HCA 400 User Manual

Release AAA-AAC
February 2005
P/N 399Z00007

Business Headquarters

Voltaire, Inc.
6 Fortune Drive, Suite 301
Billerica, MA 01821
Tel: 978-439-5400
Fax: 978-439-5401
www.voltaire.com

Israel Office

Voltaire, Ltd.
9 Hamenofim St.
Bldg. A Herzeliya
46725, Israel
Tel: +972 (9) 971-7666
Fax: +972 (9) 971-7660

Installation Voltaire HCA 400 and Voltaire HCA 400 host stack software package, which includes all computer software and associated media and printed materials, and the "online" or electronic documentation, indicates your acceptance of the terms of the End-User Software License Agreement, and creates a legal and binding agreement between you (either an individual or an entity) and Voltaire Inc. For details, please refer to the End-User Software License Agreement.

THIS DOCUMENT IS PROVIDED "AS IS" WITH NO WARRANTIES WHATSOEVER, INCLUDING ANY WARRANTY OF MERCHANTABILITY, NONINFRINGEMENT, FITNESS FOR ANY PARTICULAR PURPOSE, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION OR EXAMPLES.

Voltaire disclaims all liability, including liability for infringement of any proprietary rights, relating to use of information in this specification.

No license, expressed or implied, by written or otherwise, to any intellectual property rights is granted herein.

This document as well as the software described in it is furnished under license and may only be used or copied in accordance with the terms of the license. The information in this manual is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Voltaire. Voltaire Ltd. assumes no responsibility or liability for any errors or inaccuracies that may appear in this document or any software that may be provided in association with this document. Except as permitted by such license, no part of this document may be reproduced, stored in a retrieval system, or transmitted in any form or by any means without the express written consent of Voltaire Ltd.

Voltaire is a trademark or registered trademark of Voltaire Ltd. or its subsidiaries in the United States and other countries.

Other names and brands may be claimed as the property of others and have been designated with an asterisk (*) throughout this document.

Voltaire Part Number: [399Z00007]

About this Guide

This User Guide describes how to install and configure the Voltaire HCA 400 on Linux host systems. The HCA 400 consists of an interface card (hardware) and an InfiniBand host stack (software).

Audience

This guide is intended primarily for system administrators who are responsible for maintaining and configuring the HCA 400 family.

It is assumed the reader is familiar with InfiniBand technology and terminology.

Document Conventions

NOTE

- Text set off in this manner presents clarifying information, specific instructions, commentary, sidelights, or interesting points of information.

IMPORTANT

- Text set off in this manner indicates presents important information regarding a specific feature.

CAUTION

- Text set off in this manner indicates that failure to follow directions could result in damage to equipment or loss of information.

Document Organization

This User Guide describes how to install and configure the Voltaire HCA 400 on Linux host systems.

- *Chapter 1 – Introduction*: Provides a product overview and software components review.
- *Chapter 2 – Installing the Voltaire InfiniBand Protocol Host Stack:* Describes the HCA 400 installation process, provides an installation Quick Start, and provides detailed explanations about the installation and pre-configuration instructions.
- *Chapter 3 – Configuring the Host Stack*: Describes the host stack configuration process.
- *Chapter 4 – Protocol Configuration and usage for SDP, MPI and IPoIB:* Describes specific configuration information for the SDP, MPI, and IPoIB protocols.
- *Chapter 5 – Uninstalling the Host Stack:* Describes the procedure for uninstalling the host stack from a host.

Contents

Figures

Chapter 1 - Introduction

In This Chapter

This chapter provides introductory and general information on the HCA 400. The following information is included:

Product Overview

The HCA 400 enables you to efficiently deploy InfiniBand clusters for High Performance Computing (HPC) and Business (BIZ) applications.

Organizations around the world have embraced clusters as the most cost-effective way to provide superior performance for demanding applications. Now, high performance InfiniBand technology from Voltaire is enabling these organizations to significantly improve the performance of their computational and visualization clusters.

The HCA 400 interface card and the InfiniBand host stack are part of a family of end-to-end InfiniBand solutions from Voltaire, including the ISR 9024 and ISR 9288 InfiniBand switch routers, Voltaire IP Router (IPR and RM-4GE) and Voltaire Fibre-Channel Router (FCR and RM-4FC). Using InfiniBand solutions from Voltaire, enterprises and high performance computing clusters achieve higher performance with a lower total cost of ownership.

Host Stack Description

The host stack is a set of device drivers, related binaries, libraries and documentation, which are installed on Linux host systems.

HCA 400 Interface Card Description

The Voltaire HCA 400 interface card is installed in the host system; and provides the physical InfiniBand interface.

The HCA 400 is a dual-port, 4x InfiniBand host channel adapter card that enables PCI-X and PCI-Express based servers to access the full performance of a high speed InfiniBand fabric. It includes complete software support for a range of demanding business and HPC applications. The HCA 400 features Voltaire's TCP/IP emulation software, which enables the InfiniBand fabric to appear to the server operating system and application programs as a standard Ethernet network.

The HCA 400 features device drivers for Linux 32 and 64-bit server architectures and a comprehensive suite of upper layer protocols that are ideal for high performance computing clusters and high-performance enterprise data center applications including IBM DB2 and Oracle 10G RAC.

The HCA 400 is designed to maximize the performance of a 4x (10 Gbps) InfiniBand fabric. The hardware features remote direct memory access (RDMA) for high throughput with extremely low CPU utilization and latency. These capabilities combined with Voltaire's InfiniBand stack allow applications to run significantly faster achieving bandwidths of over 800 Megabytes/sec, minimal latency and reduced CPU utilization.

Review of Software Components

The following table lists and describes the main software components relevant to the HCA 400 and describes the functions they perform. It is recommended that you review this table prior to installation.

Voltaire Host Stack Component	Description
HCA drivers	HCA drivers provide the necessary driver support to control the HCA 400 card.
MPI (Message Passing Interface)	MPICH is an open-source, portable implementation of the Message-Passing Interface standard. It contains a complete implementation of version 1.1of the MPI standard and also provides significant parts of MPI-2, particularly in the area of parallel I/O. It is based on MPICH-1.2.5 with updates. It supports local in-memory communication, and provides support for C and Fortran applications.

Voltaire Host Stack Component	Description
SDP (Socket Direct Protocol)	SDP provides applications with an optional replacement to TCP, utilizing InfiniBand for optimal performance and reduced CPU utilization.
IPoIB	IPoIB supports legacy networks on the local InfiniBand subnet. The standard IPoIB driver model tunnels standard Ethernet frames across the subnet using InfiniBand transport services.
IB-ARP	Provides dynamic network address translation.
CM	InfiniBand Communication Manager.
GSI	General Services Interface.

Chapter 2 - Installing the Voltaire InfiniBand Protocol Host Stack

In This Chapter

This chapter describes how to install the Voltaire HCA 400 on Linux host systems. It includes both hardware installation of the HCA 400 interface card and software installation of the host stack. The following information is included:

Overview

The HCA 400 host stack requires a working fabric manager to configure the fabric. As default, the HCA 400 is configured to work with the Voltaire Fabric Manager (VFM). When working with other vendor subnet managers or with MiniSM (Mellanox Subnet Manager), make sure that the HCA 400 is configured to work with the fabric prior to the installation.

NOTE

- The installation process in the following sections must be completed on each Linux host that is going to use Voltaire InfiniBand HPC or BIZ Stack software components.
- Throughout these installation instructions, the Courier font is used to indicate messages that the install script prints to the screen. User responses are shown in `Courier`.

Quick Start

This chapter provides an overview to the Host Stack installation procedure. Detailed instructions for relevant installation sequences are described further on in the manual.

Host Stack Installation Steps

Step 1 Read the Hardware And Software Prerequisites section on page 2-3. Verify that the host complies with the minimum requirements.

Step 2 Install the operating system in compliance with the requirements stated in Hardware And Software Prerequisites section on page 2-3.

Step 3 Install the HCA 400 interface card as described on page 2-4.

Step 4 Use the standard RPM utility to install the host stack RPM package version that matches the operating system and CPU architecture.

Step 5 Configure the host stack as described in on page 3-1.

Step 6 Configure the host stack for one of the following communication protocols (refer to the Protocol Configuration and usage for SDP, MPI and IPoIB Protocols chapter on page 4-1):

- IPoIB
- SDP
- MPI

Installation Prerequisites for Hardware and Software

This chapter defines the minimum hardware and software requirements for Voltaire HCA 400 host system configurations.

Required Hardware

- Architecture – the HCA 400 supports both 32bits and 64bits architectures:
  - Xeon – i686 architecture
  - AMD – Opteron – x86_64 architecture
  - Intel - EM64T – x86_64 architecture
  - Intel – Itanium2 – IA64 architecture
- SMP – The HCA 400 supports both symmetric and NUMA multiprocessor configurations. For optimal performance, using two CPUs or more is recommended.
- Bus interface – The HCA 400 comes with either PCI-X or PCI-Express interfaces. The PCI-X HCA complies with PCI-X2.2 and PCI 2.1 (32/64 bit, 33/66 MHz). The HCA uses 3.3 volts and cannot operate in a 5-volt slot. For optimal performance, use a PCI-X 133 MHz slot.
  The PCI-Express HCA has an 8x PCI-Express interface.
- Memory – The host stack software requires a minimum of 1024 MB of RAM.
  In large clusters environment the memory requirement might be higher. For further information please advise Appendix B.

Required Software

The HCA 400 supports the Enterprise additions of Red Hat and Novell SUSE.Since parts of the host protocol stack and drivers are implemented as Linux kernel modules, the kernel version used by the host server has a significant effect on the Host Stack.

For more detailed information on Linux distribution and kernel support please consult with the HCA 400 release notes.

Installing the HCA 400 Interface Card

Refer to the safety document supplied with the interface card prior to installation.

To install the HCA 400 Interface Card:

Step 1 Ensure that the host is powered down and disconnect the host from its power source.

Step 2 Select a free PCI - slot in the host system and install the HCA 400 interface card.

Step 3 Connect the InfiniBand cable to either of the HCA ports and to the switch.

Step 4 Reconnect the host to its power source and power up the system while observing the card LEDs. If properly installed and operational, one or more of the card LEDs will light after the system powers up. If no card LEDs appear, troubleshoot and fix the problem before proceeding (the LEDs will be lit only after the software installation).

CAUTION

- The PCI-X channel adapter uses 3.3 volts. Do not attempt to insert it into any 5-volt PCI slot. For optimal performance, it is imperative that the channel adapter be placed in a 133 MHz slot

Host Stack Installation

These instructions apply to host stack software components installed on a host system.

Prior to starting the installation, obtain the host stack installation package from the Voltaire FTP or web site.

CAUTION

- At this point you should have the HCA 400 interface card installed on all systems with the cards connected to the fabric.
- Prior to installing the host stack software, make sure all InfiniBand drivers from prior versions are removed (uninstalled) before trying to install the new drivers. Uninstall information can be found in Chapter 5 of this manual.

User Authorization

An authorized user should perform installation. Please login to the system as user `root`.

Checking the Linux Kernel Version

When installing a binary package, compatibility is required between the software package and the kernel installed on the server.

To check the kernel version, use the `uname -r` command and make sure that you are using the correct version from the Voltaire website that holds all the binaries for different kernels.

On the Voltaire web site you will find all binary RPM installations for the supported Linux distributions and kernels.

Software Installation

Software installation of the host stack is performed using the Red Hat Package Manager (RPM) utility.

The installation procedure varies, depending on the type of Linux system you are installing on. Select the RPM package suitable for the installed kernel from the installation package.

The version convention is:

```
ibhost-<hpc/biz>-<version number>-<kernel version>-
<architecture>.rpm
```

The default installation installs the software components in `/usr/voltaire`.

HPC Installation

```
rpm -ivh ibhost-hpc-version_number-1{the correct
kernel}.arch.rpm
```

BIZ Installation

```
rpm -ivh ibhost-BIZ-version_number-1{the correct
kernel}.arch.rpm
```

IMPORTANT

- The HCA 400 installation updates some of the environment settings, to enable these setting please logout / login to the system.

Chapter 3 - Configuring the Host Stack

In This Chapter

This chapter describes the configuration process for the host stack. The following information is included:

Pre-Configuration Instructions

After installing the host stack, the following steps must be done in preparation to the configuration process:

- IP Address. The HCA 400 uses the Linux network scripts to configure its IPoIB network interface. When first installing the stack please configure an IP address using the IB-setup configuration utility or via the Linux network scripts (ifconfig, ifup...). The IPoIB interface can also be configured to use a dynamic IP address retrieved from a DHCP server as described in section 3-4.
- Logical Link. Make sure that you have a running Subnet Manager in your fabric. The Host Stack will not be operational without a logical link. When a logical link is present, the two LEDS on the HCA 400 interface card should be lighted (one orange, one green) for the port connected to the fabric.
- MiniSM (Mellanox Subnet Manager). If you are using MiniSM to configure the fabric, make sure that the MiniSM scans the fabric after each host stop/start.
- HCA Firmware. In certain cases, upgrading the stack software requires a firmware upgrade. Refer to the firmware update parameter on page 3-7. Refer to the release notes to determine whether a firmware update is needed.

Setting Configuration Parameters

Host stack configuration parameters are set using the `ib-setup` command.

The ib-setup is located in `/usr/voltaire/scripts` which is added to the system path for convenient use.

NOTE

- This section focuses on the interactive mode of ib-setup. In some cases it is preferable to configure the system in CLI mode (usually when working with external cluster management tools) .The ib-setup utility supports a CLI-like configuration for some of the configuration parameters. For more information, refer to ib-setup CLI mode on page 3-8.

The Opening screen shows the status of your system, InfiniBand configured parameters and active parameters.

```
====== Voltaire InfiniBand Stack Setup  ======
Version:          ibhost-v2.0.0_2: Sun Dec  7 05:16:04 PST 2003 on opt1.
System:           kernel version: 2.4.19-SMP, memory 997MB.
Hostname:         opt1.
IB configuration: AutoStart: off, SM: off.
HCA Ports status: 1 - PORT_ACTIVE, 2 - PORT_DOWN

           IPoIB (config)        IPoIB (active)
ip-addr:   192.168.0.1           192.168.0.1
netmask:                         255.255.255.0
broadcast:                       192.168.0.255
mtu:       1500                  1500

1) Auto-start       4) Firmware-update  7) Exit
2) IPoIB            5) Start
3) Fabric           6) Stop
=> 1
Would you like to enable IB stack, on system startup?
(Y/n):
```

Figure 3-1. ib-setup Opening Screen

The Opening screen features a menu-based configuration utility with the following options:

Parameter	Options	Default Value	Description
1) Auto-Start	YES/NO	YES	Determines whether the InfiniBand stack will be started automatically when the computer boots.

Parameter	Options	Default Value	Description
2) IPoIB	IP-Addr: 192.168.0.10 Net-mask: 255.255.255.0 Broadcast: 192.168.0.255 MTU: 1500	When you configure the IPoIB for DHCP, select (1) for ON, or (0) for OFF (see 2a below for explanation on DHCP). Assign an IP address, netmask, broadcast address and MTU for the IPoIB-UD interface. Each host on the InfiniBand fabric should have a distinct IP address to access the other hosts on the fabric using IPoIB-UD tunneling. The same addresses are used when using SDP. In the event that you are connected to an IP subnet using the Voltaire IP gateway, this netmask should be the same as the netmask on the IP side and the above IP address should be in the same subnet. Note: The MTU must be identical on all the interfaces.		

```
====== Voltaire InfiniBand Stack Setup  ======
Version:          ibhost-v2.0.0_2: Sun Dec  7 05:16:04 PST 2003 on opt1.
System:           kernel version: 2.4.19-SMP, memory 997MB.
Hostname:         opt1.
IB configuration: AutoStart: off, SM: off.
HCA Ports status: 1 - PORT_ACTIVE, 2 - PORT_DOWN

           IPoIB (config)        IPoIB (active)
ip-addr:   192.168.0.1           192.168.0.1
netmask:                         255.255.255.0
broadcast:                       192.168.0.255
mtu:       1500                  1500

1) Auto-start       4) Firmware-update  7) Exit
2) IPoIB            5) Start
3) Fabric           6) Stop
=> 2
Please enter DHCP mode for ipoib-ud0 [1=on 0=off] [0]:
Please enter IP address for ipoib-ud0 [192.168.0.1]:
Please enter network mask for ipoib-ud0 []:
Please enter broadcast address for ipoib-ud0 []:
Please enter network mtu for ipoib-ud0 [1500]:
```

Figure 3-2. IPoIB

Parameter	Description

2a)DHCP	DHCP support over InfiniBand assumes that you have a DHCP server in the InfiniBand fabric, or that you can reach it via the Voltaire IP gateway. The DHCP server should have a listener socket on the IPoIB interface. The DHCP server should be configured to answer the request in broadcast mode. If the DHCP server does not answer in broadcast packets, you can change the network script: `/etc/sysconfig/network-scripts/ifup`
	
	To request a broadcast answer, by adding –b to the dhcpd.
	
	To configure the stack to work with DHCP, choose the IPoIB parameter in ib-setup; the following line appears:
	
	Enter DHCP mode for IPoIB-ud0 [1=on 0=off] [0]:
	
	Enter 1 to configure the stack for DHCP or 0 for static configuration. The stack adds the following files:
	
	In Red Hat:
	
	`/etc/sysconfig/network-scripts/ifcfg-IPoIB`
	
	`/etc/sysconfig/network-scripts/ifcfg-IPoIB-ud0`
	
	In SuSE:
	
	`/etc/sysconfig/network/ifcfg-IPoIB-ud0`

Parameter	Options	Default Value	Description
3) Fabric	VFM; MINISM; ADVANCED	VFM	There are three options for fabric management configuration: VFM – When working with Voltaire Fabric manager (VFM) this option should be set. The VFM supports advanced fabric management services such as: multicast groups, address translation services (ATS) and path queries. MINISM – Choose this if your SM does not require joining to a multicast group, or when you are not using a Voltaire managed switch. ADVANCED – For expert users, lets you configure which fabric services will be supported.

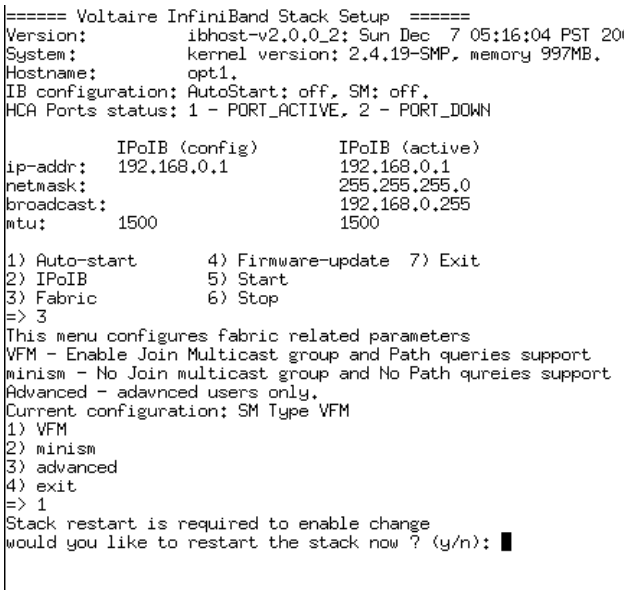

Figure 3-3. Fabric

Parameter	Options	Default Value	Description
4) Firmware Update	N/A	When choosing this option, the default file that will be used is the latest firmware version, which was modified for compatibility and performance reasons. Caution: updating the HCA with an inappropriate firmware version can cause operational problems that are time-consuming to diagnose and repair. Do not accept the proposed default firmware version until you have verified that it is appropriate for your installation. For more information about the firmware versions supported, please refer to the release notes.	

```
====== Voltaire InfiniBand Stack Setup  ======
Version:          ibhost-v2.0.0_2: Sun Dec  7 05:16:04 PST 2003 on opt1.
System:           kernel version: 2.4.19-SMP, memory 997MB.
Hostname:         opt1.
IB configuration: AutoStart: off, SM: off.
HCA Ports status: 1 - PORT_ACTIVE, 2 - PORT_DOWN

           IPoIB (config)        IPoIB (active)
ip-addr:   192.168.0.1           192.168.0.1
netmask:                         255.255.255.0
broadcast:                       192.168.0.255
mtu:       1500                  1500

1) Auto-start       4) Firmware-update  7) Exit
2) IPoIB            5) Start
3) Fabric           6) Stop
=> 4
You are about to upload new FW to the HCA
FW image is: /usr/mst/fw-23108-rel-3.0.0/fw-23108-a1-rel.host.sanmina.img
This operation would change the FW on the HCA and requires a system shutdown
 Do you wish to continue [y/n]:y
If you wish to use a different image please enter the image name, using full path:
```

Figure 3-4. Firmware-update

5) MPI	Depends on the specific package. For example on AMD Opteron system: mpi.gcc.mpd mpi.gcc.rsh mpi.pathcc.mpd mpi.pathcc.rsh mpi.pgcc.mpd mpi.pgcc.rsh	mpi.gcc.rsh	Each directory is designed to support different compiler and job initiation method. The directory name is based on the following convention: mpi.compiler-name.job-initation-method. Each option, once selected, link the relevant mpi directory to /usr/voltaire/mpi Please note that the selection should be consistent on all the nodes that are part of the same job.

Parameter	Options	Default Value	Description

```
====== Voltaire HCA400 InfiniBand Stack Setup  ======
Version:          ibhost-v3.0.0_9: Wed Dec 29 15:21:59 IST 2004 on builder.
System:           kernel version: 2.4.21-20.ELsmp, memory 3588MB.
Hostname:         as3-1.voltaire.com.
IB configuration: AutoStart: on, SM: VFM.
HCA400 Firmware version: 0x300000000.
HCA400 PCI Typte: PCI-X
HCA Ports status: 1 - PORT_ACTIVE, 2 - PORT_DOWN

           IPoIB (config)        IPoIB (active)
ip-addr:   192.168.0.11          192.168.0.11
netmask:   255.255.255.0         255.255.255.0
broadcast: 192.168.0.255         192.168.0.255
mtu:       1500                  1500

1) Auto-start       4) Firmware-update  7) MPI
2) IPoIB            5) Start            8) Exit
3) Fabric           6) Stop
=> 7

There can be several available MPI compiled versions,
to use different C & FORTRAN compilers, like
       GNU             compilers     (gcc, g77),
       Intel           compilers     (icc, ifc),
       Pathscale       compilers     (pathcc, pathf90),
       Portland Group compilers      (pgcc, pgf77, pgf90)

The versions may be built to run by remote shell (rsh)
      or MPI deamon (mpd)


Currently used version:  /usr/voltaire/mpi.pgcc.rsh

Which one would you like to use?

1) mpi.gcc.mpd     4) mpi.pathcc.rsh  7) Exit
2) mpi.gcc.rsh     5) mpi.pgcc.mpd
3) mpi.pathcc.mpd  6) mpi.pgcc.rsh
=>
```

Figure 5. MPI

6) Start	N/A		Loads the InfiniBand Stack and removes the IPoIB interface.
7) Stop	N/A		Stops the InfiniBand Stack and removes the IPoIB interface.
8) Exit	N/A		Exits the ib-setup program.

CLI Mode

When working in a clustered environment, users generally use an external cluster management tool such as PDSH (Parallel Distributed Shell) or CEXEC. These utilities can run a specific command on each node of the cluster. To enable the use of such tools to configure the host stack, the ib-setup utility supports a subset of configuration options in CLI mode.

The configuration options supported in this mode are:

- `ip`—used to configure static IP, for example:
  `ib-setup --config --ip 192.168.1.10 --netmask 255.255.0.0 --broadcast 192.168.255.255 --mtu 1500`

- `network`—used to configure an InfiniBand IP address according to the eth0, using the base network , for example:
  `ib-setup --config --network 192.168.1.0`
- `dhcp`—used to configure the stack to use dhcp to retrieve its IP address from the dhcp server, for example: ib-setup --config --dhcp
- `burn_hca`—Upload a new firmware image to the HCA; note that after the burning procedure is completed, you must shutdown the system.

`help /?`—Provides help on command usage.

Manual Start / Stop

You can use the `ib-setup` command to start the host stack, or you can use the following command line commands.

To enable/start the InfiniBand stack support: `ibhost.init start`.

To disable/stop the InfiniBand stack support: `ibhost.init stop`.

4

Chapter 4 - Protocol Configuration and usage for SDP, MPI and IPoIB Protocols

In This Chapter

This chapter describes the specific configuration process for each of the protocols supported by the host stack. Some of the supported protocols are at the user-level while some are at the kernel-level. Each protocol is intended for a specific application/use as described in this chapter. The following information is included:

Overview

Voltaire's HCA 400 enables your applications, such as HTTP and FTP, to communicate via the InfiniBand fabric in the same way as an Ethernet network would. This section discusses using the following three protocols:

- IPoIB
- SDP
- MPI

Some of these protocols are at the user-level while some are at the kernel-level. Each protocol is intended for a specific application/use.

IP over InfiniBand (IPoIB) Protocol

The IP/IB (IP over InfiniBand) software module enables standard TCP/IP connections over the InfiniBand Fabric. Existing applications currently using TCP/IP can switch to the IP/IB interface with no modifications.

Login to the system and check your IPoIB IP address (using the `ifconfig` command), make sure all IP addresses are unique. If any of the IP-addresses are the same use the ib-setup utility to modify the IP.

Use the IPoIB-UD interface as any other NIC. Use `ping` or any other TCP/IP application to check the connectivity between the InfiniBand hosts.

Sockets Direct Protocol (SDP)

The Sockets Direct Protocol (SDP) defines a standard wire protocol over InfiniBand fabric to support stream sockets (SOCK_STREAM) networking over InfiniBand. SDP utilizes various InfiniBand features (such as remote DMA (RDMA), memory windows, solicited events etc.) for high-performance zero-copy data transfers.

SDP is implemented as an address family under BSD (Berkeley Software Distribution) sockets. This design allows the user to keep using the BSD sockets semantics. To start working with SDP sockets, the user needs to call the socket() system call using Voltaire's AF_IBT (declared to be 26) address family.

There are two easy ways to start working with SDP sockets, either by changing your application to refer to the SDP directly, or by using the Voltaire socket switch (see details below).

Voltaire's SDP implementation supports the AIO (Asynchronous IO) extension. For more information regarding AIO, refer to:
http://lse.sourceforge.net/io/aio.html

Utilizing SDP using Current Applications

To use SDP with your current applications without changing the application code, you need to define the environment variable `LD_PRELOAD` and to configure the SDP-socket switch configuration file, as shown in the example below.

While using the bash command interpreter, you can use the following, in order to run an application with SDP:

```
LD_PRELOAD=/lib/sock-redirect.so <prog-name>
```

where <prog-name> is the program you would like to use with SDP.

NOTE

- When working with 64bit architectures, we provide two copies of sock-redirect.so, one for 32-bit applications and the second for 64-bit applications. The 64bit sock-redirect.so resides at: `/lib64/sock-redirect.so`

Example: Testing FTP over SDP

Server Side:

```
LD_PRELOAD=/lib/sock-redirect.so /etc/init.d/xinetd
```

Restart

-or-

```
LD_PRELOAD=/lib64/sock-redirect.so /etc/init.d/xinetd
```

Restart

This option applies to the users using the 64-bit (Opteron) option.

Client Side:

```
LD_PRELOAD=/lib/sock-redirect.so ftp <ip-address>
```

-or-

```
LD_PRELOAD=/lib64/sock-redirect.so ftp <ip-address>
on 64-bit systems
```

The IP address is the IPoIB-UD interface of the FTP server.

Verify that the FTP server is listening over SDP, using the command:

```
cat/proc/voltaire/sdp-sockets/ib-socket-netstat
```

You can also define the LD_PRELOAD environment variable as follows:

```
export LD_PRELOAD=/lib/sock-redirect.so
```

and it will apply to all commands invoked from the given bash session.

You can also add this line to the ~/.bashrc file for future sessions.

Socket Redirect – Configuration File

The SDP-sockets-switch configuration file is a text-based file, which is very sensitive to changes. It is located at /usr/voltaire/config/sock-redirect-config.txt.

It controls the behavior of the following system calls: socket(), bind() and connect().

In order to provide offloading, you should configure the LD_PRELOAD variable as described previously on page 4-3.

The format of the file is inflexible and all changes should be done carefully.

The configuration file consists of four parts; each separated with a description line. The description lines are critical to the file structure and should not be removed.

- ENABLE or DISABLE—Controls the SDP socket offloading (default: ENABLE); when this line states disable, it means that the server will not use any offloading options. All connections will be TCP only. If this flag is set to disable, the rest of the configuration file is ignored. This is applicable for both the client and the server sides.
- YES or NO—Replaces all TCP Sockets to SDP (default: YES.) The valid options are YES/NO. When set to Yes all TCP connections will be replaced with SDP connections, transparently to the applications. Using this option means that there will be no more TCP connections from the server. When set to Yes the rest of the configuration file is meaningless.

  When set to NO the logic of replacing TCP sockets with SDP sockets will be affected by the parameters, which are configured in the remaining parts of the file. This is applicable for both client and server sides.
- # Target IPs—Connects via SDP (default: list of IP addresses

  When the `replace all` flag is set to NO, Active TCP connections that connect to the IP's in this list will be replaced with SDP connections. That means each connect system call using an address from this list will be flagged and the connection will be replaced with an SDP connection.

  This is applicable for the client side as it controls the connect() system call.
- # Listen on SDP—Local ports (default: single port and range example). When the replace all flag is set to NO, TCP listening sockets (which are listening on all ports) that appear on the list will be replaced with an SDP listening socket. You can also enter a port range by typing the - character to state the port range. This option is applicable for the server side, as it controls the bind() system call.
- # Listen on SDP—Local IP addresses. (Default: example IP of the local IPoIB.)

When the replace all flag is set to NO, TCP listening sockets that are listening on any IP address from the list will be replaced with SDP sockets.

NOTE

- Configuring the sock-redirect listeners is invalid if the application is listening on address any. This option is applicable to the server side as it controls the bind() system call.

Example

The following is an example of an SDP configuration file, implementing some of the operations described previously.

```
# Voltaire Socket Offloading Switch Configuration file
#

# Enable / Disable SDP socket offloading
ENABLE

# Replace all TCP Sockets to SDP
YES

# Target IPs - Connect via SDP
192.168.0.1
192.168.0.2
192.168.0.3
192.168.0.4

# Listen on SDP - Local ports.
50
5000-7000


# Listen on SDP - Local IPs
192.168.0.22
```

Using SDP with New Code

New programs (or old ones for which you have source code) may use SDP sockets directly using the address family AF_IBT, which for now is set to 26 using socket the (2) system call. In such cases there is no need for the LD_PRELOAD environment variable. All other system calls (e.g. read, write, select, poll, send, recv, setsockopt) on the file descriptor will behave the same as with the TCP sockets.

MPI (Message Passing Interface)

MPI is a library specification for message passing, proposed as a standard by a broadly based committee of vendors, implementers, and users. It is intended for distributed computing. Voltaire's implementation of the MPI protocol allows seamless passing of messages between MPI and InfiniBand.

Voltaire's MPI SDK (Software Development Kit) provides the user with all required libraries, include files, build and execution utilities needed for working with MPI.

This section describes the content of the SDK, compiler support provided, and explanations and examples of the MPI usage.

Compiler Support

The Voltaire's MPI SDK supports several C/Fortran compilers.

For each CPU architecture, the MPI SDK supports and compiles with a different compiler as described in the list below:

GNU – supported on all systems.

Intel C++/Fortran compiler – For IA32, IA64 and EM64T.

PGI – For AMD Opteron.

PathScale – For AMD Opteron.

Directory structure and selection

The Voltaire MPI SDK consists of several directories- each compatible with a combination of certain compilers and job initiation method.

Before starting any job, please make sure that you are using the compiler and job start method of your choice.

Use the ib-setup configuration utility to choose the appropriate directory, as described in section 3-5.

NOTE

- Please make sure that the same directory is chosen on all the nodes.

MPI SDK Content

All parts of the SDK reside under /usr/voltaire/mpi.

- `/usr/voltaire/mpi/bin` – all executable binaries reside in this directory. The executable binaries are either MPI applications, such as mpi_bandwith or mpi_latency, or utilities used to run and monitor the MPI applications such as mpirun. Note that the installation adds /usr/voltaire/mpi/bin to the path variable of your environment.
- `/usr/voltaire/mpi/lib` – all of the MPI libraries resides in this directory.
- `/usr/voltaire/mpi/include` – all MPI include files that might be needed for compilation reside in this directory.
- `/usr/voltaire/mpi/examples` – in this directory you will find example code for simple MPI applications.

IMPORTANT

- In this section, we refer to the /usr/voltaire/mpi as the active MPI implementation.

Compiling your Application

Voltaire's MPI SDK contains several scripts that will help you to compile your MPI application using the precompiled libraries. These include scripts for C programs as well as Fortran programs. All these scripts are located under the /usr/voltaire/mpi/bin directory.

There are two ways to compile MPI program with Voltaire's MPI SDK:

- Using the HCA 400 provided scripts:
    - mpicc -> for C programs
    - mpicxx -> for C++ programs
    - mpif77 -> for Fortran programs
    - mpif90 -> For Fortran 90 programs

  These scripts include all of the necessary libraries and compiler flags.

  You can check these scripts' command line options by running the command with the –show option (e.g. mpicc –show).

- Using your own compiling environment and scripts:

  To keep using your own makefiles and compilation scripts you should make sure your application is linked with the appropriate libraries.
  To do that add these flags to the link stage:
  -L/usr/voltaire/mpi/lib –lmpich –L/usr/mellanox/lib –l -lmtl_common -lvapi -lmpga -lpthread

NOTE

- All MPI headers can be found at:
  /usr/voltaire/mpi/include
- All of the compilation scripts support the following linkage flags:
  noshlib – to be linked with static MPI libraries.
  shlib - to be linked with shared MPI libraries.
- When no linkage flag supplied, the default mode is linkage with shared libraries.

Running your MPI Application

The SDK enables running the MPI application in one of two methods:

- Using either RSH (Remote SHell) or SSH (Secure SHell) by running mpirun_rsh or mpirun_ssh respectively.
- Using MPD, a multi-purpose daemon that runs on every node.

Using mpirun_rsh and mpirun_ssh applications

System configuration

To start working with either of these execution utilities, you need to enable the RSH/SSH on all of your servers. Make sure that the hosts can find one another: verify that the names of all the hosts in the cluster appear in the etc/hosts files on all the other hosts in the cluster; or make sure that the NIS/DNS is set properly.

Make sure that no password is needed to open the RSH/SSH connection between the servers.

IMPORTANT

IMPORTANT

- RSH is limited in connection number in large clusters (usually over 128 nodes); for these configurations, use of the SSH utilities is recommended.

When running the MPI application using mpirun_rsh / mpirun_ssh, the user can load a configuration parameter file to the host stack, changing its default parameters. The value of the parameters has an impact on memory consumption and stack scalability. Refer to Appendix A for further information.

mpirun_rsh and mpirun_ssh command line options

mpirun_ssh / mpirun_rsh are simple applications designed to start MPI jobs on remote hosts. These applications use the following parameters:

Parameter name:	Explanation:
-np #	Number of processors
hostname1 hostname2 | -hostfile hostfile	Names of the hosts separated by space or a hostfile containing the hostname of the servers this program should run on.
Application name (optional are command line parameters for your application)	The MPI application you are planning to run

mpirun_ssh –np 2 IB1 IB2 ./mpi_bandwidth 1000 256000

This example runs the mpi_bandwidth executable using mpirun_ssh initiator on two processors (-np 2) between nodes IB1 and IB2. The mpi_bandwidth executable requires two command line parameters 1000 and 256000 (which are the number of iterations and the message size).

A different way to run the same application will be to create a hostfile, which contains the node names:

----------hostfile----------

IB1

IB2

----------hostfile----------

And then run the command

mpirun_ssh –np 2 –hostfile ./hostfile ./mpi_bandwidth 1000 256000

NOTE

The hostfile should contain the exact number of hosts you specified with the –np parameter. If the hostfile contains more hosts than specified with –np , the job will be started using the first np hosts while the others will be ignored

•

Host file usage

Each line in the hostfile file represents one process to be initiated by the mpirun_xxx application. If you wish to run more than one process on each node you should duplicate the hostname lines to indicate the number of processes you wish to initiate.

Please note that the order of process initiation might have a performance effect.

Example

----------hostfile----------

IB1

IB2

IB1

IB2

----------hostfile----------

mpirun_ssh –np 4 –hostfile ./hostfile ./xhpl

The above example runs the Linpack benchmark using 4 processors on 2 nodes.

Using MPD:

System configuration

Using the MPD as a job start mechanism requires that the daemons run on all nodes and have proper access right to other nodes.

To do so please follow these instructions:

1. Create two files called .mpd.conf and .mpdpasswd Theses files could be a simple one line file that holds your mpd password.
   For example:
   password=123456
   Please note that your MPD password should not be your Linux user password, but an arbitery password to be used to control MPD access only.
2. Place these files on all nodes in the user home directory.
3. Set protection on the file so that you have read and write privileges:
   chmod 600 .mpd.conf
   chmod 600 .mpdpasswd
4. Make sure that the active MPI directory on all nodes is mpi.compiler.mpd. For detailed instruction please refer to section 3-5 of this manual.
5. Before starting the MPD you need to create a host file that contains the names of all the nodes which participate in the job.
   To start the MPD daemons use the following command line on one of the nodes:
   mvapich.mpd.sh start hostfile /usr/voltaire/mpi/bin
6. If you choose to terminate the MPD daemons you can do so by running the following command line:
   mvapich.mpd.sh stop hostfile /usr/voltaire/mpi/bin

   It is recommended to verify that all daemons had exited properly. To do so please run ./mpdcleanup on all nodes.

Starting jobs using MPD

To start an MPI job using MPD, you need to use a simple application called mpirun_mpd.

The command line options of mpirun_mpd are similar to mpirun_ssh.

The following command line is a simple example of running mpi_bandwith with 2 processors on 2 nodes using the MPD:

mpirun_mpd -np 2 ./mpi_bandwidth 1000 100000.

Using MSTI MPI/PRO Libraries

As mentioned in the previous section, Voltaire's MPI libraries are installed at /usr/voltaire/mpi. The host stack installation adds this directory to the path. To use MSTI MPI/PRO libraries you must remove the Voltaire directory from the Path. Contact MSTI for further information.

NOTE

- On some Linux distributions there might be other MPI packages installed for Ethernet connectivity. To verify which MPI packages have been installed, use the mpirun command. To check the specific package name installed, use for example, the rpm –qf /usr/bin/mpirun command.

In order to eliminate any confusion, perform one the following:

- Remove the package that might interfere with the Voltaire MPI. For example: `rpm –e package-name`.
- Include the /usr/voltaire/mpi/bin in you PATH variable before other MPI directories. For example:
- for bash users: `export PATH=/usr/voltaire/mpi/bin:$PATH`
- for tcsh users: `set path=(/usr/voltaire/mpi/bin $path)`

5

Chapter 5 - Uninstalling the Host Stack

In This Chapter

This chapter describes the procedure for uninstalling the host stack from host systems. It includes the following information:

Uninstalling the Host Stack

This section describes how to uninstall the following InfiniBand components:

- Host Stack software using the Linux installer (Refer to page 2-5).

Uninstalling the Host Stack Software

CAUTION

- Current configuration will be removed.

Do the following:

```
ibhost.init stop
```

To query the installed package run the following command:

```
rpm –qa | grep ibhost
```

Use the result package name at the next command:

```
rpm –e {ibhost package installed}
```

```
rm –rf /usr/mellanox  /usr/voltaire  /usr/mst
```

Appendix A - Example Configuration

This Appendix provides a brief explanation how to connect and configure a two-node InfiniBand fabric.

Figure 5-1 shows a simple example of Machine A and Machine B connected via the InfiniBand fabric.

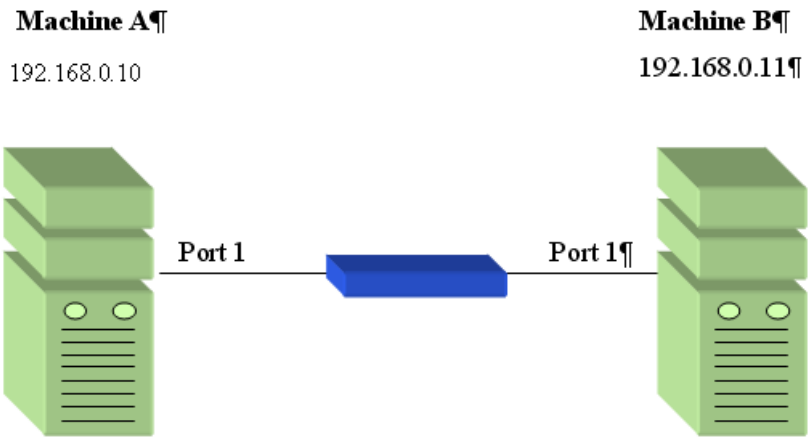

Figure 5-1. Set Up

The host stack comes with default configuration for the IPoIB interface with the 192.168.0.10 IP address.

For this example setup, reconfigure Server B IPoIB interface IP address, refer to page 3-3 of this manual , for a description on how to configure the IPoIB interface.

There is no need to run the Subnet-Manager on the host as it is part of the Voltaire InfiniBand switch.

NOTES

- Connect HCA Port #1, which is located farthest from the PCI connector, to the switch.
- Machine A and B will get their LID from the Subnet Manager, which is running on the InfiniBand switch.
- Each host has ipoib0 NIC interface.

Appendix B - Voltaire MPI Memory Consumption and Configuration

The Voltaire MPI implementation takes advantage of Infiniband RDMA technology.
Using the IB driver requires pre-allocated buffers to be allocated, mapped to the HCA, and pinned down in the RAM.
These buffers are required for both regular DMA and Remote Data Memory Access (RDMA) operations. These buffers are allocated during the job initialization process.

In general - when running an MPI job over InfiniBand the memory consumption of the system can be divided to three parts:

- System memory – Memory used by the OS.
- MPI pre-allocated memory
- Application memory that has been pinned down for RDMA purposes.

The following section describes the different components that affect the MPI pre-allocated memory size. We will suggest a simple formula to help you calculate the expected memory consumption based on to the cluster size and the number of processes you are going to use. We will also suggest some parameters that users can change to reduce the memory consumption.

Memory Consumption Calculation

The Voltaire MPI uses the Infiniband Reliable Connection (RC) mechanism for inter process communication. A reliable connection is established between every pair or processes residing on different nodes. Each RC connection uses a different set of buffers for DMA and RDMA operations.
Therefore the memory consumption of each process is a product of the number of its peer processes multiplied by resources required for each connection.

Memory Consumption per Process:

((VIADEV_PREPOST_DEPTH+10) * sizeof(VBUF) * (NP-1)) +

(VIADEV_NUM_RDMA_BUFFER * 2 * sizeof(VBUF) * (NP-1)) +

(sizeof(QP) * (NP-1))) + sizeof (CQ).

Where:

- NP – is the number of processes that participate in the Job.

- VIADEV_PREPOST_DEPTH – represents the number of buffers posted to the receive queue. Default value is 64.
- VIADEV_NUM_RDMA_BUFFER – represent the number of pre-allocated RDMA buffers, these buffers are used in the fast data path. Default value is 32.
- Size of VBUF – The size of each pre-allocated buffer. Default value is 12KB.
- Size of QP – The size of the IB queue pair. Default value around 176KB.
- Size of CQ – The size of the IB completion queue. Default value around 2052 KB.

Memory consumption on a node is a product of the number of process on the node times the memory consumption per host.

When working with more than one process per node the MPI will use shared memory for intra-node communication up to eager size threshold. The size of the shared memory is being affected by the number of process running on the same node and can be tuned with parameter file variable: SMPI_LENGTH_QUEUE

IMPORTANT

- The above description is valid for the MPI initialization allocation, during run time the MPI would pin down the application memory. This is required to support RDMA operation. The amount of memory consumed at run time is entirely depended on the application.

Memory Configuration

It is possible to limit the memory consumption of the MPI by changing the parameters described in the calculation above.

It is important to understand that changing these parameters could have an effect on the performance. The influence on performance is highly related to the application behavior and can not be described in a general way.

Most of the parameters can be changed in runtime using the mpi parameter file. The parameter file is a simple text file that holds a list of parameters and values in the following format:

PARAMETER = Value

To use the parameter file, you need to supply it in the mpirun_rsh command line.

Example:

mpirun_ssh -paramfile ./paramfile –np 4 –hostfile ./hostfile ./xhpl

All of the parameters described above, besides the size of VBUF can be controlled via the parameter file. However we do not recommend changing the QP and CQ size. Changing these parameters requires deep understanding of IB mechanisms and should be done by an advanced user only.

The VBUF size can only be changed at compilation time.

Example 1 – Running 100 Processes on 100 Nodes using 1 CPU.

Memory per process:

((VIADEV_PREPOST_DEPTH+10) * sizeof(VBUF) * (NP-1)) +

(VIADEV_NUM_RDMA_BUFFER * 2 * sizeof(VBUF) * (NP-1)) +

(sizeof(QP) * (NP-1))) + sizeof (CQ) =

((64 + 10) * 12 * 99) + (32 * 2 * 12 * 99) + (176 * 99) + 2052 =

87912 + 76032 + 17424 + 2052 = 183418 KB = ~185 MB.

Memory per node:

Since we are running 1 process on the node, the Memory consumption of the node is similar to the memory consumption of the process ~185 MB.

Example 2 – Running 100 Processes on 50 Nodes using 2 CPUs.

Memory per process:

((VIADEV_PREPOST_DEPTH+10) * sizeof(VBUF) * (NP-1)) +

(VIADEV_NUM_RDMA_BUFFER * 2 * sizeof(VBUF) * (NP-1)) +

(sizeof(QP) * (NP-1))) + sizeof (CQ) =

((64 + 10) * 12 * 99) + (32 * 2 * 12 * 99) + (176 * 99) + 2052 =

87912 + 76032 + 17424 + 2052 = 183418 KB = ~185 MB.

Memory per node:

Memory consumption of the node consists of the number of processes that run on the node multiplied by the memory consumption of each process.

Memory consumption on the node is: 185 * 2 = ~270.

Glossary

ACPI	Advanced Configuration and Power Control
AIO	Asynchronous IO
ARP	Address Resolution Protocol
BSD	Berkeley Software Distribution
CLI	Command Line Interface
CPU	Central Processing Unit
FTP	File Transfer Protocol
FW	Firmware
GE	Gigabit Ethernet
GUI	Graphical User Interface
GUID	Global Unique ID
HCA	Host Control Adaptor
HPC	High Performance Computer
IB	InfiniBand
IBTA	InfiniBand Trade Association
IP	Internet Protocol
ISR	InfiniBand Switch Router
LID	Local ID
LIDSTAT	Properties retrieved from a specific LID

LAN	Local Area Network
LMC	LID Mask Control
PDSH	Parallel Distribute Shell
RPM	Red Hat Package Manager
RSH	Remote Shell
SDP	Socket Direct Protocol
SDK	Software Development Kit
SSH	Secure Shell
SM	Subnet Manager
SWG	Software Working Group
TCP	Transmission Control Protocol
UDP	User Datagram Protocol

Index