:**

```
sc0:sms-user:> setupplatform -p cod
```

You are prompted to enter the COD parameters (headroom quantity and domain RTU information). For example:

```
sc0:sms-user:> setupplatform -p cod
PROC RTUs installed: 12
PROC Headroom Quantity (0 to disable, 8 MAX) [0]:0
PROC RTUs reserved for domain A (12 MAX) [0]: 4
PROC RTUs reserved for domain B (8 MAX) [2]: 4
PROC RTUs reserved for domain C (4 MAX) [0]: 0
PROC RTUs reserved for domain D (4 MAX) [0]:?
PROC RTUs reserved for domain E (4 MAX) [0]?
PROC RTUs reserved for domain G (4 MAX) [0]?
PROC RTUs reserved for domain H (4 MAX) [0]?
PROC RTUs reserved for domain I (4 MAX) [0]?
PROC RTUs reserved for domain J (4 MAX) [0]?
PROC RTUs reserved for domain K (4 MAX) [0]?
PROC RTUs reserved for domain L (4 MAX) [0]?
PROC RTUs reserved for domain M (4 MAX) [0]?
PROC RTUs reserved for domain N (4 MAX) [0]?
PROC RTUs reserved for domain O (4 MAX) [0]?
PROC RTUs reserved for domain P (4 MAX) [0]?
PROC RTUs reserved for domain Q (4 MAX) [0]?
PROC RTUs reserved for domain R (4 MAX) [0]?
```

Note the following about the prompts displayed:

- Instant access CPU (headroom) quantity

  The text in parentheses indicates the maximum number of instant access CPUs (headroom) allowed. The value inside the brackets is the number of instant access CPUs currently configured.

  To disable the instant access CPU (headroom) feature, `type 0`. You can disable the headroom quantity only when there are no instant access CPUs in use.

- Domain reservations

  The text in parentheses indicates the maximum number of RTU licenses that can be reserved for the domain. The value inside the brackets is the number of RTU licenses currently allocated to the domain.

2. **Verify the COD resource configuration by running the** `showplatform`**(1M) command:**

```
sc0:sms-user:> showplatform -p cod
```

For example:

```
sc0:sms-user:> showplatform -p cod

COD:
====
Chassis HostID : 5014936C37048
PROC RTUs installed: 8
PROC Headroom Quantity: 0
PROC RTUs reserved for domain A : 4
PROC RTUs reserved for domain B : 0
PROC RTUs reserved for domain C : 0
PROC RTUs reserved for domain D : 0
PROC RTUs reserved for domain E : 0
PROC RTUs reserved for domain F : 0
PROC RTUs reserved for domain G : 0
PROC RTUs reserved for domain H : 0
PROC RTUs reserved for domain I : 0
PROC RTUs reserved for domain J : 0
PROC RTUs reserved for domain K : 0
PROC RTUs reserved for domain L : 0
PROC RTUs reserved for domain M : 0
PROC RTUs reserved for domain N : 0
PROC RTUs reserved for domain O : 0
PROC RTUs reserved for domain P : 0
PROC RTUs reserved for domain Q : 0
PROC RTUs reserved for domain R : 0
```

Note – The chassis host ID is used for COD licensing purposes. If the `Chassis HostID` is listed as UNKNOWN, you must power on the centerplane support boards to obtain the Chassis host ID. In this case, allow up to one minute before rerunning the `showplatform` command to display the chassis host ID.

Monitoring COD Resources

This section describes various ways to track COD resource use and obtain COD information.

COD System Boards

You can determine which system boards in your system are COD boards by using the showboards(1M) command.

▼ To Identify COD System Boards

- **In an SC window, log in as platform administrator and type:**

```
sc0:sms-user:> showboards -v
```

The information displayed shows board assignments and test status. COD CPU boards are identified as CPU (COD).

For example:

```
sc0:sms-user:> showboards -v
Location    Pwr    Type of Board   Board Status  Test Status   Domain
--------    ---    -------------   ------------  -----------   ------
SC0         On     SC              Main              -             -
SC1         On     SC              Spare             -             -
PS0         On     PS                -               -             -
PS1         On     PS                -               -             -
.
.
.
SB0         Off    CPU             Available     Unknown       Isolated
SB1          -     Empty Slot      Available         -         Isolated
SB2         Off    CPU             Available     Unknown       Isolated
SB3          -     Empty Slot      Available         -         Isolated
SB4         On     CPU (COD)       Assigned      Unknown       A
SB5          -     Empty Slot      Available         -         Isolated
SB6         On     CPU (COD)       Active        Passed        B
SB7          -     Empty Slot      Available         -         Isolated
SB8          -     Empty Slot      Available         -         Isolated
SB9          -     Empty Slot      Available         -         Isolated
SB10         -     Empty Slot      Available         -         Isolated
SB11         -     Empty Slot      Available         -         Isolated
SB12        Off    CPU (COD)       Assigned      Unknown       C
.
.
.
```

COD Resource Usage

To obtain information on how COD resources are used in your system, use the showcodusage(1M) command.

▼ To View COD Usage By Resource

- **In an SC window, log in as a platform administrator and type:**

```
sc0:sms-user:> showcodusage -p resource
```

For example:

```
sc0:sms-user:> showcodusage -p resource
Resource:
=========
Resource     In Use  Installed  Licensed  Status
----------   ------  ---------  --------  ------
PROC              4         12        12  OK: 8 available
```

TABLE 7-3 describes the COD resource information displayed by the showcodusage(1M) command.

TABLE 7-3 showcodusage Resource Information

Item	Description
Resource	The COD resource (processor).
In Use	The number of COD CPUs currently used in the system.
Installed	The number of COD CPUs installed in the system.
Licensed	The number of COD RTU licenses installed.
Status	One of the following COD states: • OK – Indicates there are sufficient licenses for the COD CPUs in use and specifies the number of remaining COD resources available and the number of any instant access CPUs (headroom) available. • HEADROOM – The number of instant access CPUs in use. • VIOLATION – Indicates a license violation exists. Specifies the number of COD CPUs in use that exceeds the number of COD RTU licenses available. This situation can occur when you force the deletion of a COD license key from the COD license database, but the COD CPU associated with that license key is still in use.

▼ To View COD Usage by Domain

- **In an SC window, log in as a platform or domain administrator and type:**

```
sc0:sms-user:> showcodusage -p domains -v
```

The output includes the status of CPUs for all domains. For example:

```
sc0:sms-user:> showcodusage -p domains -v
Domains:
========
Domain/Resource  In Use  Installed  Reserved  Status
---------------  ------  ---------  --------  ------
A - PROC              0          4         4
     SB4 - PROC       0          4
         SB4/P0                               Unused
         SB4/P1                               Unused
         SB4/P2                               Unused
         SB4/P3                               Unused
B - PROC              4          4         4
     SB6 - PROC       4          4
         SB6/P0                               Licensed
         SB6/P1                               Licensed
         SB6/P2                               Licensed
         SB6/P3                               Licensed
C - PROC              0          4         0
    SB12 - PROC       0          4
        SB12/P0                               Unused
        SB12/P1                               Unused
        SB12/P2                               Unused
        SB12/P3                               Unused
.
.
.
```

TABLE 7-4 describes the COD resource information displayed by domain.

TABLE 7-4 `showcodusage` Domain Information

Item	Description
Domain/Resource	The COD resource (processor) for each domain. An unused processor is a COD CPU that has not yet been assigned to a domain.
In Use	The number of COD CPUs currently used in the domain.
Installed	The number of COD CPUs installed in the domain.
Reserved	The number of COD RTU licenses allocated to the domain.
Status	One of the following CPU states: • Licensed – The COD CPU has a COD RTU license. • Unused – The COD CPU is not in use. • Unlicensed – The COD CPU could not obtain a COD RTU license and is not in use.

To View COD Usage by Resource and Domain

- **In an SC window, log in as a platform administrator and type:**

```
sc0:sms-user:> showcodusage -v
```

The information displayed contains usage information by both resource and domain.

For example:

```
sc0:sms-user:> showcodusage -v
Resource:
=========
Resource  In Use  Installed  Licensed  Status
--------  ------  ---------  --------  ------
PROC           4          4        16  OK: 12 available
Domains:
========
Domain/Resource  In Use  Installed  Reserved  Status
---------------  ------  ---------  --------  ------
A - PROC              0          0         0
B - PROC              0          0         0
     SB6 - PROC       0          0
         SB6/P0                               Unused
         SB6/P1                               Unused
         SB6/P2                               Unused
         SB6/P3                               Unused
C - PROC              0          0         0
    SB12 - PROC       0          0
        SB12/P0                               Unused
        SB12/P1                               Unused
        SB12/P2                               Unused
        SB12/P3                               Unused
D - PROC              4          4         0
     SB4 - PROC       4          4
         SB4/P0                               Licensed
         SB4/P1                               Licensed
         SB4/P2                               Licensed
         SB4/P3                               Licensed
    SB16 - PROC       4          4
        SB16/P0                               Unused
        SB16/P1                               Unused
        SB16/P2                               Unused
        SB16/P3                               Unused
E - PROC              0          0         0
F - PROC              0          0         0
G - PROC              0          0         0
.
.
.
R - PROC              0          0         0
Unused - PROC         0          0        12
```

Deconfigured and Unlicensed COD CPUs

When you activate a domain that uses COD system boards, any COD CPUs that cannot obtain a COD RTU license are identified as deconfigured or unlicensed. You can determine which COD CPUs are deconfigured or unlicensed by reviewing the following items:

- Message output for a `setkeyswitch on` operation

  Any COD CPUs that did not acquire a COD RTU license are identified as deconfigured. If all the COD CPUs on a COD system board are deconfigured, the `setkeyswitch on` operation fails the COD system board, and the `setkeyswitch on` operation also fails, as the next example shows:

```
sc0:sms-user:> setkeyswitch -d A on
.
.
.
Acquiring licenses for all good processors...
Proc SB03/P0    deconfigured: no license available.
Proc SB03/P2    deconfigured: no license available.
Proc SB03/P3    deconfigured: no license available.
Proc SB03/P1    deconfigured: no license available.
No minimum system left after Check CPU licenses (for COD)! Bailing out!
.
.
.
Deconfigure Slot0: 00008
Deconfigure EXB:   00008
POST (level=16, verbose=40, -H3.0) execution time 3:08
# SMI Sun Fire 15K POST log closed Fri Jul 26 15:15:53 2002
```

- `showcodusage`(1M) command output

  To obtain the status of COD CPUs for a domain, see "To View COD Usage by Domain" on page 154. The Unlicensed status indicates that a COD RTU license could not be obtained for the COD CPU and that the CPU is not being used by the domain.

Other COD Information

TABLE 7-5 summarizes the COD configuration and event information that you can obtain through other system controller commands. For further details on these commands, refer to their descriptions in the *System Management Services (SMS) 1.6 Reference Manual*.

TABLE 7-5 Obtaining COD Component, Configuration, and Event Information

Command	Information Displayed
`showlogs`	Information about COD events, such as license violations or headroom activation, that are logged on the platform console
`showplatform -p cod`	Current COD resource configuration: • Number of instant access CPUs (headroom) in use • Domain RTU license reservations Chassis host ID

CHAPTER 8

Domain Control

This chapter addresses the functions that provide control over domain software and server hardware. Control functions are invoked at the discretion of an administrator. They are also useful to SMS for providing automatic system recovery (ASR).

Domain control functionality provides control over the software running on a domain. It includes those functions that enable a domain to be booted and interrupted. Only the domain administrator can invoke the domain control functions.

This chapter includes the following sections:

- "Booting Domains" on page 161
- "Hardware Control" on page 167

Booting Domains

This section describes the various aspects of booting the Solaris OS in a domain.

The `setkeyswitch`(1M) command is responsible for initiating and sequencing a domain boot. It powers on the domain hardware as required and invokes a POST to test and configure the hardware in the logical domain into a Sun Fire high-end system's physical hardware domain. It downloads and initiates the OpenBoot PROM as required to boot the Solaris OS on the domain.

Only domains that have their virtual keyswitch set appropriately are subject to boot control. See "Virtual Keyswitch" on page 111.

OpenBoot PROM boot parameters are stored in the domain's virtual NVRAM. The `osd`(1M) command provides those parameter values to OpenBoot PROM, which adapts the domain boot as indicated.

Certain parameters, in particular those that might not be adjustable from OpenBoot PROM itself when a domain is failing to boot, can be set by setobpparams(1M) so that they take effect at the next boot attempt.

Keyswitch Control

The domain keyswitch control (see "Virtual Keyswitch" on page 111) manually initiates domain boot.

The setkeyswitch command boots a properly configured domain when its keyswitch control is moved from the off or standby position to one of the on positions.

The setobpparams(1M) command provides a method by which a manually initiated (keyswitch control) domain boot sequence can be stopped in the OpenBoot PROM. For more information, see "Setting the OpenBoot PROM Variables" on page 115 and refer to the setobpparams man page.

Power Control

Power for the following components can be controlled using the poweron and poweroff commands.

- Fan tray
- Centerplane support board
- Expander board
- System board
- Standard PCI board
- Hot-pluggable PCI and PCI+ boards
- MaxCPU board
- wPCI board
- System controller (spare only; poweroff or resetsc can be used to power on the spare)

▼ To Power System Boards On and Off From the Command Line

Platform administrators are allowed to control power to the entire system and can execute these commands without a *location* option. Domain administrators can control power to any system board assigned to their domains. Users with only domain privileges must supply the *location* option.

- To power on a system component, type:

```
sc0:sms-user:> poweron  location
```

where *location* is the location of the system component you want to power on and, if you are a domain administrator, for which you have privileges.

For more information, refer to the `poweron`(1M) man page.

- To power off a system component, type:

```
sc0:sms-user:> poweroff  location
```

where *location* is the location of the system component you want to power off and, if you are a domain administrator, for which you have privileges.

Enter `y` or `n` after the warning message:

```
!!!WARNING!!!WARNING!!!WARNING!!!WARNING!!!WARNING!!!
!!!WARNING!!!WARNING!!!WARNING!!!WARNING!!!WARNING!!!

This will trip the breakers on PS at PS5, which must be turned on
manually!

Are you sure you want to continue to power off this component?
(yes/no)? y
```

Caution – Remove a component from the domain using DR before powering it down. Powering off the component without first removing it from the domains causes a domain stop (dstop). If you are powering off a component to replace it, use the `poweroff`(1M) command. Do not use the breakers to power off the component before it has been removed from the domain; this can also cause a dstop. After the component has been removed from the domain, using the breakers to power it down does not cause a dstop.

For more information, refer to the `poweroff`(1M) man page.

If you try to power off the system while any domain is actively running the OS, the command fails and displays a message in the message panel of the window. In that case, issuing a `setkeyswitch` *domain-id* `standby` command for the active domains gracefully shuts down the processors. Once they have shut down, you can reissue the command to power off.

If the platform loses power due to a power outage, `pcd` records and saves the last state of each domain before power was lost.

▼ To Recover From Power Failure

If you lose power to only the SC, switch on the power to the SC. Sun Fire high-end system domains are not affected by the loss of power to one SC. If you lose power to both the SC and the domains, use the following procedure to recover from the power failure. For switch locations, refer to the *Sun Fire 15K/12K System Site Planning Guide*.

Caution – Losing power to both SCs without shutting down SMS crashes the domains.

---

1. **Manually switch off the bulk power supplies on the Sun Fire high-end system as well as the power switch on the SC.**

   This prevents power surge problems that can occur when power is restored.

2. **After power is restored, manually switch on the bulk power supplies on the Sun Fire high-end system.**

3. **Manually switch on the SC power.**

   This boots the SC and starts the SMS daemons. Check your SC platform message file for completion of the SMS daemons.

   Wait for the recovery process to complete. Any domain that was powered on and running the Solaris OS returns to the OS run state. Domains at OpenBoot PROM eventually return to an OpenBoot PROM run state.

   The recovery process must finish before any SMS operation is performed. You can monitor the domain message files to determine when the recovery process has completed.

Domain-Requested Reboot

SMS reboots domains upon request from the domain management software (Solaris software or `dsmd`). The domain software requests reboot services in the following situations.

- Upon execution of a user reboot request–for example, Solaris `reboot`(1M) or the OpenBoot PROM boot command, `reset-all`.
- Upon Solaris software panic.
- Upon trapping the CPU-detected RED_mode or Watchdog Reset conditions.

Automatic System Recovery (ASR)

Automatic system recovery (ASR) consists of those procedures that restore the system to running all properly configured domains after one or more domains have been rendered inactive due to software or hardware failures or due to unacceptable environmental conditions.

SMS software supports a software-initiated reboot request as part of ASR. Every domain that crashed is automatically rebooted by `dsmd`.

Situations that require ASR are domain boots requested by domain software upon detecting failures that crash the domain (for example, panic).

There are other situations, such as detection of domain software hangs as described in "Solaris Software Hang Events" on page 208, where SMS initiates a domain boot as part of the recovery process.

The `dsmd` software ignores the OpenBoot PROM parameter, `auto-boot?`, which on systems without a service processor can prevent the system from automatically rebooting in power-on-reset situations. `dsmd` does *not* ignore keyswitch control. If the keyswitch is set to `off` or `standby`, the keyswitch setting is honored when determining whether a domain is subject to ASR reboot actions.

Domain Reboot

In general, a fast domain reboot is possible in situations where:

- No serious error has been attributed to hardware since the last boot.
- No failures have occurred that would cause SMS to question the reliability of the existing set of domain resources.

Because SMS is responsible for monitoring the hardware and detecting and responding to errors, SMS decides whether or not to request a fast reboot based upon its record of hardware errors since the last boot.

Because POST controls the hardware configuration based upon a number of inputs including, but not limited to, the blacklist data (see "Blacklist Editing" on page 168), POST decides whether or not the hardware configuration has changed so as to preclude a fast reboot. If system management has requested a fast reboot, POST verifies that the hardware configuration implied by its current inputs matches the hardware configuration used for the last boot; if it does not, POST fails the fast-POST operation. The system management software is prepared to recover from this type of POST failure by requesting a full-test (slow) domain boot.

Sun Fire high-end system management software minimizes the elapsed time taken by the part of the domain boot process that it can control.

Domain Abort or Reset

Certain error conditions can occur in a domain that require aborting the domain software or issuing a reset to the domain software or hardware. This section describes the domain abort and reset functions that are provided by dsmd.

The dsmd software provides a software-initiated mechanism to abort a domain Solaris OS, requesting that it panic to take a core image. No user intervention is needed.

SMS provides the reset(1M) command to enable the user to abort the domain software and issue a reset to the domain hardware.

Control is passed to the OpenBoot PROM after the reset command is issued. In the case of a user-interface-issued reset command, the OpenBoot PROM uses its default configuration to determine whether the domain is booted to the Solaris environment. In the case of a dsmd-issued reset command, the OpenBoot PROM provides parameters that force the domain to be booted to the Solaris OS.

The reset command normally sends a signal to all CPU ports of a specified domain. This is a hard reset and clears the hardware to a clean state. Using the -x option, however, reset can send an XIR signal to the processors in a specified domain. This is done in software and is considered a soft reset. An error message is given if the virtual key switch is in the secure position. An optional Are you sure? prompt is given by default. For example:

```
sc0:sms-user:> reset -d C
Do you want to send RESET to domain C? [y|n]:y
RESET to processor 4.1.0 initiated.
RESET to processor 4.1.1 initiated.
RESET initiated to all processors for domain: C
```

For more information, refer to the reset man page.

For information on resetting the main or spare SC see “SC Reset and Reboot” on page 176.

SMS software illuminates or darkens the indicator LEDs on LED-equipped hot-pluggable units (HPUs) as necessary to reflect the correct state when the HPU is given a power-on reset.

Hardware Control

Hardware control functions are those that configure and control the platform hardware. Some functions are invoked on the domain.

Power-On Self-Test (POST)

System Management Services software invokes POST in two contexts:

1. At domain boot time, POST is invoked to test and configure all functional hardware available to the domain.

   POST eliminates all hardware components that fail the self-test and attempts to build a bootable domain from the functionally working hardware.

   POST provides extensive diagnostics to help analyze failures. You can request that POST only verify a domain configuration, and not test it, in situations where the domain is being rebooted with no indications that a hardware failure was the cause.

2. Before a DR operation to add a system board to a domain, POST is invoked to test and configure the system board components.

   If POST indicates that the candidate system board is functional, the DR operation can safely incorporate the system board into the physical (hardware) domain.

Although POST is generally invoked automatically, there are user-visible interfaces that affect automatic POST invocations:

- You can add or remove components that you want POST to exclude from the hardware configuration by using blacklist files. These editable files are described in "Blacklist Editing" on page 168.

  This gives you finer-grained control over the hardware components that are used in a domain than is allowed by the standard domain configuration interfaces that operate on DCUs, such as system boards.

- The `setkeyswitch` command invokes POST to test and configure a domain. Nominal and maximum diagnostic test level settings are provided for use in booting the domain.
- The `addboard` and `moveboard` commands invoke POST to test and configure a system board in support of a DR operation to add that board to a running Solaris domain.
- LED-equipped FRUs with components that fail POST have the fault LED illuminated on the FRU.

Blacklist Editing

SMS supports three blacklists: one for the platform, one for the domains, and the internal automatic system recovery (ASR) blacklist.

Platform and Domain Blacklisting

The editable blacklist files specify that certain hardware resources are to be considered unusable by POST. They will not be probed for, tested, or configured in the domain interconnect.

Usually these blacklist files are empty and are not required to be present.

Blacklist capability in this context is used for resource management purposes.

Blacklisting temporarily limits the system configuration to less than all the hardware present. This has several applications, such as benchmarking, limiting memory use to make DR detach of the board faster, and varying the configuration for troubleshooting.

Sun Fire high-end system POST supports two editable canonical blacklist files, one for the platform and one for the domain, located in these two files:

`/etc/opt/SUNWSMS/config/platform/blacklist`

`/etc/opt/SUNWSMS/config/`*domain-id*`/blacklist`

The two files are considered logically concatenated.

Note – The blacklist file specifies resources based on physical location. If the component is physically moved, any corresponding blacklist entries must be changed accordingly.

The blacklist file specifies blacklisted components logically–for example, by specifying their position – and the blacklist remains on the component position through a hot-plug operation, rather than following a specific component.

▼ To Blacklist a Component

1. **Log in to the SC.**

   You must have platform administrator, domain administrator, or configurator privileges to edit the blacklist files.

2. Type the following command:

```
sc0:sms-user:> disablecomponent [-d domain-indicator] location
```

where:

`-d` *domain-indicator* Specifies the domain using one of the following:
domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.
domain-tag – Name assigned to a domain using `addtag`(1M).

location List of component locations comprising:

board-loc/proc/bank/logical-bank

board-loc/proc/bank/all-dimms-on-that-bank

board-loc/proc/bank/all-banks-on-that-proc

board-loc/proc/bank/all-banks-on-that-board

board-loc/proc

board-loc/cassette

board-loc/bus

board-loc/paroli-link

If no *domain-indicator* is specified, the platform blacklist is edited. All component locations are separated by forward slashes. The *location* forms are optional and are used to specify particular components on boards in specific locations.

Multiple *location* arguments are permitted, separated by a space.

TABLE 8-1 Valid *location* Arguments for Sun Fire High-End Servers

Location	Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K

board-loc	SB(0...17)
	IO(0...17)
	CS(0 | 1)
	EX(0...17)
		SB(0...8)
		IO(0...8)
		CS(0 | 1)
		EX(0...8)

Processor/Processor Pair (*proc*)	P(0...3)
	PP(0 | 1)
		P(0...3)
		PP(0 | 1)

bank	B	B
logical-bank	L(0 | 1)	L(0 | 1)
all-dimms-on-that-bank	D	D
all-banks-on-that-proc	B	B
all-banks-on-that-board	B	B
HsPCI cassette	C(3 | 5)V(0 | 1)	C(3 | 5)V(0 | 1)
HsPCI+ cassette	C3V(0 | 1 | 2) and C5V0	C3V(0 | 1 | 2) and C5V0
bus	ABUS | DBUS | RBUS (0 | 1)	ABUS | DBUS | RBUS (0 | 1)
paroli-link	PAR(0 | 1)	PAR(0 | 1)

Processor locations indicate single processors or processor pairs. There are four possible processors on a system board. Processor pairs on that board are procs 0 and 1, and procs 2 and 3.

Note – If you blacklist a single CPU/memprocessor in a processor pair, neither processor is used.

The MaxCPU has two processors, procs 0 and 1, and only one proc pair (PP0). `disablecomponent` exits and displays an error message if you use PP1 as a location for this board.

The HsPCI and HsPCI+ assemblies contain hot-pluggable cassettes.

There are three bus locations: address, data, and response.

Note – Do not use the `disablecomponents` command to disable centerplane support boards or a bus on the system controller.

To Remove a Component From the Blacklist

1. **Log in to the SC.**

2. Type the following command:

```
sc0:sms-user:> enablecomponent [-d domain-indicator] location
```

where:

`-d` *domain-indicator* Specifies the domain using one of the following:
domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.
domain-tag – Name assigned to a domain using `addtag`(1M).

location List of component locations consisting of:

board-loc/proc/bank/logical-bank,

board-loc/proc/bank/all-dimms-on-that-bank

board-loc/proc/bank/all-banks-on-that-proc

board-loc/proc/bank/all-banks-on-that-board

board-loc/proc

board-loc/cassette

board-loc/bus

board-loc/paroli-link

If no *domain-indicator* is specified, the platform blacklist is edited. All component locations are separated by forward slashes. The *location* forms are optional and are used to specify particular components on boards in specific locations.

Multiple *location* arguments are permitted, separated by a space.

TABLE 8-2 Valid *location* Arguments for Sun Fire High-End Servers

Location	Valid Form for Sun Fire 15K/E25K	Valid Form for Sun Fire 12K/E20K

board-loc	SB(0...17)
	IO(0...17)
	CS(0 | 1)
	EX(0...17)
		SB(0...8)
		IO(0...8)
		CS(0 | 1)
		EX(0...8)

Processor/processor pair (*proc*)	P(0...3)
	PP(0 | 1)
		P(0...3)
		PP(0 | 1)

bank	B	B
logical-bank	L(0 | 1)	L(0 | 1)
all-dimms-on-that-bank	D	D
all-banks-on-that-proc	B	B
all-banks-on-that-board	B	B
HsPCI cassette	C(3 | 5)V(0 | 1)	C(3 | 5)V(0 | 1)
HsPCI+ cassette	C3V(0 | 1 | 2) and C5V0	C3V(0 | 1 | 2) and C5V0
bus	ABUS | DBUS | RBUS (0 | 1)	ABUS | DBUS | RBUS (0 | 1)
paroli-link	PAR(0 | 1)	PAR(0 | 1)

Processor locations indicate single processors or processor pairs. There are four possible processors on a CPU/Mem board. Processor pairs on that board are: procs 0 and 1, and procs 2 and 3.

Note – If you blacklist a single CPU or memory processor in a processor pair, neither processor is used.

The MaxCPU has two processors, procs 0 and 1, and only one proc pair (PP0). The `disable component` command exits and displays an error message if you use PP1 as a location for this board.

The HsPCI and HsPCI+ assemblies contain hot-pluggable cassettes.

There are three bus locations: address, data and response.

For more information, refer to the `enablecomponent`(1M) and `disablecomponent`(1M) man pages.

ASR Blacklist

Hardware that has failed repeatedly, perhaps intermittently, must be excluded from subsequent domain configurations for many reasons. It might be some time before the component can be physically replaced. The failed component might be a subcomponent such as one processor on a CPU board. You do not want to lose the services of the rest of the component by powering it down until it can be replaced. If the hardware is broken, you do not want to waste time having POST discover that every time it runs. If the failure is intermittent, you do not want POST to pass it, only to have it fail when the OS is running.

To this end, esmd creates and edits a separate ASR blacklist file. Components that have been powered off due to environmental conditions are automatically listed and excluded from POST. The poweron, setkeyswitch, addboard, and moveboard commands query the ASR blacklist for components to exclude. Each of these commands except poweron displays a warning message. poweron instead asks whether you would like to continue or abort powering on the component. For more information, refer to the enablecomponent(1M), disablecomponent(1M,) and showcomponent(1M) man pages.

Power Control

The main SC has power control over the following components in the Sun Fire high-end system rack:

- Sun Fire high-end system boards
- HsPCI adapter slots on the Sun Fire high-end system HsPCI I/O board
- HsPCI+ adapter slots on the Sun Fire high-end system HsPCI+ I/O board
- System controllers (power off only)
- Centerplane support boards
- wPCI boards
- Expander boards
- 48V power supplies
- AC bulk power modules
- Fan trays

See "HPU LEDs" on page 176 for a description of power control in the Sun Fire high-end system I/O racks.

SMS supports the domain Solaris command interface (cfgadm(1M)) by providing the rcfgadm(1M) command to request power on or off of the HPCI adapter slots in a Sun Fire high-end system HsPCI I/O board. For more information, refer to the rcfgadm man page.

The keyswitch control interface setkeyswitch, as described in "Virtual Keyswitch" on page 111, enables the user to power on or off the hardware assigned to a domain.

All power operations are logged by the power control software.

The power control software conforms to all hardware requirements for powering on or off components. For example, SMS checks for adequate power available before powering on components. The power control interfaces will not perform a user-specified power on or power off operation if it violates a hardware requirement. Power operations that are performed contrary to hardware requirements or hardware suggested procedures are noted in the message logs.

By default, the power control software refuses to perform power operations that will affect running software. The power control user interfaces include methods to override this default behavior and forcibly complete the power operation at the cost of crashing running software. The use of these forcible overrides on power operations are noted in the message logs.

As described in "HPU LEDs" on page 176, SMS illuminates or darkens the indicator LEDs on LED-equipped HPUs, as necessary, to reflect the correct state when the HPU is powered on or off.

Fan Control

The `esmd` command provides the fan speed control for Sun Fire high-end system fans. In general, fan speeds are set to the lowest speed that provides adequate cooling, so as to minimize noise levels.

Hot-Plug Operations

Hot-plug refers to the ability to physically insert or remove a board from a powered-on platform that is actively running one or more domains without affecting those domains. During a hot-plug operation, the board is isolated from all domains.

The term for a hardware component that can be hot-plugged is hot-pluggable unit (HPU). The OK to Remove indicator LED on an HPU is illuminated when it can be safely unplugged; see "HPU LEDs" on page 176 for more information about the OK to Remove LEDs. Board presence registers indicate whether an HPU is present or absent and sense an HPU plug or unplug.

The Sun Fire high-end system HsPCI and HsPCI+ I/O assemblies are equipped with OK to Remove indicator LEDs associated with the slots into which HsPCI and HsPCI+ I/O assemblies are plugged. Each slot is equipped with a hot-plug controller that controls power to the slot and can detect presence of an adapter in the slot. However, unlike SMS support for other Sun Fire high-end system HPUs, the software that controls hot-plug for the HsPCI and HsPCI+ I/O assemblies is part of the Solaris OS on the domain.

SMS enables you to power on and off the adapter slots.

SMS software provides software interfaces, invocable from the domain, to control hardware devices associated with the adapter slots on I/O boards.

Note – For the purposes of the remaining hot-plug discussion in this section, HPUs do not include hot-pluggable I/O adapters.

SMS software provides support as necessary to enable hot-plug servicing of all HPUs in the Sun Fire high-end system rack.

Once an HPU is isolated from all domains, the only software support required for a hot-plug operation is power-off control.

Dynamic reconfiguration (DR) isolates DCUs (system boards) from a domain by DR detaching the DCU.

Unplugging

When an HPU is unplugged, the presence indicator for the HPU detects its absence, resulting in a change in hardware configuration status as described in "Hardware Configuration" on page 194.

The expected mode of user interaction during hot-unplug is as follows:

Go directly to the HPU you want to unplug.

If the HPU indicator LEDs show that it is *not* OK to Remove, request that the HPU be powered off using the `poweroff` command.

If the power-off function discovers that the HPU is in use by a domain, the power-off function fails, indicating that you first must use DR to remove the HPU from active use.

Refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide* for more information.

Plugging

The presence of a newly inserted HPU is detected and reported as a change in hardware configuration status, as described in "Hardware Configuration" on page 194.

SC Reset and Reboot

The SC supports software-initiated resets for the main and spare, providing the same functionality as external reset buttons on the system controller. Typically, an SC might be reset after failover. It is possible for the main SC software to reset the spare SC, if present, and vice versa. An SC cannot reset itself.

▼ To Reset the Main or Spare SC

The `resetsc`(1M) command sends a reset signal to the other SC. If the other SC is not present, `resetsc` exits with an error.

- **Type the following command:**

```
sc0:sms-user:> resetsc
"About to reset other SC. Are you sure you want to continue?" (y
or [n])? y
```

For more information, refer to the `resetsc` man page.

HPU LEDs

The LEDs reflect the status of the hot-pluggable units (HPUs). LEDs come in groups of three:

- The operating indicator LED is illuminated when power is on.
- The OK to Remove LED is illuminated when an HPU can be unplugged.
- The fault LED is illuminated when a hardware fault has been discovered in an HPU.

This section describes the LED control policies that are followed by SMS software for the HPUs.

Except for the system controllers, all Sun Fire high-end system HPUs are powered on and tested under control of the SMS software that runs on the main system controller.

To a certain extent, the design of the LEDs, especially their initial state upon power-on-reset, is based upon the assumption that POST is automatically initiated at power-on-reset. The only Sun Fire high-end system HPUs that meet this assumption are the system controllers. Powering on a system controller causes the processor to begin executing SC-POST code from PROM.

For all other HPUs, some are tested by POST and some are tested (or monitored) by SMS software. Although it is generally the case that testing follows shortly after power on, it is not always so.

Furthermore, it is possible that POST can be run multiple times on a power-on HPU that is being dynamically reconfigured from one domain to another. It is also possible that POST and SMS can both detect faults on the same physical HPU. These differences in power and test control between the system controllers and other Sun Fire high-end system HPUs result in different policies proposed to manage them.

The system controller provides three sets of HPU LEDs that indicate:

- The state of the SC as a whole
- The state of the CP1500 or CP2140 slot
- The state of the SC spare slot

When the Sun Fire high-end system rack is powered on, power is supplied to the system controllers. The operating indicator LED and the OK to Remove indicator LEDs are, appropriately, initialized by the hardware. All three fault LEDs are illuminated so that the fault LEDs correctly reflect a fault, should there be a problem that prevents SC-POST from running.

SMS software, upon powering off the spare system controller, extinguishes the operating indicator LED and illuminates the OK to Remove indicator LEDs on the spare system controller. SMS software cannot adjust the operating indicator or OK to Remove LEDs after powering off the main SC, where the software is running.

SC-POST does the following:

- Upon completing testing the SC with no faults found, SC-POST extinguishes the SC fault indicator LED.
- Upon completing testing the HPCI slot with no faults found, SC-POST extinguishes the SC spare slot fault LED.
- Upon completing testing the control board with no faults found at the control board, the SC main, or the SC spare slot, SC-POST extinguishes the SC fault LED.

SC-OpenBoot PROM firmware and SMS software illuminate the proper fault LEDs on the system controller after detecting a hardware error.

The following policies are used to manage LEDs on HPUs other than the system controllers.

- On every LED-equipped non-SC HPU within the Sun Fire high-end system rack, SMS assures that the operating indicator LED is steadily illuminated when power is applied to the HPU.
- On every LED-equipped non-SC HPU within the Sun Fire high-end system, SMS assures that the OK to Remove indicator LED is steadily illuminated only when the HPU can be safely unplugged. Safety considerations apply both to the person unplugging the HPU and to preserving the correct and continuing operation of Sun Fire high-end system hardware and any running software.

Note – The Sun Fire high-end system correctly illuminates the operating indicator LED and correctly darkens the OK to Remove indicator LEDs when HPUs are powered on or given a power-on-reset.

- The management of the fault LEDs and their user-visible behavior differs most between the SC and non-SC HPUs.

  On the SC, the fault LEDs are illuminated at power on, maintained on during testing, and then extinguished if no fault is found.

  Faults detected after SC-POST can cause later fault LED illumination.

  Except for the brief period when the SC is being tested by POST, the fault LEDs on the SC indicate that a fault has occurred since power on. The same is true (an illuminated fault LED indicates that a fault has been detected since power on) for non-SC HPUs. For every non-SC HPU that has LEDs within the Sun Fire high-end system, SMS ensures that the fault indicator LED is extinguished when a power on or power on reset occurs.
- When directed to do so by POST (see "Power-On Self-Test (POST)" on page 167), or the hardware monitoring software (see "Environmental Events" on page 210, "Hardware Error Events" on page 213, and "SC Failure Events" on page 215), SMS steadily illuminates the fault LED on an HPU. The fault indicator remains illuminated until the next power on or power-on-reset clears it, as described in "HPU LEDs" on page 176.

CHAPTER 9

Domain Services

Sun Fire high-end system hardware incorporates internal, private point-to-point Ethernet connections between the SC and each domain. This network, called the Management Network (MAN), is used to provide support services for each domain. This chapter describes those services.

This chapter includes the following sections:

- "Management Network Overview" on page 179
- "Management Network Services" on page 184

Management Network Overview

The Management Network (MAN) function maintains the private point-to-point network connections between the SC and each domain. No packets addressed to one domain can be routed along the network connection between the SC and another domain (FIGURE 9-1).

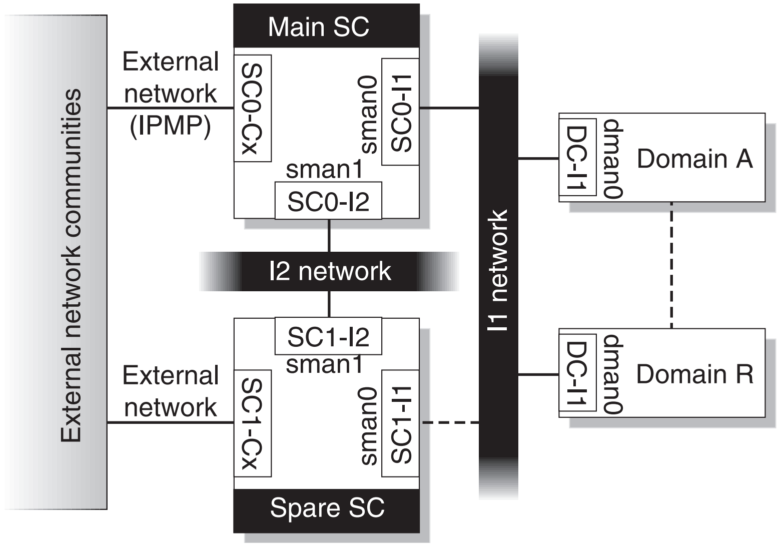


FIGURE 9-1 Management Network Overview

I1 Network

The hardware built into the Sun Fire high-end system chassis to support MAN is complex. It includes 18 Network Interface Cards (NICs) on each SC that are connected in a point-to-point fashion to NICs located on each of the 18 expander I/O slots on the Sun Fire 15K system and on each of the 9 expander I/O slots on the Sun Fire 12K system. Using this design, the number of point-to-point Ethernet links between an SC and a given DSD varies based on the number of I/O boards configured in that DSD. Each NIC from the SC connects to a hub and NIC on the I/O board. The NIC is an internal part of the I/O board and not a separate adapter card. Likewise, the Ethernet hub is on the I/O board. The hub is intelligent and can collect statistics.

All of these point-to-point links are collectively called the I1 network. Since there can be multiple I/O boards in a given domain, multiple redundant network connections from the SC to a domain are possible. FIGURE 9-2 shows a network overview of the Sun Fire E25K/15K.

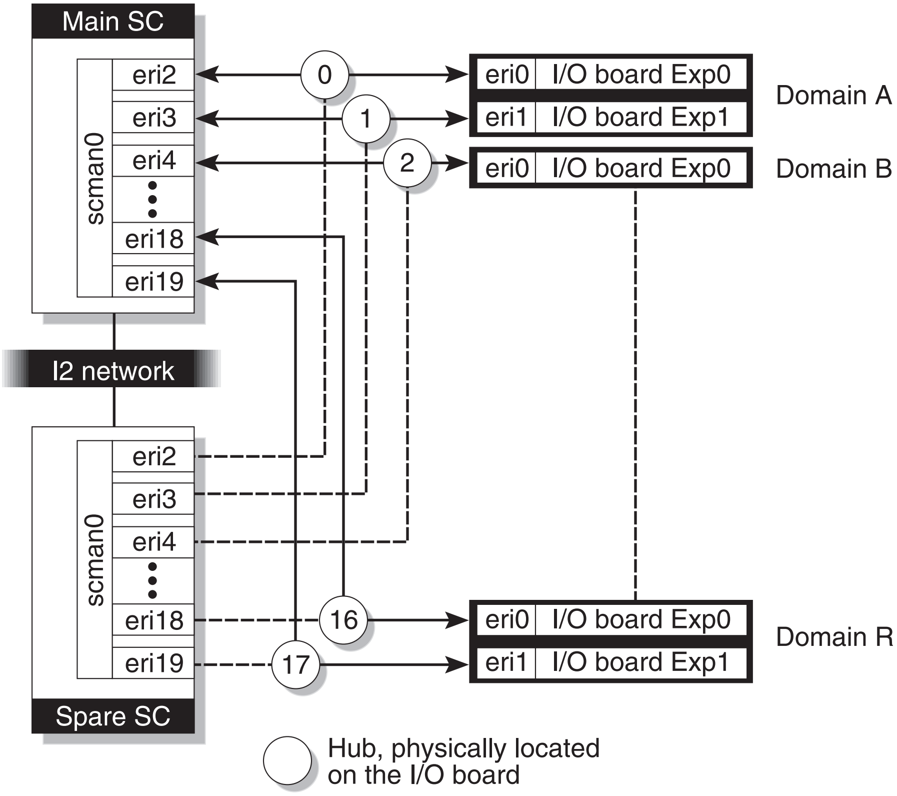


FIGURE 9-2 I1 Network Overview of the Sun Fire E25K/15K

Note – The I1 MAN network is a *private* network, not a general-purpose network. No external IP traffic should be routed across it. Access to MAN is restricted to the system controller and the domains.

On the SC, MAN software creates a meta-interface for the I1 network, presenting to the Solaris OS a single network interface, `scman0`. For more information, refer to the Solaris `scman`(1M) man page.

MAN software detects communication errors and automatically initiates a path switch, provided an alternate path is available. MAN software also enforces domain isolation of network traffic on the I1 network. Similar software operates on the domain side.

I2 Network

There is also an internal network between the two system controllers consisting of two NICs per system controller. This network is called the I2 network. It is a private SC-to-SC network and is entirely separate from the I1 network.

MAN software creates a meta-interface for the I2 network as well. This interface is presented to the Solaris software as `scman1`. As with the I1 network, I2 has a mechanism for detecting path failure and switching paths, providing an alternative is available.

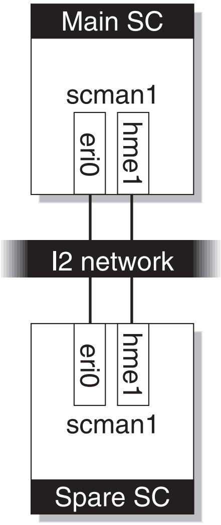


FIGURE 9-3 I2 Network Overview

The virtual network adapter on the SC presents itself as a standard network adapter. It can be managed and administered just like any other network adapter (for example, `qfe`, `hme`). The usual system administration tools such as `ndd`(1M), `netstat`(1M), and `ifconfig`(1M), can be used to manage the virtual network adapter. Certain operations of these tools (for example, changing the Ethernet address) should not be allowed for security reasons.

MAN operates and is managed as an IP network with special characteristics (for example, IP forwarding is disallowed by the MAN software). As such, the MAN operation is the same as any other IP network, with the previously noted exception. Domains can be connected to your network depending on your site configuration and security requirements. Connecting domains is not within the scope of this document–refer to the *System Administration Guide: Resource Management and Network Services.*

External Network Monitoring

External Network Monitoring for the Sun Fire high-end system provides highly available network connections from the SCs to customer networks called communities. This feature is built on top of the IP Network Multipathing (IPMP) framework provided in the Solaris 9 OS. For more information on IPMP, refer to the *System Administration Guide: IP Services.*

External networks can consist of communities. You can have zero, one, or two communities. Zero communities means external networks are not monitored. During installation, user communities are connected by physical cable to the RJ45 jacks on the SC connecting a node to the network.

For more information on connecting external networks, refer to the *Sun Fire 15K/12K System Site Planning Guide*. FIGURE 9-4 shows an external network overview.

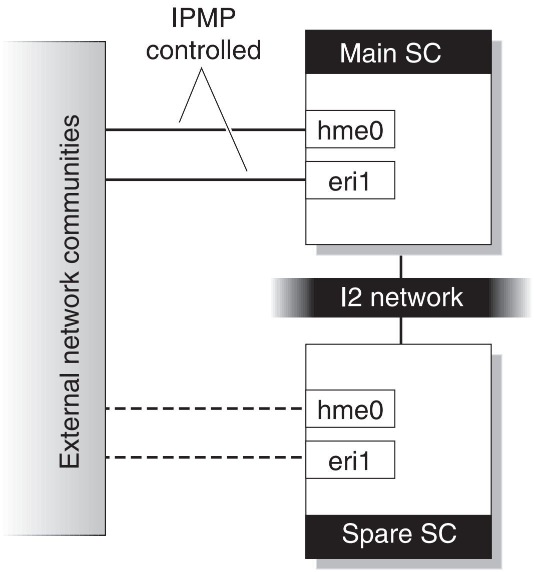


FIGURE 9-4 External Network Overview

The term *community* refers to an IP network at your site. For example, you might have an engineering community and an accounting community. A *community name* is used as the *interface group name*. An *interface group* is a group of network interfaces that attach to the same community.

Configuring External Network Monitoring requires allocating several additional IP addresses for each system controller.

The addresses can be categorized as follows:

- Test addressees – These IP addresses are assigned to the external network interfaces on each system controller. Each IP test address is used to test the health of the particular network interface to which it is assigned. One IP test address is permanently assigned to each network interface. They are permanently associated with a particular network interface. If a network interface fails, the IP test address associated with that network interface becomes unreachable.
- Failover addresses – There are two types of failover addresses:
  - SC path group specific addresses – These IP addresses are assigned to a particular interface group on each system controller. They are used to provide highly available IP connectivity to a particular system controller for a given community. The SC path group specific address is reachable as long as at least one of the network interfaces in the interface group is functioning.

Note – An SC path group-specific address is not needed if there is only one network interface in an interface group. Since there is no other network interface in the group to failover to, only the test addresses and the community failover addresses are required.

  - Community failover addresses – These IP addresses are assigned to a particular community on the MAIN SC (that is, Community C1). They are used to provide IP connectivity to the MAIN SC, either SC0 or SC1.

  All external software should reference the community failover address when communicating with the SC. This address always connects to the main SC. That way, if a failover occurs, external clients do not need to alter their configuration to reach the SC. For more information on SC failover, see Chapter 12.

MAN Daemons and Drivers

For more information on the MAN daemon and device drivers, refer to the SMS `mand`(1M) and Solaris `scman`(1M) and `dman`(1M) man pages. See also "Management Network Daemon" on page 68.

Management Network Services

The primary network services that MAN provides between the SC and the domains are:

- Domain consoles
- Message logging

- Dynamic reconfiguration (DR)
- Network boot/Solaris installation
- System controller (SC) heartbeats

Domain Console

The software running in a domain (OpenBoot PROM, `kadb`, and Solaris software) uses the system console for critical communications.

The domain console supports a login session and is secure, since the default configuration of the Solaris environment allows only the console to accept `superuser` logins. Domain console access is provided securely to remote administrators over a possibly public network.

The behavior of the console reflects the health of the software running in the domain. Character echo for user entries is nearly equivalent to that of a 9600-baud serial terminal attached to the domain. Output characters that are not echoes of user input are typically either the output from an executed command or from a command interpreter, or they might be unsolicited log messages from the Solaris software. Activity on other domains or SMS support activity for the domain do not noticeably alter the response latency of user entry echo.

You can run `kadb` on the domain's Solaris software from the domain console. Interactions with the OpenBoot PROM running on a domain use the domain console. The console can serve as the destination for log messages from the Solaris software; refer to `syslog.conf`(4). The console is available when software (Solaris, OpenBoot PROM, `kadb`) is running on the domain.

You can open multiple connections to view the domain console output. However, the default is an exclusive *locked* connection.

For more information, see "SMS Console Window" on page 11.

A domain administrator can forcibly break the domain console connection held by another domain.

You can forcibly break into the OpenBoot PROM or `kadb` from the domain console; however, it is not suggested. (This is a replacement for the physical L1-A or STOP-A key sequence available on a Sun SPARC® system with a physical console.) SMS captures console output history for subsequent analysis of domain crashes. A log of the console output for every domain is available in `/var/opt/SUNWSMS/adm/`*domain-id*`/console`.

The Sun Fire high-end system provides the hardware to either implement a shared-memory console or implement an alternate network data path for console. The hardware utilized for a shared-memory console imposes less direct latency upon console data transfers, but is also used for other monitoring and control purposes for all domains, so there is a risk of latency introduced by contention for the hardware resources.

MAN provides private network paths to securely transfer domain console traffic to the SC; see "Management Network Services" on page 184. The console has a dual-pathed nature so that at least one path provides acceptable console response latency when the Solaris software is running. The dual-pathed console is robust in the face of errors. It detects failures on one domain console path and fails over to the other domain console path automatically. It supports user-directed selection of the domain console path to use.

The `smsconfig`(1M) command is the SC configuration utility that initially configures or later modifies the hostname, IP address, and netmask settings used by management network daemon, `mand`(1M). See "Management Network Daemon" on page 68.

The `mand` daemon initializes and updates these respective fields in the platform configuration database (pcd).

The `mand` daemon is automatically started by `ssd`. The Management Network daemon runs on the main SC in *main* mode and on the spare SC in *spare* mode.

For more information, refer to the SMS `console`(1M), `mand`(1M), and `smsconfig`(1M) man pages as well as the Solaris `dman`(1M) and `scman`(1M) man pages.

Message Logging

When configured to do so, MAN transports copies of important syslog messages from the domains to disk storage on the SC. This facilitates failure analysis for crashed or unbootable domains. For more information, see "Log File Maintenance" on page 200.

Dynamic Reconfiguration

The MAN software layer is used to simplify the interface to the MAN hardware. MAN software handles the aspects of dynamic reconfiguration (DR) used by a DSD without requiring network configuration work by the domain or platform administrator.

Software in the domains using MAN need not be aware of which SC is currently the main SC. For more information on dynamic reconfiguration, refer to the *System Management Services (SMS) 1.6 Dynamic Reconfiguration User Guide*.

Network Boot and Solaris Software Installation

The SC provides network Solaris boot services to each domain.

Note – Diskless Sun Fire high-end system domains cannot be supported entirely by network services from the SC; the SC network boot service is intended primarily for recovery after a catastrophic disk failure on the domain.

When Solaris software is first installed on a domain, the network interface connecting it to the MAN is automatically created for subsequent system reboots. There are no additional tasks required by the domain administrator to configure or use MAN.

MAN is configured as a private network. A default address assignment for the Management Network is provided, using the IP address space reserved for private networks. You can override the default address assignment for MAN to handle the case where the Sun Fire high-end system is connected to a private customer network that already uses the selected MAN default IP address range.

The SC supports simultaneous network boots of domains running at least two different versions of Solaris software.

The SC provides software installation services to no more than one domain at a time.

SC Heartbeats

The I2 network supplies the intersystem controller communication. This is also called the heartbeat network. SMS failover mechanisms on the main SC use this network as one means of determining the health of the spare SC. For more information, see Chapter 12. For a description of the I2 network, see “I2 Network” on page 182.

CHAPTER 10

Domain Status Functions

Status functions return measured values that characterize the state of the server hardware or software. As such, these functions are used to provide both values for status displays and input to monitoring software that periodically polls status functions and verifies that the values returned are within normal operational limits. Monitoring and event detection functions that use the status functions are described in this chapter.

This chapter contains the following sections:

Software Status

The software state consists of status information provided by the software running in a domain. The identity of the software component currently running (for example, POST, OpenBoot PROM, or Solaris software) is available. Additional status information is available (booting, running, panicking).

SMS software provides the following commands to display the status of the software, if any, currently running in a domain:

- `showboards`
- `showdevices`
- `showenvironment`
- `showobpparams`
- `showpcimode`
- `showplatform`
- `showxirstate`

Status Commands

This section describes the SMS domain status commands.

showboards Command

The showboards(1M) command displays the assignment information and status of the DCU, including: Location, Power, Type of board, Board status, Test status, and Domain.

If no options are specified, showboards displays all DCUs, including those that are assigned or available for the platform administrator. For the domain administrator or configurator, showboards displays only DCUs for domains for which the user has privileges, including those boards that are assigned or available and in the domain's available component list.

If *domain-indicator* is specified, this command displays which DCUs are assigned or available to the given domain. If the -v option is used, showboards displays all boards, including DCUs.

For examples and more information, see "To Obtain Board Status" on page 93 and refer to the showboards man page.

showdevices Command

The showdevices(1M) command displays configured physical devices on system boards and the resources made available by these devices. Usage information is provided by applications and subsystems that are actively managing system resources. The predicted impact of a system board DR operation can be optionally displayed by performing an offline query of managed resources.

The showdevices command gathers device information from one or more Sun Fire high-end system domains. The command uses the dca(1M) as a proxy to gather the information from the domains.

For examples and more information, see "To Obtain Board Status" on page 93 and refer to the showdevices man page.

showenvironment Command

The showenvironment(1M) command displays environmental data including. Location, Sensor, Value, Unit, Age, Status. For fan trays, Power, Speed, and Fan Number are displayed. For bulk power, the Power, Value, Unit, and Status are shown.

If *domain-indicator* is specified, environmental data relating to the domain is displayed, providing that the user has domain privileges for that domain. If a domain is not specified, all domain data permissible to the user is displayed.

DCUs (for example, CPU or I/O) belong to a domain and you must have domain privileges to view their status. Environmental data relating to such things as fan trays, bulk power, or other boards are displayed without domain permissions. You can also specify individual reports for temperatures, voltages, currents, faults, bulk power status, and fan tray status with the -p option. If the -p option is not present, all reports are shown.

For examples and more information, see "Environmental Status" on page 195 and refer to the showenvironment man page.

showobpparams Command

The showobpparams(1M) command displays OpenBoot PROM bringup parameters. The showobpparams command enables a domain administrator to display the virtual NVRAM and REBOOT parameters passed to OpenBoot PROM by setkeyswitch(1M).

For examples and more information, see "Setting the OpenBoot PROM Variables" on page 115 and refer to the showobpparams man page.

showpcimode Command

The showpcimode(1m) command lists the mode settings for all the PCI-X slots on a V2HPCIX I/O board in your server. The settings are specified by the setpcimode command. A slot that returns a status of normal is running in PCI-X mode. A slot that returns a status of pci_only has been forced to run in PCI mode.

If you specify an I/O board that is not a V2HPCIX board, the command returns an error.

showplatform Command

The showplatform(1M) command displays the available component list and domain state of each domain.

A domain is identified by a *domain-tag* if one exists. Otherwise, it is identified by the *domain-id*, a letter in the set A–R. The letter set is case insensitive. The Solaris *hostname* is displayed if one exists. If a *hostname* has not been assigned to a domain, Unknown is printed.

TABLE 10-1 lists domain statuses.

TABLE 10-1 Domain Status Types

Status	Description
Unknown	The domain state could not be determined. For Ethernet addresses, the domain idprom image file does not exist. Contact your Sun service representative.
Powered Off	The domain is powered off.
Keyswitch Standby	The keyswitch for the domain is in STANDBY position.
Running Domain POST	The domain power-on self-test is running.
Loading OBP	The OpenBoot PROM for the domain is being loaded.
Booting OBP	The OpenBoot PROM for the domain is booting.
Running OBP	The OpenBoot PROM for the domain is running.
In OBP Callback	The domain has been halted and has returned to the OpenBoot PROM.
Loading Solaris	The OpenBoot PROM is loading the Solaris software.
Booting Solaris	The domain is booting the Solaris software.
Domain Exited OBP	The domain OpenBoot PROM exited.
OBP Failed	The domain OpenBoot PROM failed.
OBP in sync Callback to OS	The OpenBoot PROM is in sync callback to the Solaris software.
Exited OBP	The OpenBoot PROM has exited.
In OBP Error Reset	The domain is in OpenBoot PROM due to an error reset condition.
Solaris Halted in OBP	Solaris software is halted and the domain is in OpenBoot PROM.
OBP Debugging	The OpenBoot PROM is being used as a debugger.
Environmental Domain Halt	The domain was shut down due to an environmental emergency.
Booting Solaris Failed	OpenBoot PROM is running, boot attempt failed.
Loading Solaris Failed	OpenBoot PROM is running, loading attempt failed.
Running Solaris	Solaris software is running on the domain.
Solaris Quiesce In-Progress	A Solaris software quiesce is in progress.

TABLE 10-1 Domain Status Types *(Continued)*

Status	Description
Solaris Quiesced	Solaris software has quiesced.
Solaris Resume In-Progress	A Solaris software resume is in progress.
Solaris Panic	Solaris software has panicked, panic flow has started.
Solaris Panic Debug	Solaris software panicked, and is entering debugger mode.
Solaris Panic Continue	Exited debugger mode and continuing panic flow.
Solaris Panic Dump	Panic dump has started.
Solaris Halt	Solaris software is halted.
Solaris Panic Exit	Solaris software exited as a result of a panic.
Environmental Emergency	An environmental emergency has been detected.
Debugging Solaris	Debugging Solaris software; this is not a hung condition.
Solaris Exited	Solaris software has exited.
Domain Down	The domain is down and the setkeyswitch is in the `ON`, `DIAG`, or `SECURE` position.
In Recovery	The domain is in the midst of an automatic system recovery.

TABLE 10-2 Domain Status Types

Domain status reflects two cases. The first is that `dsmd` is busy trying to recover the domain and the second is that `dsmd` has given up trying to recover the domain. In the second case you always see “Domain Down.” In the first case you see either “Domain Down” or some other status. To recover from a “Domain Down” in *either* case, use `setkeyswitch off`, `setkeyswitch on`.

```
sc0:sms-user:> setkeyswitch off
sc0:sms-user:> setkeyswitch on
```

For examples and more information, see “To Obtain Domain Status” on page 94 and refer to the `showplatform` man page.

showxirstate Command

The showxirstate(1M) command displays CPU dump information after a reset pulse is sent to the processors. This save state dump can be used to analyze the cause of abnormal domain behavior. showxirstate creates a list of all active processors in that domain and retrieves the save state information for each processor, including its processor signature.

The showxirstate command data resides, by default, in /var/opt/SUNWSMS/adm/*domain-id*/dump.

For examples and more information, refer to the showxirstate man page.

Solaris Software Heartbeat

During normal operation, the Solaris environment produces a periodic heartbeat indicator readable from the SC. The dsmd daemon detects the absence of heartbeat updates for a running Solaris system as a hung Solaris. Hangs are not detected for any software components other than the Solaris software.

Note – The Solaris software heartbeat should not be confused with the SC-to-SC (hardware) heartbeat or the heartbeat network, both used to determine the health of failover. For more information, see "SC Heartbeats" on page 187.

The only reflection of the Solaris heartbeat occurs when dsmd detects a failure to update the Solaris heartbeat of sufficient duration to indicate that the Solaris software is hung. Upon detection of a Solaris software hang, dsmd conducts an ASR.

Hardware Status

The hardware status functions report information about the hardware configuration, hardware failures detected, and platform environmental state.

Hardware Configuration

The following hardware configuration status is available from the Sun Fire high-end system management software:

- Hardware components physically present on each board (as detected by POST)

- Hardware components not in use because they failed POST
- Presence or absence of all HPUs (for example, system boards)
- Hardware components not in use because they were on the blacklist when POST was invoked (see "Power-On Self-Test (POST)" on page 167)
- Contents of the SEEPROM for each FRU, including the part number and serial number

---

Note – The hardware configuration status available to SMS running on the SC is limited to presence or absence. It does not include information about the I/O configuration, such as where I/O adapters are plugged in and what devices are attached to those I/O adapters. Such information is available only to the software running on the domain that owns the I/O adapter.

The hardware configuration supported by functions described in this section excludes I/O adapters and I/O devices. The `showboards` command displays all hardware components that are present.

As described in "Blacklist Editing" on page 168, the current contents of the component blacklists can always be viewed and altered.

Environmental Status

The following hardware environmental measurements are available:

- Temperatures
- Power voltage and amperage
- Fan status (stopped, low-speed, high-speed, failed)
- Power status
- Faults

The `showenvironment` command displays every environmental measurement that can be taken within the Sun Fire high-end system rack.

▼ To Display the Environment Status for Domain A

1. **Log in to the SC.**

   Platform administrators can view any environment status on the entire platform. Domain administrators can see the environment status only for those domains for which they have privileges.

2. **Type the following command:**

```
sc0:sms-user:> showenvironment -d A
```

As described in "HPU LEDs" on page 176, the operating indicator LEDs on Sun Fire high-end system HPUs visibly reflect that the HPUs are powered on and the OK to remove LEDs visibly reflect those that can be unplugged.

Hardware Error Status

The dsmd daemon monitors the Sun Fire high-end system hardware operational status and reports errors. Occurrences of some errors are directly reported to the SC (for example, the error registers in every ASIC propagate to the SBBC on the SC that provides an error summary register). Although the occurrence of some errors is indicated by an interrupt delivered to the SC, some error states might require the SC to monitor hardware registers for error indications. When a hardware error is detected, esmd follows the established procedures for collecting and clearing the hardware error state.

The following types of errors can occur on Sun Fire high-end system hardware:

- Domain stops, fatal hardware errors that terminate all hardware operations in a domain
- Record stops that cause the hardware to stop collecting transaction history when a data transfer error (for example, CE ECC) occurs
- SPARC processor error conditions such as RED-state/watchdog reset
- Nonfatal ASIC-detected hardware failures

Hardware error status is generally not reported as a status. Rather, event-handling functions perform various actions when hardware errors occur such as logging errors, initiating ASR, and so forth. These functions are discussed in Chapter 11.

Note – As described in "HPU LEDs" on page 176, the fault LEDs, after POST completion, identify Sun Fire high-end system HPUs in which faults have been discovered since last powered on or submitted to a power-on reset.

SC Hardware and Software Status

Proper operation of SMS depends upon proper operation of the hardware and the Solaris software on the SC. The ability to support automatic failover from the main to the spare system controller requires properly functioning hardware and software on the spare. SMS software running on the main system controller must either be functioning sufficiently to diagnose a software or hardware failure in a manner that can be detected by the spare, or it must fail in a manner that can be detected by the spare.

Note – For failover to be supported, both SCs must be configured with identical versions of the Solaris OS and SMS software.

SC-POST determines the status of system controller hardware. It tests and configures the system controller at power-on or power-on-reset.

The SC does not boot if the SC fails to function.

If the control board fails to function, the SC boots normally, but without access to the control board devices. The level of hardware functionality required to boot the system controller is essentially the same as that required for a standalone SC.

SC-POST writes diagnostic output to the SC console serial port (TTY-A). Additionally, SC-POST leaves a brief diagnostics status summary message in an NVRAM buffer that can be read by a Solaris driver and logged or displayed when the Solaris software boots.

SC firmware and software display information to identify and service SC hardware failures.

SC firmware and software provide a software interface that verifies that the system controller hardware is functional. This selects a working system controller as the main SC in a high-availability SC configuration.

The system controller LEDs provide visible status regarding power and detected hardware faults, as described in "HPU LEDs" on page 176.

Solaris software provides a level of self-diagnosis and automatic recovery (panic and reboot). Solaris software utilizes the SC hardware watchdog logic to trap hang conditions and force an automatic recovery reboot.

Four hardware paths of communication between the SCs (two Ethernet connections, the heartbeat network, and one SC-to-SC heartbeat signal) are used in the high-availability SC configuration by each SC to detect hangs or failures on the other SC.

SMS practices self-diagnosis and institutes automatic failure recovery procedures, even in non-high-availability SC configurations.

Upon recovery, SMS software either takes corrective actions as necessary to restore the platform hardware to a known, functional configuration or reports the inability to do so.

SMS software records and logs sufficient information to enable engineering diagnosis of single-occurrence software failures in the field.

SMS software takes a noticeable interval to initialize itself and become fully functional. The user interfaces behave predictably during this interval. Any rejections of user commands are clearly identified as due to system initialization, with advice to try again after a suitable interval.

SMS software implementation uses a distributed client-server architecture. Any errors encountered during SMS initialization due to attempts to interact with a process that has not yet completed initialization are dealt with silently.

CHAPTER 11

Domain Events

Event monitoring periodically checks the domain and hardware status to detect conditions that require an action. The action taken is determined by the condition and can involve reporting the condition or initiating automated procedures to deal with it. This chapter describes the events that are detected by monitoring and the requirements with respect to actions taken in response to detected events.

This chapter includes the following sections:

Message Logging

SMS logs all significant actions other than logging or updating user monitoring displays taken in response to an event. Log messages for significant domain software events and their response actions are written to the message log file for the affected domain located in `/var/opt/SUNWSMS/adm/`*domain-id*/`messages`. Included in the log is information to support subsequent servicing of the hardware or software.

SMS writes log messages for significant hardware events to the platform log file located in `/var/opt/SUNWSMS/adm/platform/messages`. SMS writes log messages to `/var/opt/SUNWSMS/adm/`*domain-id*`/messages` for significant hardware events that can visibly affect one or more domains of the affected domains.

The actions taken in response to events that crash domain software systems include automatic system recovery (ASR) reboots of all affected domains, provided that the domain hardware (or a bootable subset thereof) meets the requirements for safe and correct operation.

SMS also logs domain console, `syslog`, event, post, and dump information and manages `sms_core` files.

Log File Maintenance

SMS software maintains SC-resident copies of all server information that it logs. Use the `showlogs`(1M) command to access log information.

The platform message log file can be accessed only by administrators for the platform, using the following command:

```
sc0:sms-user:> showlogs
```

SMS log information relevant to a configured domain can be accessed only by administrators for that domain. SMS maintains separate log files for each domain. To access the files, type the following command:

```
sc0:sms-user:> showlogs -d domain-indicator
```

where:

`-d` *domain-indicator*   Specifies the domain using:

domain-id – ID for a domain. Valid *domain-id*s are A–R and are not case sensitive.

domain-tag – Name assigned to a domain using `addtag`(1M).

SMS maintains copies of domain `syslog` files on the SC in `/var/opt/SUNWSMS/adm/`*domain-id*`/syslog`.The `syslog` information can be accessed only by administrators for that domain.

To access the information, type the following command:

```
sc0:sms-user:> showlogs -d domain-indicator -p s
```

Solaris console output logs are maintained to provide valuable insight into what happened before a domain crashed. Console output is available on the SC for a crashed domain in `/var/opt/SUNWSMS/adm/`*`domain-id`*`/console`. `console` information can be accessed only by administrators for that domain.

To access the information, type the following command:

```
sc0:sms-user:> showlogs -d domain-indicator -p c
```

XIR state dumps, generated by the `reset` command, can be displayed using `showxirstate`. For more information, refer to the `showxirstate` man page.

Domain post logs are for service diagnostic purposes and are not displayed by `showlogs` or any SMS CLI.

The `/var/tmp/sms_core.`*`daemon`* files are binaries and not viewable.

The availability of various log files on the SC supports analysis and correction of problems that prevent a domain or domains from booting. For more information, refer to the `showlogs` man page.

Note – Panic dumps for panicked domains are available in the `/var/crash` logs on the domain, not on the SC.

TABLE 11-1 lists the SMS log information types and their descriptions.

TABLE 11-1 SMS Log Type Information

Type	Description
Firmware versioning	Unsuitable configuration of firmware version at firmware invocation is automatically corrected and logged.
Power-on self test	LED fault; platform and domain messages detailing why a fault LED was illuminated.
Power control	All power operations are logged.
Power control	Power operations that violate hardware requirements or hardware suggested procedures.
Power control	Use of override to forcibly complete a power operation.
Domain console	Automatic logging of console output to a standard file.
Hardware configuration	Part numbers are used to identify board type in message logs.

TABLE 11-1 SMS Log Type Information *(Continued)*

Type	Description
Fault and error event monitoring and actions	List of all fault events or error reports written to the event log.
Event monitoring and actions	All significant environmental events (those that require taking action).
Event monitoring and actions	All significant actions taken in response to environmental events.
Domain event monitoring and actions	All significant domain software events and their response actions.
Event monitoring and actions	Significant hardware events written to the platform log.
Event monitoring and actions	All significant clock input failures, clock input switch failures, and loss or gain of phase lock.
Domain event monitoring and actions	Significant hardware events that visibly affect one or more domains are written to the domain logs.
Domain boot initiation	Initiation of each boot and the passage through each significant stage of booting a domain is written to the domain log.
Domain boot failure	Boot failures are logged to the domain log.
Domain boot failures	All ASR recovery attempts are logged to the domain log.
Domain panic	Domain panics are logged to the domain log.
Domain panic	All ASR recovery attempts are logged to the domain log.
Domain panic hang	Each occurrence of a domain hang and its accompanying information is logged to the domain log.
Domain panic	All ASR recovery attempts after a domain panic and hang are logged to the domain log.
Repeated domain panic	All ASR recovery attempts after repeated domain panics are logged to the domain message log.
Solaris OS hang events	All OS hang events are logged to the domain message log.
Solaris OS hang events	All OS hang events result in a domain panic in order to obtain a core image for analysis of the Solaris hang. This information and subsequent recovery action is logged to the domain message log.
Solaris OS hang events	SMS monitors for the inability of the domain software to satisfy the request to panic. Upon determining noncompliance with the panic request, SMS aborts the domain and initiates an ASR reboot. All subsequent recovery action is logged to the domain message file.

TABLE 11-1 SMS Log Type Information *(Continued)*

Type	Description
Hot-plug events	All HPU insertion events of system boards to a domain are logged in the domain message log.
Hot-unplug events	All HPU removals are logged to the platform message log.
Hot-unplug events	All HPU removals from a domain are logged to the domain message log.
POST-initiated configuration events	All POST-initiated hardware configuration changes are logged in `/var/opt/SUNWSMS/adm/`*domain-id*`/post`.
Environmental events	All sensor measurements outside of acceptable operational limits are logged as environmental events to the platform log file.
Environmental events	All environmental events that affect one or more domains are logged to the domain message log.
Environmental events	Significant actions taken in response to environmental events are logged to the platform message log.
Environmental events	Significant actions taken in response to environmental events within a domain are logged to the domain message log.
Hardware error events	Hardware error and related information is logged to the platform message log.
Hardware error events	Hardware error and related information within a domain is logged to the domain message file.
Hardware error events	Log entries about hardware error for which data was collected include the name of the data files.
Hardware error events	All significant actions taken in response to hardware error events are logged to the platform message log.
Hardware error events	All significant actions taken in response to hardware error events affecting a domains are logged to the domains message log.
SC failure events	All SC hardware failure and related information is logged to the platform message log.
SC failure events	The occurrence of an SC failover event is logged to the platform message log.

Log File Management

SMS manages the log files, as necessary, to keep the SC disk utilization within acceptable limits.

The message log daemon (`mld`) monitors message log size, file count *per directory*, and age every 10 minutes. The `mld` daemon executes when it reaches the first limit. TABLE 11-2 lists the MLD default settings.

TABLE 11-2 MLD Default Settings

	File Size (in Kb)	File Count	Days to Keep
SMI event log	2500	10	0
Platform messages	2500	10	0
Domain messages	2500	10	0
Domain console	2500	10	0
Domain `syslog`	2500	10	0
Domain post	20000*	1000	0
Domain dump	20000*	1000	0
sms-core.*daemon*	100000	20	0

* total per directory, not per file

Assuming 20 directories, the defaults represent approximately 4 Gbytes of stored logs.

Caution – The parameters shown in TABLE 11-2 are stored in the file `/etc/opt/SUNWSMS/config/mld_tuning`. For any changes to take effect, `mld` must be stopped and restarted. Only an administrator experienced with system disk utilization should edit this file. Improperly changing the parameters in this file could flood the disk and hang or crash the SC.

- When a log message file reaches the size limit, `mld` does the following:

  Starting with the oldest message file *x.X*, it moves that file to *x.X*+1, except when the oldest message file is `message.9` or core file is `sms_core.daemon.1`; then it starts with *x.X*-1.

  For example, `messages` becomes `messages.0`, `messages.0` becomes `messages.1` and so on up to `messages.9`. When `messages` reaches 2.5 Mbytes, then `messages.9` is deleted, all files are bumped up by one and a new empty `messages` file is created.

- When a log file reaches the file count limit, `mld` does the following:

  When `messages` or `sms_core.`*daemon* reaches its count limit, then the oldest message or core file is deleted.

- When a log file reaches the age limit, `mld` does the following:

  When any message file reaches *x* days, it is deleted.

Note – By default, the age limit (*_log_keep_days) is set to zero and not used.

- When a post*date.time.sec*.log or a *dump-name*.*date.time.sec* file reaches the file size, count, or age limit, mld deletes the oldest file in the directory.

Note – Post files are provided for service diagnostic purposes and not intended for display.

For more information, refer to the mld and showlogs man pages, and see "Message Logging Daemon" on page 69.

Domain Reboot Events

SMS monitors domain software status (see "Software Status" on page 189) to detect domain reboot events.

Domain Reboot Initiation

Since the domain software is incapable of rebooting itself, SMS software controls the initial sequence for all domain reboots. As a result, SMS is always aware of domain reboot initiation events.

SMS software logs the initiation of each reboot and the passage through each significant stage of booting a domain to the domain-specific log file.

Domain Boot Failure

SMS software detects all domain reboot failures.

Upon detecting a domain reboot failure, SMS logs the reboot failure event to the domain-specific message log.

SC resident per-domain log files are available for failure analysis. In addition to the reboot failure logs, SMS can maintain duplicates of important domain-resident logs and transcripts of domain console output, as described in "Log File Maintenance" on page 200.

Domain reboot failures are handled as follows:

- The response to reboot or reset requests is always a fast bringup procedure.
- The first attempt to recover a domain from software failure uses a quick reboot procedure.
- The first attempt to recover a domain from hardware failure uses the reboot procedure. The POST default diagnostic level is used in the reboot procedure.
- If the domain recovery fails during the POST run, dsmd retries POST at the default diagnostic level for up to six consecutive domain recovery failures after the first recovery attempt fails.
- If the domain recovery fails during the IOSRAM layout, OpenBoot PROM download and jump, OpenBoot PROM run, or Solaris software boot, dsmd reruns POST at the default diagnostic level. For subsequent failures of this type, for each recovery dsmd runs POST at a test diagnostic level higher than the previous level. The dsmd daemon retries domain recovery domain at the default level for up to six attempts after the first recovery attempt fails. All in all, dsmd tries domain recovery attempts at most seven times.
- Once the system has been recovered and Solaris software has been booted, any domain failure within four hours is treated as a repeated domain failure and is recovered by running POST at a higher diagnostic level.
- If there are no domain failures within four hours of Solaris software running, then the domain is considered successfully recovered and healthy.

  A subsequent domain hardware failure is handled by the reboot procedure.

  A subsequent domain software failure is handled by a quick reboot procedure, and the reboot or reset request is handled by the fast bringup procedure.

SMS tries all ASR methods at its disposal to boot a domain that has failed booting. All recovery attempts are logged in the domain-specific message log.

Domain Panic Events

When a domain panics, it informs dsmd so that a recovery reboot can be initiated. The panic is reported as a domain software status change (see "Software Status" on page 189).

Domain Panic

The dsmd daemon is informed when the Solaris software on a domain panics.

Upon detecting a domain panic, dsmd logs the panic event to the domain-specific message log.

SC resident per-domain log files are available to assist in domain panic analysis. In addition to the panic logs, SMS can maintain duplicates of important domain-resident logs and transcripts of domain console output, as described in "Log File Maintenance" on page 200.

In general, after an initial panic where there has been no prior indication of hardware errors, SMS requests that a fast reboot be tried to bring up the domain. For more information, see "Domain Reboot" on page 165.

The dsmd daemon handles a panic event as follows:

- If the domain recovery fails during the POST run, the dsmd daemon retries POST at the default diagnostic level for up to six consecutive domain recovery failures after the first recovery attempt fails.
- If the domain recovery fails during the IOSRAM layout, OpenBoot PROM download and jump, OpenBoot PROM run, or Solaris software boot, the dsmd daemon reruns POST at the default diagnostic level. For subsequent failures of this type, for each recovery dsmd runs POST at a test diagnostic level higher than the previous level. The dsmd daemon retries domain recovery at the default level for up to six attempts after the first recovery attempt fails. (dsmd makes a maximum of seven domain recovery attempts.)
- Once the system has been recovered and Solaris software has been booted, any domain failure within four hours is treated as a repeated domain failure and is recovered by running POST at a higher diagnostic level.
- If there are no domain failures within four hours of Solaris software startup the domain is considered successfully recovered and healthy.

  A subsequent domain hardware failure is handled by the reboot procedure.

  A subsequent domain software failure is handled by a quick reboot procedure, and the reboot or reset request is handled by the fast bringup procedure.

This recovery action is logged in the domain-specific message log.

Domain Panic Hang

The Solaris panic dump logic has been redesigned to minimize the possibility of hangs at panic time. In a panic situation, Solaris software might operate differently, either because normal functions are shut down or because it is disabled by the panic. An ASR reboot of a panicked Solaris domain is eventually started, even if the panicked domain hangs before it can request a reboot.

Since the normal heartbeat monitoring (see "Solaris Software Hang Events" on page 208) of a panicked domain might not be appropriate or sufficient to detect situations where a panicked Solaris domain does not proceed to request an ASR reboot, dsmd takes special measures as necessary to detect a domain panic hang event.

Upon detecting a panic hang event, dsmd logs each occurrence, including event information, to the domain-specific message log.

Upon detection of a domain panic hang (if any), SMS aborts the domain panic (see "Domain Abort or Reset" on page 166) and initiates an ASR reboot of the domain. dsmd logs these recovery actions in the domain-specific message log.

SC-resident log files are available to assist in panic hang analysis. In addition to the panic hang event logs, the dsmd daemon maintains duplicates of important domain-resident logs and transcripts of domain console output on the SC, as described in "Log File Maintenance" on page 200.

Repeated Domain Panic

If a second domain panic is detected shortly after recovering from a panic event, dsmd classifies the domain panic as a repeated domain panic event.

In addition to the standard logging actions that occur for any panic, the following actions are taken when attempting to reboot after the repeated domain panic event:

- With each successive repeated domain panic event, SMS attempts to run POST at a higher diagnostic test level to boot against the next untried administrator-specified degraded configuration (see "Degraded Configuration Preferences" on page 119).
- After all degraded configurations have been tried, successive repeated domain panic events continue full-test-level boots using the last specified degraded configuration.
- Upon determining that a repeated domain panic event has occurred, dsmd tries the ASR method at its disposal to boot a stable domain software environment. The dsmd daemon logs all recovery attempts in the domain-specific message log.

Solaris Software Hang Events

The dsmd daemon monitors the Solaris heartbeat described in "Solaris Software Heartbeat" on page 194 in each domain while Solaris software is running (see "Software Status" on page 189). When the heartbeat indicator is not updated for a period of time, a Solaris software hang event occurs.

The dsmd daemon detects Solaris software hangs.

Upon detecting a Solaris hang, dsmd logs the event, including event information, to the domain-specific message log.

Upon detecting a Solaris hang, dsmd requests the domain software to panic so that it can obtain a core image for analysis of the Solaris hang ("Domain Abort or Reset" on page 166). SMS logs this recovery action in the domain-specific message log.

The dsmd daemon monitors the inability of the domain software to satisfy the request to panic. Upon determining noncompliance with the panic request, the dsmd daemon aborts the domain (see "Domain Abort or Reset" on page 166) and initiates an ASR reboot. The dsmd daemon logs these recovery actions in the domain-specific message log.

Although the core image taken as a result of the panic is available for analysis only from the domain, SC-resident log files are available to assist in domain hang analysis. In addition to the Solaris hang event logs, the dsmd daemon can maintain duplicates of important domain-resident logs and transcripts of domain console output on the SC.

Hardware Configuration Events

Changes to the hardware configuration status are considered hardware configuration events. esmd detects the following hardware configuration events on a Sun Fire high-end system.

Hot-Plug Events

The insertion of a hot-pluggable unit (HPU) is a hot-plug event. The following actions take place:

- SMS detects HPU insertion events and logs each event and additional information to a platform message log file.
- If the inserted HPU is a system board in the logical configuration for a domain, SMS also logs its arrival in the domain's message log file.

Hot-Unplug Events

The removal of a hot-pluggable unit (HPU) is a hot-unplug event. The following actions take place:

- Upon occurrence of a hot-unplug event, SMS makes a log entry recording the removal of the HPU to the platform message log file.

- A hot-unplug event that detects the removal of a system board from a logical domain configuration logs it to that domain's message log file.

POST-Initiated Configuration Events

POST can run against different server components at different times due to domain-related events such as reboots and dynamic reconfigurations. As described in "Hardware Configuration" on page 194, SMS includes status from POST and identifying failed-test components. Consequently, changes in POST status of a component are considered to be hardware configuration events. SMS logs POST-initiated hardware configuration changes to the platform message log.

Environmental Events

In general, environmental events are detected when hardware status measurements exceed normal operational limits. Acceptable operational limits depend upon the hardware and the server configuration.

The `esmd` daemon verifies that measurements returned by each sensor are within acceptable operational limits. The `esmd` daemon logs all sensor measurements outside of acceptable operational limits as environmental events to the platform log file.

The `esmd` daemon also logs significant actions taken in response to an environmental event (such as those beyond logging information or updating user displays) to the platform log file.

The `esmd` daemon logs significant environmental event response actions that affect one or more domains to the log files of the affected domains.

The `esmd` daemon handles environmental events by removing from operation the hardware that has experienced the event (and any other hardware dependent upon the disabled component). Hardware can be left in service, however, if continued operation of the hardware does not harm the hardware or cause hardware functional errors.

The options for handling environmental events are dependent upon the characteristics of the event. All events have a time frame during which the event must be handled. Some events kill the domain software; some do not. Event response actions are such that `esmd` responds within the event time frame.

There are a number of responses esmd can make to environmental events, such as increasing fan speeds. In response to a detected environmental event that requires a powering off, esmd undertakes one of the following corrective actions:

- The esmd daemon uses immediate power off if there is no other option that meets the time constraints.
- If the environment event does not require immediate power off and the component is a MaxCPU board, esmd attempts to DR the endangered board out of the running domain and power it off.
- If the environment event does not require immediate power off and the component is a centerplane support board (CSB), esmd attempts to reconfigure the bus traffic to use only the other CSB and power the component off.
- Where possible, if the environment event does not require immediate power off and the component is any type of board other than a MaxCPU or CSB, esmd notifies dsmd of the environment condition and dsmd sends an orderly shutdown request to the domain. The domain flushes uncommitted memory buffers to physical storage.

If the software is still running and a viable domain configuration remains after the affected hardware is removed, dsmd attempts to recover the domain.

If either of the last two options takes longer than the allotted time for the given environmental condition, esmd immediately powers off the component regardless of the state of the domain software.

SMS illuminates the Fault indicator on any hot-pluggable unit that can be identified as the cause of an environmental event.

So long as the environmental event response actions do not include shutdown of the system controllers, all domains whose software operations were terminated by an environmental event or the ensuing response actions are subject to ASR reboot as soon as possible.

ASR reboot begins immediately if there is a bootable set of hardware that can be operated in accordance with constraints imposed by the Sun Fire high-end system to assure safe and correct operation.

Note – Loss of system controller operation (for example, by the requirement to power both SCs down) eliminates all possibility of Sun Fire high-end platform self-recovery actions being taken. In this situation, some recovery actions can require human intervention. Although an external monitoring agent might not be able to recover the Sun Fire high-end platform operation, that monitoring agent could still serve an important role in notifying an administrator about the Sun Fire high-end platform shutdown.

The following sections provide a little more detail about each type of environmental event that can occur on an Sun Fire high-end system.

Over-Temperature Events

The esmd daemon monitors temperature measurements from Sun Fire high-end systems hardware for values that are too high. There is a critical temperature threshold that, if exceeded, is handled as quickly as possible by powering off the affected hardware. High, but not critical, temperatures are handled by attempting slower recovery actions, such as a graceful shutdown or DR for the MCPU boards.

Power Failure Events

There is very little opportunity to do anything when a full power failure occurs. The entire platform, domains as well as SCs, is shut off when the plug is pulled without the benefit of a graceful shutdown. The ultimate recovery action occurs when power is restored (see "Power-On Self-Test (POST)" on page 167).

Out-of-Range Voltage Events

Power voltages for Sun Fire high-end systems are monitored to detect out-of-range events. The handling of out-of-range voltages follows the general principles outlined at the beginning of "Environmental Events" on page 210.

Under-Power Events

In addition to checking for adequate power before powering on any boards, as mentioned in "Power Control" on page 162, the failure of a power supply could leave the server inadequately powered. The system is equipped with power supply redundancy in the event of failure. The esmd daemon does not take any action (other than logging) in response to a bulk power supply hardware failure. The handling of under power events follows the general principles outlined at the beginning of "Environmental Events" on page 210.

Fan Failure Events

The esmd daemon monitors fans for continuing operation. Should a fan fail, a fan failure event occurs. The handling of fan failures follows the general principles outlined at the beginning of "Environmental Events" on page 210.

Clock Failure Events

The esmd daemon monitors clocks for continuing operation. Should a clock fail, esmd logs a message every 10 minutes. It also turns on manual override so the clock selector on that board never automatically starts using that clock. If the clock returns to good status, esmd turns off manual override and logs a message.

When phase lock is lost, the esmd daemon turns on manual override on all the boards and logs one message. When phase lock returns, esmd turns off manual override on all the boards and logs a message.

Hardware Error Events

As described in "Hardware Error Status" on page 196, the occurrence of Sun Fire high-end system hardware errors is recognized at the SC by more than one mechanism. Of the errors that are directly visible to the SC, some are reported directly by PCI interrupt to the UltraSPARC processor on the SC, and others are detected only through monitoring of the hardware registers on Sun Fire high-end systems.

There are other hardware errors that are detected by the processors running in a domain. Domain software running in the domain detects the occurrence of those errors in the domain, which then reports the error to the SC. Like the mechanism by which the SC becomes aware of the occurrence of a hardware error, the error state retained by the hardware after a hardware error is dependent upon the specific error.

The dsmd daemon performs the following functions:

- Implements the mechanisms necessary to detect all SC-visible hardware errors
- Implements domain software interfaces to accept reports of domain-detected hardware errors
- Collects hardware error data and clears the error state
- Logs the hardware error and related information as required, to the platform message log
- Logs the hardware error to the domain message log file for all affected domains

If data collected in response to a hardware error is not suitable for inclusion in a log file, the data can be saved in uniquely named files in /var/opt/SUNWSMS/adm/*domain-id*/dump on the SC.

SMS illuminates the Fault LED on any hot-pluggable unit that can be identified as the cause of a hardware error.

The actions taken in response to hardware errors (other than collecting and logging information as described previously) are twofold. First, it might be possible to eliminate the further occurrence of certain types of hardware errors by eliminating from use the hardware identified to be at fault. Second, all domains that crashed either as a result of a hardware error or were shut down as a consequence of the first type of action are subject to ASR reboot actions.

Note – Even when hardware is not shutdown or identified to be at fault, the ASR reboot actions are subject to full POST verification. POST eliminates any hardware components that fail testing from the hardware configuration.

In response to each detected hardware error and each domain-software-reported hardware error, `dsmd` undertakes the appropriate corrective actions. In some cases automatic diagnosis and domain recovery occurs (see Chapter 6), while in other instances, an ASR reboot with full POST verification is initiated for each domain brought down by a hardware error.

Note – Problems with the ASR reboot of a domain after a hardware error are detected as domain boot failure events and subject to the recovery actions described in "Domain Boot Failure" on page 205.

The `dsmd` daemon logs all significant actions, such as those beyond logging information or updating user displays taken in response to a hardware error in the platform log file. When a hardware error affects one or more domains, `dsmd` logs the significant response actions in the message log files of the affected domains.

The following sections summarize the types of hardware errors expected to be detected and handled on a Sun Fire high-end system.

Domain Stop Events

Domain stops are uncorrectable hardware errors that immediately terminate the affected domains. Hardware state dumps are taken before `dsmd` initiates an ASR reboot of the affected domains. These files are located in `/var/opt/SUNWSMS/adm/`*domain-id*`/dump`

The `dsmd` daemon logs the event in the domain message log file and also the event log file.

CPU-Detected Events

A RED_state or Watchdog reset traps to low-level domain software (OpenBoot PROM or kadb), which reports the error and requests initiation of ASR reboot of the domain.

An XIR signal (reset -x) also traps to low-level domain software (OpenBoot PROM or kadb), which retains control of the software. The domain must be rebooted manually.

Record Stop Events

Correctable data transmission errors (for example, CE ECC errors) can stop the normal transaction history recording feature of ASICs in Sun Fire high-end systems. SMS reports a transmission error as a record stop. SMS dumps the transaction history buffers of these ASICs and re-enables transaction history recording when a record stop is handled. The dsmd daemon records record stops in the domain log file.

Other ASIC Failure Events

ASIC-detected hardware failures other than domain stop or record stop include console bus errors, which might or might not impact a domain. The hardware itself does not abort any domain, but the domain software might not survive the impact of the hardware failure and could panic or hang. The dsmd daemon logs the event in the domain log file.

SC Failure Events

SMS monitors the main SC hardware and running software status as well as the hardware and running software of the spare SC, if present. In a high-availability SC configuration, SMS handles failures of the hardware or software on the main SC or failures detected in the hardware control paths (for example, console bus, or internal network connections) to the main SC by an automatic SC failover process. This cedes main responsibilities to the spare SC and leaves the former main SC as a (possibly crippled) spare.

SMS monitors the hardware of the main and spare SCs for failures.

SMS logs the hardware failure and related information to the platform message log.

SMS illuminates the Fault LED on a system controller with an identified hardware failure.

For more information, see Chapter 12.

CHAPTER **12**

SC Failover

SC failover maximizes Sun Fire high-end system uptime by adding high-availability features to its administrative operations. A Sun Fire high-end system contains two SCs. Failover provides software support to a high-availability two-SC system configuration.

The main SC provides all resources for the entire Sun Fire high-end system. If hardware or software failures occur on the main SC or on any hardware control path (for example, the console bus interface or Ethernet interface) from the main SC to other system devices, SC failover software automatically triggers a failover to the spare SC. The spare SC then assumes the role of the main and takes over all the main SC responsibilities. In a high-availability, system configuration using two SCs, SMS data, configuration, and log files are replicated on the spare SC. Active domains are not affected by this switch.

Note – For failover to be supported, both SCs must be configured with identical versions of the Solaris OS and SMS software.

This chapter includes the following sections:

Overview

In the current high-availability SC configuration, one SC acts as a “hot spare” for the other.

Failover eliminates the single point of failure in the management of the Sun Fire high-end system. The fomd daemon identifies and handles as many multiple points failure as possible. Some failover scenarios are discussed in “Failure and Recovery” on page 229.

At any time during SC failover, the failover process does not adversely affect any configured or running domains except for temporary loss of services from the SC.

In a high-availability SC system:

- If a software or hardware fault is detected on the main SC, fomd automatically fails over to the spare SC.
- If the spare SC detects that the main SC has stopped communicating with it, the spare SC initiates a takeover and assumes the role of main.

The failover management daemon (fomd(1M)) is the core of the SC failover mechanism. It is installed on both the main and spare SCs.

The fomd daemon performs the following functions:

- Determines an SC’s role (main or spare).
- Requests the general health status of the remote SC hardware and software in the form of a periodic health status message request sent over the SMS Management Network (MAN) that exists between the two SCs.
- Checks and handles recoverable and unrecoverable hardware and software faults.
- Makes every attempt to eliminate the possibility of a split-brain condition between the two SCs. (A condition is considered *split-brain* when both the SCs think they are the main SC.)
- Provides a recovery time from a main SC failure of between five and eight minutes. The recovery time includes the time for fomd to detect the failure, reach an agreement on the failure, and assume the main SC responsibilities on the spare SC.
- Logs an occurrence of an SC failover in the platform message log.

Services that would be interrupted during an SC failover include:

- All network connections
- Any SC-to-domain and domain-to-SC IOSRAM or mailbox communication
- Any process running on the main SC

You do not need to know the host name of the main SC to establish connections to it. As part of configuring SMS (refer to the `smsconfig`(1M) man page), a logical host name was created which is always active on the main SC. Refer to the *Sun Fire 15K/12K System Site Planning Guide* and the *System Management Services (SMS) 1.6 Installation Guide* for information on the creation of the logical host names in your network database.

Operations interrupted by an SC failover can be recovered after the failover completes. Reissuance of the interrupted operation causes the operation to resume and continue to completion.

All automated functions provided by `fomd` resume without operator intervention after SC failover. Any recovery actions interrupted before completion by the SC failover restarts.

Fault Monitoring

There are three types of failovers:

1. Main-initiated

   A main-initiated failover is where the `fomd` running on the main SC yields control to the spare SC in response to either an unrecoverable local hardware or software failure or an operator request.

2. Spare-initiated (takeover)

   In a spare-initiated failover (takeover), the `fomd` running on the spare determines that the main SC is no longer functioning properly.

3. Indirect-triggered takeover

   If the I2 network path between the SCs is down and there is a fault on the main, the main switches itself to the role of spare. Upon detecting this, the spare SC assumes the role of main.

In the last two scenarios, the spare `fomd` eliminates the possibility of a split-brain condition by resetting the main SC.

When either a software-controlled or a user-forced failover occurs, `fomd` deactivates the failover mechanism. This eliminates the possibility of repeatedly failing over back and forth between the two SCs.

File Propagation

One of the purposes of the `fomd` is propagation of data from the main SC to the spare SC through the interconnects that exist between the two SCs. This data includes configuration, data, and log files.

The `fomd` daemon performs the following functions:

- Propagates all native SMS files from the main to the spare SC at startup. These include all the domain data directories, the pcd configuration files, the `/etc/opt/SUNWSMS/config` directory, the `/var/opt/SUNWSMS/adm` platform and domain files, and the `.logger` files. Any user-created application files are not propagated unless specified in the `cmdsync` scripts.
- Propagates only files modified since the last propagation cycle.
- In the event of a failover, propagates all modified SMS files before the spare SC assumes its role as main.

The I2 network must be operative for the transfer of data to occur.

Note – Any changes made to the network configuration on one SC using `smsconfig -m` must be made to the other SC as well. Network configuration is not automatically propagated.

Should both interconnections between the two SCs fail, failover can still occur provided main and spare SC accesses to the high-availability SRAMs (HASRAMs) remain intact. Due to the failure of both interconnections, propagation of SMS data can no longer occur, creating the potential of stale data on the spare SC. In the event of a failover, `fomd` on the new main keeps the current state of the data, logs the state, and provides other SMS daemons and clients information about the current state of the data.

When either of the interconnects between the two SCs is healthy again, data is pulled over depending on the timestamp of each SMS file. If the timestamp of the file is earlier than the one on the SC now acting as the spare, it gets transferred over. If the timestamp of the file is later than the one on the spare SC, no action is taken.

Failover cannot occur when both of the following conditions are met:

- Both interconnects between the two SCs fail
- Access to both HASRAMs fails

This is considered a quadruple fault, and failover is disabled until at least one of the links is restored.

Failover Management

This section explains the startup, main SC, and spare SC roles.

Startup

Note – Failover between main and spare SCs with different Solaris OS versions is not a Sun-supported configuration.

For the failover software to function, both SCs must be present in the system. The determination of main and spare roles is based in part on the SC number. This slot number does not prevent a given SC from assuming either role – it only controls how it goes about doing so.

If SMS is started on one SC first, that SC becomes main. If SMS starts up on both SCs at essentially the same time, whichever SC first determines that the other SC either is not main or is not running SMS becomes main.

If SC0 is in the middle of the startup process, it queries SC1 for its role, and if the SC1 role cannot be confirmed, SC0 tries to become main. SC0 resets SC1 during this process. This is done to prevent both SCs from assuming the main role, a condition known as split brain. The reset occurs even if the failover mechanism is deactivated.

Main SC

Upon startup, the `fomd` running on the main SC begins periodically testing the hardware and network interfaces. Initially the failover mechanism is disabled (internally) until at least one status response has been received from the remote (spare) SC indicating that it is healthy.

If a local fault is detected by the main `fomd` during initial startup, failover occurs when all of the following conditions are met:

1. The I2 network was not the source of the fault.
2. The remote SC is healthy (as indicated by the health status response).
3. The failover mechanism has not been deactivated.

Spare SC

Upon startup, fomd runs on the spare SC and begins periodically testing the software, hardware, and network interfaces.

If a local fault is detected by the fomd running on the spare SC during initial startup, it informs the main fomd of its debilitated state.

Failover CLI Commands

This section describes the setfailover and showfailover commands.

setfailover Command

The setfailover command modifies the state of the SC failover mechanism. The default state is on. The following is an example of using the setfailover command:

```
# setfailover [-q] [-y|-n] [on|off|force]
```

Forcing a failover to a spare SC with a faulty clock can cause the affected domains to domain stop (dstop). The setfailover command detects faulty clocks on spare SCs and provides a second chance confirmation prompt to avoid accidentally forcing a failover to a faulty SC. However, the -q (quiet) and -y (yes to all prompts) options do not allow checking for a faulty SC.

Caution – The -q option suppresses *all* prompts, including the second chance prompt. If you use both the -q and the -y options, the failover is forced to the spare SC even if it is faulty. This forced failover could result in a Dstop if the spare SC is faulty.

The following is an example of the setfailover command detecting a faulty clock on the spare SC:

```
# setfailover force
Forcing failover. Do you want to continue (yes/no)? yes
The spare clock input on some boards might be bad. Forcing a
failover now is likely to cause the affected domains to domain stop
(Dstop).
Do you want to continue (yes/no)? no
```

TABLE 12-1 describes SC failover states.

TABLE 12-1 Options for Modifying Failover States

State	Definition
[-q]	Enables quiet mode, which suppresses all messages to stdout including prompts. When used alone, -q defaults to the -n option for all prompts. When used with either the -y or the -n option, -q suppresses all user prompts and automatically answers with either yes or no based on the option chosen.
[-y|-n]	-y automatically answers yes to all prompts. Prompts are displayed unless used with the -q option. *Use with caution*. -n automatically answers no to all prompts. Prompts are displayed unless used with the -q option.
on	Enables failover for systems that previously had failover disabled due to a failover or an operator request. This option instructs the command to attempt to re-enable failover only. If failover cannot be re-enabled, subsequent use of the showfailover command indicates the current failure that prevented the enable.
off	Disables the failover mechanism. This prevents a failover until the mechanism is re-enabled.
force	Forces a failover to the spare SC. The spare SC must be available and healthy.

Note – In the event a patch must be applied to SMS 1.6, failover must be disabled before the patch is installed. Refer to the *System Management Services (SMS) 1.6 Installation Guide.*

For more information and examples, refer to the setfailover man page.

showfailover Command

The showfailover command allows you to monitor the state and display the current status of the SC failover mechanism. The -v option displays the current status of all monitored components.

```
xc30p13-sc0:sms-svc:13> showfailover -v
SC Failover Status:      ACTIVE
Status of Shared Memory:
HASRAM (CSB at CS0): .....................................Good
HASRAM (CSB at CS1): .....................................Good
Status of xc30p13-sc0:
Role: ................................................MAIN
SMS Daemons: ........................................Good
System Clock: .......................................Good
Private I2 Network: .................................Good
Private HASRAM Network:..............................Good
Public Network..................................NOT TESTED
System Memory: ......................................38.9%
S Disk Status:
/: ..................................................17.4%
Console Bus Status:
EXB at EX1: ..............................................Good
EXB at EX2: ..............................................Good
 EXB at EX4: .............................................Good
Status of xc30p13-sc1:
Role: ...............................................SPARE
SMS Daemons: ........................................Good
System Clock: .......................................Good
Private I2 Network: .................................Good
Private HASRAM Network:..............................Good
Public Network: ................................NOT TESTED
System Memory: ......................................34.2%
Disk Status:
/: ..................................................17.1%
Console Bus Status:
EXB at EX1: .........................................Good
EXB at EX2: .........................................Good
EXB at EX4: .........................................Good
```

The -r option displays the SC role: main, spare, or unknown. For example:

```
sc0:sms-user:> showfailover -r
MAIN
```

If you do not specify an option, only the state information is displayed:

```
sc0:sms-user:> showfailover
SC Failover Status: state
```

The failover mechanism can be in one of four states: ACTIVATING, ACTIVE, DISABLED, and FAILED. TABLE 12-2 describes the four states.

TABLE 12-2 States of the Failover Mechanism

State	Definition
ACTIVATING	The failover mechanism is preparing to transition to the ACTIVE state. Failover becomes active when all tests have passed and files have been synchronized.
ACTIVE	The failover mechanism is enabled and functioning normally.
DISABLED	The failover mechanism has been disabled due to the occurrence of a failover or an operator request (`setfailover` off).
FAILED	The failover mechanism has detected a failure that prevents a failover from being possible, or failover has not yet completed activation.

In addition `showfailover` displays the state of each of the network interface links monitored by the failover processes. The display format is as follows:

```
network i/f device name: [GOOD|FAILED]
```

The `showfailover` returns a failure string describing the failure condition. Each failure string has a code associated with it. The following table defines the codes and associated failure strings.

TABLE 12-3 describes the `showfailover` command failure strings.

TABLE 12-3 `showfailover` Failure Strings

String	Explanation
None	No failure.
S-SC EXT NET	The spare SC external network interface has failed.
S-SC CONSOLE BUS	A fault has been detected on the spare SC console bus paths.
S-SC LOC CLK	The spare SC local clock has failed.
S-SC DISK FULL	The spare SC system is full.

TABLE 12-3 `showfailover` Failure Strings *(Continued)*

String	Explanation
S-SC IS DOWN	The spare SC is down or unresponsive. If this message results from the I2 network or HASRAMs being down, the spare SC could still be running. Log in to the spare SC to verify.
S-SC MEM EXHAUSTED	The spare SC memory or swap space has been exhausted.
S-SC SMS DAEMON	At least one SMS daemon could not be started or restarted on the spare SC.
S-SC INCOMPATIBLE SMS VERSION	The spare SC is running a different version of SMS software. Both SCs must be running the same version.
I2 NETWORK/HASRAM DOWN	Both interfaces for communication between the SCs are down. The main cannot tell what version of SMS is running on the spare or what its state is. It declares the spare down and logs a message to that effect. Dependent services, including file propagation, are unavailable.

For examples and more information, refer to the `showfailover` man page.

Command Synchronization

If an SC failover occurs during the execution of a command, you can restart the same command on the new main SC.

All commands and actions do the following:

- Mark the start of a command or action
- Remove or indicate the completion of a command or action
- Keep any state transition and pertinent data that SMS can use to resume the command

The `fomd` daemon provides the following support for command synchronization:

- Command sync support for `dsmd`(1M) to automatically resume ASR reboots of any or all affected domains after a failover
- Command sync support for all SMS DR-related daemons and CLIs to restart the last DR operation after a failover

The four CLI commands in SMS that require command sync support are `addboard`, `deleteboard`, `moveboard`, and `rcfgadm`.

cmdsync CLIs

The *cmdsync* commands provide the ability to initialize a script or command with a cmdsync descriptor, update an existing cmdsync descriptor execution point, or cancel a cmdsync descriptor from the spare SC's list of recovery actions. Commands or scripts can also be run in a cmdsync envelope.

In the case of an SC failover to the spare, initialization of a cmdsync descriptor on the spare SC enables the spare SC to restart or resume the target script or command from the last execution point set. These commands executes only on the main SC, and have no effect on the current cmdsync list if executed on the spare.

Commands or scripts invoked with the cmdsync commands when there is no enabled spare SC result in a no-op operation. That is, command execution proceeds as normal, but a log entry in the platform log indicates that a cmdsync attempt has failed.

initcmdsync Command

The initcmdsync(1M) command creates a cmdsync descriptor. The target script or command and its associated parameters are saved as part of the cmdsync data. The exit code of the initcmdsync command provides a cmdsync descriptor that can be used in subsequent cmdsync commands to reference the action. Actual execution of the target command or script is not performed. For more information, refer to the initcmdsync (1M) man page.

savecmdsync Command

The savecmdsync(1M) command saves a new execution point in a previously defined cmdsync descriptor. This allows a target command or script to restart execution at a location associated with an identifier. The target command or script supports the ability to be restarted at this execution point, otherwise the restart execution is at the beginning of the target command or script. For more information, refer to the savecmdsync (1M) man page.

cancelcmdsync Command

The cancelcmdsync(1M) command removes a cmdsync descriptor from the spare restart list. Once this command is run, the target command or script associated with the cmdsync descriptor is not restarted on the spare SC in the event of a failover. Take care to ensure that all target commands or scripts contain an initcmdsync command sequence as well as a cancelcmdsync sequence after the normal or abnormal termination flows. For more information, refer to the cancelcmdsync (1M) man page.

runcmdsync Command

The runcmdsync(1M) command executes the specified target command or script under a cmdsync wrapper. You cannot restart at execution points other than the beginning. The target command or script is executed through the system command after creation of the cmdsync descriptor. Upon termination of the system command, the cmdsync descriptor is removed from the cmdsync list, and the exit code of the system command returned to the user. For more information, refer to the runcmdsync (1M) man page.

showcmdsync Command

The showcmdsync(1M) command displays the current cmdsync descriptor list. For more information, refer to the showcmdsync (1M) man page.

Data Synchronization

Customized data synchronization is provided in SMS by the setdatasync(1M) command. setdatasync enables you to specify a user-created file to be added to or removed from the data propagation list.

setdatasync Command

The setdatasync list identifies the files to be copied from the main to the spare system controller (SC) as part of data synchronization for automatic failover. The specified user file and the directory in which it resides must have read and write permissions for you on both SCs. You must also have platform or domain privileges.

The data synchronization process checks the user-created files on the main SC for any changes. If the user-created files on the main SC have changed since the last propagation, they are repropagated to the spare SC. By default, the data synchronization process checks a specified file every 60 minutes; however, you can use setdatasync to indicate how often a user file is checked for modifications.

You can also use `setdatasync` to propagate a specified file to the spare SC without adding the file to the data propagation list.

Using `setdatasync backup` can slow down automatic `fomd` file propagation.

The time required to execute `setdatasync backup` is proportional to the number of files being transferred. Other factors that can affect the speed of file transfer include: the average size of files being transferred, the amount of memory available on the SCs, the load (CPU cycles and disk traffic) on the SCs, and whether the I2 network is functioning.

The following statistics assume an average file size of 200 Kbytes:

- On a lightly loaded system with a functioning I2 network, FOMD can transfer about 750 files per minute.
- On a lightly loaded system with no functioning I2 network, FOMD can transfer about 250 files per minute.

Note – There are repropagation constraints you should be aware of *before* using this command. For more information and examples, refer to the `setdatasync` (1M) man page.

`showdatasync` Command

The `showdatasync` command provides the current status of files being propagated (copied) from the main SC to its spare. The `showdatasync` command also provides the list of files registered using `setdatasync` and their status. Data propagation synchronizes data on the spare SC with data on the main SC, so that the spare SC is current with the main SC if an SC failover occurs.

For more information, refer to the `showdatasync` (1M) man page.

Failure and Recovery

In a high-availability configuration, `fomd` manages the failover mechanism on the local and remote SCs. the `fomd` daemon detects the presence of local hardware and software faults and determines the appropriate action to take.

The `fomd` daemon is responsible for detecting the faults described in TABLE 12-4.

TABLE 12-4 `fomd` Hardware and Software Fault Categories

Category	Description
a	All relevant hardware buses that are local to the SC Control board (CB)/CPU board.
b	The external network interfaces.
c	The I2 network interface between the SCs.
d	Unrecoverable software failures. This category is for those cases where an SMS software component (daemon) crashes and cannot be restarted after three attempts, the file system is full, the heap is exhausted, and so forth.

FIGURE 12-1 illustrates the failover fault categories.

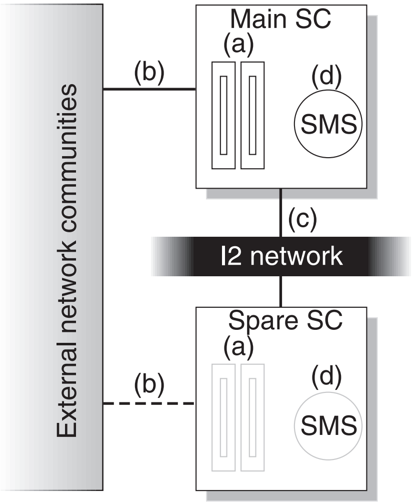


FIGURE 12-1 Failover Fault Categories

TABLE 12-5 illustrates how faults in the categories affect the failover mechanism. Assume that the failover mechanism is activated.

TABLE 12-5 Failover Fault Categories

Failure Point	Main SC	Spare SC	Failover	Notes
a	X		X	Failover to spare occurs.
a		X	Disables	No effect on the main SC, but the spare SC has suffered a hardware fault so failover is disabled.
b	X			Failover to spare.
b		X	No effect	The fact that the spare SC external network interfaces have failed does not affect the failover mechanism.
c			No effect	Main and spare SC log the fault.
d	X		X	Failover to the spare SC, assuming that it is healthy.
d		X	Disables	Failover is disabled because the spare SC is deemed unhealthy at this point.

Failover on Main SC (Main-Controlled Failover)

Events for the main `fomd` during SC failover occur in the following order:

1. Detects the fault.

2. Stops generating heartbeats.

3. Tells the remote failover software to start a takeover timer. The purpose of this timer is to provide an alternate means for the remote (spare) SC to take over if for any reason the main hangs and never reaches a count of 10.

4. Starts the SMS software in spare mode.

5. Removes the logical IP interface.

6. Enables the console bus caging mechanism.

7. Triggers propagation of any modified SMS files to the spare SC or HASRAMs.

8. Stops file propagation monitoring.

9. Shuts down main-specific daemons and sets the main SC role to UNKNOWN.

10. Logs a failover event.

11. Notifies remote (spare) failover software that it should assume the role of main. If the takeover timer expires before the spare is notified, the remote SC takes over on its own.

Events for the spare fomd during failover occur in the following order:

1. Receives message from the main fomd to assume main role, or the takeover timer expires. If the former is true, then the takeover timer is stopped.
2. Resets the old main SC.
3. Notifies hwad, frad, and mand to configure the spare fomb in the main role.
4. Assumes the role of main.
5. Starts generating heartbeat interrupts.
6. Configures the logical IP interface.
7. Disables the console bus caging mechanism.
8. Starts the SMS software in main mode.
9. Prepare the DARBs to receive interrupts.
10. Logs a role reversal event, spare to main.
11. The spare SC is now the main, and fomd deactivates the failover mechanism.

Fault on Main SC (Spare Takes Over Main Role)

In this scenario, the spare SC takes main control in reaction to loss of communication with the main SC. The most important aspect of this type of failover is the prevention of the split-brain condition. Another assumption is that the failover mechanism is not deactivated. If it has been deactivated, no takeover can occur.

The spare fomd does the following:

- Notices that the main SC is not healthy

  From the spare fomd perspective, this phenomenon can be caused by two conditions: the main SC is truly dead, or the I2 network interface is down.

  In the former case, a failover is needed (provided that the failover mechanism is activated), while in the latter it is not. To identify which is the case, the spare fomd polls for the presence of heartbeat interrupts from the main SC to determine if the main SC is still up and running. As long as heartbeat interrupts are being received, or the failover mechanism is deactivated or disabled, no failover occurs.

In the case where no interrupts are detected but the failover mechanism is deactivated, the spare fomd does not attempt to take over unless the operator manually activates the failover mechanism using the CLI command setfailover. Otherwise, if the spare SC is healthy, the spare fomd proceeds to take over the role of main.

- Initiates a takeover by resetting the remote (main) SC.

The following lists the events for the spare fomd, in order, during failover:

1. Reconfigures itself as main. This includes taking over control of the I^2C bus, configuring the logical main SC IP address, and starting up the necessary SMS software daemons.
2. Starts generating heartbeat interrupts.
3. Configures the logical IP interface.
4. Disables console bus caging.
5. Starts the SMS software in main mode.
6. Configures the DARB interrupts.
7. Logs a takeover event.
8. The spare fomd, now the main, deactivates the failover mechanism.

I2 Network Fault

The following lists the events, in order, that occur after an I2 network fault.

1. The main fomd detects the I2 network is not healthy.
2. The main fomd stops propagating files and checkpointing data over to the spare SC.
3. The spare fomd detects the I2 network is not healthy.

   From the spare fomd perspective, this phenomenon can be caused by two conditions: the main SC is truly malfunctioning, or the I2 network interface is down. In the former case, the corrective action is to fail over, while in the latter, it is not. To identify which is the case, the fomd starts polling for the presence of heartbeat interrupts from the main SC to determine if the main SC is still up and running. If heartbeat interrupts are present, the fomd keeps the spare as spare.
4. The spare fomd clears out the checkpoint data on the local disk.

Fault on Main SC (I2 Network Is Also Down)

The following lists the events, in order, that occur after a fault on the main SC.

1. The main `fomd` detects the fault.

   If the last known state of the spare SC was good, then the main `fomd` stops generating heartbeats. Otherwise, failover does not continue.

   If the access to the console bus is still available, the main failover software finishes propagating any remaining critical files to HASRAM and flushes out any or all critical state information to HASRAM.

2. The main `fomd` reconfigures the SMS software into spare mode.

3. The main `fomd` removes the logical main SC IP address.

4. The main `fomd` stops generating heartbeat interrupts.

Fault Recovery and Reboot

This section describes fault recovery and reboot precesses.

I2 Fault Recovery

The following lists the events, in order, that occur during an I2 network fault recovery.

1. The main `fomd` detects that the I2 network is healthy.

   If the spare SC is completely healthy as indicated in the health status response message, the `fomd` enables failover and, assuming that the failover mechanism has not been deactivated by the operator, does a complete re-sync of the log files and checkpointing data over to the spare SC.

2. The spare `fomd` detects that the I2 network is healthy.

   The spare `fomd` disables failover and clears out the checkpoint data on the local disk.

Reboot and Recovery

The following lists the events, in order, that occur during a reboot and recovery. A reboot and recovery scenario happens in two cases.

Main SC Receives a Master Reset or Its UltraSPARC Processor Receives a Reset

1. Assume SSCPOST passed without any problems. If SSCPOST failed and the OS cannot be booted, the main is inoperable.

2. Assume all SSC Solaris drivers attached without any problems. If the SBBC driver fails to attach, see "Fault on Main SC (Spare Takes Over Main Role)" on page 232. If any other drivers fail to attach, see "Failover on Main SC (Main-Controlled Failover)" on page 231.

3. The main `fomd` is started.

4. If the `fomd` determines that the remote SC has already assumed the main role, then see Number 5 in "Spare SC Receives a Master Reset or Its UltraSPARC Processor Receives a Reset" on page 235. Otherwise, proceed to Number 5 in this list.

5. The `fomd` configures the logical main IP address and starts up the rest of the SMS software.

6. SMS daemons start in recovery mode if necessary.

7. Main `fomd` starts generating heartbeat interrupts.

8. At this point, the main SC is fully recovered.

Spare SC Receives a Master Reset or Its UltraSPARC Processor Receives a Reset

1. Assume SSCPOST passed without any problems. If SSCPOST failed and the OS cannot be booted, the spare is inoperable.

2. Assume all SSC Solaris drivers attached without any problems. If the SBBC driver fails to attach, or any other drivers fail to attach, the spare SC is deemed inoperable.

3. The `fomd` is started.

4. The `fomd` determines that the SC is the preferred spare and assumes the spare role.

5. The `fomd` starts checking for the presence of heartbeat interrupts from the remote (initially presumed to be main) SC.

If after a configurable amount of time no heartbeat interrupts are detected, the failover mechanism state is checked. If enabled and activated, `fomd` initiates a take over. See Number 5 of "Main SC Receives a Master Reset or Its UltraSPARC Processor Receives a Reset" on page 235. Otherwise, `fomd` continues monitoring for the presence of heartbeat interrupts and the state of the failover mechanism.

6. The `fomd` starts periodically checking the hardware, software, and network interfaces.
7. The `fomd` configures the local main SC IP address.
8. At this point, the spare SC is fully recovered.

Client Failover Recovery

The following lists the events that occur during a client failover recovery. A recovery scenario happens in the following two cases.

Fault on Main SC–Recovering From the Spare SC

Clients with any operations in progress are manually recovered by checkpointing any recurring data.

Fault on Main SC (With I2 Network Down)–Recovering From the Spare SC

Since the I2 network is down, all checkpointing data is removed. Clients cannot perform any recovery.

Once you have finished with recovery, you can continue with the reboot steps.

Reboot Main SC (With Spare SC Down)

This condition is identical to "Fault on Main SC–Recovering From the Spare SC" on page 236.

Reboot of Spare SC

No recovery is necessary.

Security

All failover-specific network traffic (such as health status request or response messages and file propagation packets) is sent *only* over the interconnect network that exists between the two SCs.

CHAPTER 13

SMS Utilities

This section discusses the SMS backup, configuration, restore, and version utilities. For more information and examples of these utilities, refer to the *System Management Services (SMS) 1.6 Reference Manual* and online man pages.

This chapter includes the following sections:

SMS Backup Utility

The smsbackup creates a cpio(1) archive of files that maintain the operational environment of SMS.

Note – This utility runs on the SC and does not replace the need for routine and timely backups of SC and domain OSs and domain application data.

Whenever changes are made to the SMS environment (for example, by adding boards to or removing boards from a domain), you must run smsbackup again to maintain a current backup file for the system controller.

The name of the backup file is smsbackup.*X.X*.cpio, where *X.X* represents the active version from which the backup was taken.

The smsbackup utility saves all configuration, platform configuration database, SMS, and log files. In other words, SMS saves everything needed to return SMS to the working state it was in at the time the backup was made.

Backups are *not* performed automatically. Whenever changes are made to the SMS environment, a backup should be performed. This process can be automated by making it part of a root cron job run at periodic intervals depending on your site requirements.

The backup log file resides in /var/sadm/system/logs/smsbackup. You must specify the target location when running smsbackup.

Note – The target location must be a valid UNIX file system (UFS) directory. You cannot perform smsbackup to a tmp file system directory.

Whenever you run smsbackup, you receive confirmation that it succeeded or are notified that it failed.

You must have superuser privileges to run smsbackup. For more information and examples, refer to the smsbackup man page.

Restore SMS backup files using the smsrestore(1M) command.

SMS Restore Utility

The smsrestore utility restores the operational environment of the SMS from a backup file created by smsbackup(1M). You can use smsrestore to restore the SMS environment after the SMS software has been installed on a new disk or after hardware replacement or addition. Failover should be disabled and SMS stopped before smsrestore is performed. Refer to the "Stopping and Starting SMS" section of the *System Management Services (SMS) 1.6 Installation Guide*.

If any errors occur, smsrestore writes error messages to /var/sadm/system/logs/smsrestore.

Note – This utility runs on the SC and does not restore SC OS, domain OS, or domain application data.

The smsrestore utility cannot restore what you have not backed up. Whenever changes are made to the SMS environment (for example, by shutting down a domain), you must run smsbackup to maintain a current backup file for the system controller.

You must have superuser privileges to run smsrestore. For more information and examples, refer to the smsrestore man page.

SMS Version Utility

The smsversion(1M) utility administers adjacent, co-resident installations of SMS under the same OS. Adjacent versions of SMS are versions with sequential version numbers, such as SMS 1.4.1 and SMS 1.6. In other words, you cannot use smsversion to switch directly between SMS 1.2 and SMS 1.6 or 1.5 to 1.6.

Note – Switching versions from SMS 1.6 to an earlier installed version has SC security implications. Refer to "Switching SMS Versions" in the *System Management Services (SMS) 1.6 Installation Guide*.

The smsversion utility permits two-way SMS version-switching between sequential co-resident installations on the same OS. TABLE 13-1 notes the conditions for use.

TABLE 13-1 Switching Between SMS Versions

Condition	Explanation
New features	Features supported in the newer version of SMS (for example, SC Secure by Default functionality), might not be supported in the older version. Switching to an older version of SMS can result in the loss of those features. Also, the settings for the new features might be erased.
Flash PROM differences	Switching versions of SMS requires reflashing the CPU flash PROMs with the correct files. These files can be found in the /opt/SUNWSMS/*SMS_version*/firmware directory. Use flashupdate(1M) to reflash the PROMs after you have switched versions. Refer to the flashupdate man page and *System Management Services (SMS) 1.6 Installation Guide* for more information on updating flash PROMs.

When you switch between sequential releases of SMS (for example, 1.6 to 1.4.1), SMS must be stopped before running smsversion. Refer to "Stopping and Starting SMS" in the *System Management Services (SMS) 1.6 Installation Guide*. The smsversion utility backs up important system and domain information and switches to the target SMS version. You can switch back to the next sequential SMS version (for example, 1.6 to 1.5) at a later time.

Note – Switching between sequential SMS versions across Solaris OSs (for example, Solaris 8 and 9 OSs) is *not* supported. Once you upgrade from a Solaris 8 version of SMS to a Solaris 9 version, you cannot go back without also reinstalling the earlier version of the OS. Using the `smsversion` command to switch from Solaris 10 with SMS 1.6 back to SMS 1.5 is not supported unless the previous OS is reinstalled. Refer to the *System Management Services (SMS) 1.6 Installation Guide* for more information.

Without options, `smsversion` displays the active version and exits when only one version of SMS is installed.

If any errors occur, `smsversion` writes error messages to `/var/sadm/system/logs/smsversion`.

You must have superuser privileges to run `smsversion`. For more information and examples, refer to the `smsversion` man page.

Version Switching

Note – Switching from SMS 1.6 to an earlier installed version of SMS has SC security implications. Refer to the *System Management Services (SMS) 1.6 Installation Guide* for more information.

▼ To Switch Between Two Adjacent, Co-resident Installations of SMS

On the main SC:

1. **Make certain your configuration is stable and backed up using** `smsbackup`**.**

   Being stable means the following commands should *not* be running: `smsconfig`, `poweron`, `poweroff`, `setkeyswitch`, `cfgadm`, `rcfgadm`, `addtag`, `deletetag`, `addboard`, `moveboard`, `deleteboard`, `setbus`, `setdefaults`, `setobpparams`, `setupplatform`, `enablecomponent`, or `disablecomponent`.

2. **Deactivate failover using** `setfailover off`**.**

   On the spare SC:

3. **Run** `/etc/init.d/sms stop`**.**
4. **Run** `smsversion`**.**
5. **Run** `smsrestore`**.**

6. **If necessary, run** `smsconfig -m` **and reboot.**

   Run only `smsconfig -m` if you changed your network configuration using `smsconfig -m` *after* creating the `smsbackup` you just restored.

   On the main SC:

7. **Stop SMS using** `/etc/init.d/sms stop`**.**

   On the spare SC:

8. **If** `smsconfig -m` **was run, reboot; otherwise, run** `/etc/init.d/sms start`**.**

   When the SC comes up, it becomes the main SC.

9. **If necessary, update the CPU flash PROMs using** `flashupdate`**.**

   On the former main SC:

- **Repeat Steps 4-6 and 8.**

  On the new main SC:

- **Activate failover using** `setfailover on`**.**

  For more information refer to the *System Management Services (SMS) 1.6 Installation Guide*.

SMS Configuration Utility

The `smsconfig` utility configures the MAN networks, modifies the hostname and IP address settings used by the MAN daemon `mand`(1M), and administers domain directory access control lists (ACLs). It also displays the current configuration.

UNIX Groups

The `smsconfig` utility configures the UNIX groups used by SMS to describe user privileges. SMS uses a default set of UNIX groups installed locally on each SC. The `smsconfig` utility allows you to customize those groups using the `-g` option. You can also add users to groups using the `-a` option and remove users from groups using the `-r` option.

For information and examples on adding, removing, and listing authorized users, refer to the *System Management Services (SMS) 1.6 Installation Guide* and the `smsconfig`(1M) man page.

Access Control List (ACL)

Traditional UNIX file protection provides read, write, and execute permissions for the three user classes: file owner, file group, and other. To provide protection and isolation of domain information, access to each domain's data is denied to all unauthorized users. SMS daemons, however, are considered authorized users and have full access to the domain file systems. For example:

- `sms-esmd` needs to read the blacklist files in each domain directory: `$SMSETC/config/[A-R]`
- `sms-osd` needs to read from and write to the `bootparamdata` file in each domain: `$SMSVAR/data/[A-R]`
- `sms-dsmd` needs to write to hpost logs for every domain: `$SMSVAR/adm/[A-R]/post`

The `smsconfig` utility sets the ACL entries associated with the domain directories so that the domain administrator has full access to the domain. A plus sign (+) to the right of the mode field indicates the directories that have ACL defined.

```
domain-id:sms-user:> ls -al
total 6
drwxrwxrwx   2 root     bin          512 May 10 12:29 .
drwxrwxr-x  23 root     bin         1024 May 10 12:29 ..
-rw-rw-r--+  1 root     bin          312 May  4 16:15 blacklist
```

To add a user account to the ACL, the user must already belong to a valid SMS group as described in the *System Management Services (SMS) 1.6 Installation Guide*.

Note – UFS attributes, such as the ACL, are supported in UFS file systems only. If you restore or copy directories with ACL entries into the `/tmp` directory, all ACL entries are lost. Instead, use the `/var/tmp` directory for temporary storage of UFS files and directories.

Network Configuration

For each network, `smsconfig` can set one or more *interface* designations within that network. By default, `smsconfig` steps through the configuration of all three internal, enterprise networks (MAN, I1, and I2).

To configure an individual network, append the *net-id* to the command line. MAN *net-id*s are designated `I1`, `I2`, and `C`.

Configure a single domain within an enterprise network by specifying both the desired domain and its *net-id*. A domain can be excluded from the I1 MAN by using the word `NONE` as the MAN *hostname*.

Note – Once you have configured or changed the configuration of the MAN network, you *must* reboot the SC for the changes to take effect.

You must have superuser privileges to run `smsconfig`. For more information and examples, refer to the *System Management Services (SMS) 1.6 Installation Guide* and `smsconfig` man page, and see "Management Network Services" on page 184.

MAN Configuration

Typing `smsconfig -m` does the following:

1. Creates `/etc/hostname.scman[01]`.
2. Creates `/etc/hostname.hme0` and `/etc/hostname.eri1` according to inputs to the external network prompts of `smsconfig`.
3. Updates `/etc/netmasks` and `/etc/hosts`.
4. Sets OpenBoot PROM variable `local-mac-address?=true` (default is false).

For more information on `smsconfig`, refer to the `smsconfig`(1M) man page and see "Management Network Services" on page 184.

APPENDIX **A**

SMS man Pages

The SMS man pages are in the *System Management Services (SMS) 1.6 Reference Manual* portion of your Sun Fire high-end system documentation set as well as online (after you have installed the SMS packages).

The following is a list of SMS man pages:

- `addboard`(1M) – Assigns, connects, and configures a board to a domain
- `addcodlicense`(1M) – Adds a Capacity on Demand (COD) right-to-use (RTU) license key to the COD license database
- `addtag`(1M) – Assigns a domain name (tag) to a domain
- `cancelcmdsync`(1M) – Removes a command synchronization descriptor from the command synchronization list
- `codd`(1M) – Capacity on Demand daemon
- `console`(1M) – Accesses the domain console
- `dca`(1M) – Domain configuration agent
- `deleteboard`(1M) – Unconfigures, disconnects, and unassigns a system board from a domain
- `deletecodlicense`(1M) – Removes a COD RTU license key from the COD license database
- `deletetag`(1M) – Removes the domain name (tag) associated with the domain
- `disablecomponent`(1M) – Adds the specified component to the blacklist
- `dsmd`(1M) – Domain status monitoring daemon
- `dxs`(1M) – Domain *X* server
- `efhd`(1M) – Error and fault handling daemon
- `elad`(1M) – Event log access daemon
- `erd`(1M) – Event reporting daemon
- `enablecomponent`(1M) – Removes the specified component from the ASR blacklist

- `esmd`(1M) – Environmental status monitoring daemon
- `flashupdate`(1M) – Updates system board PROMs
- `fomd`(1M) – Failover management daemon
- `frad`(1M) – FRU access daemon
- `help`(1M) – Displays help information for SMS commands
- `hpost`(1M) – Sun Fire high-end system power-on self test (POST) control application
- `hwad`(1M) – Hardware access daemon
- `initcmdsync`(1M) – Creates a command synchronization descriptor that identifies the script to be recovered
- `kmd`(1M) – Key management daemon
- `mand`(1M) – Management network daemon
- `mld`(1M) – Message logging daemon
- `moveboard`(1M) – Moves a system board from one domain to another
- `osd`(1M) – OpenBoot PROM server daemon
- `pcd`(1M) – Platform configuration database daemon
- `poweroff`(1M) – Controls power off
- `poweron`(1M) – Controls power on
- `rcfgadm`(1M) – Remote configuration administration
- `reset`(1M) – Sends reset to all ports (CPU or I/O) of a specified domain
- `resetsc`(1M) – Sends reset to the spare SC
- `runcmdsync`(1M) – Prepares a specified script for recovery after a failover
- `savecmdsync`(1M) – Adds a marker that identifies a location in the script from which processing can be resumed after a failover
- `setbus`(1M) – Performs dynamic bus reconfiguration on active expanders in a domain
- `setcsn`(1M) – Sets the chassis serial number for a Sun Fire high-end system
- `setdatasync`(1M) – Modifies the data propagation list used in data synchronization
- `setdate`(1M) – Sets the date and time for the system controller or a domain
- `setdefaults`(1M) – Removes all instances of a previously active domain
- `setfailover`(1M) – Modifies the state of the SC failover mechanism
- `setkeyswitch`(1M) – Changes the position of the virtual keyswitch
- `setobpparams`(1M) – Sets up OpenBoot PROM variables
- `setpcimode`(1M) – Changes the settings for the PCI-X slots on a V2HPCIX I/O board in your server

- setupplatform(1M) – Sets up the available component list for domains
- showboards(1M) – Shows the assignment information and status of the system boards
- showbus(1M) – Displays the bus configuration of expanders in active domains
- showcmdsync(1M) – Displays the current command synchronization list
- showcodlicense(1M) – Displays the current COD RTU licenses stored in the COD license database
- showcodusage(1M) – Displays the current usage statistics for COD resources
- showcomponent(1M) – Displays ASR blacklist status for a component
- showdatasync(1M) – Displays the status of SMS data synchronization for failover
- showdate(1M) – Displays the date and time for the system controller or a domain
- showdevices(1M) – Displays system board devices and resource usage information
- showenvironment(1M) – Displays the environmental data
- showfailover(1M) – Displays SC failover status or role
- showkeyswitch(1M) – Displays the position of the virtual keyswitch
- showlogs(1M) – Display message log files, the event logs, or the Event Dictionary Database
- showobpparams(1M) – Displays OpenBoot PROM bringup parameters
- showpcimode(1M) – Lists the mode settings for all the PCI-X slots on a V2HPCIX I/O board in your server
- showplatform(1M) – Displays the board available component list for domains
- showxirstate(1M) – Displays CPU dump information after sending a reset pulse to the processors
- smsbackup(1M) – Backs up the SMS environment
- smsconfig(1M) – Configures the SMS environment
- smsconnectsc(1M) – Accesses a remote SC console
- smsinstall: Installs the SMS software.
- smsrestore(1M) – Restores the SMS environment
- smsupgrade: Upgrades the existing SMS software installed on a system.
- smsversion(1M) – Displays the active version of SMS software
- ssd(1M) – SMS startup daemon
- testemail(1M) – Tests the event-reporting features, which include event message logging and email notification of events
- tmd(1M) – Task management daemon
- wcapp(1M) – wPCI application daemon

APPENDIX B

Error Messages

This section discusses user-visible error messages for SMS. The types of errors and the numerical ranges are listed. To view individual errors, you must install the SMSHelp software package (SUNWSMSjh). This section contains instructions for installing SUNWSMSjh, if it was not already installed during the SMS software installation.

Each error in SMSHelp contains the error ID, the text of the message, the meaning of the message, references for further information if applicable, and recovery action to take or suggested steps for further analysis.

This chapter includes the following sections:

- "Installing SMSHelp" on page 251
- "Types of Errors" on page 256
- "Error Categories" on page 256

Installing SMSHelp

This section explains how to manually install the SUNWSMSjh package using the standard installation utility, pkgadd.

▼ To Install the SUNWSMSjh Package

1. **Log in to the SC as superuser.**

2. **Load the** `SUNWSMSjh` **package on the server:**

```
# pkgadd -d . SUNWSMSjh
```

The software briefly displays copyright, trademark, and license information for each package. Then it displays messages about `pkgadd`(1M) actions taken to install the package, including a list of the files and directories being installed. Depending on your configuration, the following messages might be displayed:

```
This package contains scripts which will be executed
with superuser permission during the process of installing this
package.

Do you want to continue with the installation of this
package [y,n,?]
```

3. **Type** `y` **at each successive prompt to continue.**

   When this portion of the installation is complete, the `SUNWSMSjh` package has been installed and the superuser prompt is displayed.

4. **Log out as superuser.**

To Start SMS Help

1. **Log in to the SC as a user with platform or domain group privileges.**
2. **In any terminal window, type:**

```
sc0:sms-user:> smshelp &
```

The SMS help browser appears. You can resize the panes if necessary, by placing the pointer to the right of the vertical scrollbar, pressing the left mouse button, and dragging to the right.

3. **Choose an error message and note its message.code.**

   Error messages are recorded in the platform and domain logs.

   The message format follows the syslog(3) convention:

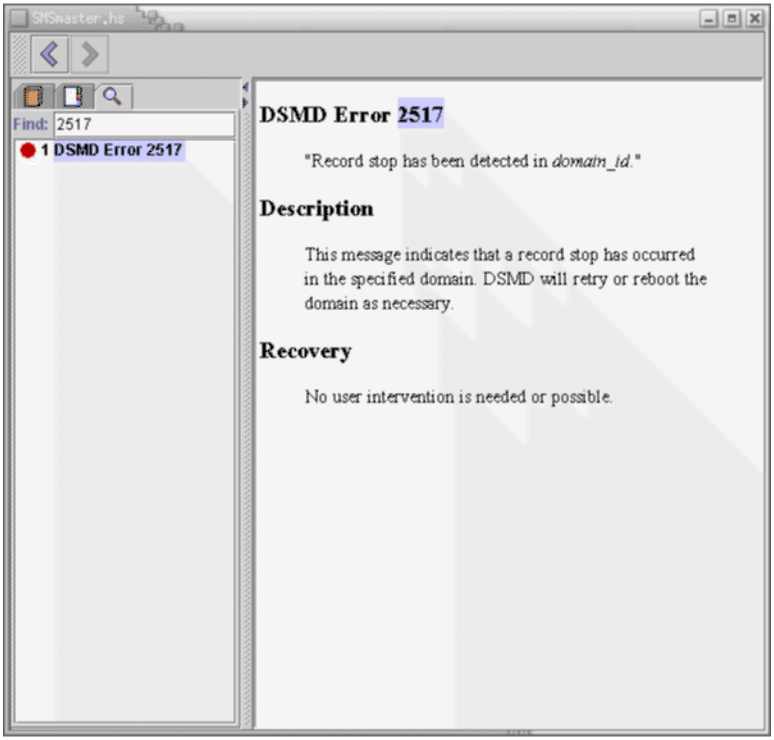
```
timestamp  host process_name  [pid]: [message_code
hight_res_timestamp level source_code_file_name
source_code_line_num] message_text
```

   For example:

```
Feb 2 18:36:14 2002 xc17-sc0 dsmd[117469]-B(): [2517
16955334989087 WARNING EventHandler.cc 121] Record stop has been
detected in domain B.
```

   Using the message_code, you can either do a quick search using the magnifying glass at the top of the browser, or you can scroll through the table of contents.

- **To do a quick search, click the magnifying glass, enter the error message number, and press Return as shown in the following example.**



DSMD Error 2517

"Record stop has been detected in *domain_id*."

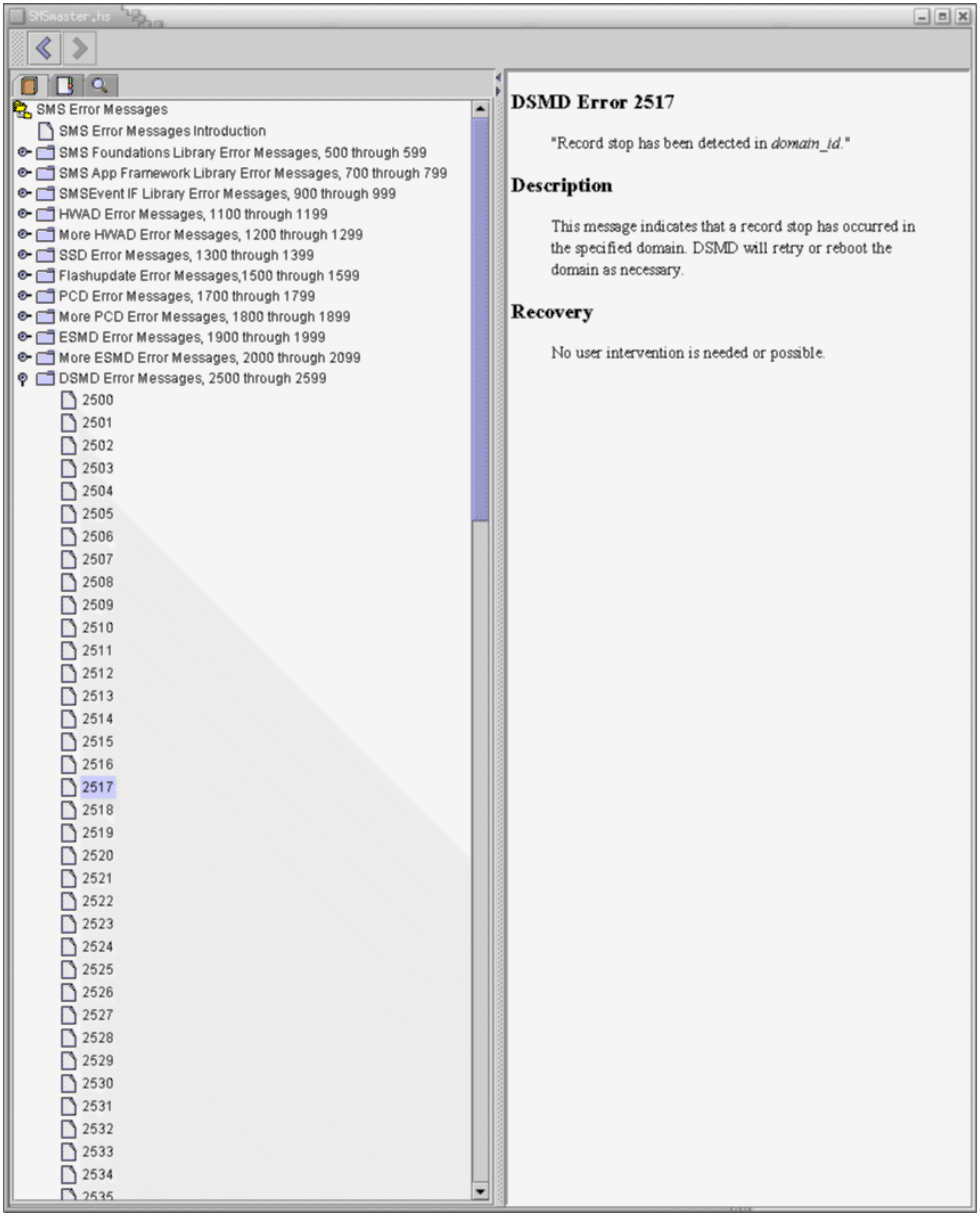
Description

This message indicates that a record stop has occurred in the specified domain. DSMD will retry or reboot the domain as necessary.

Recovery

No user intervention is needed or possible.

- **To scroll the table of contents, left-click on the message folder containing your error message, in this case, DSMD Error Messages, 2500 through 2599. Then click on error 2517, as shown in the following example.**



SMSmaster.hs
SMS Error Messages
SMS Error Messages Introduction
SMS Foundations Library Error Messages, 500 through 599
SMS App Framework Library Error Messages, 700 through 799
SMSEvent IF Library Error Messages, 900 through 999
HWAD Error Messages, 1100 through 1199
More HWAD Error Messages, 1200 through 1299
SSD Error Messages, 1300 through 1399
Flashupdate Error Messages,1500 through 1599
PCD Error Messages, 1700 through 1799
More PCD Error Messages, 1800 through 1899
ESMD Error Messages, 1900 through 1999
More ESMD Error Messages, 2000 through 2099
DSMD Error Messages, 2500 through 2599
2500
2501
2502
2503
2504
2505
2506
2507
2508
2509
2510
2511
2512
2513
2514
2515
2516
2517
2518
2519
2520
2521
2522
2523
2524
2525
2526
2527
2528
2529
2530
2531
2532
2533
2534
2535

DSMD Error 2517

"Record stop has been detected in *domain_id*."

Description

This message indicates that a record stop has occurred in the specified domain. DSMD will retry or reboot the domain as necessary.

Recovery

No user intervention is needed or possible.

Types of Errors

This section describes the six types of errors reflected in the error messages in SMS Help (TABLE B-1).

TABLE B-1 Error Types

Error	Description
EMERG	Panic conditions that would normally be broadcast to all users
ALERT	Conditions that should be corrected immediately, such as a corrupted system database
CRIT	Warnings about critical conditions, such as hard device failures
ERROR	All other errors
WARNING	Warning messages
NOTICE	Conditions that are not error conditions but might require special handling

Error Categories

TABLE B-2 shows the different error categories in SMS. Nonsequential numbering is due to error messages reserved for internal or service use.

TABLE B-2 Error Categories

Error Numbers	Message Group
0-499	Reserved for DEBUG, INFO and POST messages
500-699	Reserved for SMS Foundation Library messages
700-899	Reserved for SMS Application Framework messages
900-1099	Reserved for SMSEvent IF Library messages
1100-1299	Reserved for `HWAD` daemon and library messages
1300-1499	Reserved for `ssd` messages
1500-1699	Reserved for `flashupdate` messages
1700-1899	Reserved for `pcd` messages

TABLE B-2 Error Categories *(Continued)*

Error Numbers	Message Group
1900-2099	Reserved for `esmd` messages
2500-2699	Reserved for `dsmd` messages
2700-2899	Reserved for `addtag` messages
2900-3099	Reserved for `deletetag` messages
3100-3299	Reserved for Permissions messages
3300-3499	Reserved for *domain-tag* messages
3500-3699	Reserved for `addboard` messages
3700-3899	Reserved for `tmd` messages
4100-4299	Reserved for `showkeyswitch` messages
4300-4499	Reserved for `dca` messages
4500-4699	Reserved for `libscdr` plugin messages
4700-4899	Reserved for `osd` messages
4900-5099	Reserved for `dxs` messages
5100-5299	Reserved for `deleteboard` messages
5300-5499	Reserved for `setkeyswitch` messages
5500-5699	Reserved for `libdrcmd` messages
5700-5899	Reserved for `moveboard` messages
5900-6099	Reserved for `setupplatform` messages
6100-6299	Reserved for `power` command messages
6300-6499	Reserved for `xir` library messages
6500-6699	Reserved for `showplatform` messages
6700-6899	Reserved for `help` messages
6900-7099	Reserved for `reset` messages
7100-7299	Reserved for `showboards` messages
7300-7499	Reserved for `libshowboards` messages
7500-7699	Reserved for `autolock` messages
7700-7899	Reserved for `mand` messages
7900-8099	Reserved for `showenvironment` messages
8100-8299	Reserved for `resetsc` messages
8300-8499	Reserved for dynamic bus reconfiguration messages

TABLE B-2 Error Categories *(Continued)*

Error Numbers	Message Group
8500-8699	Reserved for `fomd` messages
8700-8899	Reserved for `kmd` messages
8900-9099	Reserved for `setdefaults` messages
9100-9299	Reserved for `mld` messages
9300-9499	Reserved for `showdevices` messages
9500-9699	Reserved for `showxirstate` messages
9700-9899	Reserved for `COD` messages
9900-10000	Reserved for `frad` messages
10100-10299	Reserved for `fruevent` messages
10300-10499	Reserved for `smsconnectsc` messages
10700-10899	Reserved for `EFE` messages
11100-11299	Reserved for `rcfgadm` messages
11300-11499	Reserved for `datasync` messages
11500-11699	Reserved for EFHD messages
11700-11899	Reserved for ELAD messages
11900-12099	Reserved for ERD messages
12100-12299	Reserved for Event Utilities messages
12300-12499	Reserved for Wcapp messages
12500-12699	Reserved for FRUID-related messages
12700-12799	Reserved for EBD error messages
50000-50099	Reserved for SMS generic messages

Glossary

A

ACL See *access control list (ACL)*.

access control list (ACL) The access control list (ACL) contains information about file and folder permissions on your system. Using an ACL enables you to define file or folder permissions for the owner, owner's group, others, and specific users and groups, and default permissions for each of these categories.

active board A board is considered active when it is in the `connected/unconfigured` state.

active board list List of boards that are in use in a domain. The `pcd`(1M) daemon keeps the state of this list.

active domain A domain running operating system (OS) software.

automatic diagnosis (AD) A software engine which is invoked when an error occurs, it then records diagnosis information as part of a FRU's component health status (CHS), which is stored in the FRUID of each component. In some instances an auto-restoration process is started and POST is re-run.

ADR See *Automated dynamic reconfiguration (ADR)*.

application-specific integrated circuit (ASIC) In the Sun Fire high-end systems, any of the large main chips in the design, including the UltraSPARC processor and data buffer chips.

arbitration stop A condition that occurs when one of the Product Name ASICs detects a parity error or equivalent fatal system error. Bus arbitration is frozen, so all bus activity stops.

ASIC See *application-specific integrated circuit (ASIC)*.

assigned board list List of components that have been assigned to a domain by a domain administrator/configurator privileged user. The `pcd`(1M) daemon keeps the state of this list.

ASR Automatic System Recovery.

auto-failover The process by which the SMS daemon, `fomd`, automatically switches SC control from the main SC to the spare in the event of hardware or software failure on the main.

Automated dynamic reconfiguration (ADR) The dynamic reconfiguration of system boards accomplished through commands that can be used to automatically `assign/unassign`, `connect/disconnect` and `configure/unconfigure` boards, and obtain board status information. You can run these commands interactively or in shell scripts.

automatic diagnosis engine A software feature that identifies hardware errors that affect the availability of a platform and its domains.

automatic system recovery (ASR) Procedures that restore the system to running all properly configured domains after one or more domains have been rendered inactive due to software or hardware failures or due to unacceptable environmental conditions.

available component list List of available components that can be assigned to a domain by a domain administrator/configurator privileged user. The `pcd`(1M) daemon keeps the state of this list. `setupplatform`(1M) updates it.

AXQ An ASIC located on the expander board in a Sun Fire high-end system.

B

BBC Boot bus controller. An ASIC used on the CPU & I/O boards (also system controller boards), that connects the boot bus to the PROM bus and the console bus.

BBSRAM See *boot bus SRAM (BBSRAM)*.

blacklist A text file that `hpost`(1M) reads when it starts up. The blacklist file specifies the Sun Fire high-end system components that are not to be used or configured into the system. Platform and domain blacklist files can be edited using the `enablecomponent` and `disablecomponent` commands. The ASR blacklist is created and edited by `esmd`.

boot bus A slow-speed, byte-wide bus controlled by the processor port controller ASICs, used for running diagnostics and boot code. UltraSPARC starts running code from boot bus when it exits reset. In the Product Name, the only component on the boot bus is the `BBSRAM`.

boot bus SRAM (BBSRAM) A 256-Kbyte static RAM attached to each processor PC ASIC. Through the PC, it can be accessed for reading and writing from JTAG or the processor. Boot bus SRAM is downloaded at various times with `hpost`(1M) and OpenBoot PROM startup code, and provides shared data between the downloaded code and the SC.

C

Capacity on Demand (COD) An option that provides additional processing resources (CPUs) provided on COD system boards that are installed on Sun Fire high-end systems. You can access the COD CPUs after you purchase the COD right-to-use (RTU) licenses for them.

cacheable address slice map (CASM) A table in the AXQ that directs cacheable addresses to the correct expander.

CASM See *cacheable address slice map (CASM)*.

Chassis HostID The serial number of the centerplane. This number is used only by the COD feature to identify the platform for COD licensing purposes.

chassis serial number A serial number that identifies a Sun Fire high-end system. The chassis serial number is printed on a label located on the front of the system chassis, near the bottom center. This number is used by your service provider to correlate hardware error events and service actions to the appropriate system.

checkpoint data A copy of the state an SC client is in at a specific execution point. Checkpoint data is periodically saved to disk.

CLI Command-line interface.

cluster A cooperative collection of interconnected computer systems, each running a separate OS image, utilized as a single, unified computing resource.

CHS Component health status.

community A customer site IP network that is physically separate from any other networks.

community name A string identifier that names a particular community. In the context of External Network Monitoring for a Sun Fire high-end system, it is used as the interface group name. See *interface group name*.

CMR Coherent Memory Replication.

cmdsync Command synchronization. Commands that work together to control recovery during SC failover. For example, `cancelcmdsync`, `initcmdsync`, and `savecmdsync`.

CPU Central processing unit.

CSB Centerplane support board

D

DARB An ASIC on the Product Name centerplane that handles data arbitration.

DARB interrupt An interrupt of the SC processor initiated by a signal from either or both DARB ASIC on the Sun Fire high-end system centerplane. DARB asserts this interrupt signal in response to three kinds of events: Dstops, Recordstops, and non-error requests for attention initiated by domain processors writing to a system register in the AXQ ASIC.

DE Diagnosis engine.

DCU See *domain configuration unit (DCU)*.

DHCP Dynamic Host Configuration Protocol.

DIMM See *dual inline memory module (DIMM)*.

DR Dynamic reconfiguration.

DSD Dynamic system domain.

dstop See *domain stop*.

disk array A collection of disks within a hardware peripheral. The disk array provides access to each of its housed disks through one or two Fibre Channel modules.

disk array controller A controller that resides on the host system and has one or two Fibre Channel modules.

disk array port A Fibre Channel module that can be connected to a disk array controller that is serviced by a driver pair; for example, `soc/pln` for SSAs.

domain A set of one or more system boards that acts as a separate system capable of booting the OS and running independently of any other domains. A machine environment capable of running its own OS. There are up to 18 domains available on the Sun Fire high-end system. Domains that share a system are characteristically independent of each other.

domain configuration unit (DCU) A unit of hardware that can be assigned to a single domain. Domains are configured from DCUs. CPU/Memory, PCI I/O, hsPCI I/O, and hsPCI+ I/O are DCUs. `csb`, `exb` boards, and the SC are not.

`domain-id` Domain ID of a domain.

`domain-tag` Domain name assigned using `addtag` (1M).

domain stop An uncorrectable hardware error that immediately terminates the affected domain.

DR See *dynamic reconfiguration (DR)*.

DRAM See *dynamic RAM(DRAM)*.

drift file The file used to record the drift (or frequency error) value computed by `xntpd`. The most common name is `ntp.drift`.

DSD Dynamic System Domain. See *domain*.

dual inline memory module (DIMM) A small printed circuit card containing memory chips and some support logic.

dynamic reconfiguration (DR) The ability to logically attach and detach system boards to and from the operating system without causing machine downtime. DR can be used in conjunction with hot-swap, which is the process of physically removing or inserting a system board. You can use DR to add a new system board, reinstall a repaired system board, or modify the domain configuration on the Sun Fire system.

dynamic RAM(DRAM) Hardware memory chips that require periodic rewriting to retain their contents. This process is called "refresh." In a Sun Fire high-end system, DRAM is used only on main memory SIMMs and on the control boards.

E

ECC Error Correction Code.

Ecache See *external cache (Ecache)*.

EEPROM Electrically Erasable Programmable Read-Only Memory.

Environmental Monitoring Systems have a large number of sensors that monitor temperature, voltage, and current. The SC daemons `esmd` and `dsmd` poll devices in a timely manner and make the environmental data available. The SC shuts down various components to prevent damage.

Ethernet address A unique number assigned to each Ethernet network adapter. It is a 48-bit number maintained by the IEEE. Hardware vendors obtain blocks of numbers that they can build into their cards. See also, *MAC address*.

external cache (Ecache) An 8-Mbyte synchronous static RAM second-level cache local to each processor module. Used for both code and data. This is a direct-mapped cache.

external network A network that requires a physical cable to connect a node to the network. In the context of a Sun Fire high-end system, it is the set of networks connected to the RJ45 jacks located on the front of each Sun Fire high-end system. See *external network interface*.

external network interface One of the RJ45 jacks located on the front of each Sun Fire high-end System Controller.

F

Fibre Channel module An optical link connection (OLC) module on a disk array controller that can be connected to a disk array port.

Fireplane Centerplane in the Sun Fire high-end system.

FPROM Flash programmable read-only memory.

FRU Field replaceable unit.

FRUID Field replaceable unit identification

G

GDCD See *global domain configuration descriptor (GDCD)*.

global domain configuration descriptor (GDCD) The description of the single configuration that `hpost`(1M) chooses. It is part of the structure handed off to OpenBoot PROM.

GUI Graphical user interface.

H

HA High availability.

HASRAM High availability SRAM.

headroom See *instant access CPUs*.

heartbeat interrupt Interruption of the normal Solaris OS indicator, readable from the SC. Absence of heartbeat updates for a running Solaris system usually indicates a Solaris hang.

hpost Host POST is the POST code that is executed by the SC. Typically this code is sourced from the SC local disk.

HPCI Hot-pluggable PCI I/O board.

HPU Hot-Pluggable Unit. A hardware component that can be isolated from a running system such that it can be cleanly removed from the system or added to the system without damaging any hardware or software.

HsPCI See *HPCI*.

I

I1 Network There are 18 network interfaces (NICs) on each SC. These are connected in a point-to-point fashion to NICs located on each of the expander I/O slots on the Sun Fire high-end system. All of these point-to-point links are collectively called the I1 network.

I^2C Inter-IC Bus. This two-wire bus is used throughout various systems to run LEDs, set system clock resources, read `thermcal` information, and so on.

I2 Network An internal network between the two system controllers consisting of two NICs per system controller. It is not a private network, and it is entirely separate from the I1 network.

IDPROM Identification PROM. Contains information specific to the Product Name internal machine, such as machine type, manufacturing date, Ethernet address, serial number, and host ID.

instant access CPUs Unlicensed COD CPUs on COD system boards installed in Sun Fire high-end systems. You can access up to a maximum of eight COD CPUs for immediate use while you are purchasing the COD right-to-use (RTU) licenses for the COD CPUs. Also referred to as *headroom*.

interface group A group of network interfaces that attach to the same community.

interface group name A string identifier that names a particular interface group. In the context of External Network Monitoring for Sun Fire high-end system, it is the name associated with a particular community.

ioctl A system call that performs a variety of control functions on devices and STREAMS. For non-STREAMS, the functions performed by this call are device-specific control functions.

IP Internet Protocol.

IP link A communication medium over which nodes communicate at the link layer. The link layer is the layer immediately below IPv4/IPv6. Examples include Ethernets (simple or bridged) or ATM networks.

IPv4 Internet Protocol version 4.

IPv6 Internet Protocol version 6. IPv6 increases the address space from 32 to 128 bits. It is backwards compatible with IPv4.

IOSRAM Input-Output Static Random-Access Memory.

IPMP IP Network Multipathing. Solaris software that provides load spreading and failover for multiple network interface cards connected to the same IP link, for example, Ethernet.

J

JTAG A serial scan interface specified by IEEE standard 1149.1. The name comes from Joint Test Action Group, which initially designed it.

JTAG+ An extension of JTAG, developed by Sun Microsystems Inc., which adds a control line to signal that board and ring addresses are being shifted on the serial data line. Often referred to simply as JTAG.

K

`kadb` An interactive kernel debugger with a user interface. For more information, refer to the `kadb`(1M) Solaris man page.

L

LAN Local area network.

LCD Liquid crystal display.

LED Light emitting diode.

LSF Load sharing facility.

M

MAC address Worldwide unique serial number assigned to a network interface. IEEE controls the distribution of MAC addresses. See also *Ethernet address*.

mailbox See *Mbox*.

MAN SMS Management Network.

MaxCPU Dual CPU board.

Mbox Message-passing mechanism between SMS software on the SC and OpenBoot PROM and the Solaris OS on the domain.

MIB Management Information Base.

metadisk A disk abstraction that provides access to an underlying group of two physical paths to a disk.

metanetwork A network abstraction that provides access to an underlying group of two physical paths to a network.

N

network interface card (NIC) Network adapter which is either internal or a separate card that serves as an interface to an IP link.

network time protocol (NTP) Network Time Protocol. Supports synchronization of Solaris time with the time service provided by a remote host.

NFS Network file system.

NIC See *network interface card (NIC)*.

NIS+ Network Information Service Plus. A secure, hierarchical network naming service.

no-domain Describes the state of a board (DCU) that is not assigned to any domain.

NTP See *network time protocol (NTP)*.

NVRAM Non-volatile read-only memory.

O

OBP See *OpenBoot PROM*.

OpenBoot PROM A layer of software that takes control of the configured Product Name from `hpost`(1M), builds some data structures in memory, and boots the operating system. IEEE 1275-compliant OpenBoot PROM.

OS Operating system.

OSR Operating system resource.

P

path group A set of two alternate paths that provide access to the same device or set of devices.

PCB Printed circuit board.

physical path The electrical path from the host to a disk or network.

platform A single physical computer.

POR Power-on-reset.

POST See *power-on self-test (POST)*.

power-on self-test (POST) A test performed by `hpost`(1M). This program takes uninitialized Product Name hardware and probes and tests its components, configures what seems worthwhile into a coherent initialized system, and hands it off to OpenBoot PROM. In the Product Name POST is implemented in a hierarchical manner with the following components: `lpost`, `spost`, and `hpost`.

PROM Programmable Read Only Memory.

R

RASS Reliability, availability, serviceability, and security.

RAM Random access memory.

RARP Reverse Address Resolution Protocol.

rstop See *Record Stop*.

Record Stop A correctable data transmission error.

RPC Remote procedure call.

RTU Right to use.

S

SA Security association.

SBBC See *BBC*.

SC System controller. The Nordica board that assists in monitoring or controlling the system.

SEEPROM Serial EEPROM.

SMP Symmetric multi-processor.

SMS System Management Services software. The software that runs on the Product Name SC and provides control/monitoring functions for the Product Name platform.

SNMP Simple Network Management Protocol.

Solaris OS Solaris operating system.

split-brain condition When both SCs think they are the main SC.

SRAM See *static RAM (SRAM)*.

SRS Sun remote services.

SST Solaris security toolkit.

static RAM (SRAM) Memory chips that retain their contents as long as power is maintained.

System Board For next-generation Sun Fire servers, there are five types of system boards, four of which can be found in the Sun Fire high-end system. The system boards are the system board, the I/O board, the WCI board, the Product Name PCI controller board, and the Product Name compact PCI controller board.

T

TCP/IP Transmission Control Protocol/Internet Protocol.

TOD Time of day.

tunnel switch The process of moving the SC/Domain communications tunnel from one I/O board to another in a domain. Typically occurring when the I/O board with the tunnel is being dynamically reconfigured out.

U

URL Uniform Resource Locator.

UltraSPARC The UltraSPARC processor is the processor module used in the Sun Fire high-end system.

V

virtual keyswitch The SC provides a virtual keyswitch for each domain which controls the bringup process for each domain. The `setkeyswitch`(1M) command controls the position of the virtual keyswitch for each domain. Possible positions are: `on`, `off`, `standby`, `diag`, and `secure`.

VCMON Voltage core (CPU) monitoring

VM Volume manager (Veritas)

W

wPCI Sun Fire Link I/O board.

X

XIR eXternally Initiated Reset. Sends a “soft” reset to the CPU in a domain. It does not reboot the domain. After receiving the reset, the CPU drops to the OpenBoot PROM prompt.

Index