# Linux® Configuration and Operations Guide

007–4633–013

COPYRIGHT

© 2003–2007, 2008, SGI. All rights reserved; provided portions may be copyright in third parties, as indicated elsewhere herein. No permission is granted to copy, distribute, or create derivative works from the contents of this electronic documentation in any manner, in whole or in part, without the prior written permission of SGI.

LIMITED RIGHTS LEGEND

The software described in this document is "commercial computer software" provided with restricted rights (except as to included open/free source) as specified in the FAR 52.227-19 and/or the DFAR 227.7202, or successive sections. Use beyond license provisions is a violation of worldwide intellectual property laws, treaties and conventions. This document is provided with limited rights as defined in 52.227-14.

TRADEMARKS AND ATTRIBUTIONS

SGI, the SGI logo, Altix, IRIX, Onyx2, Origin, and XFS are registered trademarks and NUMAflex, NUMAlink, OpenMP, Performance Co-Pilot, SGI ProPack, SGIconsole, SHMEM, and XIO are trademarks of SGI, in the United States and/or other countries worldwide.

Intel and Itanium are trademarks or registered trademarks of Intel Corporation. Linux is a registered trademark of Linus Torvalds. MIPS is a registered trademark of MIPS Technologies, Inc., used under license by SGI. Red Hat and all Red Hat-based trademarks are trademarks or registered trademarks of Red Hat, Inc. in the United States and other countries. Novell is a registered trademark, and SUSE is a trademark of Novell, Inc., in the United States and other countries.

All other trademarks mentioned herein are the property of their respective owners.

# New Features in This Manual

This update of the *Linux Configuration and Operations Guide* supports the SGI ProPack 6 for Linux software release.

## Major Documentation Changes

Updated information in "Supported Configurations" on page 17.

Editing changes throughout the manual.

This manual mostly applies to SGI Altix systems. In many instances, sections do not apply to the SGI Altix XE systems. This has been noted throughout the manual as appropriate. For more information on SGI Altix ICE 8200 systems, see the *SGI Tempo System Administrator's Guide*.

# Record of Revision

| Version | Description |
|---|---|
| 001 | October 2003<br>Original publication. |
| 002 | February 2004<br>Updated to support the SGI ProPack for Linux v2.4 release. |
| 003 | May 2004<br>Updated to support the SGI ProPack 3 for Linux release. |
| 004 | August 2004<br>Updated to support the SGI ProPack 3 for Linux Service Pack 1 release. |
| 005 | November 2004<br>Updated to support the SGI ProPack 3 for Linux Service Pack 2 release. |
| 006 | January 2005<br>Updated to support the SGI ProPack 3 for Linux Service Pack 3 release. |
| 007 | February 2005<br>Updated to support the SGI ProPack 4 for Linux release. |
| 008 | April 2005<br>Updated to support the SGI ProPack 3 for Linux Service Pack 5 release. |
| 009 | January 2006<br>Updated to support the SGI ProPack 4 for Linux Service Pack 3 release. |
| 010 | July 2006<br>Updated to support the SGI ProPack 5 for Linux release. |

| | |
|---|---|
| 011 | September 2007<br>Updated for additional changes to support the SGI ProPack 5 for Linux release. |
| 012 | April 2007<br>Updated to support the SGI ProPack 5 for Linux Service Pack 1 release. |
| 013 | May 2008<br>Updated to support the SGI ProPack 6 for Linux Service release. |

# Contents

# Procedures

# About This Guide

This guide is a reference document for people who manage the operation of SGI computer systems running SGI ProPack 6 for Linux operating system. It explains how to perform general system configuration and operations under the Linux operating system used with SGI servers and superclusters. For information about general Linux system administration, see the “Additional Reading” section of this Preface.

---

**Note:** For information on system configuration and operation on SGI ProPack 4 for Linux or SGI ProPack 5 for Linux systems, see the 007-4633-009 or 007-4633-0012 version of this manual, respectively. From the current 007-4633-013 version of this manual on the SGI Technical Publications Library, select the **additional info link**. Click on **007-4633-009** under **Other Versions :**

---

This manual contains the following chapters:

- Chapter 1, "Configuring Your System" on page 1
- Chapter 2, "System Operation" on page 35
- Chapter 3, "Kernel Tunable Parameters on SGI ProPack Servers" on page 53

## Related Publications

For a list of manuals supporting SGI Linux releases, see the *SGI ProPack 6 for Linux Start Here*.

## Additional Reading

The following sections describe publications that contain additional information that my be helpful in the administration of your system.

## Linux System Administration

Linux system administration information is available on your system at the following location:

`/usr/src/linux/Documentation`

For SLES10 systems, if you install the kernel-source package (not installed by default), then `/usr/src/linux` will be a symbolic link to the proper location.

Documentation for RHEL5 is located in the `/usr/share/doc` directory. It contains package-specific documentation. If you install the `zisofs` module, additional documentation in the form of READMEs can be found in `/usr/share/doc/zisofs-tools-1.x.x`.

Linux system administration course information is available at:

http://www.sgi.com/support/custeducation/courses/linux/sys_admin.html

Linux Documentation Project (LDP) also contains useful documentation at: http://tldp.org/.

## Intel Compiler Documentation

Documentation for the Intel compilers is located on your system in the `/docs` directory of the directory tree where your compilers are installed. If you have installed the Intel compilers, the following documentation is available:

- *Intel C++ Compiler User's Guide* (`c_ug_lnx.pdf`)
- *Intel Fortran Compiler User's Guide* (`for_ug_lnx.pdf`)
- *Intel Fortran Programmer's Reference* (`for_prg.pdf`)
- *Intel Fortran Libraries Reference* (`for_lib.pdf`)

## Other Intel Documentation

The following documents describe the Itanium (previously called "IA-64") architecture and other topics of interest:

- *Intel Itanium 2 Processor Reference Manual for Software Development and Optimization*, available online at the following location:

http://developer.intel.com/design/itanium2/manuals/251110.htm

- *Intel Itanium Architecture Software Developer's Manual*, available online at the following location:

http://developer.intel.com/design/itanium/manuals/iiasdmanual.htm

- *Introduction to Itanium Architecture*, available online at the following location:

http://shale.intel.com/softwarecollege/CourseDetails.asp?courseID=13

(secure channel required)

### Open-Source Documents

The following open-source document may be useful to you.

- *Debugging with DDD User's Guide and Reference Manual* provides information on using the DataDisplayDebugger (DDD). It is available at the following location:

  http://www.gnu.org/manual/ddd/pdf/ddd.pdf

## Obtaining Publications

You can obtain SGI documentation in the following ways:

- See the SGI Technical Publications Library at: http://docs.sgi.com. Various formats are available. This library contains the most recent and most comprehensive set of online books, release notes, man pages, and other information.
- Online versions of the *SGI ProPack 6 for Linux Start Here*, the SGI ProPack 6 release notes, which contain the latest information about software and documentation in this release, the list of RPMs distributed with SGI ProPack 6, and a useful migration guide, which contains helpful hints and advice for customers moving from earlier versions of SGI ProPack to SGI ProPack 6, can be found in the `/docs` directory on the SGI ProPack 6 for Linux CD.

  The SGI ProPack 6 release notes get installed to the following location on a system running SGI ProPack 6: `/usr/share/doc/sgi-propack-6/README.txt`.
- You can view man pages by typing man *title* on a command line.

## Conventions

The following conventions are used throughout this document:

| Convention | Meaning |
|---|---|
| `command` | This fixed-space font denotes literal items such as commands, files, routines, path names, signals, messages, and programming language structures. |
| *variable* | Italic typeface denotes variable entries and words or concepts being defined. |
| **`user input`** | This bold, fixed-space font denotes literal items that the user enters in interactive sessions. (Output is shown in nonbold, fixed-space font.) |
| [ ] | Brackets enclose optional portions of a command or directive line. |
| ... | Ellipses indicate that a preceding element can be repeated. |

## Reader Comments

If you have comments about the technical accuracy, content, or organization of this publication, contact SGI. Be sure to include the title and document number of the publication with your comments. (Online, the document number is located in the front matter of the publication. In printed publications, the document number is located at the bottom of each page.)

You can contact SGI in any of the following ways:

- Send e-mail to the following address:

  techpubs@sgi.com

- Contact your customer service representative and ask that an incident be filed in the SGI incident tracking system.

- Send mail to the following address:

  SGI
  Technical Publications
  1140 East Arques Avenue

Sunnyvale, CA 94085–4602

SGI values your comments and will respond to them promptly.

*Chapter 1*

# Configuring Your System

This chapter provides information on configuring your system and covers the following topics:

**Note:** For information on system configuration and operation on SGI ProPack 4 for Linux or SGI ProPack 5 for Linux systems, see the 007-4633-009 or 007-4633-0012 version of this manual, respectively. From the current 007-4633-013 version of this manual on the SGI Technical Publications Library, select the **additional info link**. Click on **007-4633-009** under **Other Versions :**

## PCI or PCI-X Card Hot-Plug Software

The Linux PCI/X hot-plug feature supports inserting a PCI or PCI-X card into an empty slot and preparing that card for use or deactivating a PCI or PCI-X card and then removing it from its slot, while the system is running. Hot-plug operations can be initiated using a series of shell commands.

**Note:** PCI hot-plug is **only** supported on SGI Altix 3000 series systems on the IX-Brick, PX-Brick, IA-Brick, and PA-Brick. It is **not** supported on SGI Altix XE systems or Altix 350 systems. For SGI ProPack 6 systems running RHEL5, the I3X blade (3 slot double-wide PCI-X I/O) on SGI Altix 450 or SGI Altix 4700 systems supports PCI hot-plug. PCI-X cards can be added, removed, or replaced in the I3X blade while the I3X blade is installed in the 1955 chassis and the system is operating.

This section describes hot-swap operations and covers the following topics:

- "Introduction PCI or PCI-X Card Hot-plug Operations" on page 2
- "Loading Hot-plug Software" on page 3
- "Controlling Hot-plug Operations" on page 3
- "Using Shell Commands To Control a Hot–plug Operation" on page 5

**Note:** The hot-plug feature is not configured on by default. It is a loadable module. To load it, see "Loading Hot-plug Software" on page 3.

## Introduction PCI or PCI-X Card Hot-plug Operations

A hot-swap operation is the combination of a remove and insert operation targeting the same slot. Single function cards, multi-function cards, and PCI/X-to-PCI/X bridges are supported.

A hot-plug insert operation consists of attaching a card to an SGI card carrier, inserting the carrier in an empty slot, and using software commands to initiate the software controlled power-up and initialization of the card.

A hot-plug remove operation consists of manually terminating any users of the card, and then using software commands to initiate the remove operation to deactivate and power-down the card.

The Altix system L1 hardware controller has these hot-plug restrictions, as follows:

- A 33 MHz PCI/X card cannot be inserted into an empty PCI bus
- The last card cannot be removed from a bus running at 33 MHz

If these restrictions are detected by the Linux kernel and reported to the user, the requested hot-plug operation fails.

For detailed instructions on how to install or remove a PCI or PCI-X card on the SGI Altix 3000 series systems, see “Adding or Replacing a PCI or PCI-X Card” in Chapter 12, “Maintenance and Upgrade Procedures” in *SGI Altix 3000 User’s Guide.*

For more information on the SGI L1 and L2 controller software, see the *SGI L1 and L2 Controller Software User’s Guide.*

## Loading Hot-plug Software

The hot-plug feature is not configured on by default. To load the `sgi_hotplug` module, perform the following steps:

1. Load the `sgi_hotplug` module, as follows:

   ```
   % modprobe sgi_hotplug
   ```

2. Make sure the module is loaded, as follows:

   ```
   % lsmod | grep sgi_hotplug
   sgi_hotplug           145168  0
   pci_hotplug           189124  2 sgi_hotplug,shpchp
   ```

3. Change directory (`cd`) to the `/sys/bus/pci/slots` directory and verify its contents, as follows:

   ```
   % ls -l
       total 0
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0021:00:01
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0022:00:01
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0022:00:02
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0023:00:01
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0024:00:01
       drwxr-xr-x 2 root root 0 Aug 22 10:54 0024:00:02
   ```

## Controlling Hot-plug Operations

This section describes hot-plug operations and the format of a slot name. It covers the following topics:

- "Slot Name Format" on page 3
- "Hot-plug Insert Operation" on page 4
- "Hot-plug Remove Operation" on page 5

### Slot Name Format

Hot-plug operations target a particular slot using the name of the slot. All slots that are eligible for a hot-plug operation have a directory in the hot-plug file system that is mounted at `/sys/bus/pci/slots`. The name of the target slot is based on the hardware location of the slot in the system. For the SGI ProPack 6 release, slot directories are in the form that the `lspci`(8) command uses, that is, as follows:

segment:bus:slot

Change directory (`cd`), to the `/sys/bus/pci/slots` directory and use the `ls` or `ls -l` command to view the contents of the file, as follows:

```
pci/slots> ls
0001:00:01  0002:00:01  0002:00:02  0003:00:01  0004:00:01  0004:00:02
```

Slot is part of a PCI domain. On an SGI Altix system, a *PCI domain* is a functional entity that includes a root bridge, subordinate buses under the root bridge, and the peripheral devices it controls. For more information, see "PCI Domain Support for SGI Altix Systems" on page 13.

Each `slots` directory contains two files called `path` and `power`. For example, change directory to `/sys/bus/pci/slots/0001:00:01` and perform the `ls` command, as follows:

```
slots/0001:00:01> ls
path  power
```

The `power` file provides the current hot-plug status of the slot. A value of `0` indicates that the slot is powered-down, and a value of `1` indicates that the slot is powered-up.

The `path` file is the module ID where the brick resides.

## Hot-plug Insert Operation

A hot-plug insert operation first instructs the L1 hardware controller to power-up the slot and reset the card. The L1 controller then checks that the card to be inserted is compatible with the running bus. Compatible is defined, as follows:

- The card must support the same mode as the running bus, for example, PCI or PCI-X
- The card must be able to run at the current bus speed
- That a 33 MHz card is **not** being inserted into an empty bus

Any L1 controller detected incompatibilities or errors are reported to the user and the insert operation fails.

Once the slot has been successfully powered-up by the L1 controller, the Linux hot-plug infrastructure notifies the driver of the card that the card is available and needs to be initialized. After the driver has initialized the card, the hot-plug insert operation is complete and card is ready for use.

### Hot-plug Remove Operation

Before initiating a hot-plug remove operation, the system administrator must manually terminate any processes using the target card.

**Warning:** Failure to properly terminate any outstanding accesses to the target card may result in a system failure or data corruption when the hot-plug operation is initiated.

For a hot-plug remove operation, the hot-plug infrastructure verifies that the target slot is eligible to be powered-down. The L1 hardware controller restrictions do **not** permit the last card to be removed from a bus running at 33 MHz and an attempt to remove the last card fails. The hot-plug infrastructure then notifies the driver of the card of a pending hot-plug remove operation and the driver deactivates the card. The L1 hardware controller is then instructed to power-down the slot.

Attempts to power-down a slot that is already powered-down, or power-up a slot that is already powered-up are ignored.

## Using Shell Commands To Control a Hot–plug Operation

A hot-plug operation can be initiated by writing to the target `power` file of the slot. After composing the name of the slot based on its location, change into the directory of the slot in the hot-plug virtual file system.

Change directory (`cd`), to the `/sys/bus/pci/slots` directory and use the `ls` or `ls -l` command to view the contents of the file, as follows:

```
pci/slots> ls
 0001:00:01  0002:00:01  0002:00:02  0003:00:01  0004:00:01  0004:00:02
```

For example, to target hot-plug operations to slot1, segment1 in module 001i03 (IA-Brick in position 3 of rack 1), change to directory `0001:00:01` and then perform the `ls` command, as follows:

```
slots/0001:00:01> ls
path  power
```

To query the current hot-plug status of the slot, read the `power` file of the slot, as follows:

```
slots//0001:00:01> cat power
1
```

A value of `0` indicates that the slot is powered-down, and a value of `1` indicates that the slot is powered-up.

The `path` file is the module ID were the brick resides, as follows:

```
slots/0001:00:01> cat path
module_001i03
```

To initiate an insert operation to a slot that is powered-down, write the character `1` to the `power` file of the slot, as follows:

```
slots/0001:00:01> echo 1 > power
```

Detailed status messages for the insert operation are written to the `syslog` by the hot-plug infrastructure. These messages can be displayed using the Linux `dmesg` user command, as follows:

```
slots/0001:00:01> dmesg
```

A hot-plug remove operation is initiated to a slot that is powered-up by writing the character 0 to the `power` file of the slot, as follows:

```
slots/0001:00:01> echo 0 > power
```

Detailed status messages for the remove operation are written to the `syslog` by the hot-plug infrastructure. These messages can be displayed using the Linux `dmesg` user command, as follows:

```
slots/0001:00:01> dmesg
```

## Faster SCSI Device Booting

There are three files in the `/etc/udev/rules.d` directory that can be modified to make systems boot faster that have many logical unit numbers (LUNs) for attached SCSI devices (1000 LUNS or more), as follows:

- `50-udev-default.rules`

- `58-xscsi.rules`
- `60-persistent-storage.rules`

This section describes these rules.

## `50-udev-default.rules`

---

**Note:** This section only applies to SGI Altix systems with SGI ProPack 6 running SLES10.

---

The rules in the `50-udev-default.rules` file cause SCSI drivers `sd_mod`,`osst`, `st`, `sr_mod`, and `sg` to be loaded automatically when appropriate devices are found. These rules are, as follows:

```
SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="0|7|14", RUN+="/sbin/modprobe sd_mod"
SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="1", SYSFS{vendor}=="On[sS]tream", RUN+="/sbin/modprobe osst"
SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="1", RUN+="/sbin/modprobe st"
SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="[45]", RUN+="/sbin/modprobe sr_mod"
SUBSYSTEM=="scsi_device", ACTION=="add", RUN+="/sbin/modprobe sg"
```

You can comment out all of these rules to save calls to the `modprobe`(8) command for each SCSI device to save boot time, as follows:

```
#SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="0|7|14", RUN+="/sbin/modprobe sd_mod"
#SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="1", SYSFS{vendor}=="On[sS]tream", RUN+="/sbin/modprobe osst"
#SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="1", RUN+="/sbin/modprobe st"
#SUBSYSTEM=="scsi_device", ACTION=="add", SYSFS{type}=="[45]", RUN+="/sbin/modprobe sr_mod"
#SUBSYSTEM=="scsi_device", ACTION=="add", RUN+="/sbin/modprobe sg"
```

Make sure the drivers are loaded by adding them to `INITRD_MODULES` variable in the `/etc/sysconfig/kernel` file and then run the `mkinitrd`(8) command to make sure the changes are picked up.

## `58-xscsi.rules`

---

**Note:** This section only applies to SGI Altix systems with SGI ProPack 6 running SLES10.

---

The rules in the `58-xscsi.rules` file create the `/dev/xscsi/pci...` persistent device symbolic links. The rules are, as follows:

```
# This rule creates sg symlinks for all SCSI devices (/dev/xscsi/pci..../sg)
KERNEL=="sg*", PROGRAM="/sbin/udev_xscsi %p", SYMLINK+="%c"

# This rule creates symlinks for entire disks/luns (/dev/xscsi/pci..../disc)
KERNEL=="sd*[a-z]", PROGRAM="/sbin/udev_xscsi %p", SYMLINK+="%c"

# This rule creates symlinks for disk partitions (/dev/xscsi/pci..../partX)
KERNEL=="sd*[a-z]*[0-9]", PROGRAM="/sbin/udev_xscsi %p", SYMLINK+="%c"
```

You need the rule for the `disk/luns` symbolic links for XVM in this file, (the middle rule). You can comment out the rule that creates `sg` symbolic links and the rule for disk partition symbolic links (top and bottom rules, respectively).

### 60-persistent-storage.rules

**Note:** This section only applies to SGI Altix systems with SGI ProPack 6 running SLES10.

The rules in the `60-persistent-storage.rules` file are for persistent storage links. They are not necessary and can all be commented out or you can add `GOTO="persistent_storage_end"` at the top of the file to accomplish the same.

## Filesystems Changes

The `tmpfs` filesystem memory allocations have changed for the SGI ProPack 6 release. Prior to the SGI ProPack 6 release, allocations were always done round-robin on all nodes. With SGI ProPack 6, this is now a `tmpfs` mount option. This is actually a Linux kernel change that applies to both SLES10 and RHEL5 base releases.

**Note:** To maintain SGI ProPack 4 `tmpfs` filesystem memory allocation default behavior, use the `tmpfs` filesystem `mpol=interleave` mount option.

The `tmpfs` filesystem has a `mount` option to set the NUMA memory allocation policy for all files if the `CONFIG_NUMA` flag is enabled at mount time. You can adjust this on a running system, as follows:

**`mount -o remount ...`**

The following `mount` options apply:

| | |
|---|---|
| `mpol=default` | Prefers to allocate memory from the local node |
| `mpol=prefer:Node` | Prefers to allocate memory from the given node |
| `mpol=bind:NodeList` | Allocates memory only from nodes in `NodeList` |
| `mpol=interleave` | Prefers to allocate from each node in turn |
| `mpol=interleave:NodeList` | Allocates from each node of `NodeList` in turn |

`NodeList` format is a comma-separated list of decimal numbers and ranges. The range being two hyphen-separated decimal numbers, the smallest and largest node numbers in the range. For example, `mpol=bind:0-3,5,7,9-15`.

Trying to mount a `tmpfs` filesystem with an `mpol` option will fail if the running kernel does not support NUMA architecture. It will also fail if its `Nodelist` argument specifies a node greater than or equal to `MAX_NUMNODES`.

If your system relies on that tmpfs file system being mounted, but from time to time runs a kernel built without NUMA capability (such as, a safe recovery kernel), or configured to support fewer nodes, it is advisable to omit the `mpol` option from automatic mount options. It can be added later, when the `tmpfs` is already mounted on `MountPoint` using the following:

```
mount -o remount,mpol=Policy:NodeList MountPoint
```

For more information on the `tmpfs` filesystem, see `/usr/src/linux-2.6.x.x-x/Documentation/filesystems` file on your system.

## I /O Subsystems

Although some HPC workloads might be mostly CPU bound, others involve processing large amounts of data and require an I/O subsystem capable of moving data between memory and storage quickly, as well as having the ability to manage large storage farms effectively. The XSCSI subsystem, XFS filesystem, XVM volume manager, and data migration facilities were leveraged from the IRIX operating system and ported to provide a robust, high-performance, and stable storage I/O subsystem on Linux.

The following sections describe persistent PCI-X bus numbering, persistent naming of Ethernet devices, the XSCSI subsystem, the XSCSI-SCSI subsystem, the XFS filesystem, and the XVM Volume Manager.

This section covers the following topics:

- "XSCSI Naming Systems on SGI ProPack Systems" on page 10
- "Persistent Network Interface Names" on page 13
- "PCI Domain Support for SGI Altix Systems" on page 13

### XSCSI Naming Systems on SGI ProPack Systems

This section describes XSCSI naming systems on SGI ProPack systems.

**Note:** The XSCSI subsystem on SGI ProPack 3 systems is an I/O infrastructure that leverages technology from the IRIX operating system to provide more robust error handling, failover, and storage area network (SAN) infrastructure support, as well as long-term, large system performance tuning. This subsystem was not necessary for SGI ProPack 4 systems or later. However, the XSCSI naming convention is still used on SGI ProPack 3, SGI ProPack 4, and SGI ProPack 5 and SGI ProPack 6 systems. XSCSI naming is used to provide persistent naming for devices, by using the persistent PCI bus numbering. For SGI Altix 450 and 4700 systems, see "PCI Domain Support for SGI Altix Systems" on page 13.

This section covers the following topics:

- "XSCSI Names on Non-blade Systems" on page 11
- "Domain-Based XSCSI Names" on page 12

### XSCSI Names on Non-blade Systems

For SGI ProPack 3 and SGI ProPack 4 non-blade systems, the XSCSI name has the following forms.

**For direct attached devices, the form of the XSCSI name is, as follows:**

```
/dev/xscsi/pciBB.SS.F[-C]/targetX/lunL/partT
```

**For SAN attached devices, it is, as follows:**

```
/dev/xscsi/pciBB.SS.F[-P]/nodeWWN/portP/lunL/partT
```

**Where:**

`BB` is the PCI bus number

`SS` is the Slot

`F` is the Function

`C` is the Channel ("-C" for QLA12160 HBA cards only)

**For direct attach devices:**

`X` is the target number

**For SAN attached devices:**

`WWN` is the world wide node number of the device

`P` is the port number of the device

**For either direct attach or SAN attach devices:**

`L` is the logical unit number (LUN) in `lunL`

`T` is the partition number

There are two ways of handling multiple port host bus adapter (HBA) cards in PCI. One way is to have each port be a different function. The other way is to have one function but have multiple channels off of the function. Most HBA cards use multiple functions. Therefore, most HBA cards will have differing `F` (function) numbers and the `-C` will be absent. The QLA12160 (Qlogic Parallel SCSI) uses multiple channels. Therefore, it will have one function, "0" and multiple channels, that is, "0-1" and "0-2".

An example of a direct attached device with partition 1 is, as follows:

```
/dev/xscsi/pci01.02.0/target1/lun0/part1
```

The same device attached device off of a Qlogic SCSI HBA card (or base IO9 base I/O) would be, as follows:

```
/dev/xscsi/pci01.02.0/target1/lun0/part1
```

An example of a SAN attached device is, as follows:

```
/dev/xscsi/pci22.01.1/node20000004cf2c84de/port1/lun0/part1
```

### Domain-Based XSCSI Names

All SGI Altix systems running SGI ProPack 6 for Linux use domain-based XSCSI names. The XSCSI names change to accommodate PCI Express. They basically have the same form except that the PCI numbering takes the following form:

```
/dev/xscsi/pciDDDD:BB:SS.F[-C]/...
```

**Where:**

`DDDD` is the domain number

`BB` is the Bridge number

`SS` is the slot number

`F` is the function

`C` is the Channel

An example of a direct attach device with partition 1 is, as follows:

```
/dev/xscsi/pci0001:00:03.0-1/target1/lun0/part1
```

An example of a SAN attached device with partition 1 is, as follows:

```
/dev/xscsi/pci0002:00:01.0/node20000004cf2c8d0c/port2/lun0/part1
```

Note that the device number (slot number), function `WWN`, logical unit, and port number are fixed. These will never change. However, the system bus number could change because of a hardware problem (such as the I/O brick not booting) or the system being reconfigured.

For non-PCI express host bus adapter (HBA) cards in an SGI Altix 4700 system, the domain number is equivalent to the old bus number and the bridge number is 0. For more information, see "PCI Domain Support for SGI Altix Systems" on page 13.

## Persistent Network Interface Names

**Note:** This section only applies to SGI Altix systems with SGI ProPack 6 running SLES10.

Ethernet persistent behavior changes in the SGI ProPack 6 for Linux from prior releases. This functionality is provided by the base operating system.

The basic change is that the first time a SGI ProPack 6 system is booted after installation, a `udev` rule defined in `/etc/udev/rules.d/31-net_create_names.rules` is invoked that enumerates all of the Ethernet devices on the system. It then writes another rule file called `/etc/udev/rules.d/30-net_persistent_names.rules`. This file contains a mapping of the media access control (MAC) addresses to Ethernet IP addresses. A specific physical interface is always mapped to the same Ethernet address.

When a system is rebooted, the same Ethernet addresses are always mapped back to the same MAC addresses, even if some of the interfaces have been removed in the interim. For example, if Ethernet 1 and Ethernet 3 devices were removed, a sparsely populated Ethernet space of Ethernet 0, Ethernet 2 and Ethernet 4 would result.

To re-enumerate the devices attached to your system, delete the `/etc/udev/rules.d/30-net_persistent_names.rules` file. When the system is rebooted, a new rules file is created for the current compliment of network device Ethernet addresses.

For more information, see the SGI ProPack 6 `/usr/share/doc/packages/sysconfig/README.Persistent_Interface_Names` file.

## PCI Domain Support for SGI Altix Systems

**Note:** This section does not apply to SGI Altix XE systems.

On an SGI Altix system, a *PCI domain* is a functional entity that includes a root bridge, subordinate buses under the root bridge, and the peripheral devices it controls. Separation, management, and protection of PCI domains is implemented and controlled by system software.

Previously, a PCI device was identified by bus:slot:function. A PCI device is identified by domain:bus:slot:function.

Domains (sometimes referred to as a PCI segment) are numbered from (0 to ffff). bus (0 to ff), slot (0 to 1f) and function (0 to 7).

A domain is a root bridge (with the bus being numbered zero), subordinate buses under that root bridge are buses 1-255.

In the past, a PA-brick was numbered bus 01, 02, 03, 04 (A bus for each root bridge (that is, each TIO ASIC)). With PCI domain support each root bridge is numbered with a domain/bus number 0001:00, 0002:00, 0003:00, 0004:00. If a subordinate bus is plugged into the root bridge bus, it has the same domain number as the root bridge, but a different bus number (for example, 0001:01).

For PCI Express the PCIE root complex is its own domain. Each port is a subordinate bus under that domain.

Domain numbers are allocated starting from the lowest module ID (that is, rack/slot/blade location).

## System Partitioning

This section describes how to partition an SGI ProPack server and contains the following topics:

- "Overview" on page 15
- "Advantages of Partitioning" on page 15
- "Limitations of Partitioning" on page 17
- "Supported Configurations" on page 17
- "Installing Partitioning Software and Configuring Partitions" on page 18
- "Connecting the System Console to the Controller" on page 31

This section does not apply to SGI Altix XE systems.

## Overview

A single SGI ProPack for Linux server can be divided into multiple distinct systems, each with its own console, root filesystem, and IP network address. Each of these software-defined group of processors are distinct systems referred to as a *partition*. Each partition can be rebooted, loaded with software, powered down, and upgraded independently. The partitions communicate with each other over an SGI NUMAlink connection. Collectively, all of these partitions compose a single, shared-memory cluster.

Direct memory access between partitions, sometimes referred to as global shared memory, is made available by the XPC and XPMEM kernel modules. This allows processes in one partition to access physical memory located on another partition. The benefits of global shared memory are currently available via SGI's Message Passing Toolkit (MPT) software.

It is relatively easy to configure a large SGI Altix system into partitions and reconfigure the machine for specific needs. No cable changes are needed to partition or repartition an SGI Altix machine. Partitioning is accomplished by commands sent to the system controller. For details on system controller commands, see the *SGI L1 and L2 Controller Software User's Guide.*

## Advantages of Partitioning

This section describes the advantages of partitioning an SGI ProPack server as follows:

- "Create a Large, Shared-memory Cluster" on page 15
- "Provides Fault Containment" on page 16
- "Allows Variable Partition Sizes" on page 16
- "Provide High Performance Clusters" on page 16

### Create a Large, Shared-memory Cluster

You can use SGI's NUMAlink technology and the XPC and XPMEM kernel modules to create a very low latency, very large, shared-memory cluster for optimized use of Message Passing Interface (MPI) software and logically shared, distributed memory access (SHMEM) routines. The globally addressable, cache coherent, shared memory is exploited by MPI and SHMEM to deliver high performance.

### Provides Fault Containment

Another reason for partitioning a system is fault containment. In most cases, a single partition can be brought down (because of a hardware or software failure, or as part of a controlled shutdown) without affecting the rest of the system. Hardware memory protections prevent any unintentional accesses to physical memory on a different partition from reaching and corrupting that physical memory. For current fault containment caveats, see "Limitations of Partitioning" on page 17.

You can power off and “warm swap” a failing C-brick in a down partition while other partitions are powered up and booted. For information see “Adding or Replacing a PCI or PCI-X Card” in chapter 12, “Maintenance and Upgrade Procedures” in the SGI Altix 3000 User’s Guide or see “PCI and PCI-X Cards” in Chapter 6, “Installing and Removing Customer-replaceable Units” in the SGI Altix 350 User’s Guide or see the appropriate chapter in the new *SGI Altix 4700 User’s Guide*.

### Allows Variable Partition Sizes

Partitions can be of different sizes, and a particular system can be configured in more than one way. For example, a 128-processor system could be configured into four partitions of 32 processors each or configured into two partitions of 64 processors each. (See "Supported Configurations" for a list of supported configurations for system partitioning.)

Your choice of partition size and number of partitions affects both fault containment and scalability. For example, you may want to dedicate all 64 processors of a system to a single large application during the night, but then partition the system in two 32 processor systems for separate and isolated use during the day.

### Provide High Performance Clusters

One of the fundamental factors that determines the performance of a high-end computer is the bandwidth and latency of the memory. The SGI NUMAflex technology gives an SGI ProPack partitioned, shared-memory cluster a huge performance advantage over a cluster of commodity Linux machines (white boxes). If a cluster of N white boxes, each with M CPUs is connected via Ethernet or Myrinet or InfinaBand, an SGI ProPack system with N partitions of M CPUs provides superior performance because of the significantly lower latency of the NUMAlink interconnect, which is exploited by the XPNET kernel module.

## Limitations of Partitioning

Partitioning can increase the reliability of a system because power failures and other hardware errors can be contained within a particular partition. There are still cases where the whole shared memory cluster is affected; for example, during upgrades of harware which is shared by multiple partitions.

If a partition is sharing its memory with other partitions, the loss of that partition may take down all other partitions that were accessing its memory. This is currently possible when an MPI or SHMEM job is running across partitions using the XPMEM kernel module.

Failures can usually be contained within a partition even when memory is being shared with other partitions. Starting with the SGI ProPack 4 for Linux Service Pack 3 release, XPC is invoked using normal shutdown commands such as `reboot`(8) and `halt`(8) to ensure that all memory shared between partitions is revoked before the partition resets. This is also done if you remove the XPC kernel modules using the `rmmod`(8) command. Unexpected failures such as kernel panics or hardware failures almost always force the affected partition into the KDB kernel debugger or the LKCD crash dump utility. These tools also invoke XPC to revoke all memory shared between partitions before the partition resets. XPC cannot be invoked for unexpected failures such as power failures and spontaneous resets (not generated by the operating system), and thus all partitions sharing memory with the partition may also reset.

## Supported Configurations

See the *SGI Altix 3000 User's Guide*, *SGI Altix 350 User's Guide*, *SGI Altix 450 User's Guide*, or the *SGI Altix 4700 User's Guide* for information on configurations that are supported for system partitioning. Currently, the following guidelines are valid for SGI ProPack 6 for Linux release:

- Maximum number of partitions supported is 48
- Maximum partition size is 1024 cores
- Maximum system size is 9726 cores

For additional information about configurations that are supported for system partitioning, see your sales representative.

## Installing Partitioning Software and Configuring Partitions

To enable or disable partitioning software, see "Partitioning Software" on page 18, to use the system partitioning capabilities, see "Partitioning Guidelines for SGI Altix 3000 Series Systems" on page 20 and "Partitioning a System" on page 22.

This section covers the following topics:

- "Partitioning Software" on page 18
- "Partitioning Guidelines" on page 19
- "Partitioning a System" on page 22
- "Determining If a System is Partitioned" on page 28
- "Accessing the Console on a Partitioned System" on page 29
- "Unpartitioning a System" on page 30

### Partitioning Software

SGI ProPack for Linux servers have XP, XPC, XPNET, and XPMEM kernel modules installed by default to provide partitioning support. XPC and XPNET are configured off by default in the `/etc/sysconfig/sgi-xpc` and `/etc/sysconfig/sgi-xpnet` files, respectively. XPMEM is configured on by default in the `/etc/sysconfig/sgi-xpmem` file. To enable or disable any of these features, edit the appropriate `/etc/sysconfig/` file and execute the `/etc/init.d/sgi-xp` script.

On SGI ProPack 4, SGI ProPack 5 or SGI ProPack 6 systems running SLES10, if you intend to use the cross-partition functionality of XPMEM, you will need to add `xpc` to the line in the `/etc/sysconfig/kernel` file that begins with `MODULES_LOADED_ON_BOOT`. Once that is added, you may either reboot the system or issue an `modprobe xpc` command to get the cross-partition functionality to start working. For more information on using `modprobe`, see the `modprobe`(8) man page.

To activate `xpc` on future boots on SGI ProPack 6 systems running RHEL5, you need to add a `modprobe xpc` line to the `/etc/sysconfig/modules/sgi-propack.modules` file instead of adding the module name to the `MODULES_LOADED_ON_BOOT` line.

The XP kernel module is a simple module which coordinates activities between XPC, XPMEM, and XPNET. All of the other cross-partition kernel modules require XP to function.

The XPC kernel module provides fault-tolerant, cross-partition communication channels over NUMAlink for use by the XPNET and XPMEM kernel modules.

The XPNET kernel module implements an Internet protocol (IP) interface on top of XPC to provide high-speed network access via NUMAlink. XPNET can be used by applications to communicate between partitions via NUMAlink, to mount file systems across partitions, and so on. The XPNET driver is configured using the `ifconfig` commands. For more information, see the `ifconfig`(1M) man page. The procedure for configuring the XPNET kernel module as a network driver is essentially the same as the procedure used to configure the Ethernet driver. You can configure the XPNET driver at boot time like the Ethernet interfaces by using the configuration files in `/etc/sysconfig/network-scripts`. To configure the XPNET driver as a network driver see the following procedure.

**Procedure 1-1** Setting up Networking Between Partitions

The procedure for configuring the XPNET driver as a network driver is essentially the same as the procedure used to configure the Ethernet driver (eth0), as follows:

1. Log in as root.
2. On a SGI ProPack 3 system, configure the xp0 IP address, as follows:

   ```
   netconfig -d xp0
   ```

   For a SGI ProPack 4 system , configure the xp0 IP address using `yast2`. For information on using `yast2`, see *SUSE LINUX Enterprise Server 9 Installation and Administration* manual. The driver's full name inside `yast2` is `SGI Cross Partition Network adapter`.
3. Add the network address for the xp0 interface by editing the `/etc/hosts` file.
4. Reboot your system or restart networking.

The XPMEM kernel module provides direct access to memory located on other partitions. It uses XPC internally to communicate with XPMEM kernel modules on other partitions to accomplish this. XPMEM is currently used by SGI's Message Passing Toolkit (MPT) software (MPI and SHMEM).

## Partitioning Guidelines

Partitioning rules define the set of valid configurations for a partitioned system. Fault isolation is one of the major reasons for partitioning a system. A software or hardware failure in one partition should not cause a failure in another partition. This section describes restrictions are placed on partitions to accomplish this. This section covers these topics:

- "Partitioning Guidelines for SGI Altix 3000 Series Systems" on page 20
- "Partitioning Guidelines for SGI Altix 450 Series Systems" on page 21
- "Partitioning Guidelines for SGI Altix 4000 Series Systems" on page 21

## Partitioning Guidelines for SGI Altix 3000 Series Systems

Follow these guidelines when partitioning your system:

- A partition must be made up of one or more C-bricks. The number of C-bricks in your systems determines the number of partitions you can create. The number of partitions cannot exceed the number of C-bricks your system contains. The first C-brick in each partition must have an IX-brick attached to it via the XIO connection of the C-brick.
- You need at least as many IX-bricks for base IO as partitions you wish to use.
- Each partition needs to have an IX-brick with a valid system disk in it. Since each partition is a separate running system, each system disk should be configured with a different IP address/system name, and so on.
- Each partition must have a unique partition ID number between 1 and 63, inclusively.
- All bricks in a partition must be physically contiguous. The route between any two processors in the same partition must be contained within that partition, and not through any other partition. If the bricks in a partition are not contiguous, the system will not boot.
- Each partition must contain the following components:
  - At least one C-brick for system sizes of 64 C-bricks and below, or multiples of 4 C-bricks for system sizes of 65 C-bricks and above (minimum)
  - One IX-brick (minimum)
  - One root disk
  - One console connection

### Partitioning Guidelines for SGI Altix 450 Series Systems

Partitioning guidelines on a SGI Altix 450 system are, as follows:

- The minimum granularity for a partition is one IRU (ideally with its own power supply setup). On Altix 450 systems, this means four compute blades is the minimum level of hardware isolation.
- Each partition must have the infrastructure to run as a standalone system. This infrastructure includes a system disk and console connection.
- An I/O blade belongs to the partition that the attached IRU belongs to. I/O blades cannot be shared by two partitions.
- Peripherals, such as dual-ported disks, can be shared the same way two nodes in a cluster can share peripherals.
- Partitions must be contiguous in the topology (for example, the route between any two nodes in the same partition must be contained within that partition - and not route through any other partition). This allows intra-partition communication to be independent of other partitions.
- Partitions must be fully interconnected. That is to say, for any two partitions, there is a direct route between those partitions without passing through a third. This is required to fulfill true isolation of a hardware or software fault to the partition in which it occurs.

### Partitioning Guidelines for SGI Altix 4000 Series Systems

Partitioning guidelines on a SGI Altix 4700 system are, as follows:

- The minimum granularity for a partition is one individual rack unit (IRU) (ideally with its own power supply setup). On SGI Altix 4700 systems, this means eight compute blades is the minimum level of hardware isolation.
- Each partition must have the infrastructure to run as a standalone system. This infrastructure includes a system disk and console connection.
- An I/O blade belongs to the partition to which the attached IRU belongs. I/O blades cannot be shared by two partitions.
- Peripherals, such as dual-ported disks, can be shared the same way two nodes in a cluster can share peripherals.

- Partitions must be contiguous in the topology (for example, the route between any two nodes in the same partition must be contained within that partition; and not route through any other partition). This allows intra-partition communication to be independent of other partitions.
- Quad-dense meta-routers ( 32-port routers) are a shared resource. A single quad-dense meta-router can connect to multiple partitions.
- Partitions must be fully interconnected. That is, for any two partitions, there is a direct route between those partitions without passing through a third. This is required to fulfill true isolation of a hardware or software fault to the partition in which it occurs.
- When the total system is greater than 16 IRUs (128 SHubs), it runs in coarse mode. In coarse mode, the minimum partition size is two IRUs (16 SHUBs).

## Partitioning a System

This section describes how to partition your system.

**Procedure 1-2** Partitioning a System Into Four Partitions

To partition your system, perform the following steps :

1. Make sure your system can be partitioned. See "Partitioning Guidelines for SGI Altix 3000 Series Systems" on page 20.
2. You can use the **Connect to System Controller** task of the SGIconsole Console Manager GUI to connect to the L2 controller of the system you want to partition. The L2 controller must appear as a node in the SGIconsole configuration. For information on how to use SGIconsole, see the *Console Manager for SGIconsole Administrator's Guide.*
3. Using the L2 terminal (l2term), connect to the L2 controller of the system you wish to partition. After a connection to the L2 controller, an L2> prompt appears, indicating that the L2 is ready to accept commands, for example:

   ```
   cranberry-192.168.11.92-L2>:
   ```

   If the L2 prompt does not appear, you can type `Ctrl-T`. To remain at the L2 command prompt, type **`l2`** (lowercase letter 'L') at the L2> prompt

**Note:** Each partition has its own set of the following PROM environment variables: `ConsolePath`, `OSLoadPartition`, `SystemPartition`, `netaddr`, and `root`.

For more information on using the L2 controller, see the *SGI L1 and L2 Controller Software User's Guide.*

You can partition a system from SGIconsole Console Manager system console connection, however, SGIconsole does not include any GUI awareness of partitions (in the node tree view for instance) or in the commands, and there is no way to power down a group of partitions, or get all the logs of a partitioned system, or to actually partition a system. If you partition a node that is managed by SGIconsole, make sure to edit the partition number of the node using the **Modify a Node** task. For more information, see the *Console Manager for SGIconsole Administrator's Guide.*

4. Use the L2 `sel` command to list the available consoles, as follows:

   ```
   cranberry-192.168.11.92-L2>sel
   ```

   **Note:** If the Linux operating system is currently executing, perform the proper shutdown procedures before you partition your system.

5. This step shows an example of how to partition a system into four separate partitions.

   a. To see the current configuration of your system, use the L2 `cfg` command to display the available bricks, as follows:

   ```
   cranberry-192.168.11.92-L2>cfg
        L2 192.168.11.92: - --- (no rack ID set) (LOCAL)
        L1 192.168.11.92:8:0    - 001c31
        L1 192.168.11.92:8:1    - 001i34
        L1 192.168.11.92:11:0   - 001c31
        L1 192.168.11.92:11:1   - 001i34
        L1 192.168.11.92:6:0    - 001r29
        L1 192.168.11.92:9:0    - 001r27
        L1 192.168.11.92:7:0    - 001c24
        L1 192.168.11.92:7:1    - 101i25
        L1 192.168.11.92:10:0   - 001c24
   ```

```
L1 192.168.11.92:10:1   - 101i25
L1 192.168.11.92:0:0    - 001r22
L1 192.168.11.92:3:0    - 001r20
L1 192.168.11.92:2:0    - 001c14
L1 192.168.11.92:2:1    - 101i21
L1 192.168.11.92:5:0    - 001c14
L1 192.168.11.92:5:1    - 101i21
L1 192.168.11.92:1:0    - 001c11
L1 192.168.11.92:1:1    - 001i07
L1 192.168.11.92:4:0    - 001c11
L1 192.168.11.92:4:1    - 001i07
```

b. In this step, you need to decide which bricks to put into which partitions.

You can determine which C-bricks are directly attached to IX-bricks by looking at the output from the `cfg` man. Consult the hardware configuration guide for the partitioning layout for your particular system. In the `cfg` output above, you can check the number after the IP address. For example, `001c31` is attached to `001i34` which is indicated by the fact that they both have **`11`** after their respective IP address.

---

**Note:** On some systems, you will have a rack ID in place of the IP address. `001c31` is a C–brick (designated by the `c` in `001c31`) and `001i34` is an IX-brick (designated with an `i` in `001i34`).

---

Another pair is `101i25` and `001c24`. They both have **`10`** after the IP address. The brick names containing an `r` designation are routers. Routers do not need to be designated to a specific partition number.

In this example, the maximum number of partitions this system can have is four. There are only four IX-bricks total: `001i07`, `101i21`, `101i25`, and `001i34`.

---

**Note:** Some IX-brick names appear twice. This occurs because some IX-bricks have dual XIO connections.

You do not have to explicitly assign IX-bricks to a partition. The IX-bricks assigned to a partition are inherited from the C-bricks.

---

c. When you specify bricks to L2 commands, you use a *rack.slot* naming convention. To configure the system into four partitions, do not specify the whole brick name (`001c31`) but rather use the designation 1.31 as follows:

```
cranberry-192.168.11.92-L2>1.31 brick part 1
     001c31:
     brick partition set to 1.
     cranberry-192.168.11.92-L2>1.34 brick part 1
     001#34:
     brick partition set to 1.
     cranberry-192.168.11.92-L2>1.24 brick part 2
     001c24:
     brick partition set to 2.
     cranberry-192.168.11.92-L2>101.25 brick part 2
     101#25:
     brick partition set to 2.
     cranberry-192.168.11.92-L2>1.14 brick part 3
     001c14:
     brick partition set to 3.
     cranberry-192.168.11.92-L2>101.21 brick part 3
     101i21:
     brick partition set to 3.
     cranberry-192.168.11.92-L2>1.11 brick part 4
     001c11:
     brick partition set to 4.
     cranberry-192.168.11.92-L2>1.07 brick part 4
     001#07:
     brick partition set to 4.
```

d. To confirm your settings, enter the `cfg` command again, as follows:

**Note:** This may take up to 30 seconds.

```
cranberry-192.168.11.92-L2>cfg
     L2 192.168.11.92: - --- (no rack ID set) (LOCAL)
     L1 192.168.11.92:8:0    - 001c31.1
     L1 192.168.11.92:8:1    - 001i34.1
     L1 192.168.11.92:11:0   - 001c31.1
     L1 192.168.11.92:11:1   - 001i34.1
     L1 192.168.11.92:6:0    - 001r29
     L1 192.168.11.92:9:0    - 001r27
```

```
L1 192.168.11.92:7:0    - 001c24.2
L1 192.168.11.92:7:1    - 101i25.2
L1 192.168.11.92:10:0   - 001c24.2
L1 192.168.11.92:10:1   - 101i25.2
L1 192.168.11.92:0:0    - 001r22
L1 192.168.11.92:3:0    - 001r20
L1 192.168.11.92:2:0    - 001c14.3
L1 192.168.11.92:2:1    - 101i21.3
L1 192.168.11.92:5:0    - 001c14.3
L1 192.168.11.92:5:1    - 101i21.3
L1 192.168.11.92:1:0    - 001c11.4
L1 192.168.11.92:1:1    - 001i07.4
L1 192.168.11.92:4:0    - 001c11.4
L1 192.168.11.92:4:1    - 001i07.4
```

e. The system is now partitioned. However, you need to reset each partition to complete the configuration, as follows:

```
cranberry-192.168.11.92-L2>p 1,2,3,4 rst
```

---

**Note:** You can use a shortcut to reset every partition, as follows:

```
cranberry-192.168.11.92-L2>p * rst
```

---

f. To get to the individual console of a partition, such as partition 2, enter the following:

```
cranberry-192.168.11.92-L2>sel p 2
```

For more information on accessing the console of a partition, see "Accessing the Console on a Partitioned System" on page 29.

**Procedure 1-3** Partitioning a System into Two Partitions

To partition your system, perform the following steps:

1. Perform steps 1 through 5 in Procedure 1-2, page 22.
2. To configure the system into two partitions, enter the following commands:

```
cranberry-192.168.11.92-L2>1.31 brick part 1
     001c31:
```

```
brick partition set to 1.
cranberry-192.168.11.92-L2>1.34 brick part 1
001#34:
brick partition set to 1.
cranberry-192.168.11.92-L2>1.24 brick part 1
001c24:
brick partition set to 1.
cranberry-192.168.11.92-L2>101.25 brick part 1
101#25:
brick partition set to 1.
cranberry-192.168.11.92-L2>1.14 brick part 2
001c14:
brick partition set to 2.
cranberry-192.168.11.92-L2>101.21 brick part 2
101i21:
brick partition set to 2.
cranberry-192.168.11.92-L2>1.11 brick part 2
001c11:
brick partition set to 2.
cranberry-192.168.11.92-L2>1.7 brick part 2
001#07:
brick partition set to 2.
```

3. To confirm your settings, issue the `cfg` command again, as follows:

---

**Note:** This may take up to 30 seconds.

---

```
cranberry-192.168.11.92-L2>cfg
     L2 192.168.11.92: - --- (no rack ID set) (LOCAL)
     L1 192.168.11.92:8:0    - 001c31.1
     L1 192.168.11.92:8:1    - 001i34.1
     L1 192.168.11.92:11:0   - 001c31.1
     L1 192.168.11.92:11:1   - 001i34.1
     L1 192.168.11.92:6:0    - 001r29
     L1 192.168.11.92:9:0    - 001r27
     L1 192.168.11.92:7:0    - 001c24.1
     L1 192.168.11.92:7:1    - 101i25.1
     L1 192.168.11.92:10:0   - 001c24.1
     L1 192.168.11.92:10:1   - 101i25.1
     L1 192.168.11.92:0:0    - 001r22
```

```
L1 192.168.11.92:3:0    - 001r20
L1 192.168.11.92:2:0    - 001c14.2
L1 192.168.11.92:2:1    - 101i21.2
L1 192.168.11.92:5:0    - 001c14.2
L1 192.168.11.92:5:1    - 101i21.2
L1 192.168.11.92:1:0    - 001c11.2
L1 192.168.11.92:1:1    - 001i07.2
L1 192.168.11.92:4:0    - 001c11.2
L1 192.168.11.92:4:1    - 001i07.2
```

4. Now the system has two partitions. To complete the configuration, reset the two partitions as follows:

```
cranberry-192.168.11.92-L2>p 1,2 rst
```

## Determining If a System is Partitioned

**Procedure 1-4** Determing If a System Is Partitioned

To determine whether a system is partitioned or not, perform the following steps:

1. Use the L2term to connect to the L2 controller of the system.

**Note:** If you are connected to the L2 controller, but do not have the L2 prompt, try typing the following: `CTRL-t`.

2. Use the `cfg` command to determine if the system is partitioned, as follows:

```
cranberry-192.168.11.92-L2>cfg
L2 192.168.11.92: -(no rack ID set) (LOCAL)
L1 192.168.11.92:8:0    - 001c31.1
L1 192.168.11.92:8:1    - 001i34.1
L1 192.168.11.92:11:0   - 001c31.1
L1 192.168.11.92:11:1   - 001i34.1
L1 192.168.11.92:6:0    - 001r29
L1 192.168.11.92:9:0    - 001r27
L1 192.168.11.92:7:0    - 001c24.2
L1 192.168.11.92:7:1    - 101i25.2
L1 192.168.11.92:10:0   - 001c24.2
L1 192.168.11.92:10:1   - 101i25.2
```

```
L1 192.168.11.92:0:0    - 001r22
L1 192.168.11.92:3:0    - 001r20
L1 192.168.11.92:2:0    - 001c14.3
L1 192.168.11.92:2:1    - 101i21.3
L1 192.168.11.92:5:0    - 001c14.3
L1 192.168.11.92:5:1    - 101i21.3
L1 192.168.11.92:1:0    - 001c11.4
L1 192.168.11.92:1:1    - 001i07.4
L1 192.168.11.92:4:0    - 001c11.4
L1 192.168.11.92:4:1    - 001i07.4
```

3. See the explanation of the output from the `cfg` command in Procedure 1-2.

## Accessing the Console on a Partitioned System

**Procedure 1-5** Access the Console on a Partitioned System

To access the console on a partition, perform the following steps:

1. Use the L2term to connect to the L2 controller of the system.

   **Note:** If you are connected to the L2 controller, but do not have the L2 prompt, try typing the following: `CTRL-t`.

2. To see output that shows which C-bricks have system consoles, enter the `sel` command without options on a partitioned system as follows:

```
cranberry-192.168.11.92-L2>sel

     known system consoles (partitioned)

             partition  1: 001c31 - L2 detected
             partition  2: 001c24 - L2 detected
             partition  3: 001c14 - L2 detected
             partition  4: 001c11 - L2 detected

     current system console

     console input: not defined
     console output: not filtered
```

   The output from the `sel` command shows that there are four partitions defined.

3. To get to the console of partition 2, for example, enter the following:

   ```
   cranberry-192.168.11.92-L2>sel p 2
   ```

4. To connect to the console of partition 2, enter `Ctrl-d`.

When a system is partitioned, the L2 prompt shows the partition number of the partition you selected, as follows:

```
cranberry-001-L2>sel p 2
console input: partition 2, 001c24 console0
console output: any brick partition 2
cranberry-001-L2:p2>
```

### Unpartitioning a System

**Procedure 1-6** Unpartitioning a System

To remove the partitions from a system, perform the following steps:

1. Use the L2term to connect to the L2 controller of the system.

   ---

   **Note:** If you are connected to the L2 controller, but do not have the L2 prompt, try typing the following: `CTRL-t`.

   ---

2. Shut down the Linux operating system running on each partition before unpartitioning a system.

3. To set the partition ID on all bricks to zero, enter the following command:

   ```
   cranberry-192.168.11.92-L2>r * brick part 0
   ```

4. To confirm that all the partitions on your system have been removed, enter the following command:

   ```
   cranberry-192.168.11.92-L2>cfg
   ```

   The list of bricks no longer have a dot followed by a number in their name (see "Determining If a System is Partitioned" on page 28).

5. To reset all of the bricks, enter the following command:

   ```
   cranberry-192.168.11.92-L2>r * rst
   ```

6. To get to the system console for the newly unpartitioned system, you need to reset the select setting as follows:

   ```
   cranberry-192.168.11.92-L2>sel reset
   ```

7. To get the console (assuming you still have the L2 prompt), enter `Ctrl-d`.

### Connecting the System Console to the Controller

System partitioning is an administrative function. The system console is connected to the controller as required by the configuration selected when an SGI ProPack system is installed. For additional information or recabling, contact your service representative.

## Making Array Services Operational

This section describes how to get Array Services operational on your system. For detailed information on Array Services, see chapter 3, “Array Sevices”, in the *Linux Resource Administration Guide.*

---

**Note:** This section does not apply to SGI Altix XE systems.

---

Standard Array Services is installed by default on an SGI ProPack 6 system. To install Secure Array Services, use the YaST Software Management and use the **Filter->search** function to search for secure array services by name (`sarraysvcs`).

**Procedure 1-7** Making Array Services Operational

To make Array Services operational on your system, perform the following steps:

---

**Note:** Most of the steps to install array services is now performed automatically when the array services RPM is installed. To complete installation, perform the steps that follow.

---

1. Make sure that the setting in the `/usr/lib/array/arrayd.auth` file is appropriate for your site.

   ---

   **Caution:** Changing the `AUTHENTICATION` parameter from *NOREMOTE* to *NONE* may have a negative security impact on your site.

   ---

2. Make sure that the list of machines in your cluster is included in one or more array definitions in `/usr/lib/array/arrayd.conf` file.
3. To determine if Array Services is correctly installed, run the following command:

   ```
   array who
   ```

   You should see yourself listed.

## Floating Point Assist Warnings from Applications

Some applications can generate an excessive number of kernel `KERN_WARN` "floating point assist" warning messages. This section describes how you can make these messages disappear for specific applications or for specific users.

---

**Note:** This section does not apply to SGI Altix XE systems.

---

An application generates a "floating point assist" trap when a floating point operation involves a corner case of operand value(s) that the Itanium processor cannot handle in hardware and requires kernel emulation software to complete.

Sometimes the application can be recompiled with specific options that will avoid such problematic operand value(s). Using the `-ffast-math` option with `gcc` or the `-ftz` option with the Intel compiler might be effective. Other instances can be avoided by recoding the application to avoid denormalized or non-finite operands.

When recoding or recompiling is ineffective or infeasible, the user running the application or the system administrator can avoid having these "floating point assist" syslog messages appear by using the `prctl`(1) command, as follows:

```
 % prctl --fpemu=silent command
```

The *command* and every child process of the *command* invoked with `prctl` will produce no "floating point assist" messages. The *command* can be a single application that is producing unwanted syslog messages or it may be any shell script. For example, the "floating point assist" messages for the user as a whole, that is, for all applications that the user may execute, can be silenced by changing the `/etc/passwd` entry of the user to invoke a custom script at login, instead of executing (for instance) `/bin/bash`. That custom script then launches the user's high level shell using the following:

```
prctl --fpemu=silent /bin/bash
```

Even if the syslog messages are silenced, the "assist" traps will continue to occur and the kernel software will still handle the problem. If these traps occur at a high enough frequency, the application performance may suffer and notification of these occurrences are not logged.

The syslog messages can be made to reappear by executing the `prctl` command again, as follows:

```
% prctl --fpemu=default command
```

## Unaligned Access Warnings

**Note:** This section does not apply to SGI Altix XE systems.

The section describes unaligned access warnings, as follows:

- The kernel generated unaligned access warnings in syslog and on the console, when applications do misaligned loads and stores. This is normally a sign of badly written code and an indication that the application should be fixed.
- Use the `prctl`(1) command to disable these messages on a per application basis.
- SLES10 offers a new way allowing system administrators to disable these messages on a system wide basis. This is generally discouraged, but useful for the case where a system is used to run third-party applications which cannot be fixed by the system owner.

  In order to disable these messages on a system wide level, do the following as root:

  ```
  echo 1 > /proc/sys/kernel/ignore-unaligned-usertrap
  ```

*Chapter 2*

# System Operation

This chapter describes the operation of an SGI Altix systems. It covers the following topics:

**Note:** This chapter does not apply to SGI Altix XE systems.

## Booting a System

This section describes how to boot an SGI Altix series computer system.

**Procedure 2-1** Booting a System

To boot an SGI Altix system, perform the following:

1. Obtain the system console as described in "Getting Console Access" on page 43 if you are using SGIconsole or telnet to the L2 controller as described in "Connecting to the L2 Controller" on page 42.
2. By default, when booting a menu of boot options appears. On a properly configured system (as shipped from the factory), you can boot directly to the Linux operating sytem. A system administrator can change the default using the boot mainenance menus, the extensible firmware interface (EFI) shell> command, or the `efibootmgr` command (see the `efibootmgr` options usage statement for more information).

---

**Note:** To see the boot menus properly on an SGI Altix system, make sure sure the debug switches are set to **0** or **1**.

---

A screen similar to the following appears after booting your machine:

```
EFI Boot Manager ver 1.02 [12.38]

Partition  0:                          Enabled Disabled
    CBricks         1         Nodes         2        0
    RBricks         0         CPUs          4        0
    IOBricks        1         Mem(GB)       4        0

Please select a boot option

    UnitedLinux
    ProPack
    Boot option maintenance menu


    Use the arrow keys to change option(s).  Use Enter to select an option
```

One of the menu options appears highlighted. Pressing arrow keys moves the highlight.

An example of selecting the **EFI Boot Maintenance Manager** menu is, as follows:

```
EFI Boot Maintenance Manager ver 1.02 [12.38]

Main Menu. Select an Operation


        Boot from a File
        Add a Boot Option
        Delete Boot Option(s)
        Change Boot Order

        Manage BootNext setting
        Set Auto Boot TimeOut

        Select Active Console Output Devices
        Select Active Console Input Devices
        Select Active Standard Error Devices

        Cold Reset
        Exit
```

An example of selecting the **Boot from a File** option is, as follows:

```
EFI Boot Maintenance Manager ver 1.02 [12.38]

Boot From a File.  Select a Volume


    NO VOLUME LABEL [Pci(1|1)/Scsi(Pun0,Lun1)/HD(Part1,Sig7CFD016D-A
    NO VOLUME LABEL [Pci(1|1)/Scsi(Pun0,Lun2)/HD(Part1,Sig1F9EFFAD-7
    Default Boot [Pci(1|1)/Scsi(Pun0,Lun1)]
    Default Boot [Pci(1|1)/Scsi(Pun0,Lun2)]
    Load File [Pci(4|0)/Mac(08006913DB7D)/NicName(tg0)]
    Load File [EFI Shell [Built-in]]
    Exit
```

An example of selecting Load File **EFI Shell** is, as follows:

```
    Device Path VenHw(D65A6B8C-71E5-4DF0-A909-F0D23000000099B40000002
BB40000005AB4000000A9)
EFI Shell version 1.02 [12.38]
Device mapping table
  fs0  : Pci(1|1)/Scsi(Pun0,Lun1)/HD(Part1,Sigg1)
  fs1  : Pci(1|1)/Scsi(Pun0,Lun2)/HD(Part1,Sigg2)
  blk0 : Pci(1|1)/Scsi(Pun0,Lun1)
  blk1 : Pci(1|1)/Scsi(Pun0,Lun1)/HD(Part1,Sigg1)
  blk2 : Pci(1|1)/Scsi(Pun0,Lun1)/HD(Part2,Sigg3)
  blk3 : Pci(1|1)/Scsi(Pun0,Lun1)/HD(Part3,Sigg4)
  blk4 : Pci(1|1)/Scsi(Pun0,Lun2)
  blk5 : Pci(1|1)/Scsi(Pun0,Lun2)/HD(Part1,Sigg2)
  blk6 : Pci(1|1)/Scsi(Pun0,Lun2)/HD(Part2,Sigg5)
  blk7 : Pci(1|1)/Scsi(Pun0,Lun2)/HD(Part3,Sigg6)
Shell>
```

From the system EFI shell> prompt, you can proceed with booting the system. Optional booting steps follow:

You have the option to select an EFI partition you want to load your kernel as follows. If you choose not to do this, `fs0:` is searched by default and the following prompt appears:

**`fs0:`**

If there are multiple EFI filesystems (for example, `fs1`, `fs2`, and so on), change to the one you from which you wish to load the kernel. Then perform the following:

a.

On an SGI ProPack 6 system, to boot the default kernel from the first boot disk, enter the following command at the prompt:

`Booting disk 1:`**`efi/SuSE/elilo`**

b. To boot the default kernel from the second root disk, change directory (`cd`) to `efi/sgi` (`efi/SuSE` on SGI ProPack 6) and enter the following command at the prompt:

`Booting disk 2:`**`elilo sgilinux root=/dev/xscsi/pci01.03.0-1/target2/lun0/part3`**

The preceding XSCSI path points to the seoncd disk of the primary IX-brick.

3. If the system is at the kernel debugger kdb> prompt or is not responding at all, try resetting the system from the system controller. To get to the system controller prompt, enter Ctrl-T. At the system controller (L1 or L2) prompt, enter `rst`. The EFI shell> promt appears.

4. To give up control of the console, perform one of the following:

   - For k consoles, enter Ctrl-] and then enter Ctrl-D.

- To exit a session from from SGIconsole, from the Console Manager **File** pulldown menu, choose **Exit**. To exit from the SGIconsole text-based user interface (tscm), enter **8** for quit.
- To exit a session in in either the graphical or text version of IRISconsole, enter the following: `~x`.
- To end a `telnet` session to the the L2 controller, enter the following: Ctrl-] and then Ctrl-D.

## Halting the System

This section describes how halt the Linux operating sytem and power down your system.

**Procedure 2-2** Halting the System

To halt the Linux operating sytem and power down your system, perform the following:

1. Connect to the L2> controller by following the steps in Procedure 2-4 on page 42.
2. Enter Ctrl-D to connect to the system console.
3. Enter the `halt` command.

   You can also reset the system by enter Ctrl-T to get to the L2> prompt and then enter the `rst` command to reset the system.
4. To to power on and power off individual bricks or your entire Altix system, see "Powering the System On and Off" section in the appropriate SGI Altix system hardware manual (see the "Related Publications" section in the *SGI ProPack 6 for Linux Start Here*.).

## Recovering from Machine Check Architecture Memory Errors

Starting with SGI ProPack 4 Service Pack 2, SGI Altix systems can recover from Machine Check Architecture (MCA) memory correction code (ECC) errors in user space (that is, not executing kernel code). The `mca_recovery` kernel module logs the process and memory page to the `/var/log/messages` file. The `salinfo_decoded` module logs the MCA to the `/var/log/salinfo/decoded` directory. The user code is killed.

An application the uses a page of memory and encounters an ECC error that produces the following in `/var/log/messages` file:

```
Oct 25 16:54:56 7A: kernel: OS_MCA: process [pid: 6094](errit) encounters MCA.  Oc
```

An MCA record is produced in the `/var/log/salinfo/decoded` file as shown in the following example;

```
/var/log/salinfo/decoded # ls -lrt
 total 8
 drwxr-xr-x  2 root root    6 Nov  5  2004 old
 drwxr-xr-x  4 root root   30 Jul 18 20:29 ..
 drwxr-xr-x  3 root root   67 Oct 25 16:54 .
 -rw-r--r--  1 root root 2209 Oct 25 16:54 oemdata
 -rw-r--r--  1 root root 2722 Oct 25 16:54 2005-10-25-21_54_54-cpu0-cpe.0
```

View the error message, as follows:

```
/var/log/salinfo/decoded # cat 2005-10-25-21_54_54-cpu0-cpe.0
 BEGIN HARDWARE ERROR STATE from cpe on cpu 0
 Err Record ID: 785412062707715    SAL Rev:  0.02
 Time: 2005-10-25 21:54:54    Severity 0
   Platform Memory Device Error Info Section
   Mem Error Detail
     Physical Address: 0x1301113a490 Address Mask: 0x3ffffffffffff Node: 4 Bank: 0
     OEM Specific Data

       UNCORRECTED MEMORY ERROR :module/001c14/slab/0/node :Loc DIMM1 N0_L_BUS_Y and/or DIMM1 N0_R_BUS_Y

         MBIST read address Data               ECC  SYN  FSB Bit
         0x000001301113a480 0x0000000000000000 0x00 0x00 good
         0x000001301113a488 0x0000000000000000 0x00 0x00 good
         0x000001301113a490 0x0003000000000000 0x00 0x30 multi
         0x000001301113a498 0x0000000000000000 0x00 0x00 good
         0x000001301113a4a0 0x0000000000000000 0x00 0x00 good
         0x000001301113a4a8 0x0000000000000000 0x00 0x00 good
         0x000001301113a4b0 0x0003000000000000 0x00 0x30 multi
         0x000001301113a4b8 0x0000000000000000 0x00 0x00 good
         0x000001301113a4c0 0x0000000000000000 0x00 0x00 good
         0x000001301113a4c8 0x0000000000000000 0x00 0x00 good
         0x000001301113a4d0 0x0003000000000000 0x00 0x30 multi
```

```
        0x000001301113a4d8 0x0000000000000000 0x00 0x00 good
        0x000001301113a4e0 0x0000000000000000 0x00 0x00 good
        0x000001301113a4e8 0x0000000000000000 0x00 0x00 good
        0x000001301113a4f0 0x0003000000000000 0x00 0x30 multi
        0x000001301113a4f8 0x0000000000000000 0x00 0x00 good

Platform Specific Error Info Section
  Platform Specific Error Detail
    OEM Specific Data

      UNCORRECTED ECC ERROR :module/001c14/slab/0/node :Processor received bad data from SHub

        SH_EVENT_OCCURRED                    : 0x0000000018000100
          PI Uncorrectable Error Interrupt Pending
        SH_FIRST_ERROR                       : 0x0000000000000100
          PI Uncorrectable Error Interrupt Pending
        SH_PI_ERROR_SUMMARY                  : 0x0000000020000000
          PI_UCE_INT: SHub-to-FSB Uncorrectable Data Error
        SH_PI_FIRST_ERROR                    : 0x0000000020000000
          PI_UCE_INT: SHub-to-FSB Uncorrectable Data Error
        SH_PI_ERROR_OVERFLOW                 : 0x0000000020000000
          PI_UCE_INT: SHub-to-FSB Uncorrectable Data Error
        SH_PI_UNCORRECTED_DETAIL_1           : 0x0030002602227492
          Address: 0x000001301113a490 Nasid: 0x4 Syndrome: 0x30 ECC: 0x00
        SH_PI_UNCORRECTED_DETAIL_2           : 0x0003000000000000
          Failing Dbl-word Data: 0x0003000000000000
        SH_PI_UNCOR_TIME_STAMP               : 0x8000013523cc4805
END HARDWARE ERROR STATE from cpe on cpu 0
```

The MCA recovery code does not attempt recovery if the CPU is in privileged mode (in kernel context). The MCA record will show if this is the case.

## Connecting to the L1 Controller

You can monitor the L1 controller status and error messages on the L1 controller's liquid crystal display (LCD) located on the front panel of the individual bricks. The L1 controller and L2 controller status and error messages can also be monitored at your system console. The system console allows you to monitor and manage your server or graphics system by entering L1 controller commands. You can also enter L2 controller commands to monitor and manage your system if your system has L2 controller hardware and a system console or if you are using an SGIconsole as your system console. For information on connecting to the system console, see "Getting Console Access" on page 43. For detailed information on using the L2 controller software, see the *SGI L1 and L2 Controller Software User's Guide*

**Procedure 2-3** Connecting to the L1 Controller

1. From the `Tasks` pulldown menu of SGIconsole Console Manager GUI, choose **Connect to a System Controller**.
2. To get to the L2> controller prompt, enter Ctrl -T.
3. To get back to the L1> controller prompt, enter Ctrl-D.

## Connecting to the L2 Controller

To access the L2 controller firmware, you must connect a system console such as SGIconsole or a dumb terminal, to the L2 controller. The L2 firmware is always running as long as power is supplied to the L2 controller. If you connect a system console to the L2 controller's console port, the L2 prompt appears. For instructions on connecting a console to the L2 controller, see your server or graphics system owner's guide or the *SGIconsole Hardware Connectivity Guide*.

The SGIconsole Console Manager graphical user interface (GUI) or text-based user interface (`tscm`(1)), can be used to securely access a system console and connect to an L2 controller. For information on using Console Manager to access an SGI Altix system or to access an SGI Altix system in secure mode using the `ssh`(1) command, see the *Console Manager for SGIconsole Administrator's Guide*.

Your SGI Altix system should have an L2 controller on your network that you can access. This section describes how you can connet to an L2 controller if you are not using SGIconsole.

**Procedure 2-4** Connecting to the L2 Controller

To connect to a system L2 controller, perform the following steps:

1. From the `Tasks` pulldown menu of SGIconsole Console Manager GUI, choose **Node Tasks -> Get/Steal/Spy**. Follow the instructions in the *Console Manager for SGIconsole Administrator's Guide* to connect to the console. You can also use the `tscm`(1) command line interface to Console Manager. If you do not have SGIconsole installed, proceed to the next step.

2. Use the `telnet`(1) command to connect to the L2 controller as follows:

   `telnet` **`L2-`*****system-name*****`.`*****domain-name*****`.`*****company***`.com`

   The *system-name* argument is the name of the SGI Altix system to which you want to connect. In some case, you may need to use the full domain such as *system-name.americas.sgi.com*.

3. Once connected, press the Enter key and a prompt similar to the following appears:

   ```
   system_name-001-L2>
   ```

4. To connect to the system console, enter Ctrl–D.

   If your system is partitioned, a message similar to the following appears:

   ```
   INFO: ERROR: no system console defined
   ```

   For information on working with partitioned systems, see "System Partitioning " on page 14.

---

**Note:** For detailed information on using the L2 controller software, see the *SGI L1 and L2 Controller Software User's Guide* and the *SGI Altix 3000 User's Guide.*

---

## Getting Console Access

This section describes how to access a system console.

**Procedure 2-5** Getting Console Access

1. From the `Tasks` pulldown menu of SGIconsole Console Manager GUI, choose **Node Tasks -> Get/Steal/Spy**. Follow the instructions in the *Console Manager for SGIconsole Administrator's Guide* to connect to the console. You can also use the `tscm`(1) command line interface to Console Manager. For information on using Console Manger or `tscm`(1), see *Console Manager for SGIconsole Administrator's Guide.*

   If you do not have Console Manager installed, proceed to the next step.

2. To connect to a system L2 controller, perform the steps in Procedure 2-4 on page 42.

3. Once connected, press the Enter key and a prompt similar to the following appears:

   ```
   system_name-001-L2>
   ```

4. To connect to the system console, enter Ctrl–D.

   If your system is partitioned, a message similar to the following appears:

   ```
   INFO: ERROR: no system console defined
   ```

   For information on working with partitioned systems, see "System Partitioning " on page 14.

5. To return to the L2 controller, enter Ctrl-T.

   To return to the telnet prompt, enter Ctrl-] (control -right bracket).

---

**Note:** For detailed information on using the L2 controller software, see the *SGI L1 and L2 Controller Software User's Guide.*

---

## Diskless Booting

This section describes diskless booting supported on SGI Altix systems and covers the following topics:

- "Altix Diskless Booting Overview" on page 44
- "Diskless Booting from CD" on page 45
- "Diskless Booting from the Network" on page 49

### Altix Diskless Booting Overview

This section describes an SGI approach to diskless booting of SGI Altix systems. Other approaches to diskless booting have been covered thoroughly by Linux HOWTO documents. You can use your favorite web search engine and looking for "linux diskless boot howto". You should find ample information on the concepts and methods available through Linux.

Unlike other diskless client-server models, Altix diskless systems do not rely on a server to operate after being booted. This allows a client to boot over a satellite link and continue to operate after that link has been broken.

Altix diskless booting uses a ramdisk for the root filesystem instead of NFS mounting the root filesystem. Many diskless clients operate with a root filesystem provided over a network using a remote filesystem protocol such as NFS. This approach suffers from several disadvantages, including the following:

- A strong dependence on the file server, that typically results in the client hanging if communication to the file server is lost
- Poor performance as many normal system operations (for example, opening a temporary file in `/tmp`) need several round trips over the network
- The file server needs to share writable root filesystem images for every client, which is complex and tedious to manage

The Altix root filesystem in diskless operation is fully contained in memory (that is, ramdisk). Root ramdisk uses more RAM in the client, so you will require more RAM installed to get the same level of application performance as the same machine booted by disk.

A well-equipped Altix system running diskless, for example, would have 16GB of system memory; a 4GB ramdisk for the operating system, another 4GB ramdisk for applications, such as OpenGL Performer and 8GB or more of unused ramdisk space for system memory (16GB minus 8GB plus unused ramdisk memory). Applications requiring large amounts of data can run CXFS.

Although system swapping is not required if there is sufficient memory, swapping may be added and verified as shown in the last step of Procedure 2-6, page 46.

Because the root file system is in memory, system performance is increased above that of systems with local disk.

## Diskless Booting from CD

You can use the Altix Standalone Maintenance CD without installing on hard drive to perform the following:

- Flash snprom images directly from CD.
- Boot SGI ProPack 6 for Linux from CD in about three minutes time on 2.6.5-xxxx-rtgfx kernel and modules to perform system maintenance. Once booted,

it has the same capabilities to telnet to the system as an L3 controller on CD. Note that the CD can be unmounted after booted so other CDs can be mounted.

---

**Note:** You can use any supported 2.6.x kernel.

---

**Procedure 2-6** Diskless Booting from CD

To use the CD in **rescue mode** to accomplish diskless booting, perform the following steps:

1. Boot the CD, as follows:

   ```
   fs0:\>bootia64
   ```

   Hit the `Enter` key, wait, then hit the `Enter` key again and you should see a message similar to the following:

   ```
   Uncompressing Linux... done
   Loading initrd initrd-2.6.5-7.201-rtgfx.../
   ```

2. Mount the DVD/CD, as follows:

   ```
   mount /dev/hda /media/dvd
   ```

3. Identify disks, as follows:

   ```
   dmesg | grep SCSI
   ```

4. Print out partition table, as follows:

```
machine-A# parted /dev/sda print
Disk geometry for /dev/sda: 0.000-78533.437 megabytes
Disk label type: gpt
Minor    Start       End     Filesystem  Name                    Flags
1          0.017    500.000  fat16                               boot
2        500.000  20500.000  xfs
4      20500.000  21000.000  fat16
5      21000.000  42000.000  xfs
6      42000.000  68000.009  xfs
3      68000.010  78533.421  linux-swap
Information: Don’t forget to update /etc/fstab, if necessary.
```

5. Mount the partition, as follows:

   **`mount /dev/sda2 /mnt`**

6. Determine which filesystem is the root filesystem and the mount `/boot/efi` on to it, as follows:

   **`mount /dev/sda1 /mnt/boot/efi`**

   Typical output is similar to the following:

```
/dev/sda2             20469760   6461768  14007992  32% /mnt
/dev/sda1               511712     62480    449232  13% /mnt/boot/efi
```

7. Use the `chroot`(1) command to setup the root directory for installing RPMs and so on, as follows:

   **`cd /mnt; chroot .`**

8. Backup the root filesystem to a system on your network, as follows:

```
xfsdump -l0 - /mnt | gzip -c | ssh root@fax.americas.sgi.com dd of=/bigdisk/altixbackup.dgz
```

**Note:** Answer **`yes`** and then enter the password.

**Warning:** Note that the `dd of=` string in this command will overwrite files!

9. Restore a backup from a remote system, as follows:

```
ssh root@backup.eng.sgi.com "dd if=/bigdisk/altixbackup.dgz" | gunzip -c | xfsrestore - /mnt
```

**Note:** Use the quotes to keep the `gunzip` command from running on the remote system.

10. If a DHCP server did not start the network, start it manually, as follows:

```
ifconfig eth0 149.xxx.xxx.xx netmask 255.255.xxx.xxx up
```

11. From the system booted from CD, use the `wget`(1) command to download an RPM at the location below, as follows:

    `wget` http://rpms.eng.sgi.com/package.rpm

12. Login from another system using anonymous ftp, as follows:

    `ftp` *IP address of Altix system*

    For example:

```
Name (ptc-tulip.americas:system-user): ftp
```

**Note:** Name can be ftp or anonymous.

13. Telnet from another system, as follows;

```
% telnet 169.239.221.86
Trying 169.239.221.86...
Connected to 169.239.221.86.
Escape character is '^]'.

Linux 2.6.5-7.201-rtgfx (sgialtix) (0)
```

```
bash-2.05b#
```

14. Although system swapping is not required if there is sufficient memory, swapping may be added and verified, as follows:

```
bash-2.05b# swapon -a /dev/sda3
Adding 10786176k swap on /dev/sda3.  Priority:-1 extents:1


bash-2.05b# swapon -s
Filename                                Type            Size    Used    Priority
/dev/sda3                               partition       10786176        0   -1
```

**Note:** Swap partitions can be identified using the `parted` command, see step 4.

If a swap partition does not exist, a swap partition may be defined and created, as follows:

```
bash-2.05b# mkswap /dev/sda3
Setting up swapspace version 1, size = 11045060 kB
```

**Warning:** All data in the target partition is destroyed.

Go to Supportfolio, the SGI support web site, for additional information about Altix Standalone Maintenance CD and diskless booting at: https://support.sgi.com/login

## Diskless Booting from the Network

This section describes diskless booting from the network.

Three network services required for diskless boot from network are, as follows:

- DHCP server
- TFTP server
- Anonymous FTP or HTTP server

The Boot Option Maintenance menu may be used to set a network device as the default boot device.

An example EFI boot device is, as follows:

```
[Pci(4|0)/Mac(08006913F0B4)/NicName(tg0) ]
```

When a network device is selected as the boot device, a DHCP client request is made to a DHCP server. Diskless booting requires two special parameters `next-server` and `filename` to identify the TFTP server and the boot loader file.

An example entry for these parameters in `dhcp.conf` file is, as follows:

```
next-server 169.238.221.85;
filename "/tftpboot/merged/bootia64.efi";
```

The `next-server` parameter may be the same system as the system providing the DHCP service or it may be another system. Using the parameters provided by the DHCP server, the Altix diskless client initializes the network and makes a TFTP request to the `next-server`.

The diskless client contacts the TFTP server, downloads the boot loader, and executes it. The `elilo.conf` configuration file, in the same directory as the boot loader, provides the name of a kernel and an initial ramdisk.

Three special parameters in the `elilo.conf` file establish the identity of the diskless client. Examples of these parameters are, as follows:

```
config_server=ftp://169.238.221.102/hosts
config_server=http://169.238.221.102/hosts

config_file=ITSECDEMO

hostname=tulip
```

The `config_server` parameter may define an anonymous FTP server or an HTTP server. The `config_file` parameter is equivalent to the `rc.sysinit` script in a disk boot and is executed once by `init`. The `hostname` parameter is used to identify the system name.

This setup simplifies administration by allowing common configuration settings such as desired software to be specified in `config_file` and host-specific files such as licensing and network settings to be specified by `hostname`.

The diskless client loads the operating system into memory along with the host-specific configuration files. Before control of the system is returned to `init`, DHCP client services are terminated.

When control of the boot process is returned to `init`, the system will boot in the same manner as a system with local disk. The network is re-initialized as specified in the host-specific parameters and services such as CXFS are started.

*Chapter 3*

# Kernel Tunable Parameters on SGI ProPack Servers

This section identifies and describes the settings for kernel tunable parameters appropriate for large SGI ProPack servers.

**Note:** This chapter does not apply to SGI Altix XE systems.

This information about Linux kernel tunable parameters is included in your Linux release and can be found in the following directory on an SGI ProPack 6 system:

`/usr/src/linux/Documentation/sysctl`

This section covers the following topics:

**Please note** that `sysctl` parameters are also described in the following files on your system:

```
/usr/src/linux/Documentation/filesystems/proc.txt
/usr/src/linux/Documentation/filesystems/xfs.txt
/usr/src/linux/Documentation/networking/ip-sysctl.txt
```

## CPU Scheduler `/proc/sys/sched` Directory

This section describes tunable parameters for CPU scheduling in the `/proc/sys/sched` file and only applies to SGI ProPack 3 for Linux systems. On SGI

ProPack 4 for Linux systems, tunable parameters can be found in `/proc/sys/kernel` directory.

The contents of the `/proc/sys/sched` directory is similar to the following:

```
[root@profit sched]# ls
busy_node_rebalance_ratio  idle_node_rebalance_ratio_max  min_timeslice
child_penalty              max_loadbal_rejects            sched_exec_threshold
idle_node_rebalance_ratio  max_timeslice                  sched_node_threshold
```

Do not change `min_timeslice` value to be less than 10, which is the current value for `cache_decay_ticks`, or else the scheduler's load-balancing will be adversely affected on some workloads.

**Note:** Be very careful in changing any of the values in this directory. You risk adversely affecting CPU scheduling performance.

## `/etc/sysconfig/dump` File

This file contains the configuration variables for the Linux Kernel Crash Dump (LKCD) facility that creates files in the `/var/log/dump` directory.

The following variables defined in this directory:

- `DUMP_ACTIVE`

  The `DUMP_ACTIVE` variable indicates whether the dump process is active or not. If this variable is 0, the dump kernel process is not activated.

- `DUMPDEV`

  The `DUMPDEV` variable represents the name of the dump device. It is typically the primary swap partition on the local system, although any disk device can be used.

  **Caution:** Be careful when defining this value to avoid unintended problems.

- `DUMPDIR`

  The `DUMPDIR` variable defines the location where crash dumps are saved. In that directory, a file called `bounds` is created that is the current index of the last crash dump saved. The `bounds` file is updated with an incremented index once a new crash dump or crash report is saved.

If there is an `lkcd` dump, LKCD could easily exceed multiple gigabytes in `/var`. This is why the default root filesystem is larger. For this reason, you may wish to make a separate `/var/dump` filesystem or change the configuration of `lkcd`. For more information on `lkcd`, see the `lkcd_config`(1) man page.

To save crash dumps to a different location, change the `DUMPDIR` value in `/etc/sysconfig/dump` file.

- `DUMP_SAVE`

  The `DUMP_SAVE` variable defines whether to save the memory image to disk or not. If the value is 1, the `vmcore` image is stored, and a crash report is created from the saved dump. If it is not set to 1, only a crash report is created and the dump is not saved. Use this option if you do not want your system's disk space consumed by large crash dump images.

- `DUMP_LEVEL`

  The `DUMP_LEVEL` variable has a number of possible values, as follows:

| | |
|---|---|
| `DUMP_NONE` (0) | Do nothing, just return if called. |
| `DUMP_HEADER` (1) | Dump the dump header and first 128K bytes. |
| `DUMP_KERN` (2) | Everything in `DUMP_HEADER` and kernel pages only |
| `DUMP_USED` (4) | Everything except the kernel free pages. |
| `DUMP_ALL` (8) | All memory is dumped. |

**Note:** You must use the numeric value, not the name of the variable.

- `DUMP_COMPRESS`

  The `DUMP_COMPRESS` variable indicates which compression mechanism the kernel should attempt to use for compression. The new method is not to use dump compression unless someone specifically asks for it. There are multiple types of compression available. For now, if you `modprobe dump_rle` , the `dump_rle.o` module is installed, that enables RLE compression of the dump pages. The RLE compression algorithm used in the kernel gives (on average) 40% compression of the memory image, which can vary depending on how much memory is used on the system. There are also other compression modules coming (such as `gzip`). The values for the `DUMP_COMPRESS` variable are currently, as follows:

| | |
|---|---|
| `DUMP_COMPRESS_NONE(0)` | Do not compress this dump. |
| `DUMP_COMPRESS_RLE(1)` | Use RLE compression. |
| `DUMP_COMPRESS_GZIP(2)` | Use GZIP compression. |

- `PANIC_TIMEOUT`

  The `PANIC_TIMEOUT` variable represents the timeout (in seconds) before reboot after a panic occurs. Typically, this is set to 0 on the system, which means the kernel sits and spins until someone resets the machine. This is not the preferred action if we want to recover the dump after the reboot.

The following is an example of a `/etc/sysconfig/dump` file on an SGI ProPack 3 system:

```
DUMP_ACTIVE=1
DUMPDEV=/dev/vmdump
DUMPDIR=/var/log/dump
DUMP_SAVE=1
DUMP_LEVEL=2
DUMP_FLAGS=0
DUMP_COMPRESS=0
PANIC_TIMEOUT=5
```

The following is an example of a `/etc/sysconfig/dump` file on an SGI ProPack 4 system:

```
DUMP_ACTIVE="1"
DUMPDEV="/dev/vmdump"
DUMPDIR="/var/log/dump"
DUMP_LEVEL="2"
DUMP_COMPRESS="2"
DUMP_FLAGS="0x80000004"
DUMP_SAVE="1"
PANIC_TIMEOUT="5"
BOUNDS_LIMIT=10
KEXEC_IMAGE=/boot/vmlinuz
KEXEC_CMDLINE="root console=tty0"
TARGET_HOST=""
TARGET_PORT=6688
SOURCE_PORT=6688
```

```
ETH_ADDRESS=ff:ff:ff:ff:ff:ff
DUMP_MAX_CONCURRENT=4
NETDUMP_VERBOSE=no
```

For descriptions of these SGI ProPack 4 dump parameters, see the `/etc/sysconfig/dump` file on your system.

## Resetting System Limits

To regulate these limits on a per-user basis (for applications that do not rely on `limit.h`), the `limits.conf` file can be modified. System limits that can be modified include maximum file size, maximum number of open files, maximum stack size, and so on. You can view this file is, as follows:

```
[user@machine user]# cat /etc/security/limits.conf
# /etc/security/limits.conf
#
#Each line describes a limit for a user in the form:
#
#             #
#Where:
# can be:
#        - an user name
#        - a group name, with @group syntax
#        - the wildcard *, for default entry
#
# can have the two values:
#        - "soft" for enforcing the soft limits
#        - "hard" for enforcing hard limits
#
# can be one of the following:
#        - core - limits the core file size (KB)
#        - data - max data size (KB)
#        - fsize - maximum filesize (KB)
#        - memlock - max locked-in-memory address space (KB)
#        - nofile - max number of open files
#        - rss - max resident set size (KB)
#        - stack - max stack size (KB)
#        - cpu - max CPU time (MIN)
#        - nproc - max number of processes
#        - as - address space limit
#        - maxlogins - max number of logins for this user
#        - priority - the priority to run user process with
#        - locks - max number of file locks the user can hold
#
#                 #

#*               soft    core            0
#*               hard    rss             10000
#@student        hard    nproc           20
#@faculty        soft    nproc           20
#@faculty        hard    nproc           50
#ftp             hard    nproc           0
#@student        -       maxlogins       4

# End of file
```

For instructions on how to change these limits, follow the procedure in "File Descriptor Limits for MPI Jobs" on page 58.

## File Descriptor Limits for MPI Jobs

Because of the large number of file descriptors that MPI jobs require, you might need to increase the system-wide limit on the number of open files on your Altix system. The default value for the file limit resource is 1024. You can change the file descriptor limit by editing the `/etc/security/limits.conf` file.

**Procedure 3-1** Increasing File Descriptor Limits for MPI Jobs

To change the default value for all users to 8196 file descriptors, perform the following:

1. Add the following line to `/etc/pam.d/login` file:

   ```
   session    required     /lib/security/pam_limits.so
   ```

2. Add the following lines to `/etc/security/limits.conf` file:

   ```
   *     soft    nofile      8196
   *     hard    nofile      8196
   ```

The default 1024 file descriptors allows for approximately 199 MPI processes per host. Increasing the file descriptor value to 8196, allows for more than 512 MPI processes per host.

## Understanding RSS, SIZE, and SHARE Values for MPI Jobs

You can use the `top`(1) and `ps`(1) command to view the RSS, SIZE, and SHARE values when running MPI jobs on an Altix system. In particular the RSS number for an MPI job can seem reasonable, while the SIZE and SHARE values are much higher.

The RSS value reflects the amount of memory that has been accessed and is currently resident in memory (faulted in). If swapping occurs, this value should go down. This tracks real memory usage of an application.

The SIZE value includes the faulted in memory plus adds up all the possible pages the application _could_ fault in (pages that have been previously allocated by the `mmap` function). This value includes the length of memory-mapped (`mmap`) regions, even if they are never touched.

The SHARE value includes the faulted in memory plus adds up all the possible pages the application _could_ fault in that are marked with the MAP_SHARED attribute. This value includes the length of MAP_SHARED memory-mapped (`mmap`) regions, even if they are never touched.

The reason that the SIZE and SHARE values are so high for MPI jobs, is that MPI cross-maps a significant amount of memory from each MPI process onto every other MPI process via XPMEM. This is done to allow single-copy transfers, fast MPI-2 one-sided transfers, and SHMEM capability. MPI programs currently cross-map the static region, stack, and a good portion of the private heap. All of these mappings use the MAP_SHARED attribute.

MPI programs cross-map these regions at `init` time, but none of the memory is actually touched or faulted in until the particular application accesses these mapped pages. The only "resource" the MPI programs are consuming is using up some virtual address space.

A high SIZE and/or SHARE value should not indicate any additional need for swapping, since these pages are not faulted in.

The RSS value should reasonably reflect the application memory usage, except it does not have any way to indicate shared resources.

## Memory (Swap) `sysctl` Parameters

**Note:** This section applies to SGI ProPack 3 systems only.

The following kernel parameters can be modified at runtime using the `sysctl`(8) command to affect kernel swap behavior, as follows:

- "`vm.min_swap_page_calls`" on page 60
- "`vm.oom_killer_nap_jiffies`" on page 60
- "`vm.swap_watch_interval`" on page 61
- "`vm.min_jiffies_out`" on page 61
- "`vm.print_get_swap_page`" on page 61
- "`vm.min_free_swap_pages`" on page 61

### `vm.min_swap_page_calls`

Minimum number of swap calls in a swap watch interval before a decision is made to determine if the system is out of swap space or not.

Defaults to 1000

### `vm.oom_killer_nap_jiffies`

How long the `oom_killer_thread` naps after it is first woken up and before it kills some process; also the time it naps after killing a process and before killing the next process. This is done to make sure that the system does not kill a process unless it has been out of swap space for a while. It also gives the system time to react (and perhaps stop swapping so much) after a processes has been killed and before it decides whether or not to kill another process. (Note that `oom_killer`() function checks to make sure it is still out of swap space every it time it wakes up from a "nap". If we are not out of swap space, it goes back to long term sleep waiting until `start_oom_killer`() is called again.

Defaults to 10*HZ

## vm.swap_watch_interval

How long between resets of the swap out statistics collected by the `get_swap_page`() function.

Defaults to 10*HZ

## vm.min_jiffies_out

Minimum time that the `try_to_free_pages_zone`() function has to have consistently failed before the `out_of_memory`() function will start up the `oom_killer`() function.

Defaults to 5*HZ

## vm.print_get_swap_page

If set to 1, `vm.print_get_swap_page` parameter will cause the system to log swap out statistics every `swap_watch_interval`, provided that swapping is active. This may be removed in a future release.

Defaults to 0

## vm.min_free_swap_pages

If this much swap space is free, the system will decide it is no longer out of swap (out of memory) on the next pass through the `oom_killer`() function. While settable using the `sysctl`(8) command, this is reset after each `swapon`() call to 1% of available swap pages. If some other default is desired, there has to be another `sysctl` call.

Defaults to 1% of total swap pages

## sched.child_penalty

The `sched. child_penalty` parameter controls how much or how little a forking child process inherits of one of the scheduling characteristics of the parent process, that is; its "interactivity" assessment. A forking child typically inherits only a fraction of the parent's "interactivity" assessment in order to avoid a potential denial-of-service attack on the system's CPU resource.

Each process is regularly assessed with a quantitative "interactivity" level and is assigned a value in a numerical continuum that ranges between the extremes of "totally computebound" and "executes for brief periods of time on rare occasions." If a process is deemed to be more and more "interactive," the scheduler gives it more and more of a transitory boost in priority when the process wakes up and wants the CPU resource. That is, a process that appears to be "interactive," such as a shell that responds to user keyboard inputs, is given more timely access to the CPU than a process which appears to be computebound.

This is a very heuristic assessment, and as such it is prone to approximation, confusion, and errors. One of the potential problems is the denial-of-service effect of having an interactive parent (which executes with that priority boost) being able to fork numerous children that would inherit the same high-priority "interactive" label as the parent and would themselves also preempt other lower-priority less-interactive processes.

The remedy for this potential problem is to not allow a forked child to inherit the exact same "interactive" quantitative value as the parent. Instead, a forked child is assessed a `child_penalty`, which is a percentage of the parent's "interactive" assessment. The default `child_penalty` is 50, or 50% of the parent's value.

## Load-balancing Algorithms `sysctl` Parameters

**Note:** This section applies to SGI ProPack 3 systems only.

The kernel parameters that can be modified at runtime using the `sysctl`(8) command to affect kernel load-balancing algorithms, are as follows:

```
sched.sched_loadbal_max = 1
sched.sched_exec_threshold = 1
sched.max_loadbal_rejects = 100
sched.sched_node_threshold = 125
sched.busy_node_rebalance_ratio = 10
sched.idle_node_rebalance_ratio_max = 50
sched.idle_node_rebalance_ratio = 10
```

These CPU Scheduler parameters affect the behavior of the load-balancing algorithms that attempt to equalize the runqueue lengths. This section describes these parameters.

The `sched_exec_threshold` parameter affects the aggressiveness of load-balancing at process `exec` time. When a parent process forks, the resulting child process is by default assigned to the same CPU as the parent. This is often the optimal behavior, for example, due to shared virtual address spaces. However, if the child process issues an `exec` call itself, it is reasonable to assume that the child process typically gains little advantage in executing on the CPU of the parent process. Instead, the child process should be migrated to a lesser loaded CPU. The `sched_exec_threshold` parameter is the runqueue length above which the child issuing an `exec` call will search for a lesser-loaded CPU. The default value of one (1) means that the child process will always search. This is the most aggressive behavior. Raising the value makes the search less aggressive, which trades off a slight decrease in `exec` overhead for a less load-balanced set of CPUs.

The remaining parameters described in this section control behavior of the load-balancing algorithms that periodically execute on each CPU. During each 1024Hz timer tick, the CPU Scheduler decides whether or not to spend effort to compare its runqueue length against the runqueue lengths of the other CPUs. This is to determine if one or more processes should be pull-migrated from the runqueue of another CPU onto the runqueue of this CPU. This runqueue examination can be an expensive operation in terms of system resources, especially with large numbers of CPUs. Therefore, a CPU must trade off the cost of executing it too frequently versus the inefficiency of executing it too infrequently and having the CPU remain under utilized.

At each timer tick, the first decision made is whether or not to execute the basic load-balancing algorithm at all. The more frequently a CPU performs this load-balancing scan, the more evenly balanced are the runqueues of each and every CPU relative to the other runqueues. However, there are two significant downsides to performing overly frequent load-balancing. The first is that frequent load-balancing is invasive and causes contention on the busiest CPUs’ runqueues’s spinlocks. High contention levels will affect context-switching performance and may in fact produce so much contention (especially at high CPU counts) that the system "livelocks" on the busiest CPU’s runqueue lock. The second downside to frequent load-balancing is that processes may be migrated away from local physical memory and thus may suffer substantially longer memory access latencies. The trade-off is giving a process access to more CPU cycles at the cost of having those CPU cycles be less efficient because of longer latencies. In some cases, a process is much better off remaining on a more busy CPU because the process remains close to the physical memory it can most efficiently access.

An idle CPU is more tempted to perform this relatively costly load-balancing scan than a non-idle (“busy") CPU, since the system would generally benefit (ignoring issues of NUMA memory locality) from migrating a not-currently-executing process from another CPU into this idle CPU. Every 1024Hz tick (roughly every millisecond) an idle CPU performs the load-balance scan within the node, that is , examining only the other CPU in the two-CPU Altix node. Every `idle_node_rebalance_ratio` ticks (current a default value of 10, or roughly every ten milliseconds) an idle CPU performs a load-balance scan of all nodes in the system. Therefore, increasing the `idle_node_rebalance_ratio` value makes the idle CPU full system rebalancing less frequent. Decreasing the value makes it more frequent.

If an idle CPU finds no process to pull-migrate from a busier CPU, then the delay (the "infrequency") of these idle scans is dynamically increased by one, up to a maximum value of `idle_node_rebalance_ratio_max`. Therefore, with a default maximum of 50, an idle CPU does an all-CPU load-balance scan after 10 milliseconds. If no pull-migrate occurs, the next scan occurs 11 milliseconds later, then 12 milliseconds later, and so on, up to a maximum of 50 milliseconds. When one of the scans finds a process to pull-migrate, the delay is reset to the basic `idle_node_rebalance_ratio` value, which defaults to 10. The higher the value of the `idle_node_rebalance_ratio_max` parameter, the longer it will likely be between all-CPU load-balancing scan. This means that when a "busier" CPU does emerge, the slower the other idle CPUs will be to recognize it and to off-load that "busier" CPU. Systems with larger CPU counts may well benefit from higher `idle_node_rebalance_ratio` and `idle_node_rebalance_ratio_max` values. Each individual idle CPU may be slower to see a suddenly overloaded CPU, but because there are likely to be many idle CPUs, then some idle CPU will likely recognize the overloaded CPU and perform load-balancing. Never specify an `idle_node_rebalance_ratio_max` value less than `idle_node_rebalance_ratio`.

A non-idle "busy" CPU performs the same within-the-local-node load-balancing scan at 1/100th the frequency of an idle CPU, or about every 100 milliseconds, and it performs the all-CPUs scan at a multiplier of `busy_node_rebalance_ratio` of that. With a `busy_node_rebalance_ratio` value of 10, this means an all-CPUs scan about once per second. These scan rates are definitely less frequent for a non-idle CPU than for an idle CPU. Once again, the CPU Scheduler is reluctant to migrate processes between nodes and potentially away from low-latency, local physical memory.

Once a CPU decides to perform the load-balancing scan, there are more tuning parameters that can affect behavior. The first is the `sched_node_threshold` parameter, which is the threshold ratio of "imbalance" (relative to 100) that determines whether to begin pulling processes. The default value of 125 defines an imbalance of

25%. An alternative value of 150 would define an imbalance of 50%. You should never use a value less than 100 and should avoid using values less than the default 125.

Once "busier" CPUs are identified that have processes that can be pull-migrated to this less-loaded CPU, the `sched_loadbal_max` is the number of processes that may be pull-migrated. The default value of one means that each load-balance scan will pull-migrate at most one process. Raising this value, increases the number of processes that may be migrated. The higher the value, the greater the likelihood that the load-balancing migrations may become overly aggressive. However, this is a heuristic algorithm, and some workloads might benefit from a value greater than the default of one.

Consider a system with 128 CPUs, when the runqueue workload of one of the CPUs suddenly skyrockets to 128 processes, while the other 127 CPUs are idle. With a `sched_loadbal_max` value of one, as each idle CPU executes the load-balancing scan at whatever frequency that scan occurs (as noted earlier, depending upon both fixed constants and by tuning parameters), each will pick off one process at a time from this overloaded runqueue until each CPU has a runqueue of one.

If, however, the `sched_loadbal_max` parameter is a high value, the first idle CPU to execute the load-balance algorithm would pull half of the processes of the busy CPUs; the 128–CPU system will have two CPUs with equal runqueue lengths of 64-64. The next CPU to execute the load-balance algorithm would pull 32 processes from one of these busy CPUs to equalize that one bilateral imbalance, producing runqueue lengths of 32-64-32, and then would pull 16 processes from the second CPU to equalize that bilateral imbalance, thus producing final runqueue lengths of 32-48-48. Note that most of the migrated processes will not have actually executed, but will merely have moved from waiting on one runqueue to waiting in a second runqueue. A third idle CPU does its load-balancing and produces another readjustment of 32-24-36-36. Once again, processes are migrated around in groups, from CPU to CPU, likely before actually executing. Thus, a higher `sched_loadbal_max` value may result in more active migrations and certainly in a different pattern of change of the runqueue lengths, but it is unclear whether higher values are more, or less, effective than lower value.

Finally, the `max_loadbal_rejects` parameter puts a limit the number of pull-migration candidate processes a load-balancing CPU will examine; and reject as being unsuitable, before releasing runqueue spinlocks and giving up the search. A candidate is rejected if it has not been in the runqueue of the busy CPU for a long enough time and thus is deemed as being "cache hot"; or is rejected because the candidate process's `cpus_allowed` mask (as set by CpuMemSets or by the `sys_sched_setaffinity()` system call) does not allow that process to execute on the load-balancing CPU. The higher the `max_loadbal_rejects` value, the more effort the searching CPU makes to find a process to pull-migrate. Once again, this is another trade-off; the cost of acquiring and holding valuable runqueue spinlocks for potentially longer and longer periods of time, versus the benefit of successfully load-balancing the CPU runqueues.

## Virtual Memory `hugetlb` Parameter

The `/usr/src/linux/Documentation/vm/hugetlbpage.txt` file on your system provides a brief summary of `hugetlb` page support in the Linux kernel. The Intel Itanium architecture supports multiple page sizes 4K, 8K, 64K, 256K, 1M, 4M, 16M, 256M and so on. A translation lookaside buffer (TLB) is a cache of virtual-to-physical translations. Typically, this is a very scarce resource on a processor. Operating systems try to make best use of limited number of TLB resources. This optimization is more critical now as larger physical memories (several GBs) are more available.

You can use the huge page support in Linux kernel by either using the `mmap`(2) system call or standard shared memory system calls `shmget`(2) and `shmat`(2) (see the `shmop`(2) man page).

For information on using `hugetlb`, see `/usr/src/linux/Documentation/vm/hugetlbpage.txt`.

You can also boot your system with the kernel `hugepages=` *X* parameter set, where *X* is a number of pages. Setting the `hugepages` parameter allows the kernel to allocate as many of the huge pages as possible early on. After the system is booted, it gets harder to find contiguous memory. You can also set huge pages early in the `init` scripts in `/proc/sys/vm/nr_hugepages` as described in the `/usr/src/linux/Documentation/vm/hugetlbpage.txt` file.

Some considerations on using the `hugetlb` parameter on your Altix system are, as follows:

- Starting with the SGI ProPack 3 for Linux Service Pack 1 release, allocation of `hugetlb` pages is NUMA aware. That is, pages are allocated either on the node, or as close as possible to the node where the `mmap` or `shmget`(2) system call is executed. The `hugetlb` pages are allocated and mapped into the requesting address space at `mmap`() or `shmat`() time; the `hugetlb` pages are not demand faulted into the address space as are regular pages. The `hugetlb` pages are allocated and zeroed by the thread that calls `mmap`() or `shmget`().

- Starting with the SGI Propack 3, Service Pack 1 release, `hugetlb` pages are allocated on first touch, much like how regular pages are allocated into an address space by a call to the `mmap` routine. The `hugetlb` page allocation is NUMA aware; this means that the huge page is allocated on (or as close as possible to) the node where the thread that first touched the page is executing. In previous SGI ProPack releases, tasks that allocated more `hugetlb` pages than were available at the time of the `mmap`() or `shmget`() call, caused the `mmap`() or `shmget`() call to fail. To keep this behavior the same in the ProPack 3, Service Pack 1 release, the system implements a "reservation" algorithm to ensure that once the `mmap`() or `shmget`() call has completed successfully, the system guarantees that there are sufficient pages at first-touch time to satisfy the (`delayge`) storage allocation request that occurs at first-touch time. Page reservation is not subject to NUMA allocation; that is, you cannot reserve pages on a particular node, nor does the node on which the `mmap`() or `shmget`() request executes have any influence on where the `hugetlb` pages are finally placed. Placement of `hugetlb` pages is determined by the node where the first touch occurs and the locations of the available `hugetlb` pages available at that time. The number of `hugetlb` pages currently reserved can be found in the `/proc/meminfo` file, as shown in the following example:

```
[root@revenue3 proc]# cat meminfo
        total:    used:    free:  shared: buffers:  cached:
Mem:  32301940736 4025974784 28275965952        0    98304 2333163520
Swap: 10737319936  9928704 10727391232
MemTotal:     31544864 kB
MemFree:      27613248 kB
MemShared:           0 kB
Buffers:            96 kB
Cached:        2271280 kB
SwapCached:       7200 kB
Active:         459360 kB
Inactive:      1845840 kB
HighTotal:           0 kB
HighFree:            0 kB
LowTotal:     31544864 kB
LowFree:      27613248 kB
SwapTotal:    10485664 kB
SwapFree:     10475968 kB
HugePages_Total:     0
HugePages_Free:      0
Hugepagesize:    262144 kB
```

- You can change the `hugetlb` page size can at system boot time by specifying the `hugepagesz=NNNN` parameter to the ELILO boot command prompt (this parameter can also be specified using the `append` command in the`/boot/efi/efi/sgi/elilo.conf` file). The `NNNN` parameter is the size of the `hugetlb` pages in bytes and this parameter must have a value that is a supported `hugetlb` page size for the hardware platform where the kernel is booted.

# Index