Host-Side Drivers
User Guide for Linux

September 2005 (First Edition)
Part Number 406720-001

Linux is a U.S. registered trademark of Linus Torvalds.

Host-Side Drivers User Guide for Linux

September 2005 (First Edition)
Part Number 406720-001

Contents

Preface v

Intended audience v
Typographical conventions v
Contact information vi

1: About Host-Side Drivers 1

Introduction 1
Architecture 2
Supported protocols 2
IPoIB 2
SRP 2
SDP 3
Supported APIs 3
uDAPL 3
MPI 3
HCA utilities 3

2: Installing Host-Side Drivers 5

Installing host-side drivers from the CD-ROM 5
Installing host-side drivers from an ISO over NFS 6
Upgrading host-side drivers on your Linux host 7

3: IP over IB Protocol 9

Introduction....9
Configuring IPoIB....9
Sample startup script....14
Verifying IPoIB....14
Configuring high-availability ports....14
Merging physical ports....14
Unmerging physical ports....16

4: SCSI RDMA Protocol....17

Introduction....17
Configuring SRP....17
Configuring Initiators....18
Configuring ITLs....18
Verifying SRP....21
Verifying with the CLI....21
Verifying with Element Manager....21

5: Sockets Direct Protocol....23

Configuring IPoIB interfaces....23
Specifying connection overrides....23
Converting sockets-based applications....23
Converting sockets-based applications to use SDP....24
Running a performance test on SDP....25
Sample configuration - SDP for IBM DB2 v8.1....26
Performance acceleration....26
Overview....26
Sample topology....26
Prerequisites....27
Configuring the application server....27
Configuring the database server....28
(Optional) Associating the IPC connection to IB....29
Restarting clients and servers....29
(Optional) Setting Up Non-IB connections....29
Configuring other listeners....29
Confining SDP processes....29
Troubleshooting the configuration....29
Sample configuration - OracleNet™ Over SDP for Oracle 9i....30
Performance acceleration....30
Overview....30
Sample topology....30
Configuring the application server....31
Configuring the database server....32
Setting up non-IB connections....32
Troubleshooting the configuration....32

6: Configuring uDAPL Drivers 33

About the uDAPL configuration 33
Building uDAPL applications 33
Running a uDAPL performance test 34
- Running a uDAPL throughput test 34
- Running a uDAPL latency test 35

7: Configuring MPI 37

Configuring MPI 37
Configuring SSH 37
Editing the PATH variable 39
Performing the bandwidth test 40
Performing the latency test 40

8: HCA Utilities 43

Introduction 43
hca_self_test utility 44
tvflash utility 45
- Upgrading firmware 45
- Viewing boot details 45
- Configuring the boot type 46
- Viewing card type and firmware version 46

vstat utility 47
- Viewing the GUID of the HCA 47
- Viewing basic HCA attributes with the vstat utility 49

tsinstall utility 50
ifconfig utility 50
- Bringing down an IB port 50
- Bringing up an IB port 50

9: Sample Performance Test 53

Introduction 53
Hardware and applications 53
Network topology 54
Host and switch setup 54
Running Netperf 54
IPoIB performance versus Ethernet using Netperf 56
- Performing a throughput test 56
- Performing a latency test 57

SDP performance versus IPoIB using Netperf 58

Performing a throughput test 58
Performing a latency test 59

10: Sample Test Plan 61

Overview 61
Requirements 61
Prerequisites 61
Hardware and applications 61
Network topology 62
Host and switch setup 62
IPoIB setup 63
Configuring IPoIB 63
IPoIB performance versus Ethernet using netperf 64
Performing a throughput test 64
Performing a latency test 65
SDP performance versus IPoIB using netperf 65
Configuring SDP 65
Performing a throughput test 66
Performing a latency test 66

11: Configuring Oracle RACTM with InfiniBand 67

Purpose 67
Supporting documentation 67
Configuring the cluster hardware 68
System requirements 68
Hardware requirements 68
Configuring the InfiniBand fabric 69
Installing and configuring the HCAs 69
Understanding the use of the HCA drivers 69
Configuring Fibre Channel storage 69
Configuring software 70
Installing packages 70
Installing patches 70
Installing the shared disk subsystem 70
Configuring cluster interconnect and public network 71
Installing the Oracle CRS daemon and 10g RAC 71
Configuring Oracle Net over SDP 71
Troubleshooting SRP 72
Checking the InfiniBand network interfaces 72
Checking the HCA port status 72
Verifying CRS and 10g RAC 73

Preface

This document is a guide to host drivers for servers running Linux. This document explains how to configure InfiniBand (IB) drivers and protocols, as well as how to run HCA utilities.

Intended audience

We provide this document for administrators who install, configure, and manage equipment. This document assumes that administrators have prior Ethernet, Fibre Channel (FC), and network administration experience.

Typographical conventions

The following typographic conventions are used in this manual to provide visual clues as to the purpose or application of specific text.

- **Bold** text indicates a command or keyword, or text that appears in your display.
- *Italics* indicate variables that you replace with an actual value.
- Square brackets ([,]) indicate an optional argument that you choose to include or exclude when you enter a command.
- Pipe character (|) indicates an “or” choice. For example, “**a | b**” indicates “a or b.”
- Ellipses (**…**) indicate truncated text. You will see these in long examples depicting terminal output that is too long to be shown in its entirety.

NOTE: Indicates an important point or aspect that you need to consider before continuing.

Contact information

Table 2-1: Customer Contact Information

For the name of your nearest authorized HP reseller	In the United States, refer to the HP US service locator webpage. In other locations, refer to the HP website.
For HP technical support	In the United States and Canada, call 1-800-HP-INVENT (1-800-474-6836). This service is available 24 hours a day, 7 days a week. For continuous quality improvement, calls may be recorded or monitored. Outside the United States and Canada, refer to www.hp.com.

1

About Host-Side Drivers

The following sections appear in this chapter:

- "Introduction" on page 1
- "Supported protocols" on page 2
- "Supported APIs" on page 3
- "HCA utilities" on page 3

Introduction

Host-side drivers integrate host channel adapters (HCAs) into commodity servers. After you physically install the HCA in the server, install the drivers to run InfiniBand (IB)-capable protocols. HCAs support the following protocols in the Linux environment:

- Internet Protocol over InfiniBand (IPoIB)
- SCSI RDMA Protocol (SRP)
- Sockets Direct Protocol (SDP)

HCAs support the following application program interfaces (APIs) in the Linux environment:

- User Direct Access Programming Library (uDAPL)
- Message Passing Interface (MPI)

Host-side drivers also provide utilities to help you configure and verify your HCA. These utilities provide upgrade and diagnostic features.

Architecture

Figure 1-1 displays the software architecture of the protocols and APIs that HCAs support. The figure displays upper-level protocols (ULPs) and APIs in relation to other IB software elements.

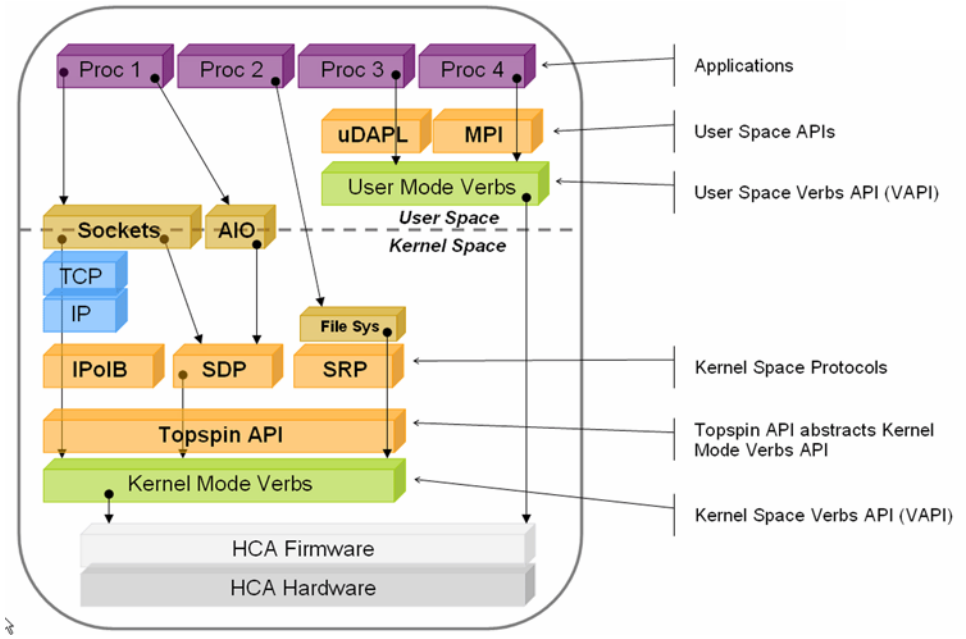

Figure 1-1: Protocol and API Architecture

Supported protocols

The term “protocol” refers to software in the networking layer in kernel space.

IPoIB

The IPoIB protocol passes IP traffic over the IB network. Both SDP and uDAPL rely on IPoIB to resolve IP addresses.

To configure IPoIB, you assign an IP address and subnet mask to each IB port. IPoIB automatically adds IB interface names to the IP network configuration. To configure IPoIB, refer to “Configuring IPoIB” on page 9.

NOTE: IPoIB supports port-to-port failover on the HCA ports.

SRP

SRP runs small computer system interface (SCSI) commands across remote direct memory access (RDMA)-capable networks so IB hosts can communicate with Fibre Channel (FC) storage devices. This protocol assigns devices and mounts file systems so that hosts can access the data on those file systems. The SRP driver is installed as part of the driver package and loads automatically when the host reboots. This protocol requires a Server Switch with a FC gateway to connect the host to FC storage.

SRP requires no host-side configuration. In conjunction with a Server Switch, SRP disguises IB-attached hosts as FC-attached hosts. The Topology Transparency feature lets FC storage communicate seamlessly with IB-attached hosts (known as SRP hosts).

NOTE: SRP supports port-to-port failover on the HCA daughter card.

SDP

SDP enables sockets-based applications to take advantage of the enhanced performance features provided by IB. It provides a high-performance, zero-copy data transfer protocol for stream-socket networking over an IB fabric. SDP achieves lower latency and higher bandwidth than IPoIB running sockets-based applications. You can configure the driver to automatically translate TCP to SDP based on source IP, destination, or application name. For configuration instructions, refer to "Sockets Direct Protocol" on page 23.

Supported APIs

The term "application program interface or API" refers to software in the networking layer in user space.

uDAPL

The uDAPL defines a set of APIs that leverages IB's RDMA capabilities. The uDAPL API is installed transparently with the driver library. To take advantage of this API, your application must explicitly support uDAPL.

This library requires no manual configuration. However, if your application supports uDAPL, it may require additional configuration changes. Refer to your application documentation for more information.

MPI

MPI is a library specification for message-passing. MPI is available on the HP Support website at http://support.hp.com. The MPI implementation lets you use either external HCA port and supports Opteron 64-bit operation. For configuration instructions, refer to "Configuring MPI" on page 37.

HCA utilities

HCA utilities provide basic tools to view HCA attributes and run preliminary troubleshooting tasks. Table 1-1 briefly describes the utilities that the HCA provides.

Table 1-1: HCA Utilities

Utility	Description
tvflash	The tvflash utility displays boot details. To use this utility, refer to "tvflash utility" on page 45.
vstat	The vstat utility displays HCA attributes. To use this utility, refer to "vstat utility" on page 47.
hca_self_test	The HCA self test utility displays the status and configuration information of the HCA. To use this utility, refer to "hca_self_test utility" on page 44.
tsinstall	The tsinstall utility installs host-side drivers on Linux servers. To use this utility, refer to "tsinstall utility" on page 50.

2

Installing Host-Side Drivers

The following sections appear in this chapter:

- "Installing host-side drivers from the CD-ROM" on page 5
- "Installing host-side drivers from an ISO over NFS" on page 6
- "Upgrading host-side drivers on your Linux host" on page 7

Installing host-side drivers from the CD-ROM

To install drivers from the CD-ROM:

1. Place the CD-ROM in your host.
2. Log in to your host.

   Example

```
host login: user-id
Password: password
You have new mail in /var/mail/user-id.
Last login: Tue Mar 29 13:29:25 from laptop.domain.com
host:~ #
```

3. Mount the CD-ROM drive.

   Example

```
host:~ # mount /media/cdrom
```

4. Navigate to the top-level directory of your CD-ROM.

   Example

```
host:~ # cd /media/cdrom
host:/media/cdrom #
```

5. Enter the **tsinstall** command.

   Example

```
host:/media/cdrom # ./tsinstall

The following kernels are installed, but do not have drivers available:
  2.6.5-7.97smp.x86_64

The following drivers are currently installed, but do not have new replacement
drivers:
  topspin-ib-mod-sles9-2.6.5-7.97smp-3.0.0-179.x86_64

Would you like to remove these drivers? (y/n) n
host:/media/cdrom #
```

Installing host-side drivers from an ISO over NFS

To install drivers from an ISO on your network:

1. Download an ISO from http://support.hp.com and copy it to your network.
2. Log in to your host.

   Example

```
host login: user-id
Password: password
You have new mail in /var/mail/user-id.
Last login: Tue Mar 29 13:29:25 from laptop.domain.com
host:~ #
```

3. Navigate to the ISO on your file system.

   Example

```
host:~ # cd /path/image/
host:/path/image #
```

4. Mount the ISO.

   Example

```
host:/path/image # mount -o loop topspin-boib-3.0.0-179.iso /mnt
```

5. Navigate to the ISO.

   Example

```
host:/path/image # cd /mnt
host:/mnt # ls
.          BUILDNUM       firmware  modules.cgz  redhat    suse
..         BUILDREVISION  linux     modules.dep  rhdd      tsinstall
.topspin   docs           modinfo   pcitable     rhdd-6.1
host:/mnt #
```

6. Enter the **tsinstall** command.

Example

```
host:/mnt # ./tsinstall

The following kernels are installed, but do not have drivers available:
  2.6.5-7.97smp.x86_64

The following drivers are currently installed, but do not have new replacement
drivers:
  topspin-ib-mod-sles9-2.6.5-7.97smp-3.0.0-179.x86_64

Would you like to remove these drivers? (y/n) n
host:/mnt #
```

Upgrading host-side drivers on your Linux host

The process to upgrade host-side drivers identically matches the process to install drivers. To upgrade drivers:

1. Place your CD in your host.
2. Log in to your host.

Example

```
host login: user-id
Password: password
You have new mail in /var/mail/user-id.
Last login: Tue Mar 29 13:29:25 from laptop.domain.com
host:~ #
```

3. Place the installation CD in your CD-ROM drive.
4. Mount the CD-ROM drive.

Example

```
host:~ # mount /media/cdrom
```

5. Navigate to the top-level directory of your CD-ROM.

Example

```
host:~ # cd /media/cdrom
host:/media/cdrom #
```

6. Enter the **tsinstall** command.

Example

```
host:/media/cdrom # ./tsinstall

The following kernels are installed, but do not have drivers available:
  2.6.5-7.97smp.x86_64

The following drivers are currently installed, but do not have new replacement
drivers:
  topspin-ib-mod-sles9-2.6.5-7.97smp-3.0.0-179.x86_64

Would you like to remove these drivers? (y/n) n
host:/media/cdrom #
```

3

IP over IB Protocol

The following sections appear in this chapter:

- "Introduction" on page 9
- "Configuring IPoIB" on page 9
- "Verifying IPoIB" on page 14

Introduction

When you configure IPoIB to run IP traffic over the IB network, the IPoIB driver starts automatically. Refer to your Linux distribution documentation for additional information about configuring IP addresses.

NOTE: To enable these IPoIB settings across reboots, you must explicitly add these settings to the networking interface startup script. For a sample script, refer to "Sample startup script" on page 14.

Configuring IPoIB requires similar steps to configuring IP on an Ethernet network. When you configure IPoIB, you assign an IP address and subnet mask to each HCA port. The first HCA port on the first HCA in the host receives the ib0 identifier. The number of the identifier increments by 1 for each HCA port in the host.

Configuring IPoIB

To configure IPoIB on your Linux host:

1. Log in to your Linux host.
2. Enter the **ifconfig** command with

- the appropriate IB interface (**ib0** or **ib1** on a host with one HCA)
- the IP address that you want to assign to the interface
- the **netmask** keyword
- the subnet mask that you want to assign to the interface

to configure an IB interface.

Example

```
host:~ # ifconfig ib0 192.168.0.176 netmask 255.255.252.0
```

3. (Optional) Enter the **ifconfig** command with the appropriate port identifier (**ib**#) argument to verify the configuration.

Example

```
1.    host:~ # ifconfig ib0
2.    ib0       Link encap:Ethernet  HWaddr 93:C1:2A:29:33:3E
3.              inet addr:192.168.0.176  Bcast:192.168.0.255  Mask:255.255.252.0
4.              UP BROADCAST RUNNING MULTICAST  MTU:2044  Metric:1
5.              RX packets:2695034 errors:0 dropped:0 overruns:0 frame:0
6.              TX packets:1195933 errors:0 dropped:0 overruns:0 carrier:0
7.              collisions:0 txqueuelen:128
8.              RX bytes:343087140 (327.1 Mb)  TX bytes:67417660 (64.2 Mb)
```

In the preceding example, the IP address of the interface appears as 192.168.0.176 (line 3).

4. Repeat step 2 and step 3 on the remaining interface(s).

Creating a subinterface

Subinterfaces divide primary (parent) interfaces to provide traffic isolation. Partition assignments distinguish subinterfaces from parent interfaces. The default Partition Key (p_key), ff:ff, applies to the primary (parent) interface.

To create a subinterface:

1. Create a partition on a Server Switch. Alternatively, you can choose to create the partition of the IB interface first, and then create the partition for the ports on the Server Switch. Refer to the *Element Manager User Guide* for information regarding partitions on the IB Server Switch.
2. Log in to your host.
3. Navigate to the /usr/local/topspin/sbin directory.

Example

```
host:~ # cd /usr/local/topspin/sbin
```

4. Enter the **ipoibcfg add** command with
   - the parent interface (**ib***X***)** to which you want to add the subinterface
   - the partition key value that has been created on the Server Switch

   to create the subinterface.

Example

```
host:~ # ./ipoibcfg add ib0 80:02
```

5. Enter the **ifconfig -a** command to verify that you created the interface (line 27 in the following example).

Example

```
1.    host:~ # ifconfig -a
2.    eth0      Link encap:Ethernet  HWaddr 00:09:6B:B5:73:D0
3.              inet addr:10.2.1.176  Bcast:10.2.3.255  Mask:255.255.252.0
4.              UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
5.              RX packets:3969163 errors:0 dropped:0 overruns:0 frame:0
6.              TX packets:17449 errors:0 dropped:0 overruns:0 carrier:0
7.              collisions:0 txqueuelen:1000
8.              RX bytes:435714894 (415.5 Mb)  TX bytes:2257959 (2.1 Mb)
9.              Interrupt:16
10.
11.   eth1      Link encap:Ethernet  HWaddr 00:09:6B:B5:73:D1
12.             BROADCAST MULTICAST  MTU:1500  Metric:1
13.             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
14.             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
15.             collisions:0 txqueuelen:1000
16.             RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
17.             Interrupt:17
18.
19.   ib0       Link encap:Ethernet  HWaddr 93:C1:2A:29:33:3E
20.             inet addr:192.168.0.176  Bcast:192.168.0.255  Mask:255.255.252.0
21.             UP BROADCAST RUNNING MULTICAST  MTU:2044  Metric:1
22.             RX packets:2695034 errors:0 dropped:0 overruns:0 frame:0
23.             TX packets:1195933 errors:0 dropped:0 overruns:0 carrier:0
24.             collisions:0 txqueuelen:128
25.             RX bytes:343087140 (327.1 Mb)  TX bytes:67417660 (64.2 Mb)
26.
27.   ib0.8002  Link encap:Ethernet  HWaddr 00:00:00:00:00:00
28.             BROADCAST MULTICAST  MTU:2044  Metric:1
29.             RX packets:0 errors:0 dropped:0 overruns:0 frame:0
30.             TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
31.             collisions:0 txqueuelen:128
32.             RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
33.
34.   ib1       Link encap:Ethernet  HWaddr DA:C1:2F:17:55:E8
35.             inet addr:192.168.4.176  Bcast:192.168.4.255  Mask:255.255.252.0
36.             UP BROADCAST RUNNING MULTICAST  MTU:2044  Metric:1
37.             RX packets:12 errors:0 dropped:0 overruns:0 frame:0
38.             TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
39.             collisions:0 txqueuelen:128
40.             RX bytes:756 (756.0 b)  TX bytes:384 (384.0 b)
41.
42.   lo        Link encap:Local Loopback
43.             inet addr:127.0.0.1  Mask:255.0.0.0
44.             UP LOOPBACK RUNNING  MTU:16436  Metric:1
45.             RX packets:362 errors:0 dropped:0 overruns:0 frame:0
46.             TX packets:362 errors:0 dropped:0 overruns:0 carrier:0
47.             collisions:0 txqueuelen:0
48.             RX bytes:43815 (42.7 Kb)  TX bytes:43815 (42.7 Kb)
```

6. Enter the **cd** command to return to the root directory.

   Example

```
host:~ # cd
```

7. Configure the new interface just as you would the parent interface (refer to "Configuring IPoIB" on page 9).

   Example

```
host:~ # ifconfig ib0.8002 192.168.5.5 netmask 255.255.252.0
```

Removing a subinterface

To remove a subinterface:

1. Bring the subinterface offline. You cannot remove a subinterface until you bring it down.

   Example

```
host:~ # ifconfig ib0.8002 down
```

2. Navigate to the /usr/local/topspin/sbin directory.

   Example

```
host:~ # cd /usr/local/topspin/sbin
```

3. Enter the **ipoibcfg** command with
   - the **del** keyword
   - the primary interface (**ib***X*) of the subinterface that you want to remove
   - the P_key of the subinterface that you want to remove

   to delete the subinterface.

   Example

```
host:~ # ./ipoibcfg del ib0 80:02
```

4. (Optional) Enter the **cd** command to return to the root directory.

   Example

```
host:~ # cd
```

5. (Optional) Enter the **ifconfig -a** command to verify that the subinterface no longer appears in the interface list.

Example

```
host:~ # ifconfig -a
eth0      Link encap:Ethernet  HWaddr 00:09:6B:B5:73:D0
          inet addr:10.2.1.176  Bcast:10.2.3.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:4034686 errors:0 dropped:0 overruns:0 frame:0
          TX packets:18863 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:443157905 (422.6 Mb)  TX bytes:2497589 (2.3 Mb)
          Interrupt:16

eth1      Link encap:Ethernet  HWaddr 00:09:6B:B5:73:D1
          BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:0 (0.0 b)  TX bytes:0 (0.0 b)
          Interrupt:17

ib0       Link encap:Ethernet  HWaddr 93:C1:2A:29:33:3E
          inet addr:192.168.0.176  Bcast:192.168.0.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:2044  Metric:1
          RX packets:2695034 errors:0 dropped:0 overruns:0 frame:0
          TX packets:1195933 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:128
          RX bytes:343087140 (327.1 Mb)  TX bytes:67417660 (64.2 Mb)

ib1       Link encap:Ethernet  HWaddr DA:C1:2F:17:55:E8
          inet addr:192.168.4.176  Bcast:192.168.4.255  Mask:255.255.252.0
          UP BROADCAST RUNNING MULTICAST  MTU:2044  Metric:1
          RX packets:12 errors:0 dropped:0 overruns:0 frame:0
          TX packets:5 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:128
          RX bytes:756 (756.0 b)  TX bytes:384 (384.0 b)

lo        Link encap:Local Loopback
          inet addr:127.0.0.1  Mask:255.0.0.0
          UP LOOPBACK RUNNING  MTU:16436  Metric:1
          RX packets:378 errors:0 dropped:0 overruns:0 frame:0
          TX packets:378 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:45023 (43.9 Kb)  TX bytes:45023 (43.9 Kb)
```

Sample startup script

The file that follows is an example file named ifcfg-ib0 that resides in /etc/sysconfig/network-scripts/ on a Linux host.

```
DEVICE=ib0
BOOTPROTO=static
IPADDR=192.168.10.1
NETMASK=255.255.255.0
ONBOOT=yes
```

Verifying IPoIB

To verify your configuration, ping the interface that you configured. To verify your configuration:

1. Log in to your host.
2. Enter the ping command with the IP address of your IB port.

   Example

```
host:~ # ping 192.168.2.9
PING 192.168.2.9 (192.168.2.9) 56(84) bytes of data.
64 bytes from 192.168.2.9: icmp_seq=1 ttl=64 time=0.033 ms
64 bytes from 192.168.2.9: icmp_seq=2 ttl=64 time=0.018 ms
64 bytes from 192.168.2.9: icmp_seq=3 ttl=64 time=0.016 ms

--- 192.168.2.9 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 1999ms
rtt min/avg/max/mdev = 0.016/0.022/0.033/0.008 ms
host:~ #
```

Configuring high-availability ports

IPoIB supports active/passive port failover high availability between two ports on the same HCA. When you enable the high availability feature, both ports on the HCA (ib0 and ib1) merge into one virtual port. If you configure high availability between the ports on the HCA, only one of the physical ports passes traffic. The other port is used as standby in the event of a failure.

Merging physical ports

To configure high availability on HCA ports in a Linux host:

1. Log in to your Linux host.
2. Navigate to the /usr/local/topspin/sbin directory.

   Example

```
host:~ sbin# cd /usr/local/topspin/sbin/
```

3. Enter the **ipoibcfg list** command to display the available interfaces. The example that follows indicates two HCAs in a single host.

   Example

```
host:~ sbin# ./ipoibcfg list
ib0 (P_Key 0xffff)
ib1 (P_Key 0xffff)
ib2 (P_Key 0xffff)
ib3 (P_Key 0xffff)
```

4. Enter the **ifconfig** command with
   - the IB identifier of the first IB port on the HCA
   - the **down** keyword

   to disable the first interface that you want to merge into one virtual port.

   Example

```
host:~ sbin# ifconfig ib0 down
```

5. Enter the **ifconfig** command with
   - the IB identifier of the second IB port on the HCA
   - the **down** keyword

   to disable the second interface that you want to merge into one virtual port.

   Example

```
host:~ sbin# ifconfig ib1 down
```

6. Enter the **ipoibcfg merge** command with
   - the IB identifier of the first IB port on the HCA
   - the IB identifier of the second IB port on the HCA

   to merge the two ports on the first HCA into one virtual HA port.

   Example

```
host:~ sbin# ./ipoibcfg merge ib0 ib1
```

7. Show the interfaces.

   NOTE: The ib1 interface no longer appears, as it is merged with ib0.

   Example

```
host:~ sbin# ./ipoibcfg list
ib0 (P_Key 0xffff)
ib2 (P_Key 0xffff)
ib3 (P_Key 0xffff)
```

8. Enter the ifconfig command with
   - the appropriate port identifier (**ib**#)
   - the **up** keyword

to enable the interface.

Example

```
host:~ sbin# ifconfig ib0 up
```

9. Assign an IP address to the merged port just as you would to a standard interface. Refer to "Configuring IPoIB" on page 9.

Unmerging physical ports

To disable active/passive high availability on the physical ports:

1. Enter the **ifconfig** command with the appropriate IB interface argument and the **down** argument to disable the high availability interface that you want to unmerge.

Example

```
host:~ sbin# ifconfig ib0 down
```

2. Enter the **ipoibcfg unmerge** with the identifier of the port that you want to unmerge.

Example

```
host:~ sbin# ./ipoibcfg unmerge ib0
```

3. Enter the **ifconfig** command with the appropriate IB interface argument and the **up** argument to enable the interfaces.

Example

```
host:~ sbin# ifconfig ib0 up
host:~ sbin# ifconfig ib1 up
```

4

SCSI RDMA Protocol

The following sections appear in this chapter:

- "Introduction" on page 17
- "Configuring SRP" on page 17
- "Verifying SRP" on page 21

Introduction

SRP requires no manual configuration on the host side. To connect an IB-attached SRP host to a storage area network (SAN), cable your SRP host to an IB fabric that includes a Server Switch with a FC gateway. Log in to the Server Switch to configure the FC connection between the SAN and the SRP host.

Configuring SRP

We provide a number of ways to configure the connection between the SAN and the SRP host. The method that you choose depends on the interfaces available to you and the global access settings on your Server Switch. The instructions in this section provide one example of how to configure the connection. For more detailed instructions, refer to the *Fibre Channel Gateway User Guide*.

To configure your IB fabric to connect a SRP host to a SAN:

1. "Configuring Initiators" on page 18
2. "Configuring ITLs" on page 18

NOTE: If you intend to manage your environment with VFrame software, do not configure ITLs.

Configuring Initiators

When you configure initiators, you assign FC WWNNs to SRP hosts so the SAN can recognize the hosts. Steps to configure initiators appear in the "Configuring ITLs" section of this chapter.

NOTE: You can configure initiators that you have not yet connected to your fabric. You can enter the GUID of the initiator into the CLI or Element Manager (EM) so the configuration works when you connect the SRP host.

Configuring ITLs

You must configure ITLs for your initiators to communicate with your storage. You can configure ITLs with the CLI or the EM graphical user interface (GUI).

- If you restricted port and LUN access when you configured global attributes, proceed to "Configure ITLs with Element Manager while global policy restrictions apply" on page 19.
- If you did not configure access, perform the steps below.

NOTE: If you enter an FC command and receive an error message that reads, "Operation temporarily failed - try again," give your FC gateway time to finish initializing, then retry the command.

Configuring ITLs with Element Manager while no global policy restrictions apply

To configure ITLs with a Linux SRP host while your port masking and LUN masking policies are unrestricted:

1. Log in to your host.
2. Enter the **/usr/local/topspin/bin/vstat --verbose | grep -i guid** command to display the host GUID.

   NOTE: Record the GUID value. You will enter it repeatedly.

3. Bring up the FC gateways on your Server Switch with the following steps:
   a. Launch EM.
   b. Double-click the FC gateway card that you want to bring up. The **Fibre Channel Card** window opens.
   c. Click the **up** radio button in the **Enable/Disable Card** field, then click the **Apply** button.
   d. (Optional) Repeat this process for additional gateways.

   The FC gateway automatically discovers all attached storage.

   NOTE: Discovered LUs remain gray (inactive) until a SRP host connects to them. Once a host connects to an LU, its icon becomes blue (active). Hosts do not stay continually connected to LUs, so the color of the icon may change.

4. From the **FibreChannel** menu of the EM, select **Storage Manager**. The **Topspin Storage Manager** window opens.

5. Click the **SRP Hosts** folder in the **Storage** navigation tree in the left-hand frame of the interface. The **SRP Hosts** display appears in the right-hand frame of the interface.
6. Click the **Define New** button in the **SRP Hosts** display. The **Define New SRP Host** window opens.

NOTE: If your host includes multiple HCAs, you must configure each individual HCA as an initiator. When you configure one HCA in a host, any other HCAs in the host are not automatically configured.

7. Select a GUID from the **Host GUID** pulldown menu in the **Define New SRP Host** window. The menu displays the GUIDs of all connected hosts that you have not yet configured as initiators.
8. (Optional) Type a description in the **Description** field in the **Define New SRP Host** window.
9. Click the **Next >** button. The **Define New SRP Host** window displays a recommended WWNN for the host and recommended WWPNs that will represent the host on all existing and potential FC gateway ports.

NOTE: You can manually configure the WWNN or WWPNs, but we *strongly recommend* that you use the default values to avoid conflicts.

10. Click the **Finish** button. The new host appears in the **SRP Hosts** display.
11. Expand the **SRP Hosts** folder in the **Storage** navigation tree, then click the host that you created. The host display appears in the right-hand frame of the interface.
12. (Optional) Click the **LUN Access** tab in the host display, then click the **Discover LUNs** button. The targets and associated LUNs that your FC gateway sees appear in the **Accessible LUNs** field.
13. Click the **Refresh** button in the **Topspin Storage Manager** window.

Configure ITLs with Element Manager while global policy restrictions apply

These instructions apply to environments where the portmask policy and LUN masking policy are both restricted. To verify that you have restricted your policies:

1. Enter the **show fc srp-global** command at the command-line interface.
2. View the **default-gateway-portmask-policy** and **default-lun-policy fields**.

If restrictions apply to either field, **restricted** appears in the field output.

To configure ITLs with a Linux SRP host while your port masking and LUN masking policies are restricted:

1. Log in to your host.
2. Enter the **/usr/local/topspin/bin/vstat --verbose | grep -i guid** command at the host CLI to display the host GUID.

NOTE: Record the GUID value. You will enter it repeatedly.

3. Bring up the FC gateways on your Server Switch with the following steps:
   a. Launch EM.
   b. Double-click the FC gateway card that you want to bring up. The **Fibre Channel Card** window opens.
   c. Click the **up** radio button in the **Enable/Disable Card** field, then click the **Apply** button.
   d. (Optional) Repeat this process for additional gateways.

   The FC gateway automatically discovers all attached storage.

   NOTE: Discovered LUs remain gray (inactive) until a SRP host connects to them. Once a host connects to a LU, its icon becomes blue (active).

4. From the **FibreChannel** menu, select **Storage Manager...**
5. Click the **SRP Hosts** folder in the **Storage** navigation tree in the left-hand frame of the interface. The **SRP Hosts** display appears in the right-hand frame of the interface.
6. Click the **Define New** button in the **SRP Hosts** display. The **Define New SRP Host** window opens.

   NOTE: If your host includes multiple HCAs, you must configure each individual HCA as an initiator. When you configure one HCA in a host, any other HCAs in the host are not automatically configured.

7. Select a GUID from the **Host GUID** pulldown menu in the **Define New SRP Host** window. The menu displays the GUIDs of all available hosts that you have not yet configured as initiators.
8. (Optional) Type a description in the **Description** field in the **Define New SRP Host** window. If you do not enter a description, your device will assign a description.
9. Click the **Next >** button. The **Define New SRP Host** window displays a recommended WWNN for the host and recommended WWPNs that will represent the host on all existing and potential FC gateway ports.

   NOTE: You can manually configure the WWNN or WWPNs, but we *strongly recommend* that you use the default values to avoid conflicts.

10. Click the **Finish** button. The new host appears in the **SRP Hosts** display.
11. Expand the **SRP Hosts** folder in the **Storage** navigation tree, then click the host that you created. The host display appears in the right-hand frame of the interface.
12. Click the **Targets** tab in the host display. Double-click the WWPN of the target that you want your host to access. The **IT Properties** window opens.
13. Click the "**...**" button ( ) next to the **Port Mask** field. The **Select Port(s)** window opens and displays two port numbers for each slot in the chassis. "Raised" port numbers represent restricted ports. "Pressed" port numbers represent accessible ports.
14. Click the port(s) to which the SAN connects to "press" them and grant the initiator access to the target through those ports, then click the **OK** button.
15. Click the **Apply** button in the **IT Properties** window, then close the window.
16. Click the **LUN Access** tab in the host display, then click the **Discover LUNs** button. The targets and associated LUNs that your FC gateway sees appear in the **Available LUNs** field.
17. Click the **LUN Access** tab, click the target that you configured in step 13, then click the **Add >** button. The target and its LUN(s) appear in the **Accessible LUNs** field in an **Inactive ITLs** folder.

18. Click the LUN that you want your host to reach, then click the **Edit ITL Properties** button. The **ITL Properties** window opens.
19. Click the "**...**" button ( ) next to the **Port Mask** field. The **Select Port(s)** window opens and displays two port numbers for each slot in the chassis. "Raised" port numbers represent restricted ports. "Pressed" port numbers represent accessible ports.
20. Click the port(s) to which the SAN connects to "press" them and grant the initiator access to the target through those ports, then click the **OK** button.
21. Click the **Refresh** button in the **Topspin Storage Manager** window.

Verifying SRP

You can verify SRP host-to-storage connections with the host CLI or with the EM GUI.

Verifying with the CLI

To verify that your host connects successfully to FC storage:

1. Log in to your SRP host.
2. Enter the **cat /proc/scsi/scsi** command to view attached storage.

Example

```
# cat /proc/scsi/scsi
Attached devices:
Host: scsi0 Channel: 00 Id: 00 Lun: 00
  Vendor: SEAGATE  Model: ST336706LC       Rev: 010A
  Type:   Direct-Access                    ANSI SCSI revision: 03
Host: scsi0 Channel: 00 Id: 01 Lun: 00
  Vendor: SEAGATE  Model: ST336706LC       Rev: 010A
  Type:   Direct-Access                    ANSI SCSI revision: 03
Host: scsi2 Channel: 00 Id: 00 Lun: 00
  Vendor: DGC      Model: RAID 1           Rev: 0099
  Type:   Direct-Access                    ANSI SCSI revision: 04
Host: scsi2 Channel: 00 Id: 00 Lun: 01
  Vendor: DGC      Model: RAID 5           Rev: 0099
  Type:   Direct-Access                    ANSI SCSI revision: 04
Host: scsi2 Channel: 00 Id: 00 Lun: 02
  Vendor: DGC      Model: RAID 5           Rev: 0099
  Type:   Direct-Access                    ANSI SCSI revision: 04
```

NOTE: This command verifies that the SRP host sees FC storage. It does not verify that the host can connect to or communicate with the storage. You must configure permissions to grant SRP hosts access to LUNs. For more information, refer to the *Fibre Channel Gateway User Guide*.

Verifying with Element Manager

To verify that your host connects successfully to FC storage:

1. Launch EM and log in to the Server Switch that connects your SRP host to FC storage.
2. Click the **FibreChannel** menu and select **Storage Manager**. The **Storage Manager** window opens.

3. Expand the SRP hosts folder in the Storage navigation tree. A list of SRP hosts appears. SRP host icons that appear in blue () successfully connect to storage.
4. (Optional) Verify LUN access with the steps that follow:
    a. Click an SRP host in the Storage navigation tree.
    b. Click the **LUN Access** tab in the right-hand frame of the display.
    c. Expand all icons in the **Accessible LUNs** field. The SRP host connects successfully to LUNs with a blue LUN icon ().

5

Sockets Direct Protocol

The following sections appear in this chapter:

- "Configuring IPoIB interfaces" on page 23.
- "Specifying connection overrides" on page 23
- "Converting sockets-based applications" on page 23
- "Running a performance test on SDP" on page 25
- "Sample configuration - SDP for IBM DB2 v8.1" on page 26
- "Sample configuration - OracleNet™ Over SDP for Oracle 9i" on page 30

Configuring IPoIB interfaces

SDP uses the same IP addresses and interface names as IPoIB. Configure the IPoIB IP interfaces, if you have not already done so (page 9).

Specifying connection overrides

Use a text editor to open the libsdp.conf file (located in /usr/local/topspin/etc). This file defines when to automatically use SDP instead of TCP. You may edit this file to specify connection overrides.

Converting sockets-based applications

Refer to "Converting sockets-based applications to use SDP" on page 24 for information on the various conversion methods.

Converting sockets-based applications to use SDP

Table 5-1 describes the three ways to convert your sockets-based applications to use SDP instead of TCP.

Table 5-1: SDP Conversion Information

Conversion Type	Method	Required Action
Explicit/ source code	Converts sockets to use SDP based on application source code. This is useful when you want full control from your application when using SDP.	1. Change your source code to use `AF_INET_SDP` instead of `AF_INET` when calling the socket ( ) system call. • `AF_INET_SDP` is defined in `/usr/local/topspin/include/sdp_sock.h`
Explicit/ application	Converts socket streams to use SDP based on the application environment.	1. Load the installed `libsdp_sys.so` library in one of the following ways: • Edit the `LD_PRELOAD` environment variable. Set this to the full path of the library you want to use and it will be preloaded. *or* • Add the full path of the library into `/etc/ld.so.preload`. The library will be preloaded for every executable that is linked with `libc`. 2. Set the application environment to include AF_INET_SDP. Example: csh setenv AF_INET_SDP sh AF_INET_SDP =1 export AF_INET_SDP

Table 5-1: SDP Conversion Information

Conversion Type	Method	Required Action

Automatic	Converts socket streams based upon destination port, listening port, or program name.	1. Load the installed `libsdp.so` library in one of the following ways:
		• Edit the `LD_PRELOAD` environment variable. Setting this to the full path of the library you want to use will cause it to be preloaded.
		or
		• Add the full path of the library into `/etc/ld.so.preload`. This will cause the library to be preloaded for every executable that is linked with `libc`.
		2. Configure the ports, IP addresses, or applications that explicitly use SDP by editing the `libsdp.conf` file.
		a. locate `libsdp.conf` (located in `/usr/local/topspin/etc`)
		b. Make the following modifications:
		• Match on Destination Port
		Syntax:
		`destination ip_addr`[/*`prefix_length`*][:*`start_port`* [-*`end_port`*]]
		Example:
		match *`destination`* 192.168.1.0/24
		• Match on Listening Port
		Syntax:
		`listen ip_addr`[/*`prefix_length`*][:*`start_port`*[-*`end_port`*]]
		Example:
		match *listen* *:5001
		• Match on Program Name
		Syntax:
		`match program program_name*`
		This uses shell type globs. *`db2*`* matches on any program with a name starting with `db2`. *`t?p`* would match on `ttcp`, etc.
		Example:
		`match program db2*`
		For more information about how `AF_INET` sockets are converted to `AF_SDP` sockets, refer to `/usr/local/topspin/etc/libsdp.conf`.

Running a performance test on SDP

To perform throughput and latency tests on SDP, refer to "SDP performance versus IPoIB using Netperf" on page 58.

Sample configuration - SDP for IBM DB2 v8.1

The following subsections appear in this section:

- "Performance acceleration" on page 26
- "Overview" on page 26
- "Sample topology" on page 26
- "Prerequisites" on page 27
- "Configuring the application server" on page 27
- "Configuring the database server" on page 28
- "(Optional) Associating the IPC connection to IB" on page 29
- "Restarting clients and servers" on page 29
- "(Optional) Setting Up Non-IB connections" on page 29
- "Configuring other listeners" on page 29
- "Confining SDP processes" on page 29
- "Troubleshooting the configuration" on page 29

Performance acceleration

By leveraging a Server Switch, it is very simple to accelerate the database to application tier through the network by utilizing the SDP between connected systems. While additional improvements can be achieved by modifying the application tier client and/or database server connection code, this is not necessary to enable much of the benefit that a database environment can achieve.

In order to realize DB2 performance improvements with IB, it is necessary to enable SDP for inter-process communications between cluster servers. This can be done non-invasively, simply by controlling which processes use the SDP libraries.

Overview

To accelerate application performance in database systems:

- an additional library is loaded for all binaries; and
- a configuration script is set up to focus the scope of the SDP acceleration to the appropriate processes.

This needs to be done on both the application server and database server in the same way.

Sample topology

The following example shows a single database server attached to a single application server.

- The client communicates through an Ethernet to IB gateway with the application server on the 10.10.1.50 IP address using normal TCP/IP communications.

- The SDP protocol communicates between the database and the application servers using the alternate 192.168.10.50 address.

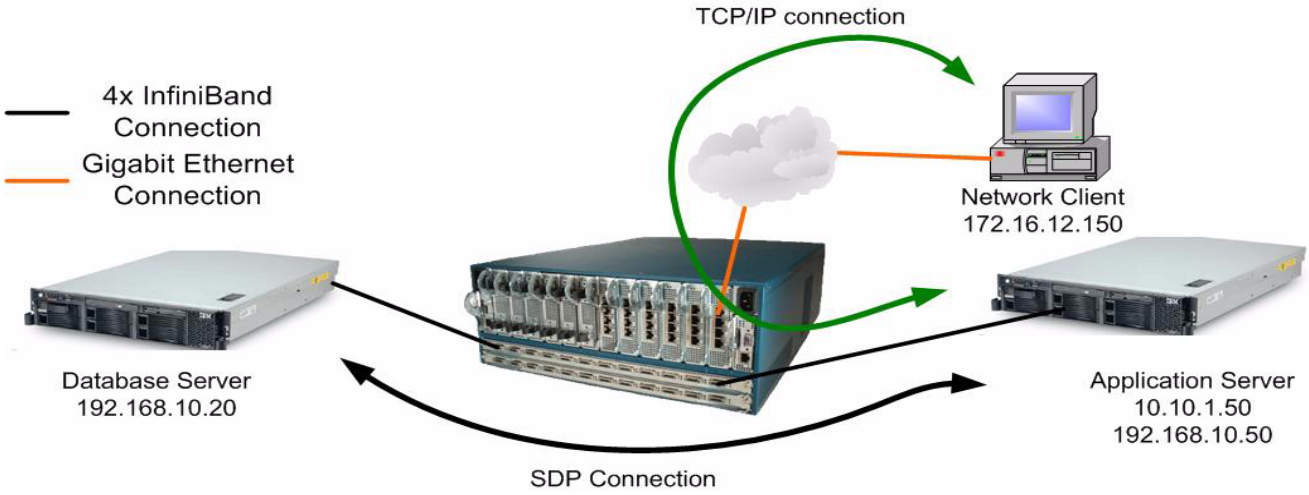

Figure 5-1: Sample Topology Using SDP

Prerequisites

Set up the IPoIB protocol, defining addresses for HCA network interfaces ib# (refer to "Configuring IPoIB" on page 9 for details). All processes that need to use SDP must use the HCA interfaces. Please refer to the DB2 documentation on configuring the DB2 configuration file **db2nodes.cfg** in order to specify the IB network interfaces as the interconnect addresses.

Configuring the application server

Set up a Preload Script

Set up a preload script in order to load the SDP library for all programs.

Example

```
# echo '/lib/libsdp.so' >> /etc/ld.so.preload
```

Add configuration lines to the SDP initialization script

Add the appropriate configuration lines to the SDP initialization script.

This sets the host to listen on port 5000 for SDP connections on the 192.168.10.50 IP address, and to use SDP for any outbound connections targeted to port 5000 on a remote host. This also assumes that the DB2 server and client are configured to connect on port 5000. Set this as appropriate to your environment.

Perform these changes on all clients and servers.

- match listen *:* program db2*
- match destination *:* program db2*

Example

```
# echo 'match listen *.* program db2*'  >>
/usr/local/topspin/etc/libsdp.conf
# echo 'match destination *:* program db2*'  >>
/usr/local/topspin/etc/libsdp.conf
```

Examine configuration files

View SDP connection information by examining the following two files:

```
/proc/topspin/sdp/conn_data
/proc/topspin/sdp/conn_main
```

These files should show you the connections coming over SDP. If there are no SDP connections, these special files will only show header information. If SDP is enabled properly on the server, you should see at least one connection in wait state on the server.

Configuring the database server

Set up a preload script

1. Set up a preload script in order to load the SDP library for all programs.
2. Add the following line to **/etc/ld.so.preload:**

   /lib/libsdp.so

Example

```
# echo '/lib/libsdp.so' >> /etc/ld.so.preload
```

In this way, all processes prefixed with “db2” will use the SDP socket calls provided in the /lib/libsdp.so library.

NOTE: A typographical error in this entry can cause error messages. Refer to “Loading error message” on page 29.

Add configuration lines to the SDP initialization script

Add the appropriate configuration lines to the SDP initialization script. This sets the host to listen on port 1521 for SDP connections, and to use SDP for any outbound connections targeted to port 1521 on a remote host. This also assumes that the server and client are configured to connect on port 1521. Set this as appropriate to your environment.

Perform these changes on all clients and servers.

Example

```
# echo 'match listen *.*program db2*'  >>
/usr/local/topspin/etc/libsdp.conf
# echo 'match destination *:* program db2*'  >>
/usr/local/topspin/etc/libsdp.conf
```

Examine Configuration Files

View SDP connection information by examining the following two files:

```
/proc/topspin/sdp/conn_data
/proc/topspin/sdp/conn_main
```

These files should show you the connections coming over SDP. If there are no SDP connections, these special files will only show header information. If SDP is enabled properly on the server, you should see at least one connection in wait state on the server.

(Optional) Associating the IPC connection to IB

This step is only necessary if multiple hosts are running DB2 in cluster mode.

When DB2 is configured, it expects a separate network interface and hostname to be used for IPC traffic; IPC traffic is one of the traffic types that can be accelerated with SDP. It is important that the hostname and IP address that are used for the IPC connection be attached to the IB fabric, especially if the application servers are not on the IB network.

Restarting clients and servers

Restart any DB2 client or server processes after the above changes have been made on all clients and servers.

(Optional) Setting Up Non-IB connections

The configuration described above states that all processes connecting on port 5000 are SDP processes. Processes communicating over SDP need to connect to other processes using SDP; mismatches will not work.

If you need to set up other connections to clients that are not IB-connected (not using SDP), use one of the following methods:

- "Configuring other listeners" on page 29

  or

- "Confining SDP processes" on page 29

Configuring other listeners

If you need to set up other connections to clients that are not IB-connected (not using SDP), you could configure other listeners using port numbers not specified in the **libsdp.conf** file.

Refer to "Examine Configuration Files" on page 28.

Confining SDP processes

As an alternative to configuring additional listeners, you could confine SDP to processes connecting over the IPoIB subnet(s) defined over the IB fabric.

Troubleshooting the configuration

glibc processes fail

A typo in the **/etc/ld.so.preload** file can cause glibc processes to fail.

If glibc processes should fail, clear out the **/etc/ld.so**.**preload** file by using echo.

```
# echo ""  > /etc/ld.so.preload
```

Loading error message

If you get an error message similar to the following, the **libsdp.so** file is either corrupt or was installed improperly:

Example Error Message

```
[root@app1 root]# ls
ls: error while loading shared libraries: /lib/libsdp.so: cannot
open shared object file: No such file or directory
```

1. Use echo to clear out the **/etc/ld.so.preload** file:

Example

```
[root@app1 root]# echo "" >> /etc/ld.so.preload
```

2. Verify the install of the host software. If it is in question, reinstall the host software and modules rpms.
3. Determine if there was a typographical error in the **ld.so.preload** entry.

NOTE: A typographical error will produce an error message like the above, but will display the library filename that might have been misspelled.

Use **echo** to correct the entry into the file, for example:

Example

```
[root@app1 root]# echo /lib/libsdp.so > /etc/ld.so.preload
```

Sample configuration - OracleNet™ Over SDP for Oracle 9i

- "Performance acceleration" on page 30
- "Overview" on page 30
- "Sample topology" on page 30
- "Configuring the application server" on page 31
- "Configuring the database server" on page 32
- "Setting up non-IB connections" on page 32
- "Troubleshooting the configuration" on page 32

Performance acceleration

By leveraging a Server Switch, it is very simple to accelerate the database to application tier through the network by utilizing the SDP protocol between connected systems. While additional improvements can be achieved by modifying the application tier client and/or database server connection code, this is not necessary to enable much of the benefit that a database environment can achieve.

Overview

To accelerate application performance in database systems:

- an additional library is loaded for all binaries; and
- a configuration script is set up to focus the scope of the SDP acceleration to the appropriate processes.

This needs to be done on both the application server and database server in the same way.

Sample topology

The following example shows a single database server attached to a single application server.

- The client communicates through a Ethernet to IB gateway with the application server on the 10.10.1.50 IP address using normal TCP/IP communications.

- The database and the application servers communicate through a Ethernet to IB gateway on an alternate 192.168.10.50 address using the SDP protocol.

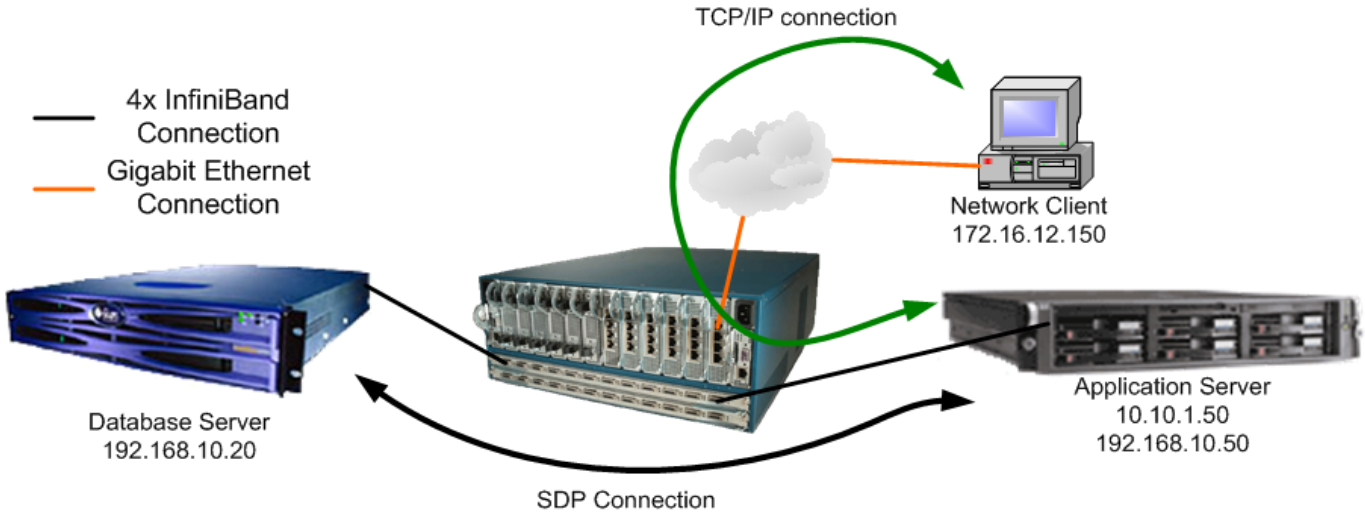

Figure 5-2: Sample Topology Using SDP

Configuring the application server

Set up a preload script

Set up a preload script in order to load the SDP library for all programs.

```
# echo '/lib/libsdp.so' >> /etc/ld.so.preload
```

Add configuration lines to the SDP initialization script

1. Add the appropriate configuration lines to the SDP initialization script.

   This sets the host to listen on port 1521 for SDP connections, and to use SDP for any outbound connections targeted to port 1521 on a remote host. This also assumes that the Oracle server and client are configured to connect on port 1521. Set this as appropriate to your environment.

   Perform these changes on all clients and servers.

```
# echo 'match listen *:1521'  >>
/usr/local/topspin/etc/libsdp.conf
# echo 'match destination *:1521'  >>
/usr/local/topspin/etc/libsdp.conf
```

2. Restart any client or server processes, such as the listener process on the database server, and any applications that leverage OracleNet for database connectivity on the application server.

Examine configuration files

View SDP connection information by examining the following two files:

```
/proc/topspin/sdp/conn_data
/proc/topspin/sdp/conn_main
```

These files should show you the connections coming over SDP. If there are no SDP connections, these special files will only show header information. If SDP is enabled properly on the server, you should see at least one connection in wait state on the server.

Configuring the database server

Set up a preload script

Set up a preload script in order to load the SDP library for all programs.

```
# echo '/lib/libsdp.so' >> /etc/ld.so.preload
```

Add configuration lines to the SDP initialization script

1. Add the appropriate configuration lines to the SDP initialization script.

   This sets the host to listen on port 1521 for SDP connections, and to use SDP for any outbound connections targeted to port 1521 on a remote host. This also assumes that the Oracle server and client are configured to connect on port 1521. Set this as appropriate to your environment.

   Perform these changes on all clients and servers.

```
# echo 'match listen *:1521'  >>
/usr/local/topspin/etc/libsdp.conf
# echo 'match destination *:1521'  >>
/usr/local/topspin/etc/libsdp.conf
```

2. Restart any client or server processes, such as the listener process on the database server, and any applications that leverage OracleNet for database connectivity on the application server.

Examine configuration files

View SDP connection information by examining the following two files:

```
/proc/topspin/sdp/conn_data
/proc/topspin/sdp/conn_main
```

These files should show you the connections coming over SDP. If there are no SDP connections, these special files will only show header information. If SDP is enabled properly on the server, you should see at least one connection in wait state on the server.

Setting up non-IB connections

The above configuration assumes that all processes connecting on port 1521 are SDP processes. Processes communicating over SDP need to connect to other processes using SDP; mismatches will not work.

Configure other listeners

If you need to set up other connections to clients that are not IB-connected (not using SDP), you could configure other listeners using port numbers not specified in the **libsdp.conf** file.

Refer to "Examine configuration files" on page 31.

Confine SDP processes

As an alternative to configuring additional listeners, you could confine SDP to processes connecting over the IPoIB subnet(s) defined over the IB fabric.

Troubleshooting the configuration

A typo in the **/etc/ld.so**.preload file can cause glibc processes to fail.

If glibc processes should fail, clear out the **/etc/ld.so.preload** file using echo.

```
# echo "”  > /etc/ld.so.preload
```

6

Configuring uDAPL Drivers

The following sections appear in this chapter:

- "About the uDAPL configuration" on page 33
- "Building uDAPL applications" on page 33
- "Running a uDAPL performance test" on page 34

About the uDAPL configuration

The uDAPL protocol is transparently installed and requires no further configuration. However, your application may require configuration for uDAPL. In addition, you may want to run the performance and latency tests that are provided with the rpms.

- "Building uDAPL applications" on page 33
- "Running a uDAPL performance test" on page 34
- Viewing a Sample uDAPL implementation (refer to"Configuring Oracle RACTM with InfiniBand" on page 67).

Building uDAPL applications

1. Verify the application requirements:
   a. Your application must support uDAPL. Refer to your application documentation for more information.
   b. uDAPL applications must include udat.h, which is located in /usr/local/topspin/include/dat.
   c. uDAPL applications must be linked against the libraries in /usr/local/topspin/lib.
2. View sample make files and C code. Refer to /usr/local/topspin/examples/dapl.

Running a uDAPL performance test

The utility to test uDAPL performance is included with the rpms after the host-side drivers are installed. The uDAPL test utility is located in the /usr/local/topspin/bin/ directory and must be run on a server and a client host.

Running a uDAPL throughput test

The throughput test measures RDMA WRITE throughput using uDAPL.

1. Start the throughput test on the server host.

   Syntax for server:

   /usr/local/topspin/bin/thru_server.x *<device_name> <RDMA size> <iterations> <batch size>*

   Example

```
[root@cdrom]# /usr/local/topspin/bin/thru_server.x ib0 262144 500 100
```

   - *ib0* is the name of the device.
   - *262144* is the size in bytes of the RDMA WRITE.
   - *500* is the numbers of RDMAs to perform for the test.
   - *100* is the number of RDMAs to perform before waiting for completions.

2. Start the throughput test on the client.

   Syntax for client:

   /usr/local/topspin/bin/thru_client.x *<server IP address> <RDMA size>*

   Example

```
[root@gcdrom]# /usr/local/topspin/bin/thru_client.x ib1 10.3.2.12 262144
```

   - *ib1* is the name of the device.
   - *10.3.2.12* is the IPoIB address of computer 1.
   - *262144* is the size in bytes of the RDMA WRITE.

3. View the throughput results.

   Example

```
RDMA throughput server started on ib0
Created an EP with ep_handle = 0x8143718
queried max_recv_dtos = 256
queried max_request_dtos = 1024
Accept issued...
Received an event on ep_handle = 0x8143718
Context = 29a
Connected!
received rmr_context = bfb78 target_address = 80ea000 segment_length = 10000
Sent  6006.243 Mb in  1.0 seconds throughput =  6003.805 Mb/sec
Sent  6006.243 Mb in  1.0 seconds throughput =  6003.001 Mb/sec
Sent  6006.243 Mb in  1.0 seconds throughput =  6004.016 Mb/sec
Sent  6006.243 Mb in  1.0 seconds throughput =  6003.127 Mb/sec
Sent  6006.243 Mb in  1.0 seconds throughput =  6001.610 Mb/sec
total secs 5 throughput 6003 Mb/sec
Received an event on ep_handle = 0x8143718
Context = 29a
```

Running a uDAPL latency test

The uDAPL latency test measures the half of round-trip latency for uDAPL sends.

1. Start the latency test on the server host.

   Syntax for server:

   /usr/local/topspin/bin/lat_server.x *<device_name> <RDMA size> <iterations> <batch size>*

   Example

```
[root@cdrom]# /usr/local/topspin/bin/lat_server.x ib0 150000 1 1
```

   - *ib0* is the name of the device.
   - *150000* is the numbers of RDMAs to perform for the test.
   - *1* is the size in bytes of the RDMA WRITE.
   - *1* is a flag specifying whether polling or event should be used. 0 signifies polling, and 1 signifies events.

2. Start the latency test on the client.

   Syntax for client:

   /usr/local/topspin/bin/lat_client.x *<server IP address> <RDMA size>*

   Example

```
[root@gcdrom]# /usr/local/topspin/bin/lat_client.x ib1 10.3.2.12 150000 1 1
```

   - *ib1* is the name of the device.
   - *10.3.2.12* is the IPoIB address of computer 1 (server device).
   - *150000* is the numbers of RDMAs to perform for the test.
   - *1* is the size in bytes of the RDMA WRITE.
   - *1* is a flag specifying whether polling or event should be used. 0 signifies polling, and 1 signifies events.

3. View the latency results.

   Example

```
Server Name: 10.3.2.12
Server Net Address: 10.3.2.12
         Connection Event: Received the correct event
Latency:    29.0 us
Latency:    29.0 us
Latency:    28.5 us
Latency:    29.5 us
Latency:    29.5 us
Latency:    29.5 us
Latency:    29.0 us
Average latency:    29.1 us
         Connection Event: Received the correct event
closing IA...
Exiting program...
```

7

Configuring MPI

The following sections appear in this chapter:

- "Configuring MPI" on page 37
- "Configuring SSH" on page 37
- "Editing the PATH variable" on page 39
- "Performing the bandwidth test" on page 40
- "Performing the latency test" on page 40

Configuring MPI

Before you can RSH or SSH MPI, you must establish a connection between two hosts so that you can run commands between the nodes without a log-in or password.

Configuring SSH

To configure SSH between two hosts so that a connection does not require a password:

1. Log in to the host that you want to configure as the local host (hereafter, "host 1"). (The second host serves as the remote host.)

Example

```
login: username
Password: password
Last login: Tue Aug 31 14:52:42 from 10.10.253.115
You have new mail.
[root@qa-bc1-blade4 root]#
```

2. Enter the **ssh-keygen -t rsa** command to generate a public/private RSA key pair. The CLI prompts you for a folder in which to store the key.

Example

```
qa-bc1-blade4:~ # ssh-keygen -t rsa
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
```

3. Press the **Enter** key to store the key in the default directory (/root/.ssh). The CLI prompts you to enter a password.

NOTE: Do not enter a password!

Example

```
Enter file in which to save the key (/root/.ssh/id_rsa):
Created directory '/root/.ssh'.
Enter passphrase (empty for no passphrase):
```

4. Press the **Return** key to bypass the password option. The CLI prompts you to re-enter the password.

Example

```
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
```

5. Press the **Return** key again (once again, omit a password). The CLI displays the fingerprint of the host.

Example

```
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
0b:3e:27:86:0d:17:a6:cb:45:94:fb:f6:ff:ca:a2:00 root@qa-bc1-blade4
qa-bc1-blade4:~ #
```

6. Move to the .ssh directory that you created.

Example

```
qa-bc1-blade4:~ # cd .ssh
qa-bc1-blade4:~/.ssh #
```

7. Copy the public key to a file.

Example

```
qa-bc1-blade4:~/.ssh # cp id_rsa.pub authorized_keys4
```

8. Log in to the host that you want to configure as the remote host (hereafter “host 2”).

Example

```
login: username
Password: password
Last login: Tue Aug 31 14:52:42 from 10.10.253.115
You have new mail.
[root@qa-bc1-blade5 root]#
```

9. Create a .ssh directory in the root directory in host 2.

Example

```
qa-bc1-blade5:~ # mkdir .ssh
```

10. Return to host 1 and copy the file from step 7 to the directory that you created in step 9.

Example

```
qa-bc1-blade4:~/.ssh # scp authorized_keys4 qa-bc1-blade5:/root/.ssh
```

11. Test your ssh connection.

Example

```
[root@qa-bc1-blade4 root]# ssh qa-bc1-blade5
Last login: Tue Aug 31 14:53:09 2004 from host

[root@qa-bc1-blade5 root]#
```

Editing the PATH variable

1. Establish RSH or SSH connections between two nodes so that you can run commands between a local and remote node without a log-in or password (refer to “Configuring SSH” on page 37).
2. Verify that you do not need to add the compiler to the PATH
3. Add, if required, the following paths to your environment PATH:
   - **/usr/local/topspin/mpi/mpich/bin**
   - **/usr/local/topspin/bin**

     The source code is provided for you; however, you do not need to view the source code to run the MPI tests. The source code and Makefile for the MPI test programs can be found in /usr/local/topspin/mpi/mpich/src/examples.

NOTE: Optionally, you can add the paths for all users by adding **export PATH=$PATH:/usr/local/topspin/mpi/mpich/bin:/usr/local/topspin/bin** to your **/etc/profile.d** script.

4. Verify that your compiler and MPI script match. Compilers reside in the **/usr/local/topspin/mpi/mpich/bin** directory. GNU compilers use mpicc and mpif77 scripts. Intel compilers use mpicc.i and mpif90.i scripts.

Performing the bandwidth test

Before you perform the bandwidth test, configure RSH or SSH on your hosts. To perform the test:

1. Log in to your local host.
2. Enter the **mpirun_ssh** (or **mpirun_rsh**) command with
   - the **-np** keyword to specify the number of processes
   - the number of processes (integer)
   - the host name of the local host
   - the host name of the remote host
   - the mpi_bandwidth command
   - the number of times to transfer the data (integer)
   - the number of bytes to transfer (integer)

   to perform the bandwidth test.

   Example

```
[root@qa-bc1-blade2 root]# /usr/local/topspin/mpi/mpich/bin/mpirun_ssh -np 2 qa-
bc1-blade2 qa-bc1-blade3 /usr/local/topspin/mpi/mpich/bin/mpi_bandwidth 1000
262144
The authenticity of host 'qa-bc1-blade2 (X.X.X.X)' can't be established.
RSA key fingerprint is 0b:57:f2:c9:dc:cb:ef:67:1c:51:3b:bf:58:8a:35:04.
Are you sure you want to continue connecting (yes/no)?
```

3. Enter **yes** at the prompt to connect to your remote host.

   Example

```
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added 'qa-bc1-blade2,10.2.1.176' (RSA) to the list of known
 hosts.
262144  241.250722
[root@qa-bc1-blade2 root]#
```

   The output (in the example, **241.250722**) represents available bandwidth in MB/sec.

Performing the latency test

Before you perform the latency test, configure RSH or SSH on your hosts. To perform the test:

1. Log in to your local host.
2. Enter the **mpiruh_ssh** command with
   - the **-np** keyword to specify the number of processes
   - the number of processes (integer)
   - the host name of the local host
   - the host name of the remote host
   - the **mpi_latency** command
   - the number of times to transfer the data (integer)
   - the number of bytes to transfer (integer)

to run the latency test.

Example

```
[root@qa-bc1-blade2 root]# /usr/local/topspin/mpi/mpich/bin/mpirun_ssh -np 2 qa-
bc1-blade2 qa-bc1-blade3 /usr/local/topspin/mpi/mpich/bin/mpi_latency 10000 1
1       6.684000
```

The output (in the example, **6.684000**) represents the latency in microseconds.

8

HCA Utilities

The following sections appear in this chapter:

- "Introduction" on page 43
- "hca_self_test utility" on page 44
- "tvflash utility" on page 45
- "vstat utility" on page 47
- "tsinstall utility" on page 50
- "ifconfig utility" on page 50

Introduction

The sections in this chapter discuss fundamental HCA features. These features address basic usability and provide starting points for troubleshooting. This chapter also includes steps that detail how to enable and disable HCA external IB ports.

The HCA includes a self-test that displays information on the health and activity of the HCA. The HCA also includes the vstat utility, which provides a number of different displays to view HCA details. The list of details can include the GUID of the HCA.

hca_self_test utility

The hca_self_test utility displays basic HCA attributes and provides introductory troubleshooting information. To run this utility:

1. Log in to your host.
2. Enter the **hca_self_test** command in the /usr/local/topspin/sbin/ directory.

Example

```
[root@qa-bc1-blade2 root]# /usr/local/topspin/sbin/hca_self_test

---- Performing InfiniBand HCA Self Test ----
Number of HCAs Detected ................ 1
PCI Device Check ....................... PASS
Host Driver Version .................... rhel3-2.4.21-4.ELsmp-2.0.0-591
Host Driver RPM Check .................. PASS
HCA Type of HCA #0 ..................... Cougar
HCA Firmware on HCA #0 ................. v3.01.0000 build 2.0.0.591
HCA Firmware Check on HCA #0 ........... PASS
Host Driver Initialization ............. PASS
Number of HCA Ports Active ............. 2
Port State of Port #0 on HCA #0 ........ UP
Port State of Port #1 on HCA #0 ........ UP
Error Counter Check .................... FAIL
    REASON: found errors in the following counters
      Errors in /proc/topspin/core/ca1/port1/counters
        Symbol error counter:                          10
Kernel Syslog Check .................... PASS
------------------ DONE ---------------------
```

Table 8-1 lists and describes the fields in the test output.

Table 8-1: Self Test Fields

Field	Description
Number of HCAs Detected	Number of HCAs on the host that the test recognizes.
PCI Device Check	Confirms that Tavor shows up correctly as a PCI device.
Host Driver Version	Version of the drivers on the host.
Host Driver RPM Check	Confirms that the rpms that are installed are compatible with the host OS.
HCA Type of HCA #0	Displays the “family” of the HCA (cougar, jaguar, or lion family).
HCA Firmware on HCA #0	Firmware version that runs on the HCA.
HCA Firmware Check on HCA #0	Displays PASS or FAIL.
Host Driver Initialization	Confirms that the IPoIB driver was installed correctly.
Number of HCA Ports Active	Number of enabled ports on the HCA.
Port State of Port #0 on HCA #0	Displays **up** or **down** to reflect the status of the port.
Port State of Port #1 on HCA #0	Displays **up** or **down** to reflect the status of the port.
Error Counter Check	Displays PASS or FAIL.
Kernel Syslog Check	Displays PASS or FAIL.

tvflash utility

The tvflash utility performs the following tasks:

- Updates the firmware on the HCA
- Displays the boot details of a remote-boot HCA (for details on the remote-boot features, refer to the *Boot over InfiniBand User Guide for Linux*)
- Views card type and firmware version

Upgrading firmware

To upgrade firmware on your host:

1. Log in to your host and copy the updated firmware binary to your local device. For the purpose of these instructions, the binary appears in the /usr/local/topspin/share/ directory.
2. Enter the tvflash command with
   - the **-h** flag
   - the number of the HCAs in the host (0 or 1 on hosts that support 2 HCAs)
   - the firmware binary file (including path)

   Example

```
[root@test root]# /usr/local/topspin/sbin/tvflash –h 0 /usr/local/topspin/share/
fw-cougar-a1-3.00.0002.bin
```

Viewing boot details

Run the **tvflash -o** command to view boot details. To run this utility:

1. Enter the **cd** command to move to the **/usr/local/topspin/sbin/** directory.

```
[root@qa-bc1-blade2 root]# cd /usr/local/topspin/sbin/
```

2. Enter the **tvflash -o** command.

```
[root@qa-bc1-blade2 sbin]# ./tvflash -o
  auto_upgrade=yes
  boot_enable_port_1=yes
  boot_enable_port_2=yes
  boot_service_scan=yes
  boot_type=disable
  boot_saved_port=2
  boot_saved_ioc_num=1
  boot_saved_dgid=fe80:0000:0000:0000:0005:ad00:0000:14aa
  boot_saved_service_name=2200:000c:5005:633c
```

Table 8-2 lists and describes the fields of the tvflash -o display.

Table 8-2: tvflash -o Field

Field	Description
boot_type	Displays disable, pxe, saved, or well_known to reflect the value that you configure.

Configuring the boot type

Configure theboot type to determine the type of remote boot that the HCA executes.l

1. Enter the **cd** command to move to the /usr/local/topspin/sbin/ directory.

```
[root@qa-bc1-blade2 root]# cd /usr/local/topspin/sbin/
```

2. Enter the **tvflash** command with
   - the **-o** argument
   - the **boot_type=** argument
   - the boot type

   to configure the boot type on a Linux HCA.

```
[root@qa-bc1-blade2 sbin]# ./tvflash -o boot_type=pxe
Writing    [==========================================================================]
```

3. (Optional) Enter the **tvflash -o** command to verify the new boot type.

```
[root@qa-bc1-blade2 sbin]# ./tvflash -o
  auto_upgrade=yes
  boot_enable_port_1=yes
  boot_enable_port_2=yes
  boot_service_scan=yes
  boot_type=pxe
  boot_saved_port=2
  boot_saved_ioc_num=1
  boot_saved_dgid=fe80:0000:0000:0000:0005:ad00:0000:14aa
  boot_saved_service_name=2200:000c:5005:633c
```

Viewing card type and firmware version

To display the type of HCA in your host and the firmware that it runs:

1. Log in to your host.
2. Enter the **tvflash** command with the **-i** flag in the /usr/local/topspin/sbin directory.

   Example

```
[root@host ~]# /usr/local/topspin/sbin/tvflash -i

HCA #0: Found MT23108, Cougar, revision A1 (firmware autoupgrade)
  Primary image is v3.01.0000 build 2.5.0.0, with label 'HCA.Cougar.A1'
  Secondary image is v3.01.0000 build 2.0.0.0, with label 'CA.Cougar.A1.Boot'
```

   The firmware that runs on the HCA appears in the **Primary image** line. The card type also appears in this line as one of the following:

   - Jaguar
   - Cougar
   - Cougar Cub
   - Lion Cub
   - Tiger

   The ASIC version appears as A1 or A0.

NOTE: Upon installation of the host drivers, the firmware is automatically updated, if needed. However, if you have outdated firmware on a previously installed HCA you can upgrade the firmware separately.

vstat utility

Use the vstat utility to view GUIDs and other HCA attributes.

Viewing the GUID of the HCA

To view your host GUID on a Linux device:

1. Log in to your host.
2. Enter the **cd** command to move to the /usr/local/topspin/bin directory

   Example

```
# cd /usr/local/topspin/bin
```

3. Enter the **vstat --verbose** command. The GUID of the HCA appears in line 26 of the example output.

NOTE: Abbreviated output appears in the following example.

Example

```
1.    [root@qa-bc1-blade2 bin]# ./vstat --verbose
2.    1 HCA found:
3.            hca_id=InfiniHost0
4.            vendor_id=0x02C9
5.            vendor_part_id=0x5A44
6.            hw_ver=0xA1
7.            fw_ver=0x300010000
8.            num_phys_ports=2
9.            max_num_qp=0xfbf8       (Maximum Number of QPs supported)
10.           max_qp_ous_wr=0xffff    (Maximum Number of oustanding WR on any WQ)
11.           flags=0x00001476
12.           max_num_sg_ent=0x3b     (Max num of scatter/gather entries for WQE other
13.    than RD)
14.           max_num_sg_ent_rd=0x3b  (Max num of scatter/gather entries for RD WQE)
15.           max_num_srq=0x3f0       (Maximum Number of SRQs supported)
16.           max_wqe_per_srq=0xffff  (Maximum Number of oustanding WR on any SRQ)
17.           max_srq_sentries=0x3b   (Maximum Number of scatter entries for SRQ WQE)
18.           max_num_cq=0xff80       (Max num of supported CQs)
19.           max_num_ent_cq=0x1ffff  (Max num of supported entries per CQ)
20.           max_num_mr=0x1fff0      (Maximum number of memory region supported)
21.           max_mr_size=0xFFFFFFFFFFFFFFFF  (Largest contigous block of memory regio
22.   n in bytes)
23.           max_pd_num=0x3ffe       (Maximum number of protection domains supported)
24.           page_size_cap=0x1000    (Largest page size supported by this HCA)
25.           max_pkeys=0x40  (Maximum number of partitions supported)
26.           node_guid=00:05:ad:00:00:01:60:44       (Node GUID for this hca)
27.           local_ca_ack_delay=0x10 (Log2 4.096usec Max. RX to ACK or NAK delay)
28.   port_lid=0x0181
```

Viewing basic HCA attributes with the vstat utility

To run the vstat utility on Linux:

1. Log in to your Linux host.
2. Enter the **/usr/local/topspin/bin/vstat** command.

Example

```
# /usr/local/topspin/bin/vstat
1 HCA found:
        hca_id=InfiniHost0
        vendor_id=0x02C9
        vendor_part_id=0x5A44
        hw_ver=0xA1
        fw_ver=0x300010000
        num_phys_ports=2
                port=1
                port_state=PORT_ACTIVE
                sm_lid=0x000a
                port_lid=0x0016
                port_lmc=0x00
                max_mtu=2048
                gid_tbl_len=32
                GID[  0]= fe:80:00:00:00:00:00:00:05:ad:00:00:00:17:18

                port=2
                port_state=PORT_ACTIVE
                sm_lid=0x000a
                port_lid=0x0002
                port_lmc=0x00
                max_mtu=2048
                gid_tbl_len=32
                GID[  0]= fe:80:00:00:00:00:00:00:05:ad:00:00:00:17:19
```

Table 8-3 lists and describes the fields in the **vstat** command output.

Table 8-3: vstat Fields

Field	Description
hca_id	ID of the HCA.
vendor_id	Vendor ID of the HCA.
vendor_pard_id	ID number of the HCA.
hw_ver	Hardware version of the HCA.
fw_ver	Firmware version that runs on the HCA.
num_phys_ports	Number of physical ports on the HCA.
port	Port on the HCA. **NOTE:** In this display, the port appears as 1 or 2, not 0 or 1.

Table 8-3: vstat Fields (Continued)

Field	Description
port_state	Displays PORT_ACTIVE or PORT_DOWN to display whether the port is up or down.
sm_lid	LID of the subnet manager on the switch that connects to the HCA.
port_lid	LID that the subnet manager assigns to the port.
port_lmc	LID mask control (LMC) of the port.
max_mtu	Maximum transmission unit (in bytes) of the port.
gid_tbl_len	Global ID (GID) table length.
GID	GID of the port.

tsinstall utility

The tsinstall utility installs host-side drivers on Linux hosts. The tsinstall utility arrives with the ISO image. You can only run the utility when you have placed the CD-ROM in your host or when you have mounted the ISO over NFS. For step-by-step instructions, refer to "Installing Host-Side Drivers" on page 5.

NOTE: Whenever you install or upgrade host-side drivers with the tsinstall utility, the utility automatically upgrades the firmware on the HCA.

ifconfig utility

When you troubleshoot, verify that your interfaces run successfully. The following sections describe how to bring up and shut down IB interfaces.

Bringing down an IB port

To bring down an IB interface on Linux:

1. Log in to your host.
2. Enter the **ifconfig** command with
   - the port that you want to disable
   - the **down** keyword

   Example

```
# ifconfig ib0 down
```

Bringing up an IB port

To bring up an IB interface on Linux:

1. Log in to your host.
2. Enter the **ifconfig** command with
   - the port that you want to disable

- the **up** keyword

Example

```
# ifconfig ib0 up
```

9

Sample Performance Test

The following sections appear in this chapter:

- "Introduction" on page 53
- "Hardware and applications" on page 53
- "Network topology" on page 54
- "Host and switch setup" on page 54
- "Running Netperf" on page 54
- "IPoIB performance versus Ethernet using Netperf" on page 56
- "SDP performance versus IPoIB using Netperf" on page 58

Introduction

This test plan requires basic knowledge of Linux administration, networking protocols, and network administration. This test of basic functionality and performance of the system should be completed in three days or less.

Hardware and applications

- Performance testing requires a minimum of two x86-based servers to demonstrate some of the basic functionality of the switch and the associated upper-level protocols (ULPs). To take advantage of the high throughput and low latency aspects of the fabric, testing requires a minimum of dual Xeon servers (approximately 2.0 GHz), each with a 133MHz PCI-X expansion bus.
- Use an Ethernet switch to network the two servers to one another. This switch can be of any speed, but a Gigabit version will provide the best platform for comparing high performance communication over Ethernet and IB.
- You can use a Topspin 90/Cisco SFS 3001 or Topspin 360/Cisco SFS 3012 in this evaluation.

- This test requires one host channel adapter in each server.
- Performance testing requires the Netperf utility. The tool and more information can be found at http://www.netperf.org, or by contacting your sales engineer for a pre-built RPM. Install the Netperf server and client on both servers in the test setup.

Network topology

The network diagram in Figure 9-1 illustrates the way two servers, a Server Switch, and an Ethernet network should be connected for basic testing.

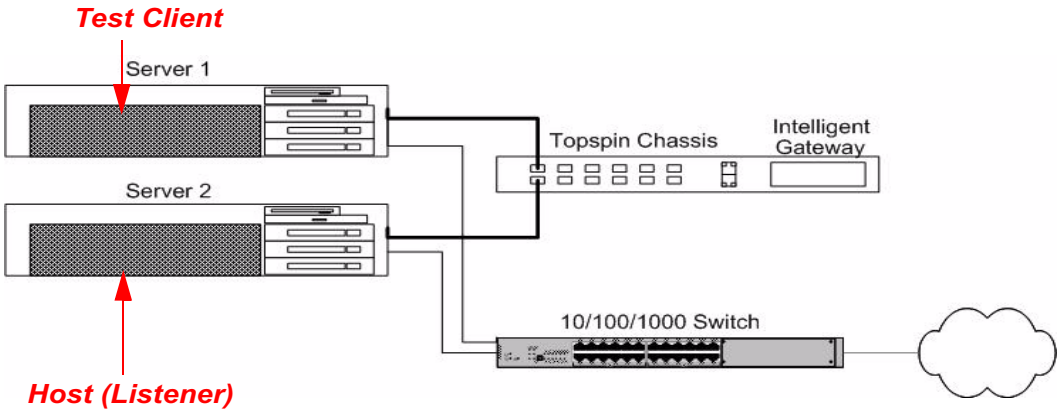

Figure 9-1: Sample Test Topology

Host and switch setup

For basic inter-fabric testing of the Server Switch, no configuration is required on the Server Switch itself; therefore, configuration instructions for the management interface of the switch do not appear here. For those details, refer to the appropriate hardware guide.

- For instructions on installing the HCAs, refer to the *Host Channel Adapter Installation Guide*. This guide details how to physically install HCAs in your hosts.
- To install the ULP drivers, please read and follow the instructions in "Installing Host-Side Drivers" on page 5.

NOTE: Do not continue on to configure the drivers after the installation, as this will be described below to suit the appropriate environment.

- Configure the IP over IB protocol. Refer to "Configuring IPoIB" on page 9.

Running Netperf

To run a Netperf performance test over the IB network:

1. Identify a host to run the Netserver application.
2. Log in to the Netserver host.

3. Enter the **netserver** command to start Netserver.

   Example

```
[root@qa-bc1a-blade3 bin]# ./netserver
Starting netserver at port 12865
[root@qa-bc1a-blade3 bin]#
```

4. Identify a host to run the Netperf application.
5. Log in to the Netperf host.
6. Enter the **netperf** command with
   - the **-v2** flag to set the verbosity level to 2
   - the **-H** flag to specify the IP address of the netserver host
   - the IP address of the netserver host
   - the **-l** flag to configure the duration (in seconds) of the test
   - the **-t** flag to configure the type of traffic (TCP stream)
   - the **TCP_STREAM** argument to specify unreturned TCP traffic
   - the **-c** to display local CPU utilization
   - the **-C** to display remote CPU utilization
   - two dashes (--) to separate global and test-specific arguments
   - the **-m** flag to configure the block size (message length) of each msg (in bytes)

to run Netperf.

Example

```
qa-bc1a-blade4:/data/software/qa/i686/bin # ./netperf -v2 -H 192.168.2.10 -l 30
-t TCP_STREAM -c -C -- -m 65536


TCP STREAM TEST to 192.168.2.10 : histogram
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^6bits/s  % S      % S      us/KB   us/KB

 87380  16384  65536    30.00      1303.34   43.54    80.17    5.473   10.078

Alignment      Offset         Bytes    Bytes       Sends   Bytes    Recvs
Local  Remote  Local  Remote  Xfered   Per                 Per
Send   Recv    Send   Recv             Send (avg)          Recv (avg)
    8       8      0       0 4.888e+09  65536.83     74579   31311.96 156096

Maximum
Segment
Size (bytes)
  1992


Histogram of time spent in send() call.
UNITS              0     1     2     3     4     5     6     7     8     9
UNIT_USEC     :    0:    0:    0:    0:    0:    0:    0:    0:    0:    0
TEN_USEC      :    0:    0:    0:    0:    0:    0:    6:    3:    7:    8
HUNDRED_USEC  :    0:  348: 3736: 56470: 1161:  562: 11056:  642:  258:  247
UNIT_MSEC     :    0:   74:    1:    0:    0:    0:    0:    0:    0:    0
TEN_MSEC      :    0:    0:    0:    0:    0:    0:    0:    0:    0:    0
TENTH_SEC     :    0:    0:    0:    0:    0:    0:    0:    0:    0:    0
>1_SECS: 0
HIST_TOTAL:      74579
```

7. Enter the **killall netperf** command to close the Netperf application.

Example

```
qa-bc1a-blade3:/data/software/qa/i686/bin # pkill netserver
```

IPoIB performance versus Ethernet using Netperf

Netperf has many options. This example just uses the basic TCP stream test for measurements.

Performing a throughput test

1. Develop a base case for comparison.
   a. On *Server 1*, run Netperf across a normal Ethernet interface.

      For the output below, we used a cross-over cable between the two servers on their Gigabit Ethernet interfaces:

      The options entered into Netperf have the following meaning:

      - “-c” and “-C” - request a report of the local and remote CPU utilization metrics.

- "-f g" - requests a report of the results in Gbps.
- "-H 192.168.10.21" - specifies the host to contact for running the test.

Example

```
# netperf -c -C -f g -H 192.168.10.21
```

2. Read the test results.

   The sample results show about wire speed over the Gigabit Ethernet link, with around 20% CPU utilization on both ends.

   Example

```
TCP STREAM TEST to 192.168.10.21
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 87380  16384  16384    10.00         0.94   19.10    23.70    1.662   2.062
```

3. Run the test over the IPoIB interface, which was previously set up.

   Example

```
# netperf -c -C -f g -H 192.168.0.2
TCP STREAM TEST to 192.168.0.2
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 87380  16384  16384    10.01         1.21   33.88    87.35    2.290   5.905
```

   The results in this example show approximately a 28% increase in throughput, but at the expense of higher CPU utilization on both the sender and receiver. This is because the native Ethernet card does TCP/IP checksumming in hardware, while the IPoIB interface must use the host CPU.

Performing a latency test

To demonstrate the latency advantage of IB compared to Ethernet, use a Netperf test called TCP request/response. This test will send a 1-byte request to the remote machine and the remote machine will issue a 1-byte response.

1. Develop a base case for comparison on *Server 1*. Add the **-t TCP_RR** option to the **netperf** command to specify this test.

   Example

```
# netperf -c -C -f g -H 192.168.10.21 -t TCP_RR
```

2. Read the results. The sample results show performance of about 5800 request/response transactions per second.

   Example

```
TCP REQUEST/RESPONSE TEST to 192.168.10.21
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
16384  87380  1       1      10.00   5787.80  4.50   7.30   7.775   12.619
```

3. Run the test over the IB interface on *Server 1*.

   Example

```
netperf -c -C -f g -H 192.168.0.2 -t TCP_RR
TCP REQUEST/RESPONSE TEST to 192.168.0.2
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
16384  87380  1       1      10.00   11629.08  18.30  19.51  15.733  16.777
```

4. Compare the results. The IB interface shows about 11,600 request/response transactions per second, which is approximately double the performance of the Gigabit Ethernet interface.

SDP performance versus IPoIB using Netperf

When you compare IPoIB over IB performance to TCP/IP over Ethernet performance ("IPoIB performance versus Ethernet using Netperf" on page 56), you notice that the higher speed and lower latency of IB create greater CPU overhead with IP. The SDP protocol maintains the advantages of IB and removes the CPU overhead. The SDP protocol sets up a reliable connection over the IB fabric, and TCP socket connections can be made without the overhead of TCP. SDP uses RDMAsemantics to transmit data between the two hosts' buffers without CPU intervention.

Performing a throughput test

1. Tell the SDP library that the next process should use SDP, and start the netserver process on *Server 2*:

   Example

```
# netserver.sdp
```

2. Run the Netperf SDP throughput test on *Server 1*.

   Example

```
# netperf.sdp -c -C -f g -H 192.168.0.2
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 65535  65535  65535    10.00         1.94   36.70    57.90    1.551   2.446
```

3. Read the test results.

   The throughput has increased approximately 50% from using IPoIB, and the CPU utilization has been significantly reduced.

Performing a latency test

In addition to the throughput test, you can also test the effect of using SDP on the request/response test:

Example

```
# netperf.sdp -c -C -f g -H 192.168.0.2 -t TCP_RR
TCP REQUEST/RESPONSE TEST to 192.168.0.2
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
65535  65535  1       1      10.00   17145.82  15.00  17.10  8.747   9.976
```

In the example above, there is approximately a 50% increase in transactions per second from the IPoIB case. In addition, there is a reduction in CPU utilization on both the transmit and receive end.

10

Sample Test Plan

The following evaluation test plan will walk you through basic setup of the IB-based switching fabric, introduce you to some of the ULPs (Upper Layer Protocols) supported on the fabric, and perform some basic tests that showcase the fabric's performance.

Overview

- "Requirements" on page 61
- "Network topology" on page 62
- "Host and switch setup" on page 62
- "IPoIB setup" on page 63
- "IPoIB performance versus Ethernet using netperf" on page 64
- "SDP performance versus IPoIB using netperf" on page 65

Requirements

Prerequisites

This test plan requires basic knowledge of Linux administration, networking protocols, and network administration. This test of basic functionality and performance of the system should be completed in threedays or less.

Hardware and applications

- A minimum of two x86-based servers are required for demonstrating some of the basic functionality of the Server Switch and the associated ULPs. To take advantage of the high

throughput and low latency aspects of the fabric, a minimum of dual Xeon servers (approximately 2.0 GHz) with 133MHz PCI-X expansion busses are required.

- An Ethernet switch should be used to network the two servers together. This switch can be of any speed, but a Gigabit version will provide the best platform for comparing high performance communication over Ethernet and IB.
- A Topspin 90/Cisco SFS 3001 or Topspin 360/Cisco SFS 3012 can be used in this evaluation.
- One HCA is required for each server.
- A utility called *netperf* is also required for performance testing. The tool and more information can be found at http://www.netperf.org, or by contacting your sales engineer for a pre-built rpm. Install the netperf server and client on both servers in the test setup.

Network topology

The network diagram in Figure 10-1 illustrates the way two servers, a Server Switch, and an Ethernet network should be connected for basic testing.

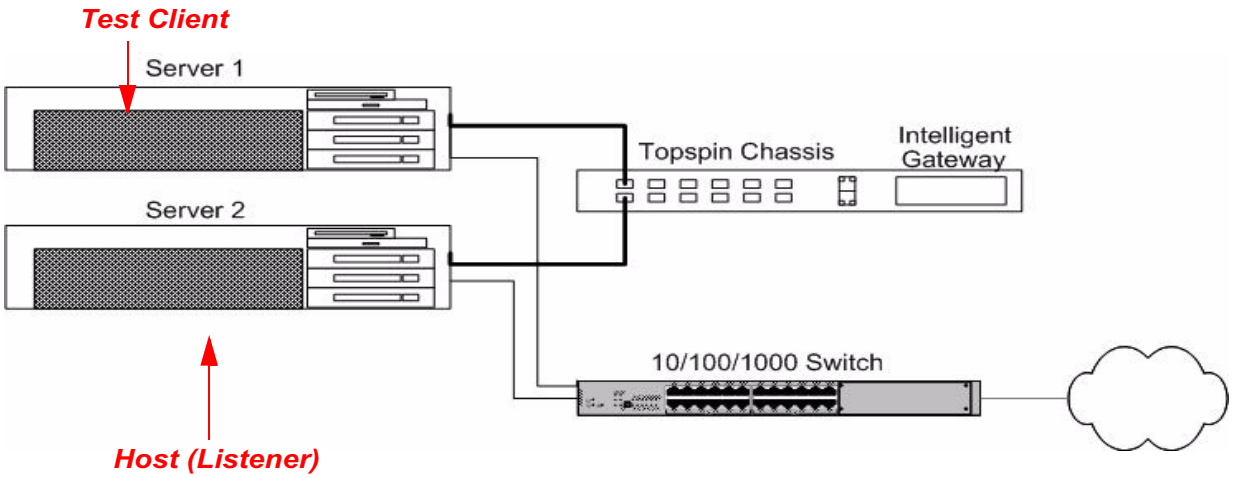

Figure 10-1: Sample Test Topology

Host and switch setup

For basic inter-fabric testing of the Server Switch, no configuration is required on the Server Switch itself; therefore, configuration of the switch's management interface can be left for later.

- For instructions on installing the HCAs, refer to the *HCA Hardware Guide*. This guide describes how to physically install the HCAs into your servers.
- To install the ULP drivers, read and follow the instructions in the *HCA Hardware Guide*.

NOTE: Do not continue on to configure the drivers after the installation, as this will be described below to suit the appropriate environment.

IPoIB setup

IPoIB is simply that–IP packets running over the IB fabric. This protocol is useful for testing connectivity into the fabric between two hosts, and also for taking advantage of the high speed fabric for “legacy” applications that are written to communicate over IP.

Configuring IPoIB

Configuration of IPoIB is similar to configuring Ethernet interfaces under Linux except the interfaces are called ib*x* (in other words, ib0, ib1, etc) instead of eth*x* (for example, eth0, eth1, etc).

To test the IPoIB interfaces, choose a subnet that is currently not routed in your network environment. For this test, we'll choose 192.168.0.0 with a netmask of 255.255.255.0 and assign “Server 1” the address 192.168.0.1 and “Server 2” 192.168.0.2.

1. On *Server 1*, use **ifconfig** to configure ib0:

Example

```
# ifconfig ib0 192.168.0.1 netmask 255.255.255.0
```

2. Verify that the interface was configured properly:

Example

```
# ifconfig ib0
ib0       Link encap:Ethernet  HWaddr 00:00:00:00:00:00
          inet addr:192.168.0.1  Bcast:192.168.0.255  Mask:255.255.255.0
          UP BROADCAST  MTU:2044  Metric:1
          RX packets:0 errors:0 dropped:0 overruns:0 frame:0
          TX packets:3 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:128
          RX bytes:0 (0.0 b)  TX bytes:126 (126.0 b)
```

3. Repeat the process on *Server 2* by configuring ib0 to 192.168.0.2.

   If the system failed to configure the interface properly, you may not have successfully installed the HCA drivers on the operating system. If the drivers did not install, it is likely due to a version mismatch between the driver suite and the installed kernel. Check your versions and make any necessary corrections.

   Refer to http://hp.com/support/

4. To test connectivity, attempt to ping *Server 2* from *Server 1*, using the **ping** command:

Example

```
# ping -c 1 192.168.0.2
PING 192.168.0.2 (192.168.0.2) from 192.168.0.1 : 56(84) bytes of data.
64 bytes from 192.168.0.2: icmp_seq=0 ttl=64 time=154 usec
--- 192.168.0.2 ping statistics ---
1 packets transmitted, 1 packets received, 0% packet loss
round-trip min/avg/max/mdev = 0.154/0.154/0.154/0.000 ms
```

If you do not receive a response from the other server, check cable connectivity. Be sure the IB cable is plugged into the correct port for ib0 on the HCA (top port on the PCI adapter card). Also, check the LEDs on both the HCA and the IB switch. For details, refer to the *Host Channel Adapter Hardware Guide*. Refer to the appropriate *Hardware Guide* for LED information on the switch; hardware guides are available at http://hp.com/support.

IPoIB performance versus Ethernet using netperf

To test the performance characteristics of IPoIB, use a tool called netperf. This utility runs on both machines with one machine listening on a TCP socket and the other connecting and sending test data. The listening program is called *netserver* while the test client is called *netperf*.

Netperf has many options, but this example just uses the basic TCP stream test for measurements.

1. Install the netperf utility on both the netperf server and client in the test setup.

   For more information, refer to the requirements in "Hardware and applications" on page 61.

2. Start the listener on *Server 2*:

Example

```
# netserver
```

Performing a throughput test

1. Develop a base case for comparison.

   a. On *Server 1*, run netperf across a normal Ethernet interface.

   For the output below, we used a cross-over cable between the two servers on their Gigabit Ethernet interfaces:

   The options entered into netperf have the following meanings:

   - "-c" and "-C" - request a report of the local and remote CPU utilization metrics.
   - "-f g" - requests a report of the results in gigabits per second.
   - "-H 192.168.10.21" - specifies the host to contact for running the test.

   Example

```
# netperf -c -C -f g -H 192.168.10.21
```

2. Read the test results.

   The sample results show about wire speed over the Gigabit Ethernet link, with approximately 20% CPU utilization on both ends.

   Example

```
TCP STREAM TEST to 192.168.10.21
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 87380  16384  16384    10.00         0.94   19.10    23.70    1.662   2.062
```

3. Run the test over the IPoIB interface, which was previously set up.

```
# netperf -c -C -f g -H 192.168.0.2
TCP STREAM TEST to 192.168.0.2
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 87380  16384  16384    10.01         1.21   33.88    87.35    2.290   5.905
```

The results in this example show approximately a 28% increase in throughput, but at the expense of higher CPU utilization on both the sender and receiver. This is because the native Ethernet card does TCP/IP check summing in hardware, while the IPoIB interface must use the host CPU.

Performing a latency test

To demonstrate the latency advantage of IB compared to Ethernet, use a netperf test called TCP request/response. This test will send a 1-byte request to the remote machine and the remote machine will issue a 1-byte response.

1. Develop a base case for comparison on *Server 1*. Add the **-t TCP_RR** option to the **netperf** command to specify this test.

Example

```
# netperf -c -C -f g -H 192.168.10.21 -t TCP_RR
```

2. Read the results. The sample results show performance of approximately 5800 request/response transactions per second.

Example

```
TCP REQUEST/RESPONSE TEST to 192.168.10.21
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
16384  87380  1       1      10.00   5787.80  4.50   7.30   7.775   12.619
```

3. Run the test over the IB interface on *Server 1*.

Example

```
netperf -c -C -f g -H 192.168.0.2 -t TCP_RR
TCP REQUEST/RESPONSE TEST to 192.168.0.2
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
16384  87380  1       1      10.00   11629.08  18.30  19.51  15.733  16.777
```

4. Compare the results. The IB interface shows approximately 11,600 request/response transactions per second, which is approximately double the performance of the Gigabit Ethernet interface.

SDP performance versus IPoIB using netperf

If you performed the steps in “IPoIB performance versus Ethernet using netperf” on page 64, you saw that it's difficult to take advantage of the high bandwidth of IB using IPoIB without sacrificing the CPU overhead associated with TCP/IP.

To solve the CPU overhead problem, the SDP can be used over the fabric. The SDP protocol sets up a reliable connection over the IB fabric, and TCP socket connections can be made without the overhead of TCP. RDMA semantics are used in the protocol, which essentially transmits data between the two host's buffers without CPU intervention.

Configuring SDP

The decision to use this protocol rather than setting up a normal TCP socket is made at the kernel level. Applications do not have to be re-written or re-compiled to take advantage of this capability. The decision to use this protocol rather than setting up a normal TCP socket is made at the kernel level.

There are a variety of methods to control how connections are configured to use SDP, as documented in /usr/local/topspin/etc/libsdp.conf.

1. Make sure processes include the SDP library when they load.

   The /etc/ld.so.preload file tells the system's dynamic linker to load the SDP library when processes are started.

   a. Create the **/etc/ld.so.preload** file if the file does not exist.

   b. Add the following line to **/etc/ld.so.preload** on both systems:

      /lib/libsdp_sys.so

2. Stop the existing netserver daemon, which expects TCP connections over a normal network socket, by stopping the netserver daemon on *Server 2,* using the **killall** command.

Example

```
# killall netserver
```

Performing a throughput test

1. Tell the SDP library that the next process should use SDP, and start the netserver process on *Server 2*:

Example

```
# netserver.sdp
```

2. Run the netperf SDP throughput test on *Server 1*.

Example

```
# netperf.sdp -c -C -f g -H 192.168.0.2
Recv   Send    Send                          Utilization       Service Demand
Socket Socket  Message  Elapsed              Send     Recv     Send    Recv
Size   Size    Size     Time     Throughput  local    remote   local   remote
bytes  bytes   bytes    secs.    10^9bits/s  % T      % T      us/KB   us/KB
 65535  65535  65535    10.00         1.94   36.70    57.90    1.551   2.446
```

3. Read the test results.

   The throughput has increased approximately 50% from using IPoIB, and the CPU utilization has been significantly reduced.

Performing a latency test

In addition to the throughput test, you can also test the effect of using SDP on the request/response test:

Example

```
# netperf.sdp -c -C -f g -H 192.168.0.2 -t TCP_RR
TCP REQUEST/RESPONSE TEST to 192.168.0.2
Local /Remote
Socket Size   Request Resp.  Elapsed Trans.   CPU    CPU    S.dem   S.dem
Send   Recv   Size    Size   Time    Rate     local  remote local   remote
bytes  bytes  bytes   bytes  secs.   per sec  % T    % T    us/Tr   us/Tr
65535  65535  1       1      10.00   17145.82  15.00  17.10  8.747   9.976
```

In the example above, there is approximately a 50% increase in transactions per second from the IPoIB case. In addition, there is a reduction in CPU utilization on both the transmit and receive end.

11

Configuring Oracle RAC™ with InfiniBand

Purpose

This chapter is intended to provide the reader with general instructions on how to:

- install a cluster
- install Oracle Real Application Clusters (RAC) (Version 10.1.0.2.0)
- configure a cluster database on Linux over IB

For further information on the benefits of Oracle RAC over IB, refer to http://www.topspin.com/solutions/pdf/RDMA White Paper.pdf.

Supporting documentation

This document is not intended to replace the Oracle installation guide, but should be used as a supplement to configure RAC to work with IB.

IB documentation can be located at:

www.topspin.com/support

The following documents may be referenced for further information:

- *Host Channel Adapter User Guide*
- *Ethernet Gateway User Guide*
- *Fibre Channel Gateway User Guide*
- *Element Manager User Guide*

Configuring the cluster hardware

The following example shows a Topspin 360/Cisco SFS 3012 with FC and Ethernet gateways, servers with HCAs, FC storage and an Ethernet router.

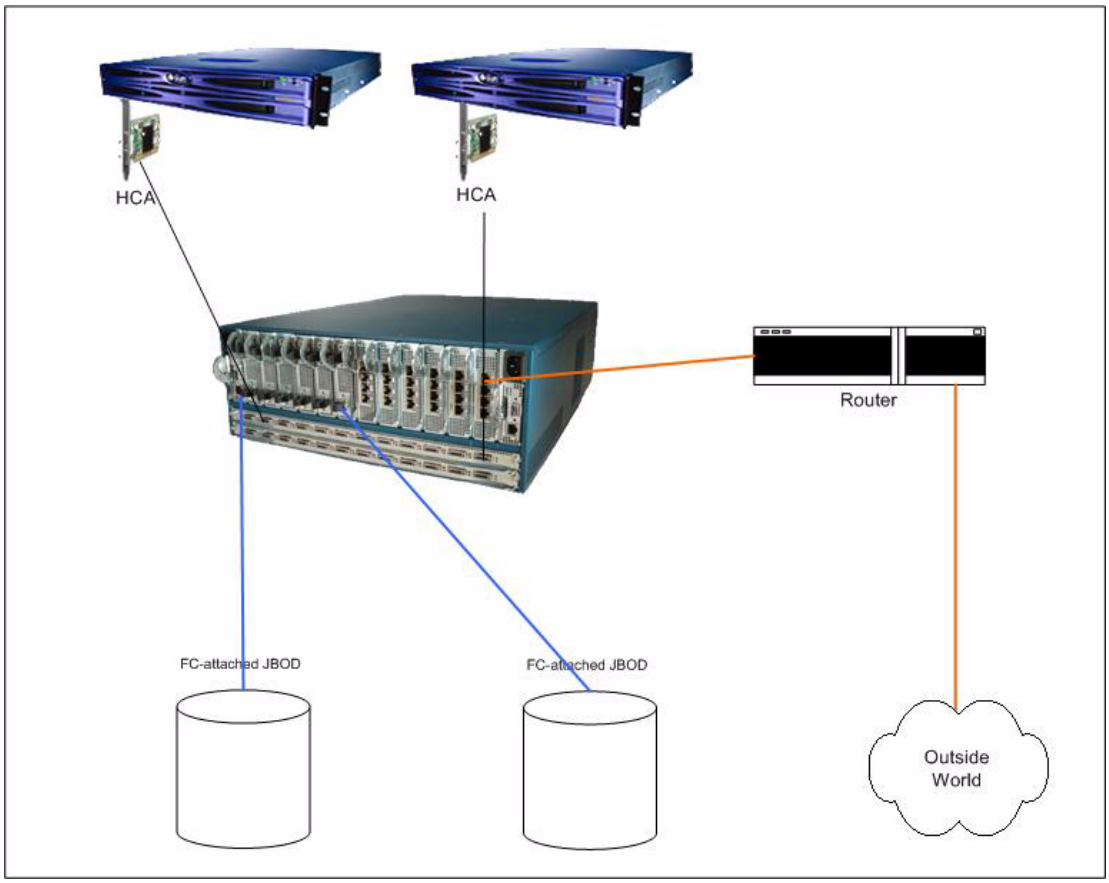

Figure 11-1: Sample Cluster Topology

System requirements

Check the RAC/Linux certification matrix for information on currently supported hardware/software.

Hardware requirements

- Refer to the RAC/Linux certification matrix for information on supported configurations
- For RAC over IB, an IB switch is required for network connectivity.

  Back-to-back connections between two hosts over an HCA is not a supported configuration.
- FC gateway
- (optional) Ethernet gateway (an Ethernet gateway is used in the configuration example for this chapter)
- Ensure that the hosts have at least the following resources:
  - 400 MB in /tmp
  - 2 GB of physical memory (RAM) (4 GB recommended)
  - 2 GB swap space

Configuring the InfiniBand fabric

Refer to the *Element Manager User Guide*.

Installing and configuring the HCAs

Refer to the *Host Channel Adapter Guide*. Once the HCA is installed and configured, all of the drivers will be installed as well.

Understanding the use of the HCA drivers

The following diagrams depict the connection between the Oracle components and the IB drivers.

- IPoIB - IP over IB driver
- CRS - Cluster Ready Services
- DB - database
- uDAPL - User Direct Access Programming Library driver
- ASM - Automatic Storage Management
- SRP - SCSI RDMA Protocol driver
- OCFS - Oracle Clustered File System
- SDP - Socket Direct Protocol driver

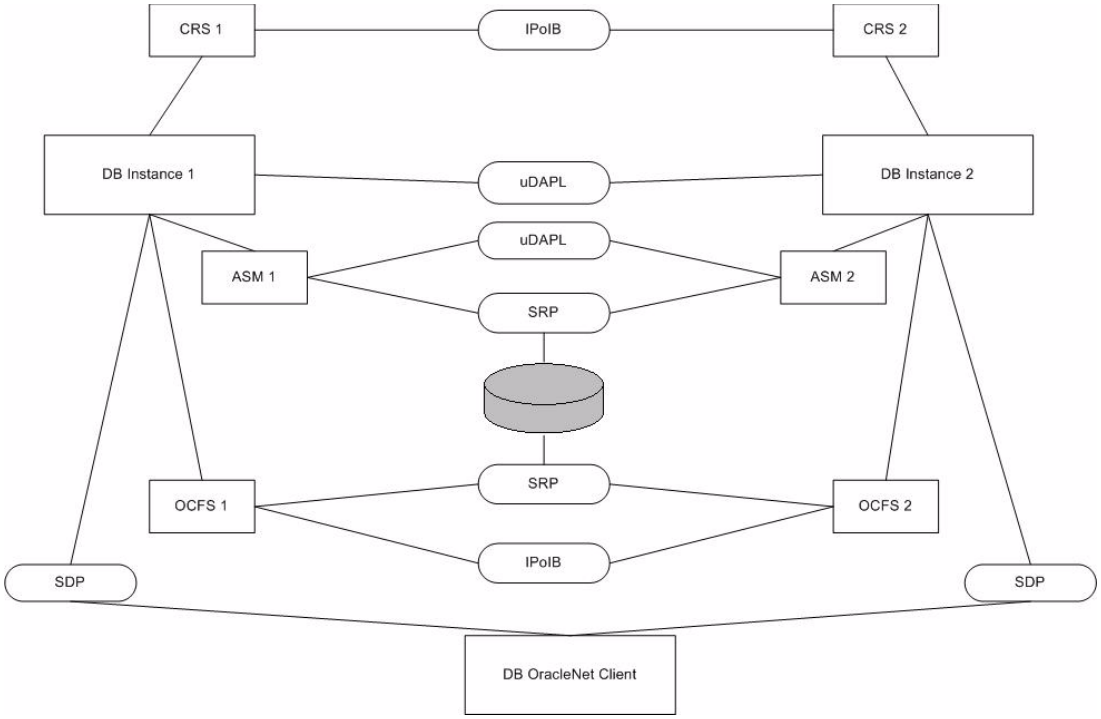

Figure 11-2: Oracle Components and IB Driver Connections

Configuring Fibre Channel storage

Refer to the *Fibre Channel User Guide* to configure the FC gateway. Oracle Cluster File system and Oracle Automatic Storage Management can be used over IB.

Configuring software

For RAC on Linux support, consult the operating system vendor and refer to the RAC/Linux certification matrix.

NOTE: All necessary drivers have been already been installed and configured with the HCA procedure.

Installing packages

1. Verify that you have make and rsh-server packages installed.

```
$rpm -q rsh-server make
rsh-server-0.17-5
make-3.79.1-8
```

2. If make and rsh-server packages are not installed, use the package manager of your choice to install them.

Installing patches

10g RAC over IB is supported only on RHEL 3.0. Consult with your operating system vendor to get the latest patch version of the kernel.

Installing the shared disk subsystem

- Installation of the disk subsystem is highly dependent on the subsystem you have chosen. If you choose not to use IB for your shared subsystem configuration, please refer to your hardware documentation for installation and configuration instructions on Linux as additional drivers and patches may be required.
- If you chose not to use IB for your FC storage, install host adapters in your cluster nodes, if you have not already done so. For information on installing host adapters, refe to the documentation that shipped with your host adapters and node hardware.
- For the configuration example in this document, we assume that the shared disk subsystem that is not using SRP is installed correctly, and that the shared disks are visible to all nodes in the cluster.
- If you choose to use raw over SRP:

  a. configure the FC gateway; and

  b. follow the usual steps to create a disk partition and map the partition to a raw device.

  For storage over IB, the fabric will need to configure the ib0 interface before setting up raw, OCFS, or ASM.
- If you are using ASM, there is no additional work required.
- If you are using OCFS, replace your load_ocfs script under /sbin with the patched load_ocfs script from the support site.

  OCFS depends on MAC address of the NIC or HCA. However, the MAC address of the HCA can be different across reboots, so the load_ocfs has been changed to incorporate the MAC address dependency.

Configuring cluster interconnect and public network

Each system will have at least IP addresses for the following:

- one address for the public network
- one address for the private cluster interconnect
- one address for a virtual IP

1. For the public network, get the addresses from your network manager.
2. For the private interconnect, use appropriate addresses.

   Host drivers require only one interface to be configured for all host drivers that you choose to use.
3. Configure the ib0 interface by creating a **ifcfg-ib0** file in /etc/sysconfig/network-scripts.

   The following is a sample script:

```
DEVICE=ib0
BOOTPROTO=static
IPADDR=192.168.5.6
NETMASK=255.255.255.0
ONBOOT=yes
```

4. Add all addresses into the file **/etc/hosts**.

   ```
   [oracle@opcbrh1 oracle]$ more /etc/hosts
   ```

```
9.25.120.143    rac1        #Oracle 10g Rac node 1 - public network
9.25.120.143    rac2        #Oracle 10g Rac node 2 - public network
1.1.1.1         rac1-ib0    #Oracle 10g Rac node 1 - interconnect, storage, CRS
daemon
1.1.1.2         rac2-ib0    #Oracle 10g Rac node 2 - interconnect, storage, CRS
daemon
```

5. Make sure that your public network is accessible through the Ethernet gateway. Refer to the *Ethernet Gateway User Guide* to configure the Ethernet gateway.
6. Make sure you can rcp to all nodes of the cluster over the ib0 interface.

   Interprocess communication is an important issue for RAC because cache fusion transfers buffers between instances using this mechanism. IPC on IB Oracle RAC 10g uses the uDAPL protocol. The uDAPL protocol will be available automatically when the rpms are installed and the ib0 interface is configured.

Installing the Oracle CRS daemon and 10g RAC

1. Install the Oracle CRS daemon and Oracle database in the usual way. However, make sure to select ib0 interface for the private interconnect.
2. Make a copy of **libskgxp10.so** in $ORACLE_HOME/lib directory and copy the new library you have received into that directory. Whenever you want to revert back to UDP you will need the original library. This might not be the actual way you do, it may be a patch install.

Configuring Oracle Net over SDP

Refer to the Oracle cookbook for SQL Net over SDP.

Troubleshooting SRP

For complete SRP installation instructions, refer to the *Host-Side Drivers User Guide*.

Checking the InfiniBand network interfaces

Check for IB network interfaces using the **ifconfig -a** command.

You should see interfaces that begin with ib (in other words, ib0, ib1).

Use the **ifconfig -a** command to display IB interfaces. If there are no ib0 and ib1 interfaces, you may create them automatically or manually. To create them automatically each time the server reboots, change the directory. The directory may be /etc/sysconfig/network-scripts.

Create one script per HCA port you wish to use (in other words, ifcfg-ib0, ifcfg-ib1). (You may copy another **ifcfg** file, modify the DEVICE and IPADDR lines, then save it as either **ifcfg-ib0** or **ifcfg-ib1**.)

To create it manually each time after booting the server, enter the **ifconfig** command with the following:

Example

```
ifconfig ib# addr netmask mask
```

- ib# is the HCA network interface getting the IP address. This may be either ib0 or ib1.
- addr is the IP address to assign the network interface.
- netmask is a mandatory keyword.
- mask is the netmask for the IP address.

Checking the HCA port status

Example

```
$ /usr/local/topspin/sbin/hca_self_test

---- Performing InfiniBand HCA Self Test ----
Number of HCAs Detected ................ 2
PCI Device Check ....................... PASS
Host Driver Version .................... rhel3-2.4.21-4.ELsmp-2.0.0-527 Host
Driver RPM Check ................... PASS HCA Type of HCA #0 .....................
CougarCub HCA Firmware on HCA #0 ................. v3.01.0000 build 2.0.0.527 HCA
Firmware Check on HCA #0 ........... PASS HCA Type of HCA #1
..................... Cougar HCA Firmware on HCA #1 ................. v3.01.0000
build 2.0.0.527 HCA Firmware Check on HCA #1 ........... PASS Host Driver
Initialization ............. PASS Number of HCA Ports Active ............. 1 Port
State of Port #0 on HCA #0 ........ UP Port State of Port #1 on HCA #0 ........
DOWN Port State of Port #0 on HCA #1 ........ DOWN Port State of Port #1 on HCA
#1 ........ DOWN Error Counter Check ..................... PASS Kernel Syslog
Check ................... PASS
------------------ DONE ---------------------
```

Verifying CRS and 10g RAC

Verify that CRS and 10g RAC are communicating over the IB fabric.

1. Reboot the server or restart CRS using the init scripts.

   Example

```
2004-04-13 17:03:14.815 [114696] >TRACE:   clsc_listen: (0x824bff8) Listening on
(ADDRESS=(PROTOCOL=tcp)(HOST=qa-rac3)(PORT=49895))

qa-rac3 equivalent in your system should be the ib0 interface.
```

2. Restart your Oracle instance. You should see the following in the alert log:

   Example

```
"cluster interconnect IPC version: Oracle UDAPL Apr 20 2004 17:08:19
IPC Vendor 1 proto 1 Version 1.0"
```

3. You can also verify the same by using the following from the sqlplus prompt and checking the udump directory for the proper trace file:

   Example

```
Oradebug setmypid
Oradebug ipc
```

Index

A

AF_INET_SDP 24

C

configure
  SDP 23

D

dapl
  directory 33
Discover LUNs 19, 20

F

failure
  glibc 29

G

glibc 29

I

IBM
  DB2 v8.1 sample config 26
introduction 1
IPoIB
  about 63
  configure for performance test 63
ITLs
  configuring 18

L

latency test
  IPoIB 57, 65
  SDP 59, 66
  uDAPL 34
LD_PRELOAD 25

N

netperf 62

P

performance test
  IPoIB 63
preload script 28

R

RDMA
  performance 34
  performance test 34
RDMA thru_client.x 34
RPMs 33
rsh 37

S

sample config
  IBM DB2 v8.1 26
sample topology
  database cluster 31
SDP
  configure 23
  glibc fails 29
  initialization script 28
  performance test 65
  vs IPoIB 65

T

TCP
    convert to SDP 24
TCP/IP checksumming 57, 64
throughput test
    IPoIB 56, 64
    SDP 58, 66
thru_server.x 34

U

uDAPL
    about 33
    application configuation 33
    sample make files 33
ULP
    performance test 61
upper layer protocols
    performance test 61