Sun HPC ClusterTools™ 5 Software Administrator's Guide

Sun Microsystems, Inc.
4150 Network Circle
Santa Clara, CA 95054 U.S.A.
650-960-1300

Part No. 817-0083-10
February 2003, Revision A

Send comments about this document to: docfeedback@sun.com

Contents

Preface

The *Sun HPC ClusterTools™ Software Administrator's Guide* explains how to configure and manage a cluster of one or more Sun™ servers running the Sun Cluster Runtime Environment (CRE) job-management software. These instructions are designed for an experienced system administrator with networking knowledge.

Several distributed resource management packages can be used with Sun CRE to provide effective resource management and utilization accounting in conjunction with the Sun MPI parallel applications:

- Sun Grid Engine SGE Version 5.3 and Sun Grid Engine Enterprise Edition SGEEE Version 5.3
- Load Sharing Facility Version 4.*x* of Platform Computing
- Portable Batch System (PBS) PBS Pro 5.*x*.*x* of Veridian

For information about these packages, see the documentation provided with each respective package.

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 one or more of the following for this information:

- *Solaris Handbook for Sun Peripherals*
- AnswerBook2™ online documentation for the Solaris™ operating environment
- Other software documentation that you received with your system

Typographic Conventions

Typeface*	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. You *must* be superuser to do this. To delete a file, type `rm` *filename*.

* The settings on your browser might differ from these settings.

Shell Prompts

Shell	Prompt
C shell	*machine-name*%
C shell superuser	*machine-name*#
Bourne shell and Korn shell	$
Bourne shell and Korn shell superuser	#

Related Documentation

Application	Title	Part Number
Sun HPC ClusterTools software documentation	*Read Me First: Guide to Sun HPC ClusterTools Software Documentation*	817-0096-10
Sun HPC ClusterTools software	*Sun HPC ClusterTools 5 Software Product Notes*	817-0081-10 817-0082-10
	Sun HPC ClusterTools 5 Software Installation Guide	817-0090-10 817-0084-10
	Sun HPC ClusterTools 5 Software Performance Guide	
	Sun HPC ClusterTools 5 Software User's Guide	
Sun MPI programming	*Sun MPI 6.0 Software Programming and Reference Guide*	817-0085-10
Sun S3L software	*Sun S3L 4.0 Software Programming Guide*	817-0086-10
	Sun S3L 4.0 Software Reference Manual	817-0087-10
Prism™ graphical programming environment	*Prism 7.0 Software User's Guide*	817-0088-10
	Prism 7.0 Software Reference Manual	817-0089-10

Accessing Sun Documentation

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

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

Sun Welcomes Your Comments

Sun is interested in improving its documentation and welcomes your comments and suggestions. You can email your comments to Sun at:

docfeedback@sun.com

Please include the part number (817-0083-10) of your document in the subject line of your email.

CHAPTER 1

Introduction

The Sun Cluster Runtime Environment (CRE) is a program execution environment that provides basic job launching and load-balancing capabilities.

This manual provides information needed to administer Sun HPC clusters on which MPI programs run under Sun CRE or any other resource manager. The topics covered are organized in the following manner:

- Chapter 2 provides quick-start instructions for getting an MPI job running on a Sun HPC cluster with newly installed Sun HPC ClusterTools software.
- Chapter 3 provides an introduction to configuring a Sun HPC cluster, using both the administration command interface, `mpadmin`, and the cluster configuration file, `hpc.conf`.
- Chapter 4 provides general information about cluster configuration, and an overview of how to run parallel programs in a batch processing environment.
- Chapter 5 provides a more detailed description of the `mpadmin` command.
- Chapter 6 provides a more comprehensive description of the `hpc.conf` configuration file.
- Chapter 7 provides guidelines for performing routine maintenance and for recognizing and troubleshooting error conditions.
- Appendix A describes the Cluster Console Manager (CCM), a set of cluster administration tools.

The balance of this chapter provides an overview of the Sun HPC ClusterTools software and the Sun HPC cluster hardware on which it runs.

Sun HPC Clusters

A Sun HPC hardware configuration can be a single Sun SMP (symmetric multiprocessor) server or multiple SMPs interconnected into a cluster. Sun HPC ClusterTools software supports parallel jobs of up to 2048 processes per job running on clusters of up to 256 nodes.

Note – An individual SMP server within a Sun HPC cluster is referred to as a *node*.

Sun HPC clusters can also be built using any Sun-supported TCP/IP interconnect, such as Ethernet, high-speed Ethernet, Gigabit Ethernet, ATM OC-3, ATM OC-12, FDDI, and HIPPI.

Cluster Runtime Environment Daemons

Sun CRE comprises two sets of daemons—the master daemons and the nodal daemons. These two sets of daemons work cooperatively to maintain the state of the cluster and manage program execution.

The master daemons consist of the daemons `tm.rdb`, `tm.mpmd`, and `tm.watchd`. They run on one node exclusively, which is called the *master node*. There are two nodal daemons, `tm.omd` and `tm.spmd`. They run on all the nodes.

Sun HPC ClusterTools Software

Sun HPC ClusterTools software is an integrated suite of parallel development tools that extend Sun's network computing solutions to high-end distributed-memory applications.

Sun HPC ClusterTools components run under the Solaris 8 (32-bit or 64-bit) and Solaris 9 Operating Environments.

Sun CRE's Integration With Batch Processing Systems

The Sun CRE environment provides close integration with several batch processing systems, also known as *distributed resource managers* (DRM). You can launch parallel jobs from a batch system to control resource allocation, and continue to use Sun CRE to monitor job status. The currently supported distributed resource managers are:

- Sun Grid Engine SGE Version 5.3 and Sun Grid Engine Enterprise Edition SGEEE Version 5.3.
- Load Sharing Facility Version 4.x of Platform Computing.
- Portable Batch System (PBS) PBS Pro 5.x.x of Veridian.

To launch a parallel job through a batch processing system, follow these general guidelines:

- First enter the batch processing environment, whether LSF, PBS, or SGE.
- Reserve resources for the parallel job and set other job control parameters.
- Invoke the `mprun` command to launch the job.

You can launch the parallel job either through a script or interactively. For details, see "Close Integration With Batch Processing Systems" on page 45.

The architecture selected to implement close integration can easily accomodate new resource managers. For this purpose it provides a Sun CRE wrapper library and a resource manager plugin interface. Both are described in the *Sun MPI Software Programming and Reference Manual*.

Sun MPI and MPI I/O

Sun MPI is a highly optimized version of the Message-Passing Interface (MPI) communications library. Sun MPI implements the MPI 2 standard. In addition, Sun MPI provides extensions such as support for multithreaded programming, MPI I/O support for parallel file I/O, and others as detailed in the Sun MPI documentation.

Sun MPI provides full F77, C, and C++ support and basic F90 support.

Loadable Protocol Modules

The Sun MPI library is capable of providing high-performance communications over several different protocols. Sun HPC ClusterTools software makes three protocols available to MPI programs: Shared Memory (SHM), Transport Control Protocol (TCP), and Remote Shared Memory (RSM).

Protocols are provided as dynamically loaded library modules, separate from the MPI library. The cluster administrator determines which protocols are available on a cluster and their relative priorities. The user need not be concerned with the details of any protocol underlying MPI communications.

Prism Environment

The Prism™ graphical programming environment allows you to develop, execute, debug, and visualize data in message-passing programs.

The Prism environment can be used with applications written in F77, F90, C, and C++.

Sun S3L

The Sun Scalable Scientific Subroutine Library (Sun S3L) provides a set of parallel and scalable functions and tools that are used widely in scientific and engineering computing. It is built on top of MPI.

Sun S3L routines can be called from applications written in F77, F90, C, and C++.

Related Tools

Sun HPC ClusterTools software provides or makes use of several related tools, including Sun compilers and the Cluster Console Manager.

Sun Compilers

Sun HPC ClusterTools software supports Forte Developer 6, update 2 and Sun One Studio 7 Compiler Collection for C, C++, and Fortran compilers.

Cluster Console Manager

The Cluster Console Manager is a suite of applications (`cconsole`, `ctelnet`, and `crlogin`) that simplify cluster administration by enabling you to initiate commands on all nodes in the cluster simultaneously. Any command entered in the CCM's master window is broadcast to all the nodes in the cluster.

These applications are described in Appendix A of this manual.

CHAPTER 2

Getting Started

This chapter introduces Sun CRE and the basic procedures required to get a Sun HPC cluster ready for use. These basic procedures include starting the Sun CRE daemons and testing the cluster's readiness. This chapter also describes the procedure for shutting down Sun CRE.

The topics covered in this chapter include:

This chapter assumes that the Sun HPC ClusterTools software, including Sun CRE, has been correctly installed and configured, as described in the *Sun HPC ClusterTools Software Installation Guide.*

Notice that the system verification procedures outlined here are identical to the "post-install" procedures recommended for Sun CRE-based clusters in the *Sun HPC ClusterTools Software Installation Guide.*

Fundamental Sun CRE Concepts

This section introduces some important concepts that you should understand in order to administer the Sun HPC ClusterTools software with Sun CRE.

Cluster of Nodes

As its name implies, the Sun Cluster Runtime Environment is intended to operate in a Sun HPC cluster—that is, in a collection of Sun symmetric multiprocessor (SMP) servers that are connected by any Sun-supported TCP/IP-capable interconnect. An SMP attached to the cluster network is referred to as a *node*.

Sun CRE manages the launching and execution of both serial and parallel jobs on the cluster nodes, which are grouped into logical sets called *partitions*. (See the next section for more information about partitions.) For serial jobs, its chief contribution is to perform load-balancing in shared partitions, where multiple processes may be competing for the same node resources. For parallel jobs, Sun CRE provides:

- A single job-monitoring and control point
- Load-balancing for shared partitions
- Information about node connectivity
- Support for spawning of MPI processes
- Support for Prism interaction with parallel jobs

Note – A *cluster* can consist of a single Sun SMP server. However, executing MPI jobs on even a single-node cluster requires Sun CRE to be running on that cluster.

Security

A Sun HPC cluster may be protected from unauthorized use by means of the standard Solaris authentication AUTH_SYS or by installing a third-party authentication product. Sun HPC ClusterTools software supports both Kerberos Version 5 and Data Encryption System (DES).

In addition to one (or none) of the above authentication methods, Sun CRE provides basic security by means of a cluster password. It operates by checking the credentials of programs that request access.

The system administrator establishes the cluster password on each node of the cluster and on any outside nodes that may access the cluster. The password should be customized immediately after installing Sun HPC ClusterTools software, as described in the installation instructions.

Partitions

The system administrator configures the nodes in a Sun HPC cluster into one or more logical sets, called *partitions.* A job is always launched on a predefined partition that is currently *enabled*, or accepting jobs. A job will run on one or more nodes in that partition, but not on nodes in any other enabled partition.

Note – The CPUs in certain Sun high-end servers, such as the Sun Fire™ 15K server, can be configured into logical "nodes," referred to as *domains*. These domains can be logically grouped to form partitions, which Sun CRE uses in the same way it deals with partitions containing other types of Sun HPC nodes.

Partitioning a cluster allows multiple jobs to execute concurrently, without the risk that jobs on different partitions will interfere with each other. This ability to isolate jobs can be beneficial in various ways. For example:

- If one job requires exclusive use of a set of nodes but other jobs need to execute at the same time, the availability of two partitions in a cluster allows both needs to be satisfied.
- If a cluster contains a mix of nodes whose characteristics differ—such as having different memory sizes, CPU counts, or levels of I/O support—the nodes can be grouped into partitions that have similar resources. Jobs that require particular resources can then be run on suitable partitions, while jobs that are less resource-dependent can be relegated to less specialized partitions.

The system administrator can selectively enable and disable partitions. Jobs can be executed only on enabled partitions. This restriction makes it possible to define many partitions in a cluster but have only a few active at any one time.

In addition to enabling and disabling partitions, the system administrator can set and unset other partition attributes that influence various aspects of how the partition functions.

It is possible for nodes in a cluster not to belong to a currently enabled partition. If a user logs in to one of these "independent" nodes and does not request a particular partition for a job, Sun CRE launches that user's job on the cluster's *default partition*. It is also possible for a node to belong to more than one partition, so long as only one is enabled at a time.

Note – Although a job cannot be run across partition boundaries, it can be run on a partition plus independent nodes. See the *Sun HPC ClusterTools Software User's Guide* for information.

Load Balancing

Sun CRE load-balances programs that execute in partitions where multiple jobs are running concurrently.

When a user launches a job in such a shared partition, Sun CRE first determines what criteria (if any) have been specified for the node or nodes on which that program is to run. It then determines which nodes within the partition meet these criteria. If more nodes meet the criteria than are required to run the program, Sun CRE starts the program on the node or nodes that are least loaded. It examines the one-minute load averages of the nodes and ranks them accordingly.

Jobs and Processes

When a serial program executes on a Sun HPC cluster, it becomes a Solaris process with a Solaris *process ID*, or *pid*.

When Sun CRE executes a distributed message-passing program it spawns multiple Solaris processes, each with its own pid.

Sun CRE also assigns a *job ID*, or *jid*, to the program. If it is an MPI job, the jid applies to the overall job. Job IDs always begin with a `j` to distinguish them from pids. Many Sun CRE commands take jids as arguments. For example, you can issue an `mpkill` command with a signal number or name and a jid argument to send the specified signal to all processes that make up the job specified by the jid.

Communication Protocols

The communication protocol modules to be used by MPI jobs are loaded at job startup. Sun HPC ClusterTools software is provided with a default configuration of the protocols SHM, TCP, and RSM, along with their relative preference rankings. The cluster administrator need not take any action to make protocols available. (The default configuration can be changed by editing the Sun HPC ClusterTools software configuration file `hpc.conf`.)

Activating the Sun HPC ClusterTools Software

The Sun HPC ClusterTools software must be activated before it can be used. You can activate the software automatically as part of the installation process, Or you can activate it later as a separate operation. You must also reactivate the software after a system shutdown.

As root, you can use either a GUI-based wizard or command-line tools to install and activate the Sun HPC ClusterTools software. The wizard and CLI tools can also be used to deactivate or remove the software. The CLI provides two additional commands for starting and stopping the Sun CRE daemons. TABLE 2-1 lists the various GUI and CLI tools.

TABLE 2-1 Sun HPC ClusterTools Software Installation Utilities

Interface	Role
GUI Utility	
`ctgui`	Graphical interface to a set of Java-based wizards, which are used to install, remove, activate, and deactivate the software on cluster nodes.
CLI Utilities	
`ctinstall`	Install software on cluster nodes.
`ctremove`	Remove software from cluster nodes.
`ctact`	Activate software.
`ctdeact`	Deactivate softare.
`ctstartd`	Start all Sun CRE daemons.
`ctstopd`	Stop all Sun CRE daemons.
`ctnfssvr`	Set up Sun HPC ClusterTools software on an NFS server.

These commands are described fully in the *Sun HPC ClusterTools Software Installation Guide* as well as in their respective man pages. Examples of the software activation and deactivation CLI commands are provided below.

Activating Specified Nodes From a Central Host

This section shows an example of software activation in which the `ctact` command is initiated from a central host.

CODE EXAMPLE 2-1

```
# ./ctact –n node1,node2 –r rsh –m node2 –k /tmp/cluster-logs –g
```

CODE EXAMPLE 2-1 activates the software on `node1` and `node2` and specifies `node2` as the master node. It uses the options `–k` and `–g` to gather log information centrally and to generate pass and fail node lists. The remote connection method is rsh.

Activating the Local Node

This section shows an example of software activation on the local node.

CODE EXAMPLE 2-2

```
# ./ctact –l –m node2
```

CODE EXAMPLE 2-2 activates the software on the local node and specifies `node2` as the master node.

Verifying Basic Functionality

To test the cluster's ability to perform basic operations, you should check that all daemons are running, set the cluster password (unless it was already set at install time), create a default partition, and run a simple job. This section explains how to perform these steps.

Note – You need to have `/opt/SUNWhpc/bin` in your path for many of the following procedures.

Check That Nodes Are Up

Run `mpinfo –N` to display information about the cluster nodes. The following is an example of `mpinfo –N` output for a two-node system:

```
# mpinfo -N
NAME   UP  PARTITION  OS     OSREL  NCPU  FMEM   FSWP   LOAD1  LOAD5  LOAD15
host1  y   -          SunOS  5.8    1      7.17  74.76  0.03   0.04   0.05
host2  y   -          SunOS  5.8    1     34.70  38.09  0.06   0.02   0.02
```

If any nodes are missing from the list or do not have a `y` (yes) entry in the `UP` column, restart their nodal daemons as described in “Activating the Sun HPC ClusterTools Software” on page 11.

Create a Default Partition

A partition is a logical group of nodes that cooperate in executing an MPI program. You can create a cluster-wide partition by running the initialization script named `part_initialize` on any node in the cluster. This superuser script resides by default in `/opt/SUNWhpc/sbin`.

/opt/SUNWhpc/sbin/part_initialize

This action creates a single partition named `all`, which includes all the nodes in the cluster as members. The `all` partition can be used in the subsequent verification tests.

Then, run `mpinfo -N` again to verify the successful creation of `all`. See below for an example of `mpinfo -N` output when the `all` partition is present.

```
# /opt/SUNWhpc/sbin/part_initialize
# mpinfo -N
NAME   UP  PARTITION  OS    OSREL  NCPU  FMEM   FSWP   LOAD1   LOAD5   LOAD15
node1  y   all        SunOS 5.8    1     7.17   74.76  0.03    0.04    0.05
node2  y   all        SunOS 5.8    1     34.69  38.08  0.00    0.00    0.01
```

Verify That Sun CRE Executes Jobs

Verify that Sun CRE can launch jobs on the cluster. For example, use the `mprun` command to execute the program `hostname` on all the nodes in the cluster, as shown below:

```
# mprun –Ns –np 0 hostname
node1
node2
```

`mprun` is the Sun CRE command that launches jobs. The combination of `–Ns` and `–np 0` ensures that Sun CRE will start one `hostname` process on each node. See the `mprun` man page for descriptions of `–Ns`, `–np`, and the other `mprun` options. In this example, the cluster contains two nodes, `node1` and `node2`, each of which returns its host name.

Note – Sun CRE does not sort or rank the output of `mprun` by default, so host name ordering may vary from one run to another.

Verifying MPI Communications

You can verify MPI communications by running a simple MPI program.

The MPI program must have been compiled by one of the compilers supported by Sun HPC ClusterTools software (listed in “Sun Compilers” on page 4).

Two simple Sun MPI programs are available in `/opt/SUNWhpc/examples/mpi`:

- `connectivity.c` – A C program that checks the connectivity among all processes and prints a message when it finishes
- `monte.f` – A Fortran program in which each process participates in calculating an estimate of π using a Monte-Carlo method

See the `Readme` file in the same directory; it provides instructions for using the examples. The directory also contains the make file, `Makefile`. The full text of both code examples is also included in the *Sun MPI Software Programming and Reference Guide*.

Stopping and Restarting Sun CRE

If you want to shut down the entire cluster with the least risk to your file systems, use the Solaris `shutdown` command.

However, if you prefer to stop and restart Sun CRE daemons without shutting down the entire cluster, use the `ctstopd` and `ctstartd` commands. These commands are described in the *Sun HPC ClusterTools Software Installation Guide* as well as in the `ctstopd`(1m) and `ctstartd`(1m) man pages.

Ordinarily, you would initiate the command from a central host, specifying the cluster nodes on which the command is to execute. Alternatively, you could specify that the command execute locally on the node where it is initiated. Examples of these two approaches follow.

Stopping and Starting Sun CRE Daemons From a Central Host

This section shows how to start Sun HPC ClusterTools software daemons from a central host.

Stop Daemons on Specified Cluster Nodes

CODE EXAMPLE 2-3

```
# ./ctstopd –N /tmp/nodelist –r rsh –k /tmp/cluster-logs –g
```

CODE EXAMPLE 2-3 stops the Sun HPC ClusterTools software daemons on the nodes listed in `/tmp/nodelist`. It uses the options `–k` and `–g` to gather log information centrally and to generate pass and fail node lists. The remote connection method is rsh.

Start Daemons on Specified Cluster Nodes

CODE EXAMPLE 2-4

```
# ./ctstartd –N /tmp/nodelist –r rsh –k /tmp/cluster-logs –g
```

CODE EXAMPLE 2-4 starts the Sun HPC ClusterTools software daemons on the nodes listed in `/tmp/nodelist`. It uses the options `–k` and `–g` to gather log information centrally and to generate pass and fail node lists. The remote connection method is rsh.

Stopping and Starting Sun CRE Daemons on the Local Node

Stop Daemons Locally

CODE EXAMPLE 2-5

```
# ./ctstopd –l
```

CODE EXAMPLE 2-5 stops the Sun HPC ClusterTools software daemons on the local node.

Start Daemons Locally

CODE EXAMPLE 2-6

```
# ./ctstartd –l
```

CODE EXAMPLE 2-6 starts the Sun HPC ClusterTools software daemons on the local node.

CHAPTER 3

Overview of Administration Controls

The Sun HPC cluster's default configuration supports execution of MPI applications. In other words, if you have started the Sun CRE daemons on your cluster and created a default partition, as described in Chapter 2, users can begin executing MPI jobs.You may, however, want to customize the cluster's configuration to the specific requirements of your site.

This chapter provides a brief overview of the features that control a cluster's configuration and behavior. These are:

- The Sun CRE daemons - introduced here and described more fully in their respective man pages
- The RSM daemon
- The `mpadmin` command - introduced here and described more fully in Chapter 6, as well as in its man page
- The cluster configuration file `hpc.conf` – introduced here and described more fully in Chapter 7, as well as in its man page
- Authentication methods

The Sun CRE Daemons

Sun CRE comprises three master daemons and two nodal daemons:

- Sun CRE master daemons
  - `tm.rdb`
  - `tm.mpmd`
  - `tm.watchd`
- Sun CRE nodal daemons

- tm.omd
- tm.spmd

This section present brief descriptions of the Sun CRE daemons. For complete information on the daemons, see their respective man pages.

Master Daemon tm.rdb

tm.rdb is the *resource database daemon*. It runs on the master node and implements the resource database used by the other parts of Sun CRE. This database represents the state of the cluster and the jobs running on it.

If you make changes to the cluster configuration, for example, if you add a node to a partition, you must restart the tm.rdb daemon to update the Sun CRE resource database to reflect the new condition.

Master Daemon tm.mpmd

tm.mpmd is the *master process-management daemon*. It runs on the master node and services user (client) requests made via the mprun command. It also interacts with the resource database via calls to tm.rdb and coordinates the operations of the nodal client daemons.

Master Daemon tm.watchd

tm.watchd is the cluster *watcher daemon*. It runs on the master node and monitors the states of cluster resources and jobs and, as necessary:

- Marks individual nodes as online or offline by periodically executing remote procedure calls (RPCs) to all the nodes.
- Clears stale resource database (rdb) locks.
- If the –Yk option has been enabled, aborts jobs that have processes on nodes determined to be down. This option is disabled by default. This option can be especially valuable on very large clusters.

Nodal Daemon `tm.omd`

`tm.omd` is the *object-monitoring daemon*. It runs on all the nodes in the cluster, including the master node, and continually updates the Sun CRE resource database with dynamic information concerning the nodes, most notably their load. It also initializes the database with static information about the nodes, such as their host names and network interfaces, when Sun CRE starts up.

The environment variable `SUNHPC_CONFIG_DIR` specifies the directory in which the Sun CRE resource database files are to be stored. The default is `/var/hpc`.

Nodal Daemon `tm.spmd`

`tm.spmd` is the *slave process-management daemon*. It runs on all the compute nodes of the cluster and, as necessary:

- Handles spawning and termination of nodal processes per requests from the `tm.mpmd`.
- In conjunction with `mprun`, handles multiplexing of `stdio` streams for nodal processes.
- Interacts with the resource database via calls to `tm.rdb`.

Spin Daemon `tm.spind`

`tm.spind` is the *spin daemon*. It runs on all the compute nodes of the cluster.

This daemon enables certain processes of a given MPI job on a shared-memory system to be scheduled at approximately the same time as other related processes. This co-scheduling reduces the load on the processors, thus reducing the effect that MPI jobs have on each other.

RSM Daemon

The `hpc_rsmd` daemon provides access services to Remote Shared Memory (RSM) resources and manages RSM communications paths on behalf of MPI jobs. If your cluster is not RSM-enabled, you can ignore this section.

An instance of `hpc_rsmd` runs on each RSM-enabled node of a cluster and:

- Provides remote shared-memory resources to MPI client processes.

- Monitors and manages the RSM paths between node pairs.
- Returns cluster configuration information to Sun CRE.

`hpc_rsmd` is started when the cluster is booted. After boot, the daemon can be stopped and restarted manually by means of the following script (executed by superuser). This script can also be used to clean up files and System V shared memory segments left behind when an `hpc_rsmd` instance exits abnormally.

```
# /etc/init.d/sunhpc.hpc_rsmd  [ start | stop | clean ]
```

`mpadmin`: Administration Interface

Sun CRE provides an interactive command interface, `mpadmin`, which you can use to administer your Sun HPC cluster. It can only be invoked by the superuser.

This section introduces `mpadmin` and shows how to use it to perform several administrative tasks:

- List the names of all nodes in the cluster
- Enable nodes
- Create and enable partitions
- Customize some aspects of cluster administration

`mpadmin` offers many more capabilities than are described in this section. See Chapter 6 for a more comprehensive description of `mpadmin`.

Introduction to `mpadmin`

The `mpadmin` command has the following syntax:

```
# mpadmin [-c command] [-f filename] [-h] [-q] [-s cluster_name] [-V]
```

When you invoke `mpadmin` with no options, it goes into interactive mode, displaying an `mpadmin` prompt. It also goes into interactive mode when invoked with the options `-f`, `-q`, or `-s`. In this mode, you can execute any number of `mpadmin` subcommands to perform operations on the cluster or on nodes or partitions.

When you invoke `mpadmin` with the `-c`, `-h`, or `-V` options, it performs the requested operation and returns to the shell level.

The mpadmin command-line options are summarized in TABLE 3-1.

TABLE 3-1 mpadmin Options

Option	Description
–c *command*	Execute single specified command.
–f *file-name*	Take input from specified file.
–h	Display help / usage text.
–q	Suppress the display of a warning message when a non-root user attempts to use restricted command mode.
–s *cluster-name*	Connect to the specified Sun HPC cluster.
–V	Display mpadmin version information.

Commonly Used mpadmin Options

This section describes the mpadmin options -c, -f, and -s.

–c *command – Execute a Single Command*

Use the –c option when you want to execute a single mpadmin command and return upon completion to the shell prompt. For example, the following use of mpadmin –c changes the location of the Sun CRE log file to /home/wmitty/cre_messages:

```
# mpadmin -c set logfile="/home/wmitty/cre_messages"
```

Most commands that are available via the interactive interface can be invoked via the –c option. See Chapter 6 for a description of the mpadmin command set and a list of which commands can be used as arguments to the –c option.

–f *file-name – Take Input From a File*

Use the –f option to supply input to mpadmin from the file specified by the *file-name* argument. The source file is expected to consist of one or more mpadmin commands, one command per line.

This option can be particularly useful in the following ways:

- It can be used following use of the mpadmin command dump, which outputs all or part of a cluster's configuration in the form of an mpadmin script. If the dump output is stored in a file, mpadmin can, at a later time, read the file via the –f option, thereby reconstructing the configuration that had been saved in the dump output file.

- The `-f` option can also be used to read `mpadmin` scripts written by the system administrator—scripts designed to simplify other cluster management tasks that involve issuing a series of `mpadmin` commands.

`-s` *cluster-name – Connect to Specified Cluster*

Use the `-s` option to connect to the cluster specified by the *cluster-name* argument. A cluster's name is the host name of the cluster's master node.

The `mpadmin` commands apply to a certain cluster, determined as follows:

- First, the cluster specified on the command line with the `-s` option.
- If no such option is specified, the command uses the value of the environment variable `SUNHPC_CLUSTER`.
- If this environment variable is not set, the command uses the cluster name supplied (usually at installation time) in `/etc/hpc_system`.

Understanding Objects, Attributes, and Contexts

To use `mpadmin`, you need to understand the concepts of *object*, *attribute*, and *context* as they apply to `mpadmin`.

Objects and Attributes

From the perspective of `mpadmin`, a Sun HPC cluster consists of a system of objects, which include

- The cluster itself
- Each node contained in the cluster
- Each partition (logical group of nodes) defined in the cluster

Each type of object has a set of attributes, which control various aspects of their respective objects. For example, a node's `enabled` attribute can be

- `set` to make the node available for use
- `unset` to prevent it from being used

Some attribute values can be operated on via `mpadmin` commands.

Note – Sun CRE sets many attributes in a cluster to default values each time it starts up. You should not change attribute values, except for the attribute modifications described here and in Chapter 6.

Contexts

`mpadmin` commands are organized into three *contexts*, which correspond to the three types of `mpadmin` objects. These contexts are summarized below and illustrated in FIGURE 3-1.

- Cluster – These commands affect cluster attributes.
- Node – These commands affect node attributes.
- Partition – These commands affect partition attributes.

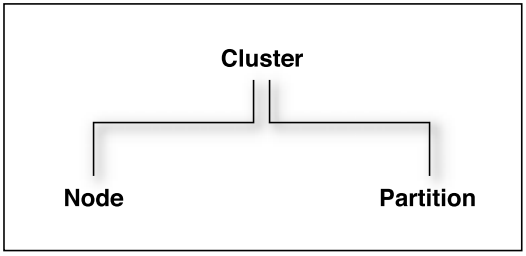


FIGURE 3-1 `mpadmin` Contexts

`mpadmin` Prompts

In interactive mode, the `mpadmin` prompt contains one or more fields that indicate the current context. TABLE 3-2 shows the prompt format for each of the possible `mpadmin` contexts.

TABLE 3-2 `mpadmin` Prompt Formats

Prompt Formats	Context
[*cluster-name*]::	Current context = Cluster
[*cluster-name*]Node::	Current context = Node, but not a specific node
[*cluster-name*]N(*node-name*)::	Current context = a specific node
[*cluster-name*]Partition::	Current context = Partition, but not a specific partition
[*cluster-name*]P(*partition-name*)::	Current context = a specific partition

Performing Sample mpadmin Tasks

To introduce the use of mpadmin, this section steps through some common tasks the administrator may want to perform. These tasks are:

- List names of nodes
- Enable nodes so that MPI jobs can run on them
- Create and enable partitions so that MPI jobs can run on them
- Customize some cluster attributes

List Names of Nodes

mpadmin provides various ways to display information about the cluster and many kinds of information that can be displayed. However, the first information you are likely to need is a list of the nodes in your cluster.

Use the list command in the Node context to display this list. In the following example, list is executed on node1 in a four-node cluster.

```
node1# mpadmin
[node0]:: node
[node0] Node:: list
    node0
    node1
    node2
    node3
[node0] Node::
```

The mpadmin command starts up an mpadmin interactive session in the Cluster context. This is indicated by the [node0]:: prompt, which contains the cluster name, node0, and no other context information.

Note – A cluster's name is assigned by Sun CRE and is always the name of the cluster's master node.

The node command on the example's second line makes Node the current context. The list command displays a list of all the nodes in the cluster.

Once you have this list of nodes, you have the information you need to enable the nodes and to create a partition. However, before moving on to those steps, you might want to try listing information from within the cluster context or the partition context. In either case, you would follow the same general procedure as for listing nodes.

If this is a newly installed cluster and you have not already run the `part_initialize` script (as described in “Create a Default Partition” on page 8), the cluster contains no partitions. If, however, you *did* run `part_initialize` and have thereby created the partition `all`, you might want to perform the following test.

```
node1# mpadmin
[node0]:: partition
[node0] Partition:: list
    all
[node0] Partition::
```

To see what nodes are in partition `all`, make `all` the current context and execute the `list` command. The following example illustrates this; it begins in the `Partition` context (where the previous example ended).

```
[node0] Partition:: all
[node0] P[all]:: list
    node0
    node1
    node2
    node3
[node0] P[all]::
```

Enabling Nodes

A node must be in the enabled state before MPI jobs can run on it.

Note that enabling a partition automatically enables all its member nodes, as described in the next section.

To enable a node manually, make that node the current context and set its `enabled` attribute. Repeat for each node that you want to be available for running MPI jobs.

The following example illustrates this, using the same four-node cluster used in the previous examples.

```
node1# mpadmin
[node0]:: node0
[node0] N[node0]:: set enabled
[node0] N[node0]:: node1
[node0] N[node1]:: set enabled
[node0] N[node1]:: node2
[node0] N[node2]:: set enabled
[node0] N[node2]:: node3
[node0] N[node3]:: set enabled
[node0] N[node3]::
```

Note the use of a shortcut to move directly from the `Cluster` context to the `node0` context without first going to the general `Node` context. You can explicitly name a particular object as the target context in this way so long as the name of the object is unambiguous—that is, it is not the same as an `mpadmin` command.

`mpadmin` accepts multiple commands on the same line. The previous example could be expressed more succinctly as:

```
node1# mpadmin
[node0]:: node0 set enabled node1 set enabled node2 set enabled node3
set enabled
[node0] N[node3]::
```

To disable a node, use the `unset` command in place of the `set` command.

Creating and Enabling Partitions

You must create at least one partition and enable it before you can run MPI programs on your Sun HPC cluster. Even if your cluster already has the default partition `all` in its database, you will probably want to create other partitions with different node configurations to handle particular job requirements.

There are three essential steps involved in creating and enabling a partition:

- Use the `create` command to assign a name to the partition.
- Set the partition's `nodes` attribute to a list of the nodes you want to include in the partition.
- Set the partition's `enabled` attribute, which automatically enables all the partition's member nodes.

Once a partition is created and enabled, you can run serial or parallel jobs on it. A serial program runs on a single node of the partition. Parallel programs are distributed to as many nodes as Sun CRE determines appropriate for the job.

Note – There are no restrictions on the number or size of partitions, so long as no node is a member of more than one enabled partition.

Example: Creating a Two-Node Partition

The following example creates and enables a two-node partition named `part0`. It then lists the member nodes to verify the success of the creation.

```
node1# mpadmin
[node0]:: partition
[node0] Partition:: create part0
[node0] P[part0]:: set nodes=node0 node1
```

```
[node0] P[part0]:: set enabled
[node0] P[part0]:: list
    node0
    node1
[node0] P[part0]::
```

Example: Two Partitions Sharing a Node

The next example shows a second partition, `part1`, being created. One of its nodes, `node1`, is also a member of `part1`.

```
[node0] P[part0]:: up
[node0] Partition:: create part1
[node0] P[part1]:: set nodes=node1 node2 node3
[node0] P[part1]:: list
    node1
    node2
    node3
[node0] P[part1]::
```

Because `node1` is shared with `part0`, which is already enabled, `part1` is not being enabled at this time. This illustrates the rule that a node can be a member of more than one partition, but only one of those partitions can be enabled at a time.

Note the use of the `up` command. The `up` command moves the context up one level, in this case, from the context of a particular partition (that is, from `part0`) to the general `Partition` context.

Example: Shared versus Dedicated Partitions

Sun CRE can configure a partition to allow multiple MPI jobs to be running on it concurrently. Such partitions are referred to as *shared* partitions. Sun CRE can also configure a partition to permit only one MPI job to run at a time. These are called *dedicated* partitions.

In the following example, the partition `part0` is configured to be a dedicated partition and `part1` is configured to allow shared use by up to four processes.

```
node1# mpadmin
[node0]:: part0
[node0] P[part0]:: set max_total_procs=1
[node0] P[part0]:: part1
[node0] P[part1]:: set max_total_procs=4
[node0] P[part1]::
```

The max_total_procs attribute defines how many processes can be active on each node in the partition for which it is being set. In this example, it is set to 1 on part0, which means only one process can be running at a time. It is set to 4 on part1 to allow up to four processes to run on that partition.

Note again that the context-changing shortcut (introduced in "Enabling Nodes" on page 20) is used in the second and fourth lines of this example.

Customizing Cluster Attributes

Two cluster attributes that you may wish to modify are logfile and administrator.

Changing the logfile *Attribute*

The logfile attribute allows you to log Sun CRE messages in a separate file from all other system messages. For example, if you enter

```
[node0]:: set logfile=/home/wmitty/cre-messages
```

Sun CRE will output its messages to the file /home/wmitty/cre-messages. If logfile is not set, Sun CRE messages will be passed to syslog, which will store them with other system messages in /var/adm/messages.

Note – A full path name must be specified when setting the logfile attribute.

Changing the administrator *Attribute*

You can set the administrator attribute to specify, say, the email address of the system administrator. To do this:

```
[node0]:: set administrator="root@example.com"
```

Note the use of double quotes.

Quitting mpadmin

Use either the quit or exit command to quit an mpadmin interactive session. Either causes mpadmin to terminate and return control to the shell level.

For example:

```
[node0]:: quit
node1#
```

Cluster Configuration File hpc.conf

When Sun CRE starts up, it updates portions of the resource database according to the contents of a configuration file named hpc.conf. This file is organized into functional sections, which are summarized here and illustrated in TABLE 3-3.

- ShmemResource section defines certain shared memory attributes.
- MPIOptions section defines certain MPI parameters.
- CREOptions section controls the logging of system events, the handling of daemon core files, the authentication options, and placing restrictions on mprun's ability to run programs in batch mode.
- PMODULES section names and locates the protocol modules that are available for communication on the cluster.
- PM sections list and rank the protocol modules that are available for communication on the cluster.
- HPCNodes section is not used by Sun CRE. It applies only in an LSF-based runtime environment.

You can change any of these aspects of your cluster's configuration by editing the corresponding parts of the hpc.conf file. Default settings are in effect if you make no changes to the hpc.conf file as provided.

To illustrate the process of customizing the hpc.conf file, this section explains how to:

- Prepare for editing hpc.conf
- Control MPI communication attributes
- Update the Sun CRE database

Note – The hpc.conf file is provided with the Sun HPC ClusterTools software.

Note – You may never need to make any changes to hpc.conf. If you do wish to make changes beyond those described in this section, see Chapter 6 for a fuller description of this file.

TABLE 3-3 General Organization of the `hpc.conf` File

```
# Begin ShmemResource
# ...
# End ShmemResource

# Begin MPIOptions Queue=
# ...
# End MPIOptions

# Begin CREOptions Server=
# ...
# End CREOptions

# Begin HPCNodes
# ...
# End HPCNodes

Begin PMODULES
...
End PMODULES

Begin PM=shm
...
End PM

Begin PM=rsm
...
End PM

Begin PM=tcp
...
End PM
```

Preparing to Edit `hpc.conf`

Perform the steps described below to stop the Sun CRE daemons and copy the `hpc.conf` template.

Stop the Sun CRE Daemons

Stop the Sun CRE daemons on all cluster nodes. For example, to stop the Sun CRE daemons on the cluster nodes, `node1` and `node2` from a central host, enter

```
# ./ctstopd –n node1,node2 –r connection_method
```

where *connection_method* is rsh, ssh, or telnet. Or, you can specify a nodelist file instead of listing the nodes on the command line.

```
# ./ctstopd –N /tmp/nodelist –r connection_method
```

where */tmp/nodelist* is absolute path to a file containing the names of the cluster nodes, with each name on a separate line. Comments and empty lines are allowed. For example, if the cluster contains the nodes node1 and node2, a nodelist file for the cluster could look like the following:

CODE EXAMPLE 3-1

```
# Sample Node List

node1
node2
```

Copy the `hpc.conf` Template

The Sun HPC ClusterTools software distribution includes an `hpc.conf` template, which is stored, by default, in `/opt/SUNWhpc/examples/rte/hpc.conf.template`.

Copy the template from its installed location to `/opt/SUNWhpc/etc/hpc.conf` and edit it as described below in this section.

When you have finished editing `hpc.conf`, you need to update the Sun CRE database with the new configuration information. This step is described in “Updating the Sun CRE Database” on page 34.

Specifying MPI Options

The `MPIOptions` section provides a set of options that control MPI communication behavior in ways that are likely to affect message-passing performance. It contains a template showing some general-purpose option settings, plus an example of alternative settings for maximizing performance. These examples are shown in TABLE 3-4.

- **G**eneral-purpose, multiuser settings – The template in the `MPIOptions` section is designed for general-purpose use at times when multiple message-passing jobs will be running concurrently.
- Performance settings – The second example is designed to maximize the performance of message-passing jobs when only one job is allowed to run at a time.

Note – The first line of the template contains the phrase "`Queue=hpc`." This is because the queue-based LSF workload management run-time environment uses the same `hpc.conf` file as does Sun CRE. For LSF, the settings apply only to the specified queue. For Sun CRE, the settings apply across the cluster.

The options in the general-purpose template are the same as the default settings for the Sun MPI library. In other words, you do not have to uncomment the general-purpose template to have its option values be in effect. This template is provided in the `MPIOptions` section so that you can see what options are most beneficial when operating in a multiuser mode.

TABLE 3-4 `MPIOptions` Section Example

```
# The following is an example of the options that affect the run
# time environment of the MPI library.  The listings below are
# identical to the default settings of the library. The "Queue=hpc"
# phrase makes this an LSF-specific entry, and only for the Queue
# named hpc. These options are a good choice for a multiuser queue.
# To be recognized by CRE, the "Queue=hpc" needs to be removed.
#
# Begin MPIOptions Queue=hpc
# coscheduling  avail
# pbind        avail
# spindtimeout   1000
# progressadjust   on
# spin      off
#
# shm_numpostbox           16
# shm_shortmsgsize        256
# rsm_maxsegsize      1048576
# rsm_numpostbox           15
# rsm_shortmsgsize        401
# rsm_maxstripe            2
# rsm_links          wrsm0,1
# maxprocs_limit   2147483647
# maxprocs_default       4096
#
# End MPIOptions

# The listing below is a good choice when trying to get maximum
# performance out of MPI jobs that are running in a Queue that
# allows only one job to run at a time.
#
# Begin MPIOptions Queue=performance
# coscheduling             off
# spin                      on
# End MPIOptions
```

If you want to use the `performance` settings, do the following:

- Delete the comment character (#) from the beginning of each line of the performance example, including the `Begin MPIOptions` and `End MPIOptions` lines.
- On Sun CRE-based clusters, delete the "`Queue=performance`" phrase from the `Begin MPIOptions` line.

The resulting template should appear as follows:

```
Begin MPIOptions
coscheduling            off
spin                    on
End MPIOptions
```

The significance of these options is discussed in Chapter 6.

Updating the Sun CRE Database

When you have finished editing `hpc.conf`, update the Sun CRE database with the new information. For example, to start the daemons on cluster nodes `node1` and `node2` from a central host, enter

```
# ./ctstartd –n node1,node2 –r connection_method
```

where *connection_method* is rsh, ssh, or telnet. Or, you can specify a nodelist file instead of listing the nodes on a command line.

```
# ./ctstopd –N /tmp/nodelist –r connection_method
```

where */tmp/nodelist* is absolute path to a file containing the names of the cluster nodes, with each name on a separate line.

Authentication and Security

Sun CRE provides basic security by means of a cluster password, which is stored in a key file on each node.

In addition, you can set up further methods of guarding the cluster against access by unauthorized users or programs. Sun CRE supports UNIX system authentication (via `rhosts`), as well as two third-party mechanisms for authentication: Data Encryption Standard (DES) and Kerberos Version 5.

Setting the Sun CRE Cluster Password

Sun CRE uses a root-read-only key file to control access to the cluster. The key file must exist on every node of the cluster, and the contents of all the key files must be identical. In addition, the key file must be placed on any node outside the cluster that might access the cluster (that is, on any node that may execute the command `mprun -c` *cluster_name*).

The key resides in `/etc/hpc_key.`*cluster_name* on each node.

The installation procedure creates a default key file on each node of the cluster. A strongly recommended step in the post-installation procedure is to customize the key immediately after installing the Sun HPC ClusterTools software. The key should be 10-20 alphanumeric characters.

The administrator can change the key at any time. As superuser, run the `set_key` script on each node in the cluster and on any nodes outside the cluster that may access the cluster:

```
# /etc/opt/SUNWhpc/HPC5.0/etc/set_key
```

It is preferable to stop the Sun CRE daemons before changing the key, since a current MPI job might fail.

To guarantee that the cluster key is set identically on every node, you should use the Cluster Console Manager tools (described in Appendix A) to update all the key files at once.

Note – The cluster password security feature exists in addition to the “current” authentication method, as specified in the `hpc.conf` file and described below.

Establishing the Current Authentication Method

Authentication is established in the configuration file `hpc.conf` in the section `CREOptions`.

TABLE 3-5 `CREOptions` Section Example

```
Begin CREOptions
 ...
auth_opt              sunhpc_rhosts
End CREOptions
```

The value of the `auth_opt` option is one of:

- sunhpc_rhosts – Contains list of trusted hosts, the default
- rhosts – Standard UNIX system authentication
- des – DES-based authentication
- krb5 - Kerberos 5 authentication

To change authentication methods, stop all Sun CRE daemons, edit the hpc.conf file, and then restart the Sun CRE daemons. See "Preparing to Edit hpc.conf" on page 30.

Note – Since authentication methods limit the time that can elapse between the initiation of a remote procedure call (RPC) and the system's response, administrators should ensure that the nodes of the Sun HPC cluster and the machines from which users submit jobs are closely synchronized. For example, you can synchronize the machines by setting all system clocks to the same time using the Solaris date command.

Setting Up the Default Authentication

When authentication option rhosts is in use, any Sun CRE operation (such as mpadmin or mprun) attempted by the superuser will be allowed only if three items:

- The requesting host
- The master Sun CRE host
- The hosts on which any mprun operation will execute

appear in one of the following files:

- The /etc/sunhpc_rhosts file, if it has been installed (the default)
- The default .rhosts file (if the sunhpc_rhosts file is not created at installation, or if it has been deleted)

The sunhpc_rhosts file's contents are visible only to the superuser.

To allow superuser access from hosts outside the cluster, the node name must be added to the /etc/sunhpc_rhosts file.

If the /etc/sunhpc_rhosts file is not used (or has been removed), the .rhosts file on each node must be updated to include the name of every node in the cluster. Using .rhosts assumes trusted hosts. For information on trusted hosts, see the man page for hosts.equiv.

Setting Up DES Authentication

In order to use DES authentication with Sun CRE, host keys must exist for each host in the cluster and `/etc/.rootkey` must exist for each node of the cluster. User keys must exist on all hosts that will be used to communicate with the cluster using Sun CRE commands, as well as on each node of the cluster (including the master), for each user who is to access the cluster. Inconsistent key distribution will prevent correct operation.

To set up DES authentication, you must ensure that all hosts in the system, and all users, have entries in both the `publickey` and `netname` databases. Furthermore, the entries in `/etc/nsswitch.conf` for both `publickey` and `netid` databases must point to the correct place. For further information, see the Solaris man pages for `publickey(4)`, `nsswitch.conf(4)`, and `netid(4)`.

After all new keys are in place, you need to restart the DES keyserver `keyserv`. You must also establish `/etc/.rootkey` on each node, as described in the man page `keylogin(1)`.

When the DES setup is complete, restart the Sun CRE daemons (see “Stopping and Restarting Sun CRE” on page 14).

It is recommended that you use one of the Cluster Console Manager tools (`cconsole`, `ctelnet`, or `crlogin`) to issue identical commands to all the nodes at the same time. For information about the Cluster Console Manager, see Appendix A.

Note – While DES authentication is in use, users must issue the `keylogin` command before issuing any commands beginning with `mp`, such as `mprun` or `mpps`.

Setting Up Kerberos Authentication

To set up Kerberos 5 authentication, the administrator registers a host principal (`host`) and a Sun CRE (`sunhpc`) principal with an instance for each node that is to be used as a Sun CRE client. In addition, each host must have `host` and `principal` entries in the appropriate `keytabs`.

For example: consider a system consisting of three nodes (`node0`, `node1`, and `node2`), in Kerberos realm `example.com`. Nodes `node0` and `node1` will be used as Sun CRE servers and all three nodes will be used as Sun CRE clients. The database should include the following principals as well as principals for any end-users of Sun CRE services, created using the `addprinc` command in `kadmin`:

```
sunhpc/node0@example.com
sunhpc/node1@example.com
sunhpc/node2@example.com
```

```
host/node0@example.com
host/node1@example.com
host/node2@example.com
```

The `sunhpc` and `host` principals should have entries in the default `keytab` (created using the `ktadd` command in `kadmin`).

Any user who wishes to use Sun CRE to execute programs must first obtain a ticket granting ticket via `kinit`.

For further information on Kerberos version 5, see the Kerberos documentation.

CHAPTER 4

Cluster Configuration Notes

This chapter examines various issues that may have some bearing on choices you make when configuring your Sun HPC cluster. The discussion is organized into three general topic areas:

- Nodes
- Interconnects
- Close Integration With Batch Processing Systems

Nodes

Configuring a Sun SMP or cluster of SMPs to use Sun HPC ClusterTools software involves many of the same choices seen when configuring general-purpose compute servers. Common issues include the number of CPUs per machine, the amount of installed memory, and the amount of disk space reserved for swapping.

Because the characteristics of the particular applications to be run on any given Sun HPC cluster have such a large effect on the optimal settings of these parameters, the following discussion is necessarily general in nature.

Number of CPUs

Since Sun MPI programs can run efficiently on a single SMP, it can be advantageous to have at least as many CPUs as there are processes used by the applications running on the cluster. This is not a necessary condition since Sun MPI applications can run across multiple nodes in a cluster, but for applications with very large interprocess communication requirements, running on a single SMP may result in significant performance gains.

Memory

Generally, the amount of installed memory should be proportional to the number of CPUs in the cluster, although the exact amount depends significantly on the particulars of the target application mix.

Computationally intensive Sun HPC ClusterTools software applications that process data with some amount of locality of access often benefit from larger external caches on their processor modules. Large cache capacity allows data to be kept closer to the processor for longer periods of time.

Swap Space

Because Sun HPC Clustertools software applications are, on average, larger than those typically run on compute servers, the swap space allocated to Sun HPC clusters should be correspondingly larger. The amount of swap should be proportional to the number of CPUs and to the amount of installed memory. Additional swap should be configured to act as backing store for the shared memory communication areas used by Sun HPC ClusterTools software in these situations.

Sun MPI jobs require large amounts of swap space for shared memory files. The sizes of shared memory files scale in stepwise fashion, rather than linearly. For example, a two-process job (with both processes running within the same SMP) requires shared memory files of approximately 35 Mbytes. A 16-process job (all processes running within the same SMP) requires shared memory files of approximately 85 Mbytes. A 256-process job (all processes running within the same SMP) requires shared memory files of approximately 210 Mbytes.

Interconnects

One of the most fundamental issues to be addressed when configuring a cluster is the question of how to *connect* the nodes of the cluster. In particular, both the type and the number of networks should be chosen to complement the way in which the cluster is most likely to be used.

Note – For the purposes of this discussion, the term *default network* refers to the network associated with the standard host name. The term *parallel application network* refers to an optional second network, operating under the control of Sun CRE.

In a broad sense, a Sun HPC cluster can be viewed as a standard LAN. Operations performed on nodes of the cluster will generate the same type of network traffic that is seen on a LAN. For example, running an executable and accessing directories and files will cause NFS traffic, while remote login sessions will cause network traffic. This kind of network traffic is referred to here as *administrative* traffic.

Administrative traffic has the potential to tax cluster resources. This can result in significant performance losses for some Sun MPI applications, unless these resources are somehow protected from this traffic. Fortunately, Sun CRE provides enough configuration flexibility to allow you to avoid many of these problems.

The following sections discuss some of the factors that you should consider when building a cluster for Sun HPC ClusterTools software applications.

Sun HPC ClusterTools Internode Communication

Several Sun HPC ClusterTools software components generate internode communication. It is important to understand the nature of this communication in order to make informed decisions about network configurations.

Administrative Traffic

As mentioned earlier, a Sun HPC cluster generates the same kind of network traffic as any UNIX-based LAN. Common operations like starting a program can have a significant network impact. The impact of such administrative traffic should be considered when making network configuration decisions.

When a simple serial program is run within a LAN, network traffic typically occurs as the executable is read from a NFS-mounted disk and paged into a single node's memory. In contrast, when a 16- or 32-process parallel program is invoked, the NFS server is likely to experience approximately simultaneous demands from multiple nodes—each pulling pages of the executable to its own memory. Such requests can often result in large amounts of network traffic. How much traffic occurs will depend on various factors, such as the number of processes in the parallel job, the size of the executable, and so forth.

Sun CRE-Generated Traffic

Sun CRE uses the cluster's default network interconnect to perform communication between the daemons that perform resource management functions. Sun CRE makes heavy use of this network when Sun MPI jobs are started, with the load being roughly proportional to the number of processes in the parallel jobs. This load is in

addition to the start-up load described in the previous section. Sun CRE will generate a similar load during job termination as the Sun CRE database is updated to reflect the expired MPI job.

There is also a small amount of steady traffic generated on this network as Sun CRE continually updates its view of the resources on each cluster node and monitors the status of its components to guard against failures.

Sun MPI Interprocess Traffic

Parallel programs use Sun MPI to move data between processes as the program runs. If the running program is spread across multiple cluster nodes, then the program generates network traffic.

Sun MPI uses the network that Sun CRE instructs it to use, which can be set by the system administrator. In general, Sun CRE instructs Sun MPI to use the fastest network available so that message-passing programs obtain the best possible performance.

If the cluster has only one network, then message-passing traffic shares bandwidth with administrative and Sun CRE functions. This results in performance degradation for all types of traffic, especially if one of the applications is performing significant amounts of data transfer, as message-passing applications often do. You should understand the communication requirements associated with the types of applications to be run on the Sun HPC cluster in order to decide whether the amount and frequency of application-generated traffic warrants the use of a second, dedicated network for parallel application network traffic. In general, a second network will significantly assist overall performance.

Prism Traffic

The Prism graphical programming environment is used to tune, debug, profile, and visualize data from Sun MPI programs running within the cluster. As the Prism program itself is a parallel program, starting it generates the same sort of Sun CRE traffic that invocation of other applications generates.

Once the Prism environment has been started, two kinds of network traffic are generated during a debugging session. The first, which has been covered in preceding sections, is traffic created by running the Sun MPI code that is being debugged. The second kind of traffic is generated by the Prism program itself and is routed over the default network along with all other administrative traffic. In general, the amount of traffic generated by the Prism program itself is small, although viewing performance analysis data on large programs and visualizing large data arrays can cause transiently heavy use of the default network.

Parallel I/O Traffic

Sun MPI programs can make use of the parallel I/O capabilities of Sun HPC ClusterTools software, but not all such programs do so. You need to understand how distributed multiprocess applications that are run on the Sun HPC cluster will make use of parallel I/O to understand the ramifications for network load.

Applications can use parallel I/O to read from and write to standard UNIX file systems, generating NFS traffic on the default network, on the network being used by the Sun MPI component, or some combination of the two. The type of traffic that is generated depends on the type of I/O operations being used by the applications. Collective I/O operations generate traffic on the Sun MPI network, while most other types of I/O operations involve only the default network.

Network Characteristics

Bandwidth, latency, and performance under load are all important network characteristics to consider when choosing interconnects for a Sun HPC cluster. These are discussed in this section.

Bandwidth

Bandwidth should be matched to expected load as closely as possible. If the intended message-passing applications have only modest communication requirements and no significant parallel I/O requirements, then a fast, expensive interconnect may be unnecessary. On the other hand, many parallel applications benefit from *large pipes* (high-bandwidth interconnects). Clusters that are likely to handle such applications should use interconnects with sufficient bandwidth to avoid communication bottlenecks. Significant use of parallel I/O would also increase the importance of having a high-bandwidth interconnect.

It is also a good practice to use a high-bandwidth network to connect large nodes (nodes with many CPUs) so that communication capabilities are in balance with computational capabilities.

An example of a low-bandwidth interconnect is the 10-Mbit/s Ethernet. Examples of higher-bandwidth interconnects include ATM and switched FastEthernet.

Latency

The *latency* of the network is the sum of all delays a message encounters from its point of departure to its point of arrival. The significance of a network's latency varies according to the communication patterns of the application.

Low latency can be particularly important when the message traffic consists mostly of small messages—in such cases, latency accounts for a large proportion of the total time spent transmitting messages. Transmitting larger messages can be more efficient on a network with higher latencies.

Parallel I/O operations are less vulnerable to latency delays than small-message traffic because the messages transferred by parallel I/O operations tend to be large (often 32 Kbytes or larger).

Performance Under Load

Generally speaking, better performance is provided by switched network interconnects, such as ATM and Fibre Channel. Network interconnects with collision-based semantics should be avoided in situations where performance under load is important. Unswitched 10-Mbit/s and 100-Mbit/s Ethernet are the two most common examples of this type of network. While 10-Mbit/s Ethernet is almost certainly not adequate for any Sun HPC ClusterTools software application, a switched version of 100-Mbit/s Ethernet may be sufficient for some applications.

Notes on RSM Setup

The `hpc_rsmd` daemon, which is thread-safe, receives requests from client processes by means of door calls. While running, the `hpc_rsmd` creates and destroys RSM backing store using System V shared memory. The directory `/var/hpc/rsm` is reserved specifically for `hpc_rsmd` usage and should not have any user files stored in it. This directory will contain the `hpc_rsmd_door` file, which is used by RSM protocol modules (PMs) to contact the daemon.

The daemon `hpc_rsmd` has a polling thread that it uses to determine the state of known RSM paths and to determine if unallocated segments should be reaped. The periodicity of the path-monitor thread is 60 seconds.

The RSM PM is activated only by a call into the MPI library made by an application. It has an MT-safe version to be used by the `libmpi_mt.so` library. Its memory usage is dynamic, dependent on the number of processes it has set up to communicate with and the memory it has exported for use in the communications.

When Sun HPC ClusterTools software is installed, certain system parameters in `/etc/system` are set to enable RSM communication. You might wish to change these values to suit other requirements of your site. The default settings established at installation time are the following:

```
set shmsys:shminfo_shmseg=0x800
set shmsys:shminfo_shmmni=0x1000
set shmsys:shminfo_shmmax=0xffffffffffffffff
```

If you decrease any of these values, then the system might not be able to run MPI jobs using RSM. Also, you may need to increase these values beyond the defaults if other applications that use System V shared memory will be running in tandem with MPI jobs.

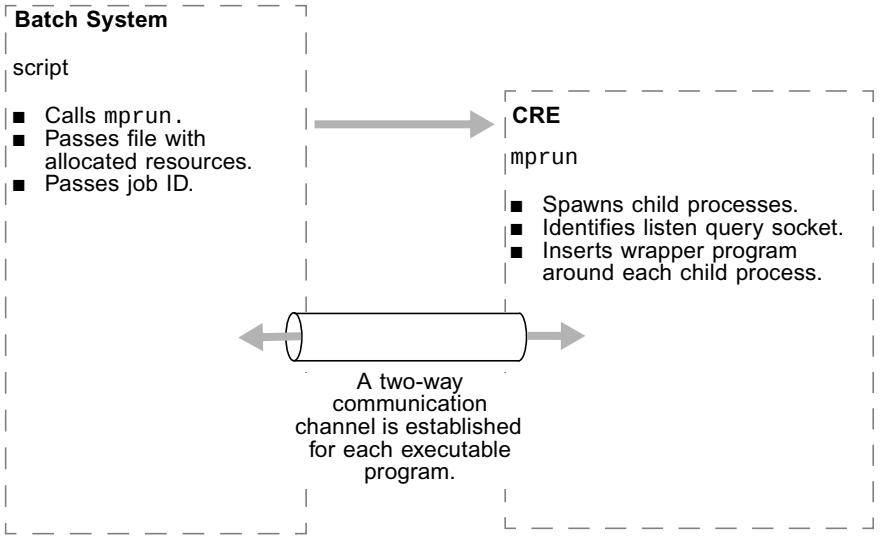
Close Integration With Batch Processing Systems

As mentioned in Chapter 1, Sun CRE provides close integration with several distributed resource managers. In that integration, Sun CRE retains most of its original functions, but delegates others to the resource manager. This section describes:

How Close Integration Works

The integration process is similar for all three, with some individual differences. The batch system, whether SGE, LSF, or PBS, launches the job through a script. The script calls `mprun`, and passes it a host file of the resources that have been allocated for the job, plus the job ID assigned by the batch system.

The Sun CRE environment continues to perform most of its normal parallel-processing actions, but its child processes do not fork any executable programs. Instead, each child process identifies a communications channel (specifically, a listen query socket) through which it can be monitored by the Sun CRE environment while running in the batch system.



Before releasing its child processes to the batch system, the Sun CRE environment inserts around it a wrapper that identifies its listen query socket. That wrapper program keeps the executable programs fully connected to the Sun CRE environment while they are running. And the Sun CRE environment can clean up when a socket closes because a program has stopped running.

A user can also invoke a similar process interactively, without a script. Instructions for script-based and interactive job launching are provided in the *Sun HPC ClusterTools Software User's Guide.*

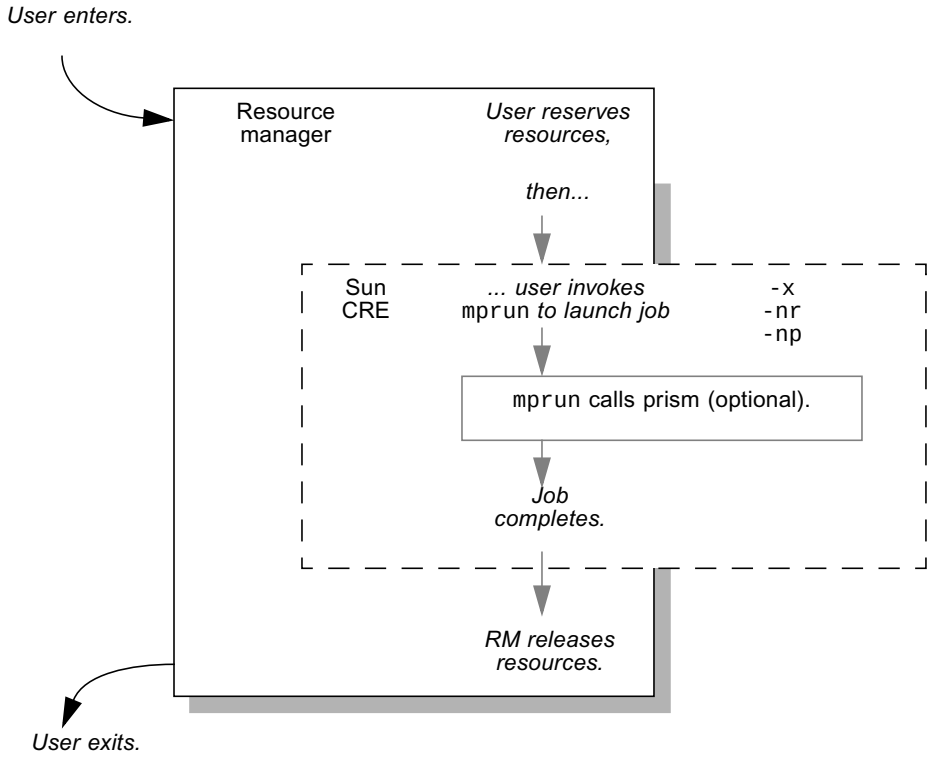
How Close Integration Is Used

The exact instructions vary from one resource manager to another, and are affected by Sun CRE's configuration, but they all follow these general guidelines:

1. The job can be launched either interactively or through a script. Instructions for both are provided in the *Sun HPC ClusterTools Software User's Guide* and the following manpages:
   - `lsf_cre(1)`
   - `pbs_cre(1)`
   - `sge_cre(1)`
2. Users enter the batch processing environment before launching jobs with `mprun`.
3. Users reserve resources for the parallel job and set other job control parameters from within their resource manager.

4. Users invoke the `mprun` command with the applicable resource manager flags. Those flags are described in the *Sun HPC ClusterTools Software User's Guide* and the `mprun(1)` manpage.

Here is a diagram that summarizes the user interaction:



Instructions for Enabling Close Integration

To enable close integration with any resource manager, you must:

- Make sure that both the Sun CRE environment and the RM are installed on the same cluster.
- Set up the `hpc.conf` file to manage the `mprun` command in a batch environment
- Set up the `sunhpc.allow` configuration file, if required.
- Configure each resource manager to work with Sun CRE.

Instructions are provided in these sections:

▼ How to Configure the `hpc.conf` File

Resource managers affect only the CREOptions section of the `hpc.conf` file. Here is an example of that section:

```
Begin CREOptions
enable_core           on
corefile_name         directory and file name of core file
syslog_facility       daemon
auth_opt              sunhpc_rhosts
default_rm            cre
allow_mprun           *
End CREOptions
```

Only two fields affect integration with resource managers: the `default_rm` field and the `allow_mprun` field.

1. **To select a default resource manager, change the `default_rm` field.**

   By selecting a default resource manager, you can save users the trouble of entering the `-x` flag each time they invoke the `mprun` command (as described in *Sun HPC ClusterTools Software User's Guide*). Simply change the value the field to one of these:

   ```
   default_rm lsf
   default_rm pbs
   default_rm sge
   ```

2. **To restrict `mprun`'s ability to run programs in the batch processing environment, add or change the `allow_mprun` field.**

   By default, the value of that field is:

   ```
   allow_mprun *
   ```

   To place no restrictions on `mprun`, either set the value to `*` or remove the field.

   To place restrictions on `mprun`, follow these steps:

a. Create the `sunhpc.allow` **file.**

The `sunhpc.allow` file specifies the restrictions placed on `mprun` by each resource manager. See "How to Configure the `sunhpc.allow` File" on page 49.

b. Change the value of the `allow_mprun` **field.**

Add the names of the resource managers that do *not* place restrictions on `mprun`.

▼ How to Configure the `sunhpc.allow` File

The `sunhpc.allow` file specifies the restrictions that a batch system will place on `mprun`'s ability to call other programs while running in a batch processing environment.

By default, `sunhpc.allow` is ignored. It is only examined when the `hpc.conf` file asks for restrictions (see "How to Configure the `hpc.conf` File" on page 48).

The `sunhpc.allow` file is a text file with one filename pattern per line. Each filename patter specifies the directories whose programs `mprun` is allowed to launch. For example:

```
/opt/SUNWhpc/*
/opt/SUNWhpc/bin/*
```

The example above uses an asterisk to denote all subdirectories or files. For a list of valid pattern-matching symbols, see `fnmatch(1)`.

Before launching a program, `mprun` examines the `sunhpc.allow` file and tries to match the program's absolute pathname to the patterns in the file. If the program's absolute pathname matches one of the patterns, the program is launched. If it does not match, the program is not launched, and an error message is displayed to the user, listing the absolute path that had no match in the file.

The `sunhpc.allow` file must be located on the master node and owned by root. It is read only at startup. If you add new entries to it, you must restart the master node daemons.

▼ How to Configure LSF For Close Integration

The previous requirements for close integration with LSF are no longer necessary. However, to maintain compatibility with Sun HPC ClusterTools 4 software, you must modify three LSB queues so they use the `sunhpc` utility as the job starter instead of the `pam -t` utility. The queues are:

- hpc queue
- hpc-batch queue
- hpc-pfs queue

Here is how their definitions in lsb.queues should look:

CODE EXAMPLE 4-1 hpc Queue

```
Begin Queue
    QUEUE_NAME             = hpc  # Added by SunHPC RTE pkg
    PRIORITY               = 45
    NICE                   = 0
    PREEMPTION             = PREEMPTIVE
    NEW_JOB_SCHEDULE_DELAY = 0
    INTERACTIVE            = ONLY
    JOB_STARTER            = /opt/SUNWhpc/lib/sunhpc
    DESCRIPTION            = Sun HPC ClusterTools software
                             interactive queue (uses
sunhpc                       as a job starter)
End Queue
```

CODE EXAMPLE 4-2 hpc-batch Queue

```
Begin Queue
    QUEUE_NAME             = hpc-batch  #Added by SunHPC RTE pkg
    PRIORITY               = 43
    NICE                   = 10
    PREEMPTION             = PREEMPTIVE
    JOB_ACCEPT_INTERVAL    = 1
    INTERACTIVE            = NO
    JOB_STARTER            = /opt/SUNWhpc/lib/sunhpc
    DESCRIPTION            = Sun HPC ClusterTools software batch
                             queue (uses sunhpc as a job starter)
End Queue
```

CODE EXAMPLE 4-3 hpc-pfs Queue

```
Begin Queue
    QUEUE_NAME             = hpc-pfs #Added by SunHPC RTE pkg
    PRIORITY               = -10
    USERS                  = root
    PREEMPTION             = PREEMPTIVE
    INTERACTIVE            = NO
    JOB_STARTER            = /opt/SUNWhpc/lib/sunhpc
    DESCRIPTION            = Sun HPC ClusterTools queue for
                             pfs daemons only
End Queue
```

▼ How to Configure PBS For Close Integration

To enable close integration with PBS, follow these procedures.

1. **Verify that these directories were created (or linked to) on each node in the cluster**

   Sun CRE integration software assumes the existence of these directories.

   You can install PBS software in these directories or create links from your installation to these directories.

```
/opt/OpenPBS # binaries
/usr/spool/PBS # configuration files
```

2. **Ensure that a** `nodes` **file exists on the master server.**

   The `nodes` file must contain a list of entries for each node and the number of processors in each node. For example,

```
hpc-u2-8 np=2  # first node, 2 processors
hpc-u2-9 np=2  # second node, 2 processors
```

   Name the file `/usr/spool/PBS/server_priv/nodes`

   To allow the clusters to be overloaded, set np to a larger value. For example:

```
hpc-u2-8 np=5000
hpc-u2-9 np=5000
```

▼ How to Configure SGE For Close Integration

The main thing you have to do is make sure each Sun CRE node is configured as an SGE execution host. Simply define a parallel environment for each queue in the SGE cluster that wil be used as a Sun CRE node. Follow these steps.

1. **Create a configuration file with the following contents.**

TABLE 4-1 SGE Configuration

pe_name	cre
user_lists	NONE
xuser_lists	NONE
start_proc_args	/bin/true
stop_proc_args	/bin/true
allocation_rule	$round_robin
control_slaves	TRUE
job_is_first_task	FALSE
queue_list	*<sge-queue sge-queue ...>*
slots	*<sum of all queue slots>*

In `user_lists`,the `NONE` value enables all users and excludes no users.

The `control_slaves` value must be `TRUE`, or SGE will prohibit parallel jobs, causing `qrsh` to exit with an error message that does not indicate the cause.

The `job_is_first_task` value must be `FALSE`, or `mprun` will count as one of the slots. If that happens, only n-1 processes will start, and the job will fail.

2. **Initialize the configuration file.**

Use the `qconf` command. For example, if you named the configuration file `pe.q`, enter:

```
% qconf -Ap pe.q
```

3. **Execute the following command for each queue that you named in the `queue_list` field of the configuration file.**

```
% qconf -mqattr qtype "BATCH INTERACTIVE PARALLEL" queue-name
```

CHAPTER 5

mpadmin: Detailed Description

This chapter describes the Sun CRE cluster administration command interface, mpadmin. Topics covered include:

- mpadmin syntax, the subcommands it supports, and other aspects of mpadmin functionality
- How to use mpadmin to perform various cluster administration tasks

mpadmin Syntax

The mpadmin command has six optional arguments, as follows:

mpadmin [**-c** *command*] [**-f** *filename*] [**-h**] [**-q**] [**-s** *cluster_name*] [**-V**]

When you invoke mpadmin with the -q or -s option or no option, mpadmin goes into the interactive mode, displaying the mpadmin prompt. In this mode, you can execute any number of mpadmin subcommands until you quit the interactive session.

Note – In the rest of this discussion, mpadmin subcommands are referred to as mpadmin commands or simply as commands.

When you invoke mpadmin with the -c, -f, -h, or -V option, mpadmin performs the requested operation and then returns to the shell level. For command arguments, you can specify most of the subcommands that are available within the mpadmin interactive environment.

Command-Line Options

TABLE 6-1 provides summary definitions of the `mpadmin` command-line options. This section describes their use.

TABLE 5-1 `mpadmin` Options

Option	Description
`-c` *command*	Executes single specified command.
`-f` *file-name*	Takes input from specified file.
`-h`	Displays help / usage text.
`-q`	Suppresses the display of a warning message when a non-superuser attempts to use restricted command mode.
`-s` *cluster-name*	Connects to the specified Sun HPC cluster.
`-V`	Displays `mpadmin` version information.

`-c` *command* – Single Command Option

Use the `-c` option when you want to execute a single `mpadmin` command and return automatically to the shell prompt. For example, the following use of `mpadmin -c` changes the location of the Sun CRE log file to `/home/wmitty/cre_messages`:

```
# mpadmin -c set logfile="/home/wmitty/cre_messages"
```

Note – Most commands that are available via the interactive interface can be invoked via the `-c` option. See “`mpadmin` Command Overview” on page 73 for an overview of the `mpadmin` command set and a list of which commands can be used as arguments to the `-c` option.

`-f` *file-name* – Take Input From a File

Use the `-f` option to supply input to `mpadmin` from the file specified by the *file-name* argument.

-h – Display Help

The -h option displays help information about mpadmin.

-q – Suppress Warning Message

Use the -q option to suppress a warning message when a non-root user attempts to invoke a restricted command.

-s *cluster-name* – Connect to Specified Cluster

Use the -s option to connect to the cluster specified by the *cluster-name* argument.

-V – Version Display Option

Use the -V option to display the version of mpadmin.

mpadmin Objects, Attributes, and Contexts

Before examining the set of mpadmin commands further, it is useful to understand three concepts that are central to the mpadmin interface: *objects*, *attributes*, and *contexts*.

mpadmin Objects and Attributes

From the perspective of mpadmin, a Sun HPC cluster consists of a system of objects, which include

- The cluster itself
- Each node contained in the cluster
- Each partition (logical group of nodes) defined in the cluster

Each type of object has a set of attributes whose values can be operated on via `mpadmin` commands. These attributes control various aspects of their respective objects, such as: whether a node is enabled or disabled (that is, whether it can be used or not), the names of partitions, and which nodes a partition contains.

Note – Sun CRE sets most cluster object attributes to default values each time it boots up. With few exceptions, *do not* change these system-defined values.

`mpadmin` Contexts

`mpadmin` commands are organized into three *contexts*, which correspond to the three types of `mpadmin` objects. These contexts are illustrated in FIGURE 6-1 and summarized below.

- `Cluster` – These commands affect cluster attributes.
- `Partition` – These commands affect partition attributes.
- `Node` – These commands affect node attributes.

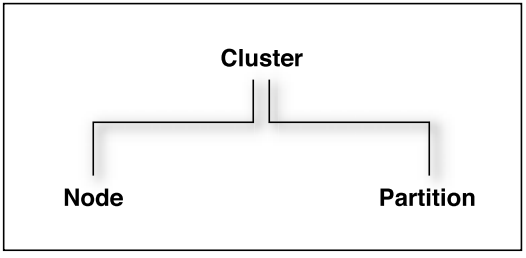


FIGURE 5-1 The `mpadmin` Contexts

Except for `Cluster`, each context is nested in a higher context: `Node` within `Cluster` and `Partition` within `Cluster`.

The `mpadmin` prompt uses one or more fields to indicate the current context. TABLE 6-2 shows the prompt format for each of the possible `mpadmin` contexts.

TABLE 5-2 `mpadmin` Prompt Formats

Prompt Formats	Context
[*cluster-name*]::	Current context = Cluster
[*cluster-name*]Node::	Current context = Node, but not a specific node
[*cluster-name*]N(*node-name*)::	Current context = a specific node
[*cluster-name*]Partition::	Current context = Partition, but not a specific partition
[*cluster-name*]P(*partition-name*)::	Current context = a specific partition

`mpadmin` Command Overview

This section describes the subcommands that `mpadmin` provides.

Types of `mpadmin` Commands

`mpadmin` provides commands for performing the following operations:

- Configuration control – These commands are used to create and delete `mpadmin` objects (nodes and partitions).
- Attribute control – These commands are used to set and reset attribute values.
- Context navigation – These commands are used to change the current context to a different context.
- Information retrieval – These commands are used to display object and attribute information.
- Miscellaneous.

Configuration Control

A Sun HPC cluster contains one or more named partitions. Each partition contains some number of specific nodes.

Sun CRE automatically creates the cluster and node objects based on the contents of the `hpc.conf` file. Partitions are the only kind of object that you are required to create and manage.

Use the `delete` command to remove partitions, but no other types of cluster objects. You remove nodes from a Sun HPC cluster by editing the `hpc.conf` file.

create

Usage:

`:: create` *object-name*

Available In: Node, Partition

The `create` command creates a new object with the name *object-name* and makes the new object the current context. Partitions can only be created from within the Partition context.

The following example creates the partition `part0`.

```
[node0] Partition:: create part0
[node0] P(part0)::
```

As the second line in the example shows, `part0` becomes the new context.

delete

Usage:

`:: delete` [*object-name*]

Available In: Node, Partition

The `delete` command deletes the object specified by the *object-name* argument. The object being deleted must either be contained in the current context or must be the current context. The first example shows a partition contained in the current context being deleted.

```
[node0] Partition:: delete part0
[node0] Partition::
```

If the current context is the object to be deleted, the *object-name* argument is optional. In this case, the context reverts to the next higher context level.

```
[node0] P(part0):: delete
[node0] Partition::
```

Attribute Control

Each `mpadmin` object has a set of attributes that can be modified. Use the `set` command to specify a value for a given attribute. Use `unset` to delete an attribute.

Note – Sun CRE requires most attributes to have their default values. Be certain to limit your attribute changes to those described in this chapter.

`set`

Usage:

`:: set` *attribute*[=*value*]

Available In: Cluster, Node, Partition

The `set` command sets the specified attribute of the current object.

You must be within the context of the target object to set its attributes. For example, to change an attribute of a specific partition, you must be in that partition's context.

To set a literal or numeric attribute, specify the desired value. The following example sets the node attribute for partition `part0`. Setting a partition's node attribute identifies the set of nodes that are members of that partition.

```
[node0] P(part0):: set nodes=node1 node2
[node0] P(part0)::
```

To change the value of an attribute that has already been set, simply set it again. The following example adds `node3` to partition `part0`.

```
[node0] P(part0):: set nodes=+node3
[node0] P(part0)::
```

As shown by this example, if the value of an attribute is a list, items can be added to or removed from the list using the + and - symbols, without repeating items that are already part of the list.

To set a Boolean attribute, specify the name of the Boolean attribute to be activated. Do not include =*value* in the expression. The following example enables partition `part0`.

```
[node0] P(part0):: set enabled
[node0] P(part0)::
```

Note – If you mistakenly set a Boolean attribute to a value—that is, if you follow a Boolean attribute's name with the =*value* field, `mpadmin` ignores the value assignment and simply considers the attribute to be active.

`unset`

Usage:

`:: unset` *attribute*

Available In: Cluster, Node, Partition

The `unset` command deletes the specified attribute from the current object. You must be within the context of an object to unset any of its attributes.

The following example disables the partition `part0` (that is, makes it unavailable for use).

```
[node0] P(part0):: unset enabled
[node0] P(part0)::
```

Note – Remember, you cannot use the `set` command to set Boolean attributes to the logical 0 (inactive) state. You must use the `unset` command.

Context Navigation

By default, `mpadmin` commands affect objects that are in the current context—that is, objects that are in the same context in which the command is invoked. For example, if the command `list` is invoked in the Node context, `mpadmin` lists all the nodes in the cluster. If `list` is invoked in the Partition context, it lists all the partitions in the cluster, as shown below:

```
[node0] Partition:: list
          part0
          part1
          part2
[node0] Partition::
```

`mpadmin` provides several context navigation commands that enable you to operate on objects and attributes outside the current context.

`current`

Usage:

`:: current` *object-name*

Available In: Cluster, Node, Partition

The `current` command changes the current context to the context of the object specified by *object-name*. The target object must exist. That is, if it is a partition, you must already have used the `create` command to create it. If the target object is a cluster or node, it must have been created by Sun CRE.

The following example changes the current context from the general Node context to the context of a specific node, `node1`.

```
[node0] Node:: current node1
[node0] N(node1)::
```

If the name of the target object does not conflict with an `mpadmin` command, you can omit the `current` command. This is illustrated by the following example, where `node1` is the name of the target object.

```
[node0] Node:: node1
[node0] N(hpc-node1)::
```

This works even when the object is in a different context.

```
[node0] Partition:: node1
[node0] N(node1)::
```

Note – The `current` command must be used when the name of the object is the same as an `mpadmin` command. For example, if you have a partition named `Partition`, its name conflicts with the command `Partition`. In this case, to make the object `Partition` the current context, you would need to include the `current` command to make it clear that the `Partition` term refers to the object and is not an invocation of the command.

---

top

Usage:

`:: top`

Available In: Node, Partition

The `top` command moves you to the Cluster context. The following example moves from the Partition context to the Cluster context.

```
[node0] Partition:: top
[node0]::
```

up

Usage:

`:: up`

Available In: Node, Partition

The `up` command moves you up one level from the current context. The following example moves from the Node context to the context of Cluster. (Since there are only two levels in the object hierarchy, the action of the `up` command is the same as that of the `top` command.)

```
[node0] N[node2] Node:: up
[node0] ::
```

node

Usage:

```
:: node
```

Available In: Cluster

The node command moves you from the Cluster context to the Node context.

```
[node0]:: node
[node0] Node::
```

partition

Usage:

```
:: partition
```

Available In: Cluster, Node

The partition command moves you from the Cluster or Node context to the Partition context.

```
[node0]:: partition
[node0] Partition::
```

Information Retrieval

The information retrieval commands display information about

- The specified object
- If no object is specified, the current context

dump

Usage:

:: dump [*object-name*]

Available In: Cluster, Node, Partition

The dump command displays the current state of the attributes of the specified object or of the current context. The object can be

- The entire cluster
- A specific partition
- All partitions in the cluster
- A specific node
- All nodes in the cluster

The `dump` command outputs objects in a specific order that corresponds to the logical order of assignment when a cluster is configured. For example, nodes are output before partitions because, when a cluster is configured, nodes must exist before they can be assigned to a partition.

The `dump` command executes in this hierarchical manner so it can be used to back up cluster configurations in a format that allows them to be easily restored at a later time.

The following example shows the `dump` command being used in this way. In this example, it is invoked using the `-c` option on the `mpadmin` command line, with the output being directed to a backup file.

```
# mpadmin -c dump > sunhpc.configuration
```

Later, when it was time to restore the configuration, `mpadmin` could read the backup file as input, using the `-f` option.

```
# mpadmin -f sunhpc.configuration
```

If you wanted to modify the configuration, you could edit the backup file before restoring it.

The following example shows the `dump` command being used to output the attribute states of the partition `part0`.

```
[node0] Partition:: dump part0
        set nodes = node1 node2 node3
        set max_total_procs = 4
        set name = part0
        set enabled
        unset no_login
[node0] Partition::
```

Note – Each attribute is output in the form of a `set` or `unset` command so that the `dump` output functions as a script.

If you are within the context of the object whose attributes you want to see, you do not have to specify its name.

```
[node0] P(part0):: dump
        set nodes = node1 node2 node3
        set max_total_procs = 4
        set enabled
        set name = part0
[node0] P(part0)::
```

list

Usage:

`:: list`

Available In: Cluster, Node, Partition

The `list` command lists all of the defined objects in the current context. The following example shows that there are three partitions defined in the Partition context.

```
[node0] Partition:: list
        part0
        part1
        part2
[node0] Partition::
```

show

Usage:

`:: show` [*object-name*]

Available In: Cluster, Node, Partition

The `show` command displays the current state of the attributes of the specified object *object-name*, which must be in the current context. The following example displays the attributes for the partition `part0`.

```
[node0] Partition:: show part0
        set nodes = node0 node1 node2 node3
        set max_total_procs = 4
        set name = part0
        set enabled
        unset no_login
[node0] Partition::
```

If, in the above example, you attempted to `show node1`, the operation fails because `node1` is not in the current context.

Miscellaneous Commands

This section describes the `mpadmin` commands `connect`, `echo`, `help`, and `quit`/`exit`.

connect

Usage:

`:: connect` *cluster-name*

Available In: Cluster

In order to access any objects or attributes in a Sun HPC cluster, you must be *connected* to the cluster.

However, connecting to a cluster ordinarily happens automatically, so you are not likely to ever need to use the `connect` command.

The environment variable `SUNHPC_CLUSTER` names a default cluster. If no other action is taken to override this default, any `mpadmin` session will connect to the cluster named by this environment variable.

If you issue the `mpadmin` command on a node that is part of a cluster, you are automatically connected to that cluster, regardless of the `SUNHPC_CLUSTER` setting.

If you are not logged in to the cluster you want to use and you do not want to use the default cluster, you can use the `mpadmin -s` option, specifying the name of the cluster of interest as an argument to the option.

Note – When Sun CRE creates a cluster, it always names it after the hostname of the cluster's master node—that is, the node on which the master daemons are running. Therefore, whenever you need to specify the name of a cluster, use the hostname of the cluster's master node.

The following example shows the `connect` command being used to connect to a cluster whose master node is `node0`.

```
[hpc-demo]:: connect node0
[node0]::
```

echo

Usage:

:: echo *text–message*

Available In: Cluster, Node, Partition

The `echo` command prints the specified text on the standard output. If you write a script to be run with `mpadmin -f`, you can include the `echo` command in the script so that it will print status information as it executes.

```
[node0]:: echo Enabling part0 and part1
Enabling part0 and part1
[node0]::
```

help

Usage:

:: help [*command*]

Available In: Cluster, Node, Partition

When invoked without a command argument, the `help` command lists the `mpadmin` commands that are available within the current context. The following example shows `help` being invoked at the `Cluster` level.

```
[node0]:: help
connect <cluster-name>  connect to a Sun HPC cluster
set <attribute>[=value] set an attribute in the current context
unset <attribute        delete an attribute in the current context
show                    show attributes in current context
dump                    show all objects on the cluster
node                    go to the node context
partition               go to the partition context
echo ...                print the rest of the line on std output
quit                    quit mpadmin
help [command]          show information about command command
? [command]             show information about command command
[node0]::
```

To get a description of a particular command, enter the command name as an argument to `help`.

If you specify a context command (`node` or `partition`), `mpadmin` lists the commands available within that context.

```
[node0]:: help node
current <node>           set the current node for future commands
create <node>            create a new node with the given name
delete [node]            delete a node
list                     list all the defined nodes
show [node]              show a node's attributes
dump [node]              show attributes for a node
set <attribute>[=value]  set the current node's attribute
unset <attribute>        delete the current node's attribute
up                       go up to the Cluster level command prompt
top                      go up to the Cluster level command prompt
echo ...                 print the rest of the line on std output
help [command]           show information about command command
? [command]              show information about command command
[node0]::
```

The "?" character is a synonym for `help`.

quit/exit

Usage:

:: quit

:: exit

Available In: Cluster, Node, Partition

Entering either quit or exit causes mpadmin to terminate and return you to the shell level.

Example:

```
[node0]:: quit
#
```

Example:

```
[node0] N(node2):: exit
#
```

Additional mpadmin Functionality

This section describes other functionality provided by mpadmin.

Multiple Commands on a Line

Because mpadmin interprets its input, if you issue more than one command on a line, mpadmin will execute them sequentially in the order they appear.

The following example shows how to display a list of nodes when not in the Node context. The `node` command switches to the Node context and the `list` command generates a list for that context.

```
[node0]:: node list
        node0
        node1
        node2
        node3
[node0] Node::
```

The following example sets the `enabled` attribute on partition `part1`. The `part1` entry acts as a command that switches the context from `part0` to `part1` and the `set` command turns on the `enabled` attribute.

```
[node0] P[part0]:: part0 set enabled
[node0] P(part0)::
```

Command Abbreviation

You can abbreviate commands to the shortest string of at least two letters so long as it is still unique within the current context.

```
[node0] Node:: pa
[node0] Partition:: li
      part0
      part1
      part2
      part3
[node0] Partition:: part2
[node0] P(part2):: sh
      set enabled
      set max_total_procs = 4
      set name = part2
      set nodes = node0 node1
[node0] P(part2)::
```

Note – The names of objects cannot be abbreviated.

Using mpadmin

This section explains how to use mpadmin to perform the principal administrative tasks involved in setting up and maintaining a Sun HPC cluster. It describes the following tasks:

- Logging in to the cluster
- Customizing cluster-level attributes
- Managing nodes
- Managing partitions

Note on Naming Partitions and Custom Attributes

You can assign names to partitions and to *custom attributes*. Custom attributes are attributes that are not part of the default Sun CRE database; they are discussed in “Setting Custom Attributes” on page 101.

Names must start with a letter and are case sensitive. The following characters can be used:

```
ABCDEFGHIJKLMNOPQRSTUVWXYZ
abcdefghijklmnopqrstuvwxyz
0123456789-_.
```

The only limit to name length is the limit imposed by Solaris on host names—it is ordinarily set at 256 characters.

Note – Do not begin an attribute name with the characters mp_. This starting sequence is reserved by Sun CRE.

Nodes and partitions have separate name spaces. Thus, you can have a partition named Parallel that contains a node named Parallel.

Logging In to the Cluster

It is assumed that you are logged in to a node that is part of the cluster you want to set up. If the node you are logged in to is not part of any cluster, set the SUNHPC_CLUSTER environment variable to the name of the target cluster. For example,

```
# setenv SUNHPC_CLUSTER node0
```

makes `node0` the default cluster. Remember, a cluster's name is the same as the host name of its master node.

Once you are connected to the cluster, you can start using `mpadmin` to perform the administrative tasks described below.

When you start up an `mpadmin` interactive session, you begin at the Cluster level. TABLE 6-3 lists the `mpadmin` commands that can be used in the Cluster context.

TABLE 5-3 Cluster-Level `mpadmin` Commands

Command	Synopsis
`connect` *cluster-name*	Connect to a Sun HPC cluster named *cluster-name*. You will not need to use this command.
`show`	Show cluster attributes.
`dump`	Show all objects in the Sun HPC cluster.
`set` *attribute[=value]*	Set a cluster-level attribute.
`unset` *attribute*	Delete a cluster-level attribute.
`node`	Enter the `Node` context.
`partition`	Enter the `Partition` context.
`echo` ...	Print the rest of the line on the standard output.
`quit / exit`	Quit `mpadmin`.
`help` [*command*] `/ ?`	Show information about commands.

Customizing Cluster-Level Attributes

This section describes various Cluster-level attributes that you may want to modify. TABLE 6-4 lists the attributes that can be changed in the Cluster context.

TABLE 5-4 Cluster-Level Attributes

Attribute	Kind	Description
`default_interactive_partition`	Value	Specifies the default partition.
`logfile`	Value	Specifies an optional output file for logging Sun CRE daemon error messages.
`administrator`	Value	Specifies an email address for the system administrator(s).

`default_interactive_partition`

This attribute specifies the default partition for running MPI jobs. Its value is used by the command `mprun`, which is described in the *Sun HPC ClusterTools Software User's Guide*.

For example, to make a partition named `part0` the default partition, enter the following in the Cluster context:

```
[node0]:: set default_interactive_partition=part0
```

When a user executes a program via `mprun`, Sun CRE decides where to run the program, based on the following criteria:

1. Check for the command-line `-p` option. If a partition is specified, execute the program in that partition. If the specified partition is invalid, the command fails.
2. Check to see if the `MPRUN_FLAGS` environment variable specifies a default partition. If so, execute the program in that partition. If the specified partition is invalid, the command fails.
3. Check to see if the `SUNHPC_PART` environment variable has a value set. If it specifies a default partition, execute the program in that partition. If the specified partition is invalid, then check to see if the user is logged in to any partition. If so, execute the program in that partition.
4. Check to see if the user is logged in to a partition. Execute the program in that partition.

5. If none of these checks yields a partition name, check for the existence of the `default_interactive_partition` attribute. If it specifies a partition, execute the program in that partition.

logfile

The `logfile` attribute allows you to log Sun CRE messages in a file separate from all other system messages. For example, if you enter:

```
[node0]:: set logfile=/home/wmitty/cre-messages
```

Sun CRE outputs its messages to the file `/home/wmitty/cre-messages`. If `logfile` is not set, Sun CRE messages are passed to `syslog`, which stores them with other system messages in `/var/adm/messages`.

Note – A full path name must be specified when setting the `logfile` attribute.

---

administrator

Set the `administrator` attribute to identify the system administrator. For example, to specify the email address of the system administrator:

```
[node0]:: set administrator="root@example.com"
```

Note the use of double quotation marks.

Managing Nodes

Ordinarily, the only administrative action that you need to take with nodes is to enable them for use. Or, if you want to temporarily make a node unavailable for use, disable it.

Other node-related administrative tasks—such as naming the nodes, identifying the master node, setting memory and process limits, and setting the node's partition attribute—are either handled by Sun CRE automatically or are controlled via partition-level attributes.

Node Commands

The table below lists the `mpadmin` commands that can be used at the Node level.

TABLE 5-5 Node-Level `mpadmin` Commands

Command	Synopsis
`current` *node*	Set the context to the specified node for future commands.
`create` *node*	Create a new node with the given name.
`delete` [*node*]	Delete a node.
`list`	List all the defined nodes.
`show` [*node*]	Show a node's attributes.
`dump` [*node*]	Show the attributes of the node.
`set` *attribute*[=*value*]	Set the specified attribute of the current node.
`unset` *attribute*	Delete the specified attribute of the current node.
`up`	Move to the next higher level (`Top`) command context.
`top`	Move to the `Top` level command context.
`echo` ...	Print the rest of the line on the standard output.
`help` [*command*]	Show information about commands (`?`).

Node Attributes

Nodes are defined by many attributes, most of which are not accessible to `mpadmin` commands. Although you are not able to affect these attributes, it can be helpful to know of their existence and meaning; hence, they are listed and briefly described in TABLE 6-6.

TABLE 6-7 lists the Node-level attributes that *can* be set via `mpadmin` commands. However, `enabled` and `max_total_procs` are the only node attributes that you can safely modify.

TABLE 5-6 Node Attributes That Cannot Be Set by the System Administrator

Attribute	Kind	Description
`cpu_idle`	Value	Percent of time CPU is idle.
`cpu_iowait`	Value	Percent of time CPU spent in I/O wait state.
`cpu_kernel`	Value	Percent of time CPU spent in kernel state.
`cpu_type`	Value	Type of CPU, for example, `sparc`.
`cpu_user`	Value	Percent of time CPU spends running user's program.
`load1`	Value	Load average for the past minute.
`load5`	Value	Load average for the past five minutes.

TABLE 5-6 Node Attributes That Cannot Be Set by the System Administrator *(Continued)*

Attribute	Kind	Description
load15	Value	Load average for the past 15 minutes.
manufacturer	Value	Manufacturer of the node, e.g., Sun_Microsystems.
mem_free	Value	Node's available RAM (in Mbytes).
mem_total	Value	Node's total physical memory (in Mbytes).
name	Value	Name of the node; this is predefined and must not be set via mpadmin.
ncpus	Value	Number of CPUs in the node.
offline	Boolean	Set automatically by the system if the tm.spmd daemon on the node stops running or is unresponsive; if set, prevents jobs from being spawned on the node.
os_arch_kernel	Value	Node's kernel architecture (same as output from arch -k, for example, sun4u).
os_name	Value	Name of the operating system running on the node.
os_release	Value	Operating system's release number.
os_release_maj	Value	Operating system's major release number.
os_release_min	Value	Operating system's minor release number,.
serial_number	Value	Hardware serial number or host ID.
swap_free	Value	Node's available swap space (in Mbytes).
swap_total	Value	Node's total swap space (in Mbytes).
update_time	Value	When this information was last updated.

TABLE 5-7 Node Attributes That Can Be Set by the System Administrator

Attribute	Kind	Description
enabled	Boolean	Set if the node is enabled, that is, if it is ready to accept jobs.
master	Boolean	Specify node on which the master daemons are running as an argument to mprun.
max_locked_mem	Value	Maximum amount of shared memory allowed to be locked down by Sun MPI processes (in Kbytes).
max_total_procs	Value	Maximum number of Sun HPC ClusterTools software processes per node.

TABLE 5-7 Node Attributes That Can Be Set by the System Administrator *(Continued)*

Attribute	Kind	Description
`min_unlocked_mem`	Value	Minimum amount of shared memory not to be locked down by Sun MPI processes (in Kbytes).
`partition`	Value	Partition of which node is a member.
`shmem_minfree`	Value	Fraction of swap space kept free for non-MPI use.

`enabled`

The attribute `enabled` is set by default when Sun CRE daemons start on a node. Unsetting it prevents new jobs from being spawned on the node.

A partition can list a node that is not enabled as a member. However, jobs will execute on that partition as if that node were not a member.

`master`

Note – You must not change this node attribute. Sun CRE automatically sets it to the hostname of the node on which the master Sun CRE daemons are running. This happens whenever the Sun CRE daemons start.

`max_locked_mem` *and* `min_unlocked_mem`

Note – You should not change these node attributes. They are described here so that you can interpret their values when node attributes are displayed via the `dump` or `show` commands.

The `max_locked_mem` and `min_unlocked_mem` attributes limit the amount of shared memory available to be locked down for use by Sun MPI processes. Locking down shared memory guarantees maximum speed for Sun MPI processes by eliminating delays caused by swapping memory to disk. However, locking physical memory can have undesirable side effects because it prevents that memory from being used by other processes on the node.

The Solaris software provides two related tunable kernel parameters:

- `tune_t_minasmem`, which is similar to `min_unlocked_mem`
- `pages_pp_maximum`, which is similar to `max_locked_mem`

The Sun CRE parameters impose limits only on MPI programs, while the kernel parameters limit all processes. Also, the kernel parameter units are pages rather than Kbytes. Refer to your Solaris documentation for more information about `tune_t_minasmem` and `pages_pp_maximum`.

max_total_procs

You limit the number of `mprun` processes allowed to run concurrently on a node by setting this attribute to an integer.

```
[node0] P(part0):: set max_total_procs=10
[node0] P(part0)::
```

If `max_total_procs` is set at the node level, that value overrides any value set at the partition level.

By default, `max_total_procs` is `unset`. Sun CRE does not impose any limit on the number of processes allowed on a node.

partition

Note – There is no need to set this attribute. Sun CRE sets it automatically if the node is included in any partition configuration(s).

A node can belong to multiple partitions, but only one of those partitions can be enabled at a time. No matter how many partitions a node belongs to, the `partition` attribute shows only one partition name—that name is always the name of the enabled partition, if one exists for that node.

shmem_minfree

Note – You should not change this node attribute. It is described here so that you can interpret its value when node attributes are displayed via the `dump` or `show` commands.

The `shmem_minfree` attribute reserves some portion of the `/tmp` file system for non-MPI use.

For example, if `/tmp` is 1 Gbyte and `shmem_minfree` is set to 0.2, any time free space on `/tmp` drops below 200 Mbytes (1 Gbyte * 0.2), programs using the MPI shared memory protocol will not be allowed to run.

This attributeexists on both nodes and partitions. If they are set to different values, the node attribute overrides the partition attribute.

Deleting Nodes

If you permanently remove a node from the Sun HPC cluster, you should then delete the corresponding node object from the Sun CRE resource database.

Recommendations

Before deleting a node:

- Remove it from any enabled partition by unsetting its `partition` attribute (automatically removing the node from the partition's `nodes` attribute list), or by removing it from the partition's `nodes` attribute list. See "Partition Attributes" on page 97 for details.
- Wait for any jobs running on it to terminate, or stop them using the `mpkill` command, which is described in the *Sun HPC ClusterTools Software User's Guide*.

Using the `delete` *Command*

To delete a node, use the `delete` command within the context of the node you want to delete.

```
[node0] N(node3):: delete
[node0] Node::
```

Managing Partitions

Partitions are logical collections of nodes that work cooperatively to run programs on the Sun HPC cluster. An MPI job can run on a single partition or on the combination of a single partition and one or more nodes that are not members of any partition. MPI jobs cannot run in multiple partitions.

You must create a partition and enable it before you can run MPI programs on your Sun HPC cluster. Once a partition is created, you can configure it to meet the specific needs of your site and enable it for use.

Once a partition is created and enabled, you can run serial or parallel jobs on it. Serial programs run on a single node of a partition. Parallel programs run on any number of nodes of a partition in parallel.

Sun CRE performs load balancing on shared partitions. When you use `mprun` to execute a program on a shared partition, Sun CRE automatically runs it on the least-loaded nodes that satisfy any specified resource requirements.

Partitions are mutable. That is, after you create and configure a partition, you can change it if your site requirements change. You can add nodes to a partition or remove them. You can change a partition's attributes. Also, since you can enable and disable partitions, you can have many partitions defined and use only a few at a time according to current needs.

There are no restrictions on the number or size of partitions, so long as no node is a member of more than one enabled partition.

Partition Commands

TABLE 6-8 lists the mpadmin commands that can be used within the partition context.

TABLE 5-8 Partition-Level `mpadmin` Commands

Command	Synopsis
`current` *partition*	Set the context to the specified partition for future commands.
`create` *partition*	Create a new partition with the given name.
`delete` [*partition*]	Delete a partition.
`list`	List all the defined partitions.
`show` [*partition*]	Show a partition's attributes.
`dump` [*partition*]	Show the attributes of a partition.
`set` *attribute*[= *value*]	Set the current partition's attribute.
`unset` *attribute*	Delete the current partition's attribute.
`up`	Move up one level in the context hierarchy.
`top`	Move to the top level in the context hierarchy.
`echo` ...	Print the rest of the line on the standard output.
`help` [*command*]	Show information about the command *command*.
`?` [*command*]	Show information about the command *command*.

Viewing Existing Partitions

Before creating a new partition, you might want to list the partitions that have already been created. To do this, use the `list` command from within the Partition context.

```
[node0] Partition:: list
        part0
        part1
[node0] Partition::
```

Creating a Partition

To create a partition, use the `create` command, followed by the name of the new partition. “Note on Naming Partitions and Custom Attributes” on page 86 discusses the rules for naming partitions.

For example:

```
[node0] Partition:: create part0
[node0] P(part0)::
```

The `create` command automatically changes the context to that of the new partition.

At this point, your partition exists by name but contains no nodes. You must assign nodes to the partition before using it. You can do this by setting the partition’s `nodes` attribute. Note these prerequisites:

- Nodes have to exist in the Sun CRE database before you can add them to partitions.
- A node must be enabled for it to be an active member of a partition. If a node is configured as a partition member, but is not enabled, it will not participate in jobs that run on that partition.

Configuring Partitions

You can configure partitions by setting and deleting their attributes using the `set` and `unset` commands. The table below TABLE 5-9shows the commonly used partition attributes.

You can combine these attributes in any way that makes sense for your site.

TABLE 5-9 Common Partitions and Their Attributes

Partition Type	Relevant Attributes	Recommended Value
Login	`no_logins`	not set
Shared	`max_total_procs`	not set or set greater than 1
Dedicated	`max_total_procs` `no_logins`	=1 set
Serial	`no_mp_jobs`	set
Parallel	`no_mp_jobs`	not set

Partitions, once created, can be enabled and disabled. This lets you define many partitions but use just a few at a time. For instance, you might want to define a number of shared partitions for development use and dedicated partitions for executing production jobs, but have only a subset available for use at a given time.

Partition Attributes

TABLE 6-10 lists the predefined partition attributes. To see their current values, use the `mpadmin show` command.

TABLE 5-10 Predefined Partition Attributes

Attribute	Kind	Description
`enabled`	Boolean	Set if the partition is enabled, that is, if it is ready to accept logins or jobs.
`max_locked_mem`	Value	Maximum amount of shared memory allowed to be locked down by MPI processes (in Kbytes).
`max_total_procs`	Value	Maximum number of simultaneously running processes allowed on each node in the partition.
`min_unlocked_mem`	Value	Minimum amount of shared memory that may not be locked down by MPI processes (in Kbytes).
`name`	Value	Name of the partition.
`no_logins`	Boolean	Disallow logins.
`no_mp_tasks`	Boolean	Disallow multiprocess parallel jobs.
`nodes`	Value	List of nodes in the partition.
`shmem_minfree`	Value	Fraction of swap space kept free for non-MPI use.

`enabled`

Set the `enabled` attribute to make a partition available for use.

By default, the `enabled` attribute is *not* set when a partition is created.

max_locked_mem *and* min_unlocked_mem

You should not change these partition attributes.

max_total_procs

To limit the number of simultaneously running mprun processes allowed on each node in a partition, set the max_total_procs attribute in a specific Node context or in the Partition context.

```
[node0] P(part0):: set max_total_procs=10
[node0] P(part0)::
```

You can set max_total_procs if you want to limit the load on a partition. For example, if max_total_procs is set to 3 and there are three nodes in the partition, then the maximum mprun -np value for programs running in that partition is 9. By default, max_total_procs is unset.

If max_total_procs is set at the node level, that value overrides any value set at the partition level.

Sun CRE does not impose any limit on the number of processes allowed on a node.

name

The name attribute is set when a partition is created. To change the name of a partition, set its name attribute to a new name.

```
[node0] P(part0):: set name=part1
[node0] P(part1)::
```

See “Note on Naming Partitions and Custom Attributes” on page 86 for partition naming rules.

no_logins

To prohibit users from logging in to a partition, set the no_logins attribute.

```
[node0] P(part1):: set no_logins
[node0] P(part1)::
```

no_mp_jobs

To prohibit multiprocess parallel jobs from running on a partition—that is, to make a serial partition—set the no_mp_jobs attribute.

```
[node0] P(part1):: set no_mp_jobs
[node0] P(part1)::
```

nodes

To specify the nodes that are members of a partition, set the partition's nodes attribute.

```
[node0] P(part1):: set nodes=node1
[node0] P(part1):: show
        set nodes = node1
        set enabled
[node0] P(part1)::
```

The value you give the nodes attribute defines the entire list of nodes in the partition. To add a node to an already existing node list without retyping the names of nodes that are already present, use the + (plus) character.

```
[node0] P(part1):: set nodes=+node2 node3
[node0] P(part1):: show
        set nodes = node0 node1 node2 node3
        set enabled
[node0] P(part1)::
```

Similarly, you can use the – (minus) character to remove a node from a partition.

To assign a range of nodes to the nodes attribute, use the : (colon) syntax. This example assigns to part0 all nodes whose names are alphabetically greater than or equal to node0 and less than or equal to node3:

```
[node0] P(part1):: set nodes = node0:node3
[node0] P(part1)::
```

Setting the nodes attribute of an enabled partition has the side effect of setting the partition attribute of the corresponding nodes. Continuing the example, setting the nodes attribute of part1 affects the partition attribute of node2:

```
[node0] P(part1):: node node2
[node0] N(node2):: show
        set partition = part1
[node0] N(node2)::
```

A node cannot be a member of more than one enabled partition. If you try to add a node that is already in an enabled partition, mpadmin returns an error message.

```
[node0] P(part1):: show
        set nodes = node0 node1 node2 node3
        set enabled
[node0] P(part1):: current part0
[node0] P(part0):: set enabled
[node0] P(part0):: set nodes=node1
mpadmin: node1 must be removed from part1 before it can be added
to part0
```

Unsetting the nodes attribute of an enabled partition has the side effect of unsetting the partition attribute of the corresponding node.

Unsetting the nodes attribute of a disabled partition removes the nodes from the partition but does not change their partition attributes.

shmem_minfree

Use the shmem_minfree attribute to reserve some portion of the /tmp file system for non-MPI use.

For example, if /tmp is 1 Gbyte and shmem_minfree is set to 0.2, any time free space on /tmp drops below 200 Mbytes (1 Gbyte * 0.2), programs using the MPI shared memory protocol will not be allowed to run.

This attribute exists on both nodes and partitions. If both are set, the node's shmem_minfree attribute overrides the partition's shmem_minfree attribute.

Enabling Partitions

A partition must be enabled before users can run programs on it. Before enabling a partition, you must disable any partitions that share nodes with the partition that you are about to enable.

To enable a partition, set its `enabled` attribute.

```
[node0] P(part0):: set enabled
[node0] P(part0)::
```

Now the partition is ready for use.

Enabling a partition has the side effect of setting the `partition` attribute of every node in that partition.

If you try to enable a partition that shares a node with another enabled partition, `mpadmin` prints an error message.

```
[node0] P(part1):: show
        set nodes = node1 node2 node3
        set enabled
[node0] P(part1):: current part2
[node0] P(part2):: show
        set nodes = node1
[node0] P(part2):: set enabled
mpadmin: part1/node1: partition resource conflict
```

Disabling Partitions

To disable a partition, unset its `enabled` attribute.

```
[node0] P(part0):: unset enabled
[node0] P(part0)::
```

Now the partition can no longer be used.

Any jobs that are running on a partition when it is disabled will continue to run. After disabling a partition, you should either wait for any running jobs to terminate or stop them using the `mpkill` command. This is described in the *Sun HPC ClusterTools Software User's Guide.*

Deleting Partitions

Delete a partition when you do not plan to use it anymore.

Note – Although it is possible to delete a partition without first disabling it, you should disable the partition by unsetting its `enabled` attribute before deleting it.

To delete a partition, use the `delete` command in the context of the partition you want to delete.

```
[node0] P(part0):: delete
[node0] Partition::
```

Setting Custom Attributes

Sun HPC ClusterTools software does not limit you to the attributes listed. You can define new attributes as desired.

For example, if a node has a special resource that will not be flagged by an existing attribute, you may want to set an attribute that identifies that special characteristic. In the following example, node `node3` has a frame buffer attached. This feature is captured by setting the custom attribute `has_frame_buffer` for that node.

```
[node0] N(node3):: set has_frame_buffer
[node0] N(node3)::
```

Users can then use the attribute `has_frame_buffer` to request a node that has a frame buffer when they execute programs. For example, use the following `mprun` command lines to select a node with or without a frame buffer, respectively:

```
% mprun -R "has_frame_buffer"
% mprun -R "\!has_frame_buffer"
```

See “Note on Naming Partitions and Custom Attributes” on page 86 for restrictions on attribute names.

CHAPTER 6

hpc.conf Configuration File

This chapter discusses the Sun HPC ClusterTools software configuration file `hpc.conf`, which defines various attributes of a Sun HPC cluster. A single `hpc.conf` file is shared by all the nodes in a cluster. It resides in `/opt/SUNWhpc/etc`.

Note – This configuration file is also used on LSF-based clusters, but it resides in a different location.

The `hpc.conf` file is organized into functional sections, which are summarized below and illustrated in CODE EXAMPLE 6-1.

- `ShmemResource` section defines certain shared memory attributes.
- `MPIOptions` section defines certain MPI parameters.
- `CREOptions` section controls the logging of system events, the handling of daemon core files, the authentication options, the default resource manager, and the controls placed on `mprun`'s ability to launch programs in batch mode.
- `PMODULES` section names and locates the protocol modules that are available for communication on the cluster.
- `PM` sections list and rank the protocol modules that are available for communication on the cluster.
- `HPCNodes` section is not used by Sun CRE. It applies only in an LSF-based runtime environment.

Sun HPC ClusterTools software is distributed with an `hpc.conf` template, which resides by default in `/opt/SUNWhpc/examples/rte/hpc.conf.template`. If you wish to customize the configuration settings, you should copy this template file to `/opt/SUNWhpc/etc/hpc.conf` and edit it.

Each configuration section is bracketed by a `Begin/End` keyword pair and, when a parameter definition involves multiple fields, the fields are separated by spaces.

CODE EXAMPLE 6-1 General Organization of the `hpc.conf` File

```
# Begin ShmemResource
# ...
# End ShmemResource

# Begin MPIOptions Queue=
# ...
# End MPIOptions

# Begin CREOptions Server=
# ...
# End CREOptions

# Begin HPCNodes
# ...
# End HPCNodes

Begin PMODULES
...
End PMODULES

Begin PM=shm
...
End PM

Begin PM=rsm
...
End PM

Begin PM=tcp
...
End PM
```

Note – When any changes are made to `hpc.conf`, the system should be in a quiescent state. To ensure that it is safe to edit `hpc.conf`, shut down the nodal and master Sun CRE daemons as described in “Stopping and Restarting Sun CRE” on page 14. If you change the `PMODULES` or `PM=rsm` sections, you must also stop the RSM daemon `hpc_rsmd`. See “RSM Daemon” on page 19.

ShmemResource Section

The ShmemResource section provides the administrator with two parameters that control allocation of shared memory and swap space: MaxAllocMem and MaxAllocSwap. This special memory allocation control is needed because some Sun HPC ClusterTools software components use shared memory.

CODE EXAMPLE 6-2 shows the ShmemResource template that is in the hpc.conf file that is shipped with Sun HPC ClusterTools software.

CODE EXAMPLE 6-2 ShmemResource Section Example

```
#Begin ShmemResource
#MaxAllocMem  0x7fffffffffffffff
#MaxAllocSwap 0x7fffffffffffffff
#End ShmemResource
```

To set MaxAllocMem and/or MaxAllocSwap limits, remove the comment character (#) from the start of each line and replace the current value, 0x7fffffffffffffff, with the desired limit.

Guidelines for Setting Limits

The Sun HPC ClusterTools software internal shared memory allocator permits an application to use swap space, the amount of which is the smaller of:

- The value (in bytes) given by the MaxAllocSwap parameter
- 90% of available swap on a node

If MaxAllocSwap is not specified, or if zero or a negative value is specified, 90% of a node's available swap is used as the swap limit.

The MaxAllocMem parameter can be used to limit the amount of shared memory that can be allocated. If a smaller shared memory limit is not specified, the shared memory limit is 90% of available physical memory.

The following Sun HPC ClusterTools software components use shared memory:

- Sun CRE uses shared memory to hold cluster and job table information. Its memory use is based on cluster and job sizes and is not controllable by the user. Shared memory space is allocated for Sun CRE when it starts up and is not affected by MaxAllocMem and MaxAllocSwap settings. This ensures that Sun CRE can start up no matter how low these memory-limit variables have been set.

- MPI uses shared memory for communication between processes that are on the same node.
- Sun S3L uses shared memory for storing data. An MPI application can allocate parallel arrays whose subgrids are in shared memory. This is done with the utility `S3L_declare_detailed()`.

Note – Sun S3L supports a special form of shared memory known as *Intimate Shared Memory* (ISM), which reserves a region in physical memory for shared memory use. What makes ISM space special is that it is not swappable and, therefore, cannot be made available for other use. For this reason, the amount of memory allocated to ISM should be kept to a minimum.

Note – Shared memory and swap space limits are applied per-job on each node.

If you have set up your system for dedicated use (only one job at a time is allowed), you should leave `MaxAllocMem` and `MaxAllocSwap` undefined. This allows jobs to maximize use of swap space and physical memory.

If, however, multiple jobs will share a system, you may want to set `MaxAllocMem` to some level below 50% of total physical memory. This reduces the risk of having a single application lock up physical memory. How much below 50% you choose to set it depends on how many jobs you expect to be competing for physical memory at any given time.

Note – When users make direct calls to `mmap`(2) or `shmget`(2), they are not limited by the `MaxAllocMem` and `MaxAllocSwap` variables. These utilities manipulate shared memory independently of the `MaxAllocMem` and `MaxAllocSwap` values.

`MPIOptions` Section

The `MPIOptions` section provides a set of options that control MPI communication behavior in ways that are likely to affect message-passing performance. It contains a template showing some general-purpose option settings, plus an example of alternative settings for maximizing performance. These examples are shown in CODE EXAMPLE 6-3.

- **G**eneral-purpose, multiuser settings – The template in the `MPIOptions` section is designed for general-purpose use at times when multiple message-passing jobs will be running concurrently.

- Performance settings – The second example is designed to maximize the performance of message-passing jobs when only one job is allowed to run at a time.

Note – The first line of the template contains the phrase "`Queue=hpc`." This line indicates a queue in the LSF batch runtime environment, which uses the same `hpc.conf` file as Sun CRE. For LSF, the settings apply only to the specified queue. For Sun CRE, the settings apply across the cluster.

The options in the general-purpose template are the same as the default settings for the Sun MPI library. In other words, you do not have to uncomment the general-purpose template to have its option values be in effect. This template is provided in the `MPIOptions` section so that you can see what options are most beneficial when operating in a multiuser mode.

If you want to use the `performance` settings, do the following:

- Delete the comment character (#) from the beginning of each line of the performance example, including the `Begin MPIOptions` and `End MPIOptions` lines.
- On Sun CRE-based clusters, delete the "`Queue=performance`" phrase from the `Begin MPIOptions` line.

The resulting template should appear as follows:

```
Begin MPIOptions
coscheduling            off
spin                    on
End MPIOptions
```

CODE EXAMPLE 6-3 `MPIOptions` Section Example

```
# The following is an example of the options that affect the run time
# environment of the MPI library.  The listings below are identical to
# the default settings of the library.  The "Queue=hpc" phrase makes
# this an LSF-specific entry, and only for the Queue named hpc.  These
# options are a good choice for a multiuser Queue.  To be recognized
# by CRE, the "Queue=hpc" needs to be removed.
#
# Begin MPIOptions Queue=hpc
# coscheduling  avail
# pbind        avail
# spindtimeout   1000
# progressadjust   on
# spin      off
#
# shm_numpostbox           16
# shm_shortmsgsize        256
# rsm_maxsegsize      1048576
# rsm_numpostbox           15
# rsm_shortmsgsize        401
# rsm_maxstripe            2
# rsm_links          wrsm0,1
# maxprocs_limit   2147483647
# maxprocs_default       4096
#
# End MPIOptions

# The listing below is a good choice when trying to get maximum
# performance out of MPI jobs that are running in a Queue that
# allows only one job to run at a time.
#
# Begin MPIOptions Queue=performance
# coscheduling             off
# spin                      on
# End MPIOptions
```

TABLE 6-1 provides brief descriptions of the MPI runtime options that can be set in `hpc.conf`. Each description identifies the default value and describes the effect of each legal value.

Some MPI options not only control a parameter directly, they can also be set to a value that passes control of the parameter to an environment variable. Where an MPI option has an associated environment variable, TABLE 6-1 names the environment variable.

TABLE 6-1 MPI Runtime Options

Option	Values		Description
	Default	Other	
coscheduling	avail		Allows spind use to be controlled by the environment variable MPI_COSCHED. If MPI_COSCHED=0 or is not set, spind is not used. If MPI_COSCHED=1, spind must be used.
		on	Enables coscheduling; spind is used. This value overrides MPI_COSCHED=0.
		off	Disables coscheduling; spind is not to be used. This value overrides MPI_COSCHED=1.
pbind	avail		Allows processor binding state to be controlled by the environment variable MPI_PROCBIND. If MPI_PROCBIND=0 or is not set, no processes will be bound to a processor. This is the default. If MPI_PROCBIND=1, all processes on a node will be bound to a processor.
		on	All processes will be bound to processors. This value overrides MPI_PROCBIND=0.
		off	No processes on a node are bound to a processor. This value overrides MPI_PROCBIND=1.
spindtimeout	1000		When polling for messages, a process waits 1000 milliseconds for spind to return. This equals the value to which the environment variable MPI_SPINDTIMEOUT is set.
		integer	To change the default timeout, enter an integer value specifying the number of milliseconds the timeout should be.
progressadjust	on		Allows user to set the environment variable MPI_SPIN.
		off	Disables user's ability to set the environment variable MPI_SPIN.

TABLE 6-1 MPI Runtime Options *(Continued)*

Option	Values		Description
	Default	Other	
shm_numpostbox	16		Sets to 16 the number of postbox entries that are dedicated to a connection endpoint. This equals the value to which the environment variable MPI_SHM_NUMPOSTBOX is set. (See the *Sun HPC ClusterTools Software Performance Guide* for details.)
		integer	To change the number of dedicated postbox entries, enter an integer value specifying the desired number.
shm_shortmsgsize	256		Sets to 256 the maximum number of bytes a short message can contain. This equals the default value to which the environment variable MPI_SHM_SHORTMSGSIZE is set.
		integer	To change the maximum-size definition of a short message, enter an integer specifying the maximum number of bytes it can contain.
rsm_numpostbox	15		Sets to 15 the number of postbox entries that are dedicated to a connection endpoint. This equals the value to which the environment variable MPI_RSM_NUMPOSTBOX is set.
		integer	To change the number of dedicated postbox entries, enter an integer specifying the desired number.
rsm_shortmsgsize	401		Sets to 401 the maximum number of bytes a short message can contain when sent via RSM without using buffers. This equals the value to which the environment variable MPI_RSM_SHORTMSGSIZE is set.
		integer	To change the maximum-size definition of a short message, enter an integer specifying the maximum number of bytes it can contain.

TABLE 6-1 MPI Runtime Options *(Continued)*

Option	Values		Description
	Default	Other	
rsm_maxstripe	2		Sets to 2 the maximum number of interfaces per stripe that can be used. (The default is the number of interfaces in the cluster, with a maximum of 64.) This equals the value to which the environment variable MPI_RSM_MAXSTRIPE is set.
		integer	To change the maximum number of stripes that can be used, enter an integer specifying the desired limit.
rsm_links	wrsm0,1		Defines the controllers/links that can be used on each node for RSM communication.
rsm_maxsegsize	(see note at right)		Sets to 17179869184 the maximum size segment that can be created for communication purposes.
		integer	To change the maximum size of an RSM segment, enter an integer specifying the desired limit.
maxprocs_default	4096		Sets to 4096 the number of processes an MPI process may be connected to at any one time. It includes processes in the same MPI job and processes in jobs that are currently connected to the MPI process. This equals the value to which the environment variable MPI_MAXPROCS is set.
		integer	To change the maximum number of processes an MPI process may be connected to at any one time, enter an integer specifying the desired limit. The value may not exceed the setting for the option maxprocs_limit.

TABLE 6-1 MPI Runtime Options *(Continued)*

Option	Values		Description
	Default	Other	
maxprocs_limit		*integer*	The maximum process table size a user may set MPI_MAXPROCS to. If the option maxprocs_default is not set, the user is able to specify a value up to MAX_INT.
spin	off		Sets the MPI library spin policy to spin nonaggressively. This equals the value to which the environment variable MPI_SPIN is set.
		on	Sets the MPI library to spin aggressively.

Setting MPI Spin Policy

An MPI process often has to wait for a particular event, such as the arrival of data from another process. If the process checks (*spins*) for this event continuously, it consumes CPU resources that may be deployed more productively for other purposes.

The administrator can direct that the MPI process instead register events associated with shared memory or remote shared memory (RSM) message passing with the spin daemon `spind`, which can spin on behalf of multiple MPI processes (*coscheduling*). This frees up multiple CPUs for useful computation. The `spind` daemon itself runs at a lower priority and backs off its activities with time if no progress is detected.

The `SUNWrte` package implements the `spind` daemon, which is not directly user callable.

The cluster administrator can control spin policy in the `hpc.conf` file. The attribute `coscheduling`, in the `MPIOptions` section, can be set to `avail`, `on`, or `off`.

- `avail` (the default) means that spin policy is determined by the setting of the environment variable `MPI_COSCHED`. If `MPI_COSCHED` is set to zero or is not set, `spind` is not used. If `MPI_COSCHED` is set to one, `spind` must be used.
- `on` means that `spind` must be used by MPI processes that wish to block on shared-memory communication. This value overrides `MPI_COSCHED=0`.
- off means that `spind` cannot be used by MPI processes. This value overrides `MPI_COSCHED=1`.

The cluster administrator can also change the setting of the attribute `spindtimeout`, indicating how long a process waits for `spind` to return. The default is 1000 milliseconds.

For tips on determining spin policy, see the man page for `MPI_COSCHED`. In general, the administrator may wish to force use of `spind` for heavily used development partitions where performance is not a priority. On other partitions, the policy could be set to `avail`, and users can set `MPI_COSCHED=0` for runs where performance is needed.

`CREOptions` Section

The CREOptions section controls the behavior of Sun CRE in logging system events, handling daemon core files, and authenticating users and programs.

The template `hpc.conf` file contains the default settings for these behaviors for the current cluster. These settings are shown in CODE EXAMPLE 6-4.

CODE EXAMPLE 6-4 `CREOptions` Section Example

```
Begin CREOptions
enable_core       off
corefile_name     core
syslog_facility   daemon
auth_opt          sunhpc_rhosts
max_pub_names     256
default_rm        cre
allow_mprun       *
End CREOptions
```

Specifying the Cluster

The cluster is specified by appending the tag Server=*master-node-name* to the `Begin CREOptions` line:

`Begin CREOptions Server=`*master-node-name*

If the node name supplied does not match the name of the current master node, then this section is ignored.

It is possible to have two `CREOptions` sections. The section without a tag is always processed first. Then the section with a matching *master-node-name* adds to or overrides the previous settings.

Logging System Events

By default, Sun CRE uses the `syslog` facility to log system events, as indicated by the entry `syslog_facility daemon`. Other possible values are `user`, `local0`, `local1`, `...`, `local7`. See the man pages for `syslog(2)`, `syslogd(8)`, and `syslog.conf(5)` for information on the `syslog` facility.

Note – In rare cases, the Sun CRE daemons may log errors to the default system log. This occurs when an error is generated before the system has read the value of `syslog_facility` in `hpc.conf`.

Enabling Core Files

By default, core files are disabled for Sun CRE daemons. The administrator may enable core files by changing `enable_core off` to `enable_core on`. The administrator may also specify where daemon core files are saved by supplying a value for `corefile_name`. See `coreadm(1M)`for the possible naming patterns. For example:

```
corefile_name /var/hpc/core.%n.%f.%p
```

This would cause any core files to be placed in `/var/hpc` with the name `core` modified by node name (`%n`), executable file name (`%f`), and process ID (`%p`). (Note that only daemon core files are affected by the `CREOptions` section; user programs are not affected.)

Enabling Authentication

Authentication may be enabled by changing the `auth_opt` value from `sunhpc_rhosts` (the default) to `rhosts`, `des`, or `krb5`. These values indicate DES or Kerberos Version 5, respectively. See “Authentication and Security” on page 34 for additional steps needed to establish the chosen authentication method.

Changing the Maximum Number of Published Names

By default, a single job may publish a maximum of 256 names. To increase or reduce that number, change this line:

`max_pub_names` *maximum-number-of-names*

Identifying A Default Resource Manager

If you have only one resource manager installed, you can save users the trouble of entering the -x *resource-manager* option each time they use they `mprun` command by specifying a default. Enter this line:

`default_rm` *resource-manager-name*

Limiting `mprun`'s Ability to Launch Programs in Batch Mode

If you need to restrict `mprun`'s ability to launch programs while running in batch mode, use the `allow_mprun` field. By default, the field is set to:

```
allow_mprun *
```

The asterisk indicates that no restrictions have been placed on mprun. The asterisk is equivalent to having no entry in the `CREOptions` section of `hpc.conf`.

If you need to set restrictions, you must:

- Change the value of `allow_mprun`
- Create the file `sunhpc.allow` to specify the restrictions

Instructions are provided in "How to Configure the `hpc.conf` File" on page 48, and "How to Configure the `sunhpc.allow` File" on page 49.

`HPCNodes` Section

This section is used only in a cluster that is using LSF as its workload manager, not Sun CRE. Sun CRE ignores the `HPCNodes` section of the `hpc.conf` file.

`PMODULES` Section

The `PMODULES` section provides the names and locations of the protocol modules (PMs) which the run-time system is to discover and make available for use for communication in the cluster.

When a Sun CRE-based cluster is being started, an instance of the daemon `tm.omd` is started on each node. This daemon is responsible for discovering various information about a node, including which PMs are available for that node. The `tm.omd` daemon looks in the `hpc.conf` file for a list of PMs that may be available. It then opens the PMs, and calls an interface discovery function to find out if the PM has interfaces that are up and running. This information is returned to the `tm.omd` and stored away in the cluster database.

The `PMODULES` section of the `hpc.conf` file lists each PM by name and gives the location (or default location) where it may be found. The template `hpc.conf` looks like this:

```
# PMODULE LIBRARY
Begin PMODULES
shm      ()
rsm      ()
tcp      ()
End PMODULES
```

Three PMs are shipped with Sun HPC ClusterTools software and included in the template `hpc.conf` file. These are:

- `shm` PM, used for on-node communication
- `tcp` PM, used (typically) for internode communication on TCP-IP-compatible interconnects
- `rsm` PM, used (typically) for internode communication on the Sun Fire™ Link interconnect

The template `hpc.conf` file specifies the location of all three PMs as `()`, which indicates the default location. The default location is `/opt/SUNWhpc/lib` for 32-bit daemons, and `/opt/SUNWhpc/lib/sparcv9` for 64-bit daemons.

The administrator has the option of putting PM libraries in a location other than the default. This is useful, for instance, when a new user-defined PM is being developed. For PMs located in a directory other than the default, the administrator must put the absolute pathname in the `hpc.conf` file. For example:

```
# PMODULE LIBRARY
Begin PMODULES
tcp      ()
shm      /home/jbuffett/libs
End PMODULES
```

In this example, the tcp libraries are located in the default location and the shm libraries are located in /home/jbuffett/libs. In a 64-bit environment, sparcv9 is automatically added to the pathname. Thus, this hpc.conf entry indicates that the shm PM libraries would be found in /home/jbuffett/libs/sparcv9.

PM Section

The hpc.conf file contains a PM section for each available protocol module. The section gives standard information (name of interface and its preference ranking) for the PM, along with additional information for some types of PMs.

The name of the PM being described appears on the same line as the keyword PM with an equal sign and no spaces between them. This example shows the PM sections provided for the shm and rsm PMs.

```
# SHM settings
# NAME  RANK
Begin PM=shm
shm     5
End PM
# RSM settings
# NAME  RANK  AVAIL
Begin PM=rsm
wrsm    20    1
End PM
```

The NAME and RANK columns must be filled in for all PMs. The shm PM requires only these two standard items of information; the rsm PM has an additional field called AVAIL.

NAME Column

The name of the interface indicates the controller type and, optionally, a numbered interface instance. Interface names not ending with a number are wildcards; they specify default settings for all interfaces of that type. The name can be can be between 1 and 32 characters in length.

If interfaces are specified by name after a wildcard entry, the named entries take precedence.

RANK Column

The rank of an interface is the order in which that interface is preferred over other interfaces, with the lowest-ranked interface the most preferred. That is, if an interface with a rank of 0 is available when a communication operation begins, it will be selected for the operation before interfaces with ranks of 1 or greater. Likewise, an available rank 1 interface will be used before interfaces with a rank of 2 or greater.

Note – Because `hpc.conf` is a shared, cluster-wide configuration file, the rank specified for a given interface will apply to all nodes in the cluster.

Network ranking decisions are usually influenced by site-specific conditions and requirements. Although interfaces connected to the fastest network in a cluster are often given preferential ranking, raw network bandwidth is only one consideration. For example, an administrator might decide to dedicate one network that offers very low latency, but not the fastest bandwidth, to all communication within a cluster and use a higher-capacity network for connecting the cluster to other systems.

Rank can also be specified for interface instances within a `PM` section. For example, consider a customized `hpc.conf` entry like this:

```
# RSM Settings
# NAME     RANK  AVAIL
Begin PM=rsm
wrsm       15    1
wrsm0      10    1
wrsm1      20    1
wrsm2      30    1
wrsm3      40    1
End PM=rsm
```

If controllers `wrsm0` and `wrsm1` could be used to establish connections to the same process, `wrsm0` would always be chosen, since it has the lower ranking number.

Note – If multiple interfaces have the same ranking, the `rsm` PM will use the MPI options `rsm_links` and `rsm_maxstripe` to reduce the number of interfaces and use the remaining interface(s) for the connection. If more than one interface is resolved for making the connection, the `rsm` PM will stripe the connection, using all interfaces resolved.

AVAIL Column

The rsm PM section contains an additional column headed AVAIL. This value indicates whether a controller is (1) or is not (2) available for RSM communication.

Configuring Out Controllers

The AVAIL column can be used to configure out one or more controllers, perhaps for maintenance on the network. If all controllers on the network are going to be unavailable, the administrator can change the availability of the wildcard entry to 0. If only certain controllers will be unavailable, the administrator can leave the wildcard entry set to 1 but add entries set to 0 availability for named instances. (Remember to stop the Sun CRE daemons and the RSM daemon hpc_rsmd when editing the hpc.conf file and restart them afterward.)

```
# RSM Settings
# NAME     RANK  AVAIL
Begin PM=rsm
wrsm       15    1
wrsm2      20    0
wrsm3      15    0
End PM=rsm
```

Note – Alternatively, the administrator can use the option rsm_links in the MPIOptions section to configure out one or more controllers. See “Configuring Out Network Controllers” on page 119.

Enabling Software Striping

Another use for the AVAIL column is to enable software-controlled striping of messages over network controllers. When a message is submitted for transmission over the network, the rsm PM distributes the message over as many network interfaces as are available with the same preference ranking, up to the limit of 8 links.

Note – Software-controlled message striping is most useful for interconnect technology that does not support hardware-controlled striping. The hardware striping performed by some interconnects is generally preferable to the software-controlled striping described here.

In striped communication, a message is split into smaller packets and transmitted in two or more parallel streams over a set of network controllers that have been logically combined into a *stripe-group*.

The AVAIL column allows the administrator to include individual network interfaces in a (software) *stripe-group pool*. Members of this pool are available to be included in logical stripe groups as long as they have the same preference ranking. These stripe groups are formed on an as-needed basis, selecting interfaces from this stripe-group pool.

To include an interface in a stripe-group pool, set its AVAIL value to 1. To exclude an interface from the pool, specify 0.

Stripe-group membership is optional so you can reserve some network bandwidth for non-striped use (assuming the network has another PM enabled). To do so, simply set AVAIL to 0 on the network interface(s) you wish to reserve in this way.

TCP-IP PM Section

The PM section provided for the tcp PM in the template hpc.conf file contains the standard NAME and RANK columns, along with several placeholder columns that are not used at this time. The default TCP settings (and placeholders) are shown in CODE EXAMPLE 6-5.

CODE EXAMPLE 6-5 PM=tcp Section Example

```
# TCP settings
# NAME  RANK    MTU     STRIPE  LATENCY BANDWIDTH
Begin PM=tcp
midn    0      16384  0       20      150
idn1    0      16384  0       20      150
wrsmd   25     32768  0       20      150
mscid   30     32768  0       20      150
scid    40     32768  0       20      150
mba     50     8192   0       20      150
ba      60     8192   0       20      150
mfa     70     8192   0       20      150
fa      80     8192   0       20      150
macip   90     8192   0       20      150
acip    100    8192   0       20      150
manfc   110    16384  0       20      150
anfc    120    16384  0       20      150
mbf     130    4096   0       20      150
bf      140    4096   0       20      150
mbe     150    4096   0       20      150
be      160    4096   0       20      150
mqfe    163    4096   0       20      150
qfe     167    4096   0       20      150
mhme    170    4096   0       20      150
hme     180    4096   0       20      150
mle     190    4096   0       20      150
le      200    4096   0       20      150
msmc    210    4096   0       20      150
smc     220    4096   0       20      150
lo      230    4096   0       20      150
End PM
```

The template hpc.conf file identifies the network interfaces that are included in the TCP PM section. The networks with the prefix “m” are for Enterprise 10000 alternate pathing support, and should be used in preference to the underlying interface (thus their lower ranking).

Note – Inclusion of any network interface in this file does not imply that Sun Microsystems supports, or intends to support, that network.

Propagating hpc.conf Information

Whenever hpc.conf is changed, the Sun CRE database must be updated with the new information. After all required changes to hpc.conf have been made, restart the Sun CRE daemons on all cluster nodes. For example, to start the daemons on cluster nodes node1 and node2 from a central host, enter

```
# ./ctstartd -n node1,node2 -r connection_method
```

where *connection_method* is rsh, ssh, or telnet. Or, you can specify a nodelist file instead of listing the nodes on a command line.

```
# ./ctstopd -N /tmp/nodelist -r connection_method
```

where */tmp/nodelist* is absolute path to a file containing the names of the cluster nodes, with each name on a separate line.

CHAPTER 7

Maintenance and Troubleshooting

This chapter describes some procedures you can use for preventive maintenance and troubleshooting. The topics covered are:

Cleaning Up Defunct Sun CRE Jobs

One preventive maintenance practice that can be beneficial is the routine cleanup of defunct jobs. There are several types of such jobs:

- Jobs that have exited, but still appear in `mpps` output
- Jobs that have not terminated, but need to be removed
- Jobs that have orphan processes

Removing Sun CRE Jobs That Have Exited

When a job does not exit cleanly, it is possible for all of a job's processes to have reached a final state, but the job object itself to not be removed from the Sun CRE database. The following are two indicators of such incompletely exited jobs:

- A process (identified by `mpps`) in the `EXIT`, `SEXIT`, `FAIL`, or `CORE` state
- A Prism main window that will not close or exit

If you see a job in one of these defunct states, perform the following steps to clear the job from the Sun CRE database:

1. **Execute** `mpps -e` **again in case Sun CRE has had time to update the database (and remove the job).**

2. **If the job is still running, kill it, specifying its job ID.**

   % **mpkill** *jid*

   If `mpps` continues to report the killed job, use the `-C` option to `mpkill` to remove the job object from the Sun CRE database. This must be done as superusee from the master node.

   # **mpkill -C** *jid*

Removing Sun CRE Jobs That Have Not Terminated

The second type of defunct job includes jobs that are waiting for signals from processes on nodes that have gone off line. The `mpps` utility displays such jobs in states such as `RUNNING`, `EXITING`, `SEXTNG`, or `CORNG`.

Note – If the job-killing option of `tm.watchd` (`-Yk`) is enabled, Sun CRE handles such situations automatically. This section assumes this option is not enabled.

Kill the job using:

% **mpkill** *jid*

There are several variants of the `mpkill` command, similar to the variants of the Solaris `kill` command. You may also use:

% **mpkill -9** *jid*

or

% **mpkill -I** *jid*

If these do not succeed, execute `mpps -pe` to display the unresponsive processes. Then, execute the Solaris `ps` command on each of the nodes listed. If those processes still exist on any of the nodes, you can remove them using `kill -9` *pid*.

Once you have eliminated defunct jobs, data about the jobs may remain in the Sun CRE database. As superuser from the master node, use `mpkill -C` to remove this residual data.

Killing Orphaned Processes

When the `tm.watchd` –`Yk` option has been enabled, the watch daemon marks processes `ORPHAN` if they run on nodes that have gone off line. If the node resumes communication with the Sun CRE daemons, the watch daemon will kill the `ORPHAN` processes. If not, you will have to kill the processes manually using the Solaris `kill` command. Otherwise, such processes will continue to consume resources.

Symptoms of orphaned processes can be detected by examining error log files or `stdout`, if you are running from a terminal. You can also search for such errors as `RPC: cannot connect`, or `RPC: timout`. These errors will appear under `user.err` priority in `syslog`.

Note – If an `mprun` process becomes unresponsive on a system, even where `tm.watchd` –`Yk` has been enabled, it may be necessary to use `Ctrl-c` to kill `mprun`.

Cleaning Up After RSM Failures

The daemon `hpc_rsmd`, which is started when the cluster is booted, manages access to remote shared memory services on behalf of MPI processes. If an instance of `hpc_rsmd` exits abnormally, you can use the following script to clean up any files and System V shared memory segments that it leaves behind.

`/etc/init.d/sunhpc.hpc_rsmd  [ start | stop | clean ]`

Using Diagnostics

The following sections describe Solaris diagnostics that may be useful in troubleshooting various types of error conditions.

Using Network Diagnostics

You can use `/usr/sbin/ping` to check whether you can connect to the network interface on another node. For example:

% **`ping hpc-node3`**

tests (over the default network) the connection to `hpc-node3`.

You can use `/usr/sbin/spray` to determine whether a node can handle significant network traffic. `spray` indicates the amount of dropped traffic. For example:

```
% spray –c 100 hpc-node3
```

sends 100 small packets to `hpc-node3`.

Checking Load Averages

You can use `mpinfo –N` or, if Sun CRE is not running, `/usr/bin/uptime`, to determine load averages. These averages can help to determine the current load on the machine and how quickly it reached that load level.

Using Interval Diagnostics

The diagnostic programs described below check the status of various parameters. Each accepts a numerical option that specifies the time interval between status checks. If the interval option is not used, the diagnostics output an average value for the respective parameter since boot time. Specify the numerical value at the end of the command to get current information.

Use `/usr/bin/netstat` to check local system network traffic. For example:

```
% netstat –ni 3
```

checks and reports traffic every three seconds.

Use `/usr/bin/iostat` to display disk and system usage. For example:

```
% iostat –c 2
```

displays percentage utilizations every two seconds.

Use `/usr/bin/vmstat` to generate additional information about the virtual memory system. For example:

```
% vmstat –S 5
```

reports on swapping activity every five seconds.

It can be useful to run these diagnostics periodically, monitoring their output for multiple intervals.

Interpreting Sun CRE Error Messages

This section presents sample error messages and their interpretations.

- The following error message usually indicates that all the nodes in a Sun CRE partition are marked down—that is, their node daemons are not running:

  ```
  No nodes in partition satisfy RRS:
  ```

- Under certain circumstances, when a user attempts to kill a job, Sun CRE may log error messages of the following form on the master node:

  ```
  Aug 27 11:02:30 ops2a tm.rdb[462]: Cond_set: unable to
  connect to ops2a/45126: connect: Connection refused
  ```

If these errors can be correlated to jobs being killed, then they can be safely ignored. One way to check this correlation would be to look at the accounting logs for jobs that were signaled during this time.

- The following error message indicates that no partitions have been set up:

  ```
  mprun: unique partition: No such object
  ```

- When there is stale job information in the Sun CRE database, an error message of the following form may occur:

  ```
  Query returned excess results:
  a.out: (TMTL UL) TMRTE_Abort: Not yet initialized
  The attempt to kill your program failed
  ```

  This might happen, for example, when `mpps` shows running processes that are actually no longer running.

  Use the `mpkill –C` *nn* command to clear out such stale jobs.

Note – Before removing the job's information from the database, the `mpkill –C` option verifies that the processes of the job are in fact no longer running.

Anticipating Common Problems

This section presents some guidelines for preventing and troubleshooting common problems.

- When running multiprocess jobs (`–np` equal to 0 or greater than 1) in a cluster with NFS-mounted file systems, you should take steps to limit core dumps to zero. This can be done with the Solaris `limit` command. See the `limit(1)` man page for additional information.
- The Sun CRE resource database daemon (`tm.rdb`) does not remove missing interfaces after a client daemon is restarted. Instead, the `mpinfo –Nv` command will show them marked as `down`.
- The contents of the `/var/adm/messages` file are local to each node. Any daemon messages are logged only on the node where that daemon runs. By default, Sun CRE daemon messages are stored in `/var/adm/messages` along with other messages handled by `syslog`. Alternatively, Sun CRE messages can be written to a file specified by the `mpadmin logfile` command.
- Use shell I/O redirection instead of `mprun –I` options whenever possible. Using shell redirection reduces the likelihood of problems involving standard I/O.
- If `mprun` is signaled too soon after it has been invoked, it exits without stopping the job's processes. If this happens, use `mpkill –9` *jid* to kill such a job.
- Sun CRE does not pass supplemental group ID information to remote processes. You must use the `–G` *gid* option with `mprun` to run with the group permissions of that group. You must be a member of that group.
- Sun CRE RPC timeouts in Sun MPI code are logged to `syslog`, but the default `syslog.conf` file causes these messages to be dropped. If you want to see these errors, modify your `/etc/syslog.conf` file so that messages of the priority `user.err` are not dropped. Note that this does not apply to RPC timeouts occurring in the Sun CRE daemons themselves. By default, these are logged to `/var/adm/messages`.

Note – If you have set the Cluster-level attribute `logfile`, all error messages generated by user code will be handled by Sun CRE (not `syslog`) and will be logged in a file specified by an argument to `logfile`.

Sun CRE RPC timeouts in user code are generally not recoverable. The job might continue to run, but processes probably will not be able to communicate with each other. There are two ways to deal with this:

- Enable the `tm.watchd` job killing option (`–Yk`), which will automatically kill jobs when nodes go off line. This will catch most of these cases, since RPC timeouts usually coincide with `tm.watchd` marking the node as off line.
- Monitor RPC errors from user codes by looking for `syslog` messages of priority `user.err`. Then use `mpkill` to kill the associated job manually.

- If a file system is not visible on all nodes, users can encounter a `permission denied` message when attempting to execute programs from such a file system. Watch for errors caused by non-shared file systems like `/tmp`, which exist locally on all nodes. This can show up when users attempt to execute programs from `/tmp`, and the program does not exist in the `/tmp` file systems of all nodes.
- If you execute `mpkill` with the `–C` option (this option is available only to the system administrator), you should look for and remove leftover files on the master node. The file names for large files are of the form:

  `/tmp/.hpcshm_mmap.`*jid*`.*`

  Smaller files will have file names of the form:

  `/tmp/.hpcshm_acf.`*jid*`.*`

The Sun MPI shared memory protocol module uses these files for interprocess communication on the same node. These files consume swap space.

Understanding Protocol-Related Errors

Errors may occur at cluster startup or at program initialization because of problems finding or loading protocol modules. Such errors are not fatal to the runtime environment (that is, to the Sun CRE daemons), but they do mean that the protocol in question is not available for communication on the cluster.

This section describes some error conditions that may occur in relation to protocol modules(PMs) and the RSM daemon `hpc_rsmd`.

Errors When Sun CRE Daemons Load Protocol Modules

The errors below are generated when the Sun CRE daemons first start up. These errors can occur because of problems in the hpc.conf file, or because of problems loading the PMs.

All these errors are considered nonfatal to the daemon, but the PM that causes the error will not be usable. The errors below cause the Sun CRE daemons to generate calls to `syslog` that result in self-explanatory error messages.

- Error: PM listed in `hpc.conf` cannot be found.
- Error: Library that PM depends on could not be found.
- Error: PM name has more than 4 characters.
- Error: Attempting to load 32-bit PM into 64-bit daemon.

The daemons cause a warning to be generated when there are duplicate PM entries in the PMODULES section of hpc.conf. If there are multiple PM entries with the same name, then only the first one is loaded.

Errors When Protocol Modules Discover Interfaces

The errors below are generated at program startup when a PM attempts interface discovery.

These errors are nonfatal to the Sun CRE daemons, but they may mean that the PM causing the error will not be usable. Appropriate error strings are generated by syslog.

- Error: Malformed interface entry in the hpc.conf file. This example indicates a missing RANK column entry for the hme interface for the tcp PM.

  ```
  -WARNING- Problem detected initializing tcp PM: PM=tcp entry
  hme is missing tokens
  ```

- Error: Missing interface entry in the hpc.conf file. If an entry is missing entirely, then default values are used if available. This example indicates that there is no entry for the hme interface for the tcp PM.

  ```
  -WARNING- Problem detected initializing tcp PM: Interface
  hme0 has missing or broken entry in hpc.conf. Will use: Rank=
  1000,stripe=0, mtu=1500 latency=-1, bandwidth=-1
  ```

- Error: Malformed interface in the hpc.conf file. This example indicates that there are too many columns for the hme interface for the tcp PM.

  ```
  -WARNING- Problem detected initializing tcp PM: PM=tcp entry
  hme has extra tokens
  ```

Errors When the RSM Protocol Module Reads MPI Options

During the startup of an MPI job, the RSM PM interprets values of options in the MPIOptions section of hpc.conf.

If the MPIOptions section does not exist, the RSM PM uses the default values. The PM ignores any option names it does not recognize. If an option is given a value that is either malformed (for instance, a character instead of a number) or out of bounds, the PM uses a default value and causes a message like the following to be generated:

```
-WARNING- ignoring rsm_shortmsgsize=10 in hpc.conf, using default
value of 384 bytes. Please contact system administrator.
```

The `rsm_links` option has other error messages associated with it, indicating an invalid instance number for an interface type or an invalid interface name. In both cases, the invalid item is ignored. The message generated is similar to the following:

```
-WARNING- ignoring rsm_links entry "hpc-comm0wrsm0,1" in hpc.conf...
```

Action of the RSM Daemon

The RSM daemon `hpc_rsmd` writes messages to `syslog` upon detection of unusual events. It writes three different levels of `syslog` messages: error, warning, and information.

Error events that result in the termination of a running MPI job are recorded in the `syslog` at error level. Error events that do not result in the termination or a running MPI job are recorded in the `syslog` at warning level. All other `syslog` messages are written at information level. The `syslogd` can be configured manually to enable or limit the output of these levels.

During `hpc_rsmd` initialization, configuration information is written to `syslog` at information level. An error causes a messages to be written at the `syslog` error level, and the `hpc_rsmd` exits.

After initialization, the `hpc_rsmd` will never spontaneously exit. If the `hpc_rsmd` detects a recoverable error in a request while it is being sent, it will retransmit the request. If the `hpc_rsmd` detects a non-recoverable error while a request is being sent, it will abort the request and return an error to the MPI process that made the request. This, in turn, will cause the MPI job to abort. However, the `hpc_rsmd` path monitor will periodically ping all paths it believes are connected and, upon failure of a ping, will mark the path not available for use.

The RSM PM does not detect error signals itself. In the event that the RSM PM is initialized when the RSM daemon `hpc_rsmd` is not running, the RSM PM prints out an appropriate error message and abort the MPI job.

Recovering From System Failure

Recovering from system failure involves rebooting Sun CRE and recreating the Sun CRE resource database.

The `sunhpc.cre_master reboot` and `sunhpc.cre_node reboot` commands should be used only as a last resort, if the system is not responding (for example, if programs such as `mprun`, `mpinfo`, or `mpps` hang).

▼ To Reboot Sun CRE:

1. **Run** `sunhpc.cre_master reboot` **on the master node:**

   # **/etc/init.d/sunhpc.cre_master reboot**

2. **Run** `sunhpc.cre_node reboot` **on all the nodes (including the master node if it is running** `tm.spmd` **and** `tm.omd`**):**

   # **/etc/init.d/sunhpc.cre_node reboot**

   The procedure attempts to save the system configuration (in the same way as using the `mpadmin dump` command), kill all the running jobs, and restore the system configuration. Note that the Cluster Console Manager applications may be useful in executing commands on all the nodes in the cluster simultaneously. For information about the Cluster Console Manager applications, see Appendix A.

Note – `sunhpc.cre_master reboot` saves the existing `rdb-log` and `rdb-save` files in `/var/hpc/rdb–log.1` and `/var/hpc/rdb–save.1`. The `rdb-log` file is a running log of the resource database activity and `rdb-save` is a snapshot of the database taken at regular intervals.

To recover the Sun CRE after a partial failure—that is, when some but not all daemons have failed, it is possible to clean up bad database entries without losing the configuration information. For example, run the following commands on the master node to clear out the dynamic data while preserving the configuration.

```
# /opt/SUNWhpc/sbin/ctstartd –l
# /etc/init.d/sunhpc.cre_master reboot
```

The `ctstartd` command is necessary in case some of the daemons are not running. The `–l` option causes the command to run on the local system; in this case, the master node.

Configuring Out Network Controllers

During maintenance or replacement of Sun Fire Link high-performance cluster interconnect hardware, the cluster administrator may wish to configure out the controller(s) that have become unavailable. This can be done in the `hpc.conf` file in either of two ways: by changing the `PM=rsm` section or by changing the `rsm_links` option in the `MPIOptions` section. This section describes both methods.

Using the `PM` Section

Stop the Sun CRE and RSM daemons when editing a `PM` section of `hpc.conf` and restart them when finished. See “Stopping and Restarting Sun CRE” on page 14 and “RSM Daemon” on page 19.

If all controllers on the same network are going to be unavailable, the administrator can add a new line to the `PM=rsm` section that names the controller instance and sets its `AVAIL` field to 0.

```
# RSM Settings
# NAME     RANK  AVAIL
Begin PM=rsm
wrsm       15    1
wrsm0      15    0
End PM=rsm
```

Once the daemons are restarted, controller 0 will be unavailable on all nodes on the cluster. The wildcard entry `wrsm` is still set to 1, indicating that all controllers on that network other than controller 0 will be available.

Using the `MPIOptions` Section

By editing the `rsm_links` option in the `MPIOptions` section, the administrator can either remove a controller instance across all nodes or remove a controller instance from one or more specified nodes.

Using the `MPIOptions` method does not require you to stop and restart the daemons. However, any MPI jobs that are currently running are not notified to not use the controller(s) that are being configured out, and this may cause a job to abort.

To remove an instance across all nodes, set the `rsm_links` option to exclude that instance. For example, suppose that a cluster has controllers `wrsm0` and `wrsml` and you wish to configure out `wrsm0`. To do so, set the value of `rsm_links` to `wrsm1` only.

```
rsm_links wrsm1
```

To remove a controller instance only on specified nodes, use the syntax *node.controller* to indicate which controllers on the node(s) in question should remain available. For example, consider a three-node cluster with controllers `wrsm0` and `wrsm1` on all three nodes. The following entry has the effect of removing controller `wrsm0` from `node1`:

```
rsm_links wrsm1 node0.wrsm0 node2.wrsm0
```

This entry specifies that controller `wrsm1` is available on all nodes, while controller `wrsm0` is available only on nodes `node0` and `node2`.

APPENDIX **A**

Cluster Console Manager Tools

This appendix describes a set of cluster administration tools that are installed with the Sun HPC ClusterTools software release.This toolset, called the Cluster Console Manager, allows you to issue commands to all nodes in a cluster simultaneously through a graphical user interface. The CCM offers three modes of operation:

- `cconsole` – This interface provides access to each node's console port through terminal concentrator links. To use this tool, the cluster nodes must be connected to terminal concentrator ports.
- `ctelnet` – This interface initiates simultaneous `telnet` sessions over the network to all nodes in the cluster. Note that if passwords are required, every node must be able to accept the same password.
- `crlogin` – This interface uses `rlogin` to log you in to every node in the cluster. Note that if you launch `crlogin` while logged in as superuser, all `rlogin` sessions are done as superuser. Likewise, if `crlogin` is launched from an ordinary user prompt, all remote logins are done as user.

Each of these modes creates a command entry window, called the *Common Window*, and a separate console window, called a *Term Window*, for each node. Each command typed in the Common Window is echoed in all Term Windows (but not in the Common Window). Every Term Window displays commands you issue as well as system messages logged by its node.

Note – If the cluster nodes are not connected to a terminal concentrator, only `ctelnet` and `crlogin` can be used, not `cconsole`.

Launching Cluster Console Tools

All Cluster Console tools are launched using the same command-line form:

% *tool_name cluster_name*

where *tool_name* is `cconsole`, `ctelnet`, or `crlogin`, and *cluster_name* is a name given to the cluster. For example,

- Launch `ctelnet` by entering:

  % **ctelnet hpc_cluster**

- Launch `crlogin` by entering:

  % **crlogin hpc_cluster**

- Launch `cconsole` by entering:

  % **cconsole hpc_cluster**

If you want to use `cconsole` to monitor messages generated while rebooting the cluster nodes, you will need to launch it from a machine outside the cluster. If you launch it from a cluster node, it will be disabled when the node from which it is launched reboots.

Note – Because `cconsole` accesses the console ports of every node in the cluster, no other accesses to any console in the cluster will be successful while the `cconsole` session is active.

All three Cluster Console commands take the standard X/Motif command-line arguments.

Common Window

The Common Window is the primary window used by the system administrator to send input to all the nodes. This window has a menu bar with three menus and a text field for command entry. The Common Window is always displayed when the Cluster Console is launched.

The Common Window menu bar has three menus:

- Hosts
- Options
- Help

In this manual, the Cluster Console term *Hosts* refers to Sun HPC cluster nodes.

Hosts Menu

The Hosts menu displays a list of the nodes contained in the cluster, plus two other entries, Select Hosts and Exit. TABLE A-1 describes these menu choices.

TABLE A-1 Cluster Console Menu Entries

Entry	Function
Host toggle buttons	Selects whether or not the host gets input from the Common Window text field. There is a separate toggle button for each node currently connected to the Cluster Console. ON — Enables input from the Common Window text field to the node. OFF — Disables input from the Cluster Console.
Select Hosts	Displays the Select Hosts dialog window.
Exit	Quits the Cluster Console program.

Select Hosts Dialog

The Select Hosts dialog enables you to add or delete nodes during the current Cluster Console session. The scrolled text window in the Select Hosts dialog displays a list of the nodes that are currently connected to the Cluster Console.

There are three Select Hosts dialog buttons, which are described in TABLE A-2.

TABLE A-2 Select Hosts Dialog Buttons

Entry	Function
Insert	Opens a Term Window and establishes a connection to the specified hosts(s). Adds the host(s) specified in the Hostname text field to the list of accessible hosts. The inserted host name(s) are displayed in the hosts list in the scrolled text window and in the Common Window.
Remove	Deletes the host selected in the Hosts list in the scrolled text window.
Dismiss	Closes the Select Hosts dialog.

.

▼ To Add a Single Node

1. **Enter the** *hostname* **in the Hostname text field.**

2. **Select Insert.**

   Entering a valid host name opens a Term Window for the specified host and establishes a connection to that host. The name of the selected host appears in the scrolled text window and in the hosts list on the Hosts menu in the Common Window.

▼ To Add All Nodes in a Cluster

1. **Enter the** *clustername* **in the Hostname text field.**

2. **Select Insert.**

   The Cluster Console automatically expands the cluster name into its constituent host names and then opens one Term Window for each node. A connection is established for each of the constituent host names. The Cluster Console automatically displays the names of the hosts in the cluster in the scrolled text window and in the hosts list on the Hosts menu in the Common Window.

▼ To Remove a Node

1. **Select the name of the host in the list in the scrolled text window.**

2. **Select Remove.**

   This closes the corresponding Term Window and disconnects the host. The name of the removed host disappears from the scrolled text window and from the hosts list on the Hosts menu in the Common Window.

Options Menu

The Options menu has one entry, Group Term Window; see TABLE A-3 for a description.

TABLE A-3 Group Term Window Entry

Entry	Function
Group Term Windows	This is a toggle button that groups and ungroups the Common Window and the Term Window. ON — Group: the Term Windows follow the Common Window when the Common Window is moved. OFF — Ungroup: the Term Windows and the Common Window move independently.

Help Menu

The Help menu has three entries; see TABLE A-4 for a description.

TABLE A-4 Help Menu

Entry	Function
Help	Displays a Help window—the interface to the Sun online help system.
About	Displays the About box, which contains information on the Cluster Console application, such as version number.
Comments	Displays the Comments box, which allows you to enter comments about the software and send them to the development team.

.

Text Field

The text field is where you enter commands that you want to have executed simultaneously on multiple nodes. The state of the host toggle buttons under the Hosts menu determines which nodes receive this input.

Term Windows

The Term Window is just like a normal terminal window. To type on only one host, move the cursor to the Term Window of the desired host and type directly into it.

The Cluster Console Term Windows are like other terminal programs, such as `xterm`, `cmdtool`, and `shelltool`, except that they can also receive input from the Common Window. The Term Windows use VT220 terminal emulation.

The environment variable `TERM` informs your editor of your terminal type. If you are having display problems from `vi` or any other tools, set the environment variable using the appropriate commands for your shell.

The Term Window contains additional functionality, which you can access by positioning the pointer over the Term Window and pressing the right mouse button. This displays the menu described in TABLE A-5.

TABLE A-5 Term Window Menu Entries

Entry	Function
Disable / Enable Scroll Bar	Toggles the scroll bar display on and off in the Term Window.
Exit This Window	Closes the current Term Window.

.

Using the Cluster Console

To issue commands to multiple nodes simultaneously:

- **Position the cursor in the text field of the Common Window and enter your command.**

  Every keystroke entered in this field is sent to all hosts that are currently selected for input.

To issue commands to a single node:

- **Position the cursor in the corresponding Term Window and enter your command.**

  Alternatively, you can turn off all hosts in the Hosts menu, except the one you want to access. Then issue your commands from the Common Window.

Administering Configuration Files

Two configuration files are used by Cluster Console: `/etc/clusters` and `/etc/serialports`. These files are created automatically.

The `clusters` File

The `clusters` configuration file maps a cluster name to the list of hostnames that make up the cluster. Each line in this database corresponds to a cluster. The format is:

clustername hostname–1 hostname–2 `[...]` *hostname–n*

For example:

```
cities          chartres izmir tampico inchon essen sydney
```

The `clusters` file is used to map cluster names to host names on the command line and in the Select Hosts dialog.

The `serialports` File

The `serialports` file maps each host name to the terminal concentrator and the terminal concentrator serial port to which it is connected. Each line in this database specifies a separate serial port using the format:

hostname terminal_concentrator serial_port

For example:

```
chartres        cities-tc 5002
izmir           cities-tc 5003
```

The `serialports` file is used by `cconsole` to determine which terminal concentrator and serial ports to connect to for the various cluster nodes that have been specified on the command line or the Select Hosts dialog.

Index

D

E

H

I

J

K

O

P

R

S

T

U

V