Oracle Maximum
Availability Architecture

An Oracle White Paper
October 2012

InfiniBand Network Troubleshooting Guidelines and Methodologies

ORACLE

EXECUTIVE OVERVIEW **3**

NETWORK BUILDING BLOCKS **4**

OPEN SYSTEMS INTERCONNECT (OSI) 4
NETWORK BUILDING HARDWARE 5
DEBUGGING AT HARDWARE LEVEL 5
NETWORK BUILDING SOFTWARE 6
INFINIBAND SUBNET MANAGER 7
INFINIBAND PARTITION 7
DEBUGGING AT SOFTWARE LEVEL 8

HEALTH CHECK GUIDELINES **9**

IMPORTANCE OF NTP SERVICE 9
LOG ROTATION AND ARCHIVES 9

CONFIGURATION BLUEPRINTS **10**

SAVING INFINIBAND TOPOLOGY 10
INFINIBAND DIAGNOSTICS TOOLS 12
Host Specific Commands *13*
Description of perfquery counters *16*
InfiniBand Switch Specific Commands *17*
GLOBAL INFINIBAND COMMANDS 21
ANALYSIS FROM IBDAGNET LOGS 23

NETWORK PACKET CAPTURES **26**

SCREENSHOTS OF IBDUMP PCAPS IN WIRESHARK 27
Node Description Query *27*
Performance Counter Query *28*
IPv4 ICMP (ping) on IPoIB *29*
SDP Handshakes *30*
ARP Request Failures *31*

TROUBLESHOOTING SCENARIOS **32**

USE CASE ANALYSIS #1 32
USE CASE ANALYSIS #2 32
USE CASE ANALYSIS #3 33
USE CASE ANALYSIS #4 33
USE CASE ANALYSIS #5 34
USE CASE ANALYSIS #6 35
USE CASE ANALYSIS #7 35
USE CASE ANALYSIS #8 36
USE CASE ANALYSIS #9 36
USE CASE ANALYSIS #10 36
USE CASE ANALYSIS #11 37
USE CASE ANALYSIS #12 38
USE CASE ANALYSIS #13 38

Executive Overview

Oracle Maximum Availability Architecture (MAA) is Oracle's best practices blueprint for implementing high availability. Network technologies are critical and key component to overall infrastructure of any modern computing environment. Oracle's Engineered Systems are built upon one of the most advance, high speed and efficient networking technologies like 10 Gigabit Ethernet and 40 Gigabit InfiniBand. This paper focuses on advanced network troubleshooting methodologies and best practice guidelines. It introduces various building blocks, common problem areas, diagnostics commands with brief descriptions, and sample outputs. It also contains a set of important scenarios with problem statements and a methodical approach to troubleshoot towards possible root causes.

Network Building Blocks

Before explaining network troubleshooting methodologies, a discussion about what components build a network can serve as background of further exploration. At high level, there are two categories: *software* and *hardware*. Each category can be further broken down into sub categories and products. It is essential to have a good understanding of these different functional areas prior to dealing with problems and attempting a diagnosis. Our ultimate goal is to find the root cause of a problem and so that you can resolve the issue.

In complex environments, a problem originating in a specific area can remain invisible until its higher level impact is visible from another area. All levels within hardware and software interoperate with each other to deliver desired functionalities. Open Systems Interconnect (OSI) model is a key while we traverse through different layers of hardware, firmware and software troubleshooting by analyzing different protocols and messages.

Open Systems Interconnect (OSI)

In the OSI model diagram below, you can see that lower three layers (1-3) have close association with the hardware and operating systems' networking driver stack, whereas the upper four layers (4-7) have more close association with applications. This knowledge is critical to remember while you troubleshoot and perform analysis at various phases. Problems at lower levels are usually inherited by upper levels but not vice versa.

OSI Layers	OSI Model			Example
7	Host Layer	Data	Application	Web Server
6			Presentation	SSL
5			Session	IPC
4		Segment	Transport	UDP, TCP, SDP, RDS
3	Media Layer	Packets	Network	IP Addressing in v4/v6 format
2		Frames	Data Link	MAC, GUID
1		Bits	Physical	Network Hardware

This document will be more focused on lowest three layers (1-3) then the others. These three layers define the media layer which deals with hardware, firmware, operating systems' network stack, and most importantly the Internet Protocol. What

happens in the upper four layers is difficult to generalize, and usually the areas are very specific to applications and their usage models.

Network Building Hardware

The network lets computing platforms exchange information over certain protocols. In other words, the network binds these independent computing platforms together to form a distributed computing environment. Each participating end point requires some kind of network hardware such as an InfiniBand HCA installed in a PCIe expansion slot or a Gigabit Ethernet port on its motherboard. A port or receptacle on such a hardware device connects to outside world using an appropriate cable. A switch is connected to the other end of the cable. Similarly, other hosts will also connect, and this scenario creates a commonly seen *star topology* at the physical layer. In some rare cases, a network port on a host may connect directly with another host's network interface port creating a point-to-point topology.

From a troubleshooting perspective it is important to understand these hardware layers, as sometimes the problems may be hidden in these areas. Quite different from the software, some of the hardware is mechanically sensitive, and any kind of damage can interrupt the functionality. When such problems are present in a system, they may not be immediately visible to an end user, but as a consequence it can trigger an error state in the application layers. You wouldn't be physically watching the hardware for a bent pin or broken connector or overheated section inside a rack all the time! The computing infrastructure is designed to deliver a set of services out of highly complex software running on an equally complex set of hardware.

Debugging at hardware level

Usually the troubleshooting phase does not begin here, but it is an important checklist item when things are not resolved by running diagnostic commands. We will go into more details later in this document, but now we will quickly look at what can be done in the hardware layers in order to find the cause of a problem, or otherwise confirm that our hardware is working as expected. Some of the hardware checks require a physical presence around the installation area, while others may be checked using sensors and ILOM plug-ins.

Cables connect two end points forming up a smallest link, while an end-to-end software application path may involve one or more such links. We usually deal with copper or optical fiber cables attached to hosts using associated connectors. An RJ45 type Ethernet cable may be one of the simplest cables as compared to the specialty InfiniBand cables.

A problem may be simply reported as a failure to ping between two hosts. With all diagnosis done at Operating System and application levels, you may discover that the Ethernet cable slipped out of its port receptacle due to broken latch, one of the wires came loose due to a bad crimp, or the cable was plugged into a wrong port. All cables are built around industry standard specifications. Ethernet cables may be more robust physically, but InfiniBand copper cables have strict specifications which define the minimum bend radius that they can tolerate. During physical handling, if these are bending with a lower radius then it can cause internal damage leading to errored conditions in the infrastructure.

It is fair to expect some periodic maintenance or update work on our installed hardware systems. During such physical maintenance, a network interface card such as IB-HCA may be displaced in its physical PCIe slot. While it may not appear to be faulty, this will cause disruption in functionality. A displaced card in the expansion slot may be a very simple problem to fix, but what if someone dislodged a small component from the PCB while doing upgrades or maintenance work. Depending on what came off, the card could become completely non-functional, continue to work with some performance impact or reduced life time, or in the worst case there is data corruption.

Another hardware area to inspect is the actual port receptacles where cables are seated. InfiniBand cables need to be handled properly while seating or re-seating in ports. Due to the nature of mechanical parts, improper handling can lead to damage to the receptacle port or cable connector. Often times these ports and connectors may not latch properly and stay loose, which can also result in errors or disruption of functionality. This also applies to network switch hardware as well as host's network interfaces.

Each network interface device has its own specific firmware which interacts with operating system-based drivers for proper enablement of functionality. Network switches also have certain active hardware components which require their own firmware. In the context of this document, it is important to understand the firmwares' presence in the environment so that we can ensure the correct versions are installed. This should be one of the check-list items in the troubleshooting phase. In context of networking inside Oracle's Engineered Systems, we have firmware for IB HCA and 10GbE NIC. Oracle InfiniBand switches have a few components inside which require a firmware, e.g. InfiniScale-IV and Bridge-X, but these are managed using an overall firmware bundle package instead of independent firmwares for easy management. Some of the specialized InfiniBand active optical cables may also need their own firmware. These are usually hard burnt in component's respective PROMs.

Hardware faults can be complex, and in order to reach a complete root cause analysis, sometimes we may need specialized diagnostic tools and equipment. With limited scope of this guide book, we will not go into those complex areas. When a troubleshooting flow brings us in this direction and the problem is not recoverable at the site of actual installation, then we probably need to replace the hardware. Faulty hardware can be further diagnosed and analyzed in engineering labs with proper facilities.

Fault identification poses a challenge during diagnosis and troubleshooting phase. It is important to run through a hardware validation checklist at the appropriate time during a troubleshooting phase. A validation is always made against a pre-determined and expected result.

Network Building Software

Just like we have software to deliver the desired business services, at the infrastructure level we have networking software. First in this category is a device driver. Device drivers are very specific to the hardware that needs to be enabled in an operating system. They create a binding between respective hardware and upper-layer protocols and applications.

To illustrate an example, we cannot see the interface named 'eth0' in Linux without a proper device driver installed in the operating system. A list of such device drivers can be seen in the output of Linux command `lsmod`. For InfiniBand, such device drivers are part of a software package called *Open Fabrics Enterprise Distribution* or *OFED*. This package also includes various configuration utilities, diagnostics tools, APIs, libraries, test tools, and service binaries. OFED is very tightly integrated with the kernel and operating system's legacy network stack. It leverages several critical components like TCP/IP drivers, configuration, and analysis tools.

Next in this category is the software that drives network switches. This discussion may overlap with firmware, but it is important to understand what lies under the hood driving switches. A firmware is simply a locked-down black box version of software. On Ethernet and InfiniBand, both switches run on their respective firmwares and expose a user interface for administration and configurations. The InfiniBand switch software also contains a customized version of OFED allowing you to have fabric-level management capabilities. One of the most critical components in the InfiniBand switch is the *Subnet Manager* which is also known as OpenSM or Open Source Subnet Manager. This will be discussed in more detail in a later section. If the switch has any specialized hardware, then it will also have an associated software component. For example, the

InfiniBand Gateway Switch includes Bridge-X devices for virtual Ethernet over InfiniBand functions. So its software, or shall we say firmware, also includes Bridge-X Manager or BXM.

InfiniBand Subnet Manager

The Subnet Manager is the most critical piece of software in any InfiniBand network. In Oracle's Engineered Systems, it runs on a set of InfiniBand switches with the highest firmware revision for compatibility reasons. One of the most visible functions of the subnet manager is to assign Local Identifiers to all entities in a fabric and create a routing table for them. Please note that this is not a layer-3 routing table as per the OSI model but remains at layer-2. Virtual Lane 15 (VL15) is used to exchange management packets over an unreliable datagram service. After an initial discovery and activation of the fabric, the subnet manager periodically scans and updates the information if needed. For high availability and redundancy reasons, there can be multiple instances of the subnet manager. Only one instance takes ownership of master role, while all others remain on standby. The configuration file for the subnet manager takes a few user configurable options through a command line interface. Switches designated as spine are set with higher priority as compared to the leaf switches. Controlled handover is set to *true* on all InfiniBand switches. When two or more switches have same priority then the master role preference is given to the switch with lower GUID.

The subnet manager also log events periodically based on logging level 3 set in the configuration. Some of the messages are self explanatory while others may be cryptic. You can use the command `showsmlog` to display the subnet manager logs. Table 1 describes a few of the error codes.

TABLE 1. SUBNET MANAGER ERROR CODES

SUBNET MANAGER ERROR CODE	DESCRIPTION
ERR 1B11	Indicates failure to create a new multicast group due to missing parameters in the join request. It is common to see these messages during a subnet initialization and activation phase. After a successful retry, these events are not logged.
ERR 1B10	Indicates a failure because the requester does not want to be a full member.
ERR 5411 and ERR 3113	Indication of a message transfer timeout. These types of error messages represent a single event and usually appear in pair. This message types are used for SM-SM synchronization, for instance standby SM doing master SM alive checking.
ERR 3315	Unknown remote side for node due to unresponsive SMA.
ERR 3809	Failed to find source physical port for trap received.
ERR 7502	Indicates inability to locate port object by lid.
ERR 7503	LID is out of range due to stale cache information.

InfiniBand Partition

The InfiniBand partition is closely managed with subnet manager. It defines logical groups of nodes within a fabric. Think of a multi-homed computing platform to understand InfiniBand partitions. In the Ethernet world, InfiniBand partitioning is

very similar to VLANs. By adding an InfiniBand HCA to multiple partitions, you enable its capability to communicate over different network paths. The entire configuration is performed at the master subnet manager which can propagate the final committed configuration to all standby switches. First, a partition key must be defined and then you add HCA port GUIDs as members. There are two types of memberships: Limited and Full.

TABLE 2. INFINIBAND PARTITION MEMBERSHIP

INFINIBAND PARTITION MEMBERSHIP	ROLE AND DESCRIPTION
Limited	Allowed to communicate to full members in same partition and not with another limited member.
Full	Allowed to communicate to all members within same partition.
Both	Special membership for Oracle Virtual Server (OVS) nodes to allow limited and full memberships to guest virtual machines hosted on its resources.

Debugging at Software Level

Most of the time, the diagnosis and troubleshooting phase begin at the software level. These cases could be as simple as a user error, a syntax error in configuration, or they can be more challenging to debug as their actual cause may be elsewhere and effects can propagate to other areas. Sometimes these issues are not confined to a certain area and you need to account for interoperability challenges. Such troubleshooting scenarios are discussed in more detail throughout this document.

Before you evaluate various commands and troubleshooting scenarios, it's important to understand some of the key software functions that build an InfiniBand fabric. The following sections review subnet manager and partitions.

It's important to make sure you have the correct version of software for proper compatibility and interoperability of components in a network infrastructure.

Health Check Guidelines

At a very high level, there are two types of diagnostic commands. Some return a Boolean style result and others are very subjective in nature. The first category is easy and straightforward. These outputs are interpreted as pass or fail without much complex analysis. For example, `sminfo` returns information about the master subnet manager. Depending on the case, this output may be sufficient, but you may need to further analyze if the returned information about the master subnet manager is as expected or not.

The following sections emphasize configuration blueprints. That simply means that you need to have a pre-determined state of the network configuration against which you can cross check the current state during a troubleshooting phase.

When a problem arises, it is important to have a good knowledge about and familiarity with different diagnostic tools and commands available for the technologies in use, and it's a mix of various technologies in our environments, including InfiniBand and 10GbE for Layer 1-2, TCP/IP for Layer 3-4. Hardware components have their own set of commands as well. Once you have such knowledge, you can apply and categorize it while troubleshooting in a methodical approach.

Importance of NTP service

NTP is a time synchronization service and it is very important to have this in place right from the beginning. We always have several components in a clustered computing environment and they all should synchronize their system clocks with a common source. During troubleshooting, analyze and co-relate events logged in these individual components. Often times these event logs are within milliseconds, so the order and sequence plays a very critical role.

Log rotation and archives

All hosts and switches implement a mechanism to rotate log files on a periodic basis. The computing hosts may have several past archives, but the NM2 switches only hold one last archive due to disk space limitations. It may be useful to copy the log files outside of NM2 during a troubleshooting phase. This can be done periodically by using secure copy (`scp`) or similar from an external machine, a cronjob from external machine, etc.

Configuration Blueprints

Blueprints are specific and may be unique to your installation. It is critical to have such configuration blueprints in order to cross-check and validate a snapshot against an expected configuration. Several commands used in troubleshooting produce very subjective outputs, and these need to be validated against a pre-determined and known state. If you are using a default configuration as delivered from factory, then such blueprints may already be available. However, in most cases, you have some deviation from a standard factory configuration. IP addresses, hostnames, and external connections are the most common examples.

It is our recommendation and best practice to save a known good state of your configuration after first installation. These become very useful later when a troubleshooting need arises.

Saving InfiniBand Topology

We will provide a basic scenario on how to save critical information on InfiniBand fabric.

iblinkinfo.pl - report link info for all links in the fabric

```
[root@scab01db01 ~]# iblinkinfo.pl -R
Switch 0x0021283a866aa0a0 SUN DCS 36P QDR scab01sw-ib3:
          18    1[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>             [  ] "" ( )
          18    2[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      41    1[  ] "scab01cel01 C 192.168.40.29 HCA-1" ( )
          18    3[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      42    2[  ] "scab01cel04 C 192.168.75.14 HCA-1" ( )
          18    4[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      46    2[  ] "scab01cel03 C 192.168.40.31 HCA-1" ( )
          18    5[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      32    2[  ] "scab01cel06 C 192.168.40.34 HCA-1" ( )
          18    6[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      36    2[  ] "scab01cel05 C 192.168.40.33 HCA-1" ( )
          18    7[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       2    2[  ] "scab01db01 S 192.168.40.21 HCA-1" ( )
          18    8[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      25    2[  ] "scab01cel07 C 192.168.40.35 HCA-1" ( )
          18    9[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      33    2[  ] "scab01db03 S 192.168.40.23 HCA-1" ( )
          18   10[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      52    2[  ] "scab01db02 S 192.168.40.22 HCA-1" ( )
          18   11[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>             [  ] "" ( )
          18   12[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      31    2[  ] "scab01db04 S 192.168.40.24 HCA-1" ( )
          18   13[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   14[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   14[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   13[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   15[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   16[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   16[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   15[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   17[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   18[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   18[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   17[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   19[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      19    1[  ] "scab01cel13 C 192.168.40.41 HCA-1" ( )
          18   20[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      21    1[  ] "scab01cel14 C 192.168.40.42 HCA-1" ( )
          18   21[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      29    1[  ] "scab01cel11 C 192.168.75.21 HCA-1" ( )
          18   22[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      26    1[  ] "scab01cel12 C 192.168.40.40 HCA-1" ( )
          18   23[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      43    1[  ] "scab01cel09 C 192.168.40.37 HCA-1" ( )
          18   24[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      23    1[  ] "scab01cel10 C 192.168.40.38 HCA-1" ( )
          18   25[  ] ==( 4X 2.5 Gbps   Down/Disabled)==>             [  ] "" ( )
          18   26[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      39    1[  ] "scab01cel08 C 192.168.40.36 HCA-1" ( )
          18   27[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      11    1[  ] "scab01db06 S 192.168.40.26 HCA-1" ( )
          18   28[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      10    1[  ] "scab01db07 S 192.168.40.27 HCA-1" ( )
          18   29[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>             [  ] "" ( )
          18   30[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      48    1[  ] "scab01db05 S 192.168.40.25 HCA-1" ( )
          18   31[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       1   31[  ] "SUN DCS 36P QDR scab01sw-ib2" ( )
          18   32[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      38    2[  ] "scab01cel02 C 192.168.40.30 HCA-1" ( )
          18   33[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       5    1[  ] "MT25408 ConnectX Mellanox Technologies" ( )
          18   34[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      16    2[  ] "MT25408 ConnectX Mellanox Technologies" ( )
          18   35[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       7    2[  ] "MT25408 ConnectX Mellanox Technologies" ( )
          18   36[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      14    1[  ] "MT25408 ConnectX Mellanox Technologies" ( )
Switch 0x002128469727a0a0 SUN DCS 36P QDR scab01sw-ib2:
           1    1[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>             [  ] "" ( )
           1    2[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      45    2[  ] "scab01cel01 C 192.168.40.29 HCA-1" ( )
           1    3[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       9    1[  ] "scab01cel04 C 192.168.75.14 HCA-1" ( )
           1    4[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      35    1[  ] "scab01cel03 C 192.168.40.31 HCA-1" ( )
           1    5[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      13    1[  ] "scab01cel06 C 192.168.40.34 HCA-1" ( )
           1    6[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       4    1[  ] "scab01cel05 C 192.168.40.33 HCA-1" ( )
           1    7[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       8    1[  ] "scab01db01 S 192.168.40.21 HCA-1" ( )
```

```
           1    8[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      34    1[  ] "scab01cel07 C 192.168.40.35 HCA-1" ( )
           1    9[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      28    1[  ] "scab01db03 S 192.168.40.23 HCA-1" ( )
           1   10[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      51    1[  ] "scab01db02 S 192.168.40.22 HCA-1" ( )
           1   11[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>            [  ] "" ( )
           1   12[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       3    1[  ] "scab01db04 S 192.168.40.24 HCA-1" ( )
           1   13[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   14[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   14[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   13[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   15[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   16[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   16[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   15[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   17[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   18[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   18[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   17[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   19[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      20    2[  ] "scab01cel13 C 192.168.40.41 HCA-1" ( )
           1   20[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      22    2[  ] "scab01cel14 C 192.168.40.42 HCA-1" ( )
           1   21[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      30    2[  ] "scab01cel11 C 192.168.75.21 HCA-1" ( )
           1   22[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      27    2[  ] "scab01cel12 C 192.168.40.40 HCA-1" ( )
           1   23[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      44    2[  ] "scab01cel09 C 192.168.40.37 HCA-1" ( )
           1   24[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      24    2[  ] "scab01cel10 C 192.168.40.38 HCA-1" ( )
           1   25[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      37    2[  ] "scab01db08 S 192.168.40.28 HCA-1" ( )
           1   26[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      40    2[  ] "scab01cel08 C 192.168.40.36 HCA-1" ( )
           1   27[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      17    2[  ] "scab01db06 S 192.168.40.26 HCA-1" ( )
           1   28[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      12    2[  ] "scab01db07 S 192.168.40.27 HCA-1" ( )
           1   29[  ] ==( 4X 2.5 Gbps   Down/ Polling)==>            [  ] "" ( )
           1   30[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>       6    2[  ] "scab01db05 S 192.168.40.25 HCA-1" ( )
           1   31[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      18   31[  ] "SUN DCS 36P QDR scab01sw-ib3" ( )
           1   32[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      15    1[  ] "scab01cel02 C 192.168.40.30 HCA-1" ( )
           1   33[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      49    1[  ] "MT25408 ConnectX Mellanox Technologies" ( )
           1   34[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      50    1[  ] "MT25408 ConnectX Mellanox Technologies" ( )
           1   35[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      47    2[  ] "MT25408 ConnectX Mellanox Technologies" ( )
           1   36[  ] ==( 4X 10.0 Gbps Active/  LinkUp)==>      53    2[  ] "MT25408 ConnectX Mellanox Technologies" ( )
```

ibdiagnet –ls 10 –lw 4x –vlr

It is very important to save /tmp/ibdiagnet.* after this run.

ibnetdiscover - discover InfiniBand topology

```
vendid=0x2c9
devid=0x673c
sysimgguid=0x2c903000a740f
caguid=0x2c903000a740c
Ca      2 "H-0002c903000a740c"          # "scac01cel14 C 192.168.40.92 HCA-1"
[2](2c903000a740e)      "S-002128468d20a0a0"[20]                # lid 113 lmc 0 "SUN DCS 36P
QDR scac01sw-ib2 10.133.40.117" lid 14 4xQDR
[1](2c903000a740d)      "S-002128468d57a0a0"[20]                # lid 112 lmc 0 "SUN DCS 36P
QDR scac01sw-ib3 10.133.40.118" lid 33 4xQDR

vendid=0x2c9
devid=0x673c
sysimgguid=0x2c903000a77b3
caguid=0x2c903000a77b0
Ca      2 "H-0002c903000a77b0"          # "scac01cel13 C 192.168.48.253 HCA-1"
[2](2c903000a77b2)      "S-002128468d20a0a0"[19]                # lid 58 lmc 0 "SUN DCS 36P
QDR scac01sw-ib2 10.133.40.117" lid 14 4xQDR
[1](2c903000a77b1)      "S-002128468d57a0a0"[19]                # lid 57 lmc 0 "SUN DCS 36P
QDR scac01sw-ib3 10.133.40.118" lid 33 4xQDR

vendid=0x2c9
devid=0x673c
sysimgguid=0x2c903000a73bb
caguid=0x2c903000a73b8
Ca      2 "H-0002c903000a73b8"          # "scac01db04 S 192.168.10.103 HCA-1"
[1](2c903000a73b9)      "S-002128468d20a0a0"[12]                # lid 17 lmc 0 "SUN DCS 36P
```

```
QDR scac01sw-ib2 10.133.40.117" lid 14 4xQDR
[2](2c903000a73ba)      "S-002128468d57a0a0"[12]                # lid 35 lmc 0 "SUN DCS 36P
QDR scac01sw-ib3 10.133.40.118" lid 33 4xQDR
```

ibswitches - show InfiniBand switch nodes in topology

```
[root@scab01db01 ~]# ibswitches
Switch  : 0x0021283a866aa0a0 ports 36 "SUN DCS 36P QDR scab01sw-ib3.us.oracle.com" enhanced
port 0 lid 18 lmc 0
Switch  : 0x002128469727a0a0 ports 36 "SUN DCS 36P QDR scab01sw-ib2.us.oracle.com" enhanced
port 0 lid 1 lmc 0
```

ibhosts – displays hosts with channel adapters in InfiniBand fabric

```
[root@scab01db01 ~]# ibhosts
Ca      : 0x0021280001cef69a ports 2 "MT25408 ConnectX Mellanox Technologies"
Ca      : 0x0021280001cef60a ports 2 "MT25408 ConnectX Mellanox Technologies"
Ca      : 0x0021280001cef68e ports 2 "MT25408 ConnectX Mellanox Technologies"
Ca      : 0x0021280001cef626 ports 2 "MT25408 ConnectX Mellanox Technologies"
Ca      : 0x00212800013e6982 ports 2 "scab01cel02 C 192.168.40.30 HCA-1"
Ca      : 0x00212800013e69ee ports 2 "scab01db05 S 192.168.40.25 HCA-1"
Ca      : 0x00212800013e695e ports 2 "scab01db07 S 192.168.40.27 HCA-1"
Ca      : 0x00212800013e699a ports 2 "scab01db06 S 192.168.40.26 HCA-1"
Ca      : 0x00212800013e6986 ports 2 "scab01cel08 C 192.168.40.36 HCA-1"
Ca      : 0x00212800013e69a6 ports 2 "scab01db08 S 192.168.40.28 HCA-1"
Ca      : 0x00212800013e6966 ports 2 "scab01cel10 C 192.168.40.38 HCA-1"
Ca      : 0x00212800013e6a06 ports 2 "scab01cel09 C 192.168.40.37 HCA-1"
Ca      : 0x00212800013e696e ports 2 "scab01cel12 C 192.168.40.40 HCA-1"
Ca      : 0x00212800013e697e ports 2 "scab01cel11 C 192.168.75.21 HCA-1"
Ca      : 0x00212800013e698e ports 2 "scab01cel14 C 192.168.40.42 HCA-1"
Ca      : 0x00212800013e6962 ports 2 "scab01cel13 C 192.168.40.41 HCA-1"
Ca      : 0x00212800013e69ae ports 2 "scab01db04 S 192.168.40.24 HCA-1"
Ca      : 0x00212800013e69f2 ports 2 "scab01db02 S 192.168.40.22 HCA-1"
Ca      : 0x00212800013e69ca ports 2 "scab01db03 S 192.168.40.23 HCA-1"
Ca      : 0x00212800013e69fe ports 2 "scab01cel07 C 192.168.40.35 HCA-1"
Ca      : 0x00212800013e6a02 ports 2 "scab01cel05 C 192.168.40.33 HCA-1"
Ca      : 0x00212800013e697a ports 2 "scab01cel06 C 192.168.40.34 HCA-1"
Ca      : 0x00212800013e69d6 ports 2 "scab01cel03 C 192.168.40.31 HCA-1"
Ca      : 0x00212800013e695a ports 2 "scab01cel04 C 192.168.75.14 HCA-1"
Ca      : 0x00212800013e6a0e ports 2 "scab01cel01 C 192.168.40.29 HCA-1"
Ca      : 0x00212800013e6a16 ports 2 "scab01db01 S 192.168.40.21 HCA-1"
```

InfiniBand Diagnostics Tools

We can break down available InfiniBand commands into various levels. This will simplify our troubleshooting approach in the later sections. In general some of these are system-specific and some are global commands. System-specific commands can also be broken down by type of system, operating system, etc.

Host Specific Commands

A host is an entity where the InfiniBand link terminates from an overall packet flow perspective. Usually this host has a dual-port IB HCA. On Linux, the following commands are available to perform health checks of this end point.

lspci - utility for displaying information about all PCI buses in the system and all devices connected to them

```
[root@scae01cn01 ~]# lspci | grep -i InfiniBand
19:00.0 InfiniBand: Mellanox Technologies MT26428 [ConnectX VPI PCIe 2.0 5GT/s - IB QDR /
10GigE] (rev b0)

[root@scae01cn01 ~]# lspci -vvv -s 19:00.0
19:00.0 InfiniBand: Mellanox Technologies MT26428 [ConnectX VPI PCIe 2.0 5GT/s - IB QDR /
10GigE] (rev b0)
        Subsystem: Mellanox Technologies MT26428 [ConnectX VPI PCIe 2.0 5GT/s - IB QDR /
10GigE]
        Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr+ Stepping- SERR+
FastB2B-
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort-
>SERR- <PERR-
        Latency: 0, Cache Line Size: 256 bytes
        Interrupt: pin A routed to IRQ 32
        Region 0: Memory at df600000 (64-bit, non-prefetchable) [size=1M]
        Region 2: Memory at de800000 (64-bit, prefetchable) [size=8M]
        Capabilities: [40] Power Management version 3
                Flags: PMEClk- DSI- D1- D2- AuxCurrent=0mA PME(D0-,D1-,D2-,D3hot-,D3cold-)
                Status: D0 PME-Enable- DSel=0 DScale=0 PME-
        Capabilities: [48] Vital Product Data
        Capabilities: [9c] MSI-X: Enable+ Mask- TabSize=256
                Vector table: BAR=0 offset=0007c000
                PBA: BAR=0 offset=0007d000
        Capabilities: [60] Express Endpoint IRQ 0
                Device: Supported: MaxPayload 256 bytes, PhantFunc 0, ExtTag-
                Device: Latency L0s <64ns, L1 unlimited
                Device: AtnBtn- AtnInd- PwrInd-
                Device: Errors: Correctable+ Non-Fatal+ Fatal+ Unsupported-
                Device: RlxdOrd- ExtTag- PhantFunc- AuxPwr- NoSnoop-
                Device: MaxPayload 128 bytes, MaxReadReq 512 bytes
                Link: Supported Speed unknown, Width x8, ASPM L0s, Port 8
                Link: Latency L0s unlimited, L1 unlimited
                Link: ASPM Disabled RCB 64 bytes CommClk- ExtSynch-
                Link: Speed unknown, Width x8
        Capabilities: [100] Unknown (14)
```

ibv_devices - lists RDMA capable devices installed in the system

```
[root@scae01cn01 ~]# ibv_devices
    device                 node GUID
    ------              ----------------
    mlx4_0              0021280001a0a384
```

ibv_devinfo - query RDMA capable devices in the system

```
[root@scae01cn01 ~]#  ibv_devinfo
hca_id: mlx4_0
        transport:                      InfiniBand (0)
        fw_ver:                         2.7.8130
        node_guid:                      0021:2800:01a0:a384
        sys_image_guid:                 0021:2800:01a0:a387
        vendor_id:                      0x02c9
        vendor_part_id:                 26428
        hw_ver:                         0xB0
        board_id:                       SUN0170000009
        phys_port_cnt:                  2
                port:   1
                        state:                  PORT_ACTIVE (4)
                        max_mtu:                2048 (4)
                        active_mtu:             2048 (4)
                        sm_lid:                 61
                        port_lid:               53
                        port_lmc:               0x00
                        link_layer:             IB


                port:   2
                        state:                  PORT_ACTIVE (4)
                        max_mtu:                2048 (4)
                        active_mtu:             2048 (4)
                        sm_lid:                 61
                        port_lid:               54
                        port_lmc:               0x00
                        link_layer:             IB
```

mstflint - FW (firmware) burning and flash memory operations tool for Mellanox Infiniband HCAs and Ethernet NIC cards

```
[root@scae01cn01 ~]# mstflint -d mlx4_0 q
Image type:      ConnectX
FW Version:      2.7.8130
Device ID:       26428
Chip Revision:   B0
Description:     Node             Port1            Port2            Sys image
GUIDs:           0021280001a0a384 0021280001a0a385 0021280001a0a386 0021280001a0a387
MACs:                                 002128a0a384     002128a0a385
Board ID:         (SUN0170000009)
VSD:
PSID:            SUN0170000009
```

ibstat - queries basic status of InfiniBand devices installed in the system

```
[root@scae01cn01 ~]# ibstat
CA 'mlx4_0'
        CA type: MT26428
        Number of ports: 2
        Firmware version: 2.7.8130
        Hardware version: b0
        Node GUID: 0x0021280001a0a384
        System image GUID: 0x0021280001a0a387
        Port 1:
                State: Active
                Physical state: LinkUp
                Rate: 40
                Base lid: 53
                LMC: 0
                SM lid: 61
                Capability mask: 0x02510868
                Port GUID: 0x0021280001a0a385
                Link layer: IB
        Port 2:
                State: Active
                Physical state: LinkUp
                Rate: 40
                Base lid: 54
                LMC: 0
                SM lid: 61
                Capability mask: 0x02510868
                Port GUID: 0x0021280001a0a386
                Link layer: IB
```

ibstatus

```
[root@scae01cn01 ~]# ibstatus
Infiniband device 'mlx4_0' port 1 status:
        default gid:     fe80:0000:0000:0000:0021:2800:01a0:a385
        base lid:        0x35
        sm lid:          0x3d
        state:           4: ACTIVE
        phys state:      5: LinkUp
        rate:            40 Gb/sec (4X QDR)
        link_layer:      IB

Infiniband device 'mlx4_0' port 2 status:
        default gid:     fe80:0000:0000:0000:0021:2800:01a0:a386
        base lid:        0x36
        sm lid:          0x3d
        state:           4: ACTIVE
        phys state:      5: LinkUp
        rate:            40 Gb/sec (4X QDR)
        link_layer:      IB
```

perfquery - queries InfiniBand port's performance counters

```
[root@scae01cn01 ~]# perfquery
# Port counters: Lid 53 port 1
PortSelect:......................1
CounterSelect:...................0x0400
SymbolErrors:....................0
LinkRecovers:....................0
LinkDowned:......................0
RcvErrors:.......................0
RcvRemotePhysErrors:.............0
RcvSwRelayErrors:................0
XmtDiscards:.....................0
XmtConstraintErrors:.............0
RcvConstraintErrors:.............0
CounterSelect2:..................0x00
LinkIntegrityErrors:.............0
ExcBufOverrunErrors:.............0
VL15Dropped:.....................0
XmtData:.........................1915835586
RcvData:.........................4294967295
XmtPkts:.........................27742207
RcvPkts:.........................106026118
```

Description of perfquery counters

TABLE 3. PERFQUERY COUNTERS

PERFQUERY COUNTER	DESCRIPTION
SymbolErrorCounter	Total number of minor link errors detected on one or more physical lanes.
LinkErrorRecovery	Total number of times the Port Training state machine has successfully completed the link error recovery process.
LinkDownedCounter	Total number of times the Port Training state machine has failed the link error recovery process and downed the link.
PortRcvErrors	Total number of packets containing an error that were received on the port. • Local physical errors • Malformed data or link packet errors • Packets discarded due to buffer overrun
PortRcvRemotePhysicalErrors	Total number of packets marked with the EBP delimiter received on the port.
PortRcvSwitchRelayErrors	Total number of packets received on the port that were discarded because they could not be forwarded by the switch relay.
PortXmitDiscards	Total number of outbound packets discarded by the port because the port is down or congested. • Output port is not in the active state • Packet length exceeded Neighbor MTU

	• Switch Lifetime Limit exceeded • Switch HOQ Lifetime Limit exceeded This may also include packets discarded while in VL Stalled State.
PortXmitConstraintErrors	Total number of packets not transmitted from the switch physical port for the following reasons: • FilterRawOutbound is true and packet is raw • PartitionEnforcementOutbound is true and packet fails partition key check or IP version check
PortRcvConstraintErrors	Total number of packets received on the switch physical port that are discarded for the following reasons: • FilterRawInbound is true and packet is raw • PartitionEnforcementInbound is true and packet fails partition key check or IP version check
LocalLinkIntegrityErrors	The number of times that the count of local physical errors exceeded the threshold specified by LocalPhyErrors.
ExcessiveBufferOverRunErrors	The number of times that OverrunErrors consecutive flow control update periods occurred, each having at least one overrun error.
VL15Dropped	Number of incoming VL15 packets dropped due to resource limitations (e.g., lack of buffers) in the port.
PortXmitData	Total number of data octets, divided by 4, transmitted on all VLs from the port. This includes all octets between (and not including) the start of packet delimiter and the VCRC, and may include packets containing errors.
PortRcvData	Total number of data octets, divided by 4, received on all VLs at the port. This includes all octets between (and not including) the start of packet delimiter and the VCRC, and may include packets containing errors.
PortXmitPkts	Total number of packets transmitted on all VLs from the port. This may include packets with errors.
PortRcvPkts	Total number of packets, including packets containing errors and excluding link packets, received from all VLs on the port.
PortXmitWait	The number of ticks during which the port selected by PortSelect had data to transmit but no data was sent during the entire tick either because of insufficient credits or because of lack of arbitration.

InfiniBand Switch Specific Commands

env_test – Oracle InfiniBand Switch self diagnostic environment test. Performs test on various internal components and prints a report.

```
[root@scae01sw-ib02 ~]# env_test
Environment test started:
Starting Environment Daemon test:
Environment daemon running
Environment Daemon test returned OK
Starting Voltage test:
Voltage ECB OK
Measured 3.3V Main = 3.27 V
Measured 3.3V Standby = 3.37 V
Measured 12V = 11.90 V
Measured 5V = 5.02 V
```

```
Measured VBAT = 3.04 V
Measured 1.0V = 1.01 V
Measured I4 1.2V = 1.22 V
Measured 2.5V = 2.50 V
Measured V1P2 DIG = 1.17 V
Measured V1P2 ANG = 1.17 V
Measured 1.2V BridgeX = 1.22 V
Measured 1.8V = 1.78 V
Measured 1.2V Standby = 1.19 V
Voltage test returned OK
Starting PSU test:
PSU 0 present OK
WARNING PSU 1 present AC Loss
PSU test returned 1 faults
Starting Temperature test:
Back temperature 28
Front temperature 25
SP temperature 40
Switch temperature 53, maxtemperature 55
Bridge-0 temperature 42, maxtemperature 44
Bridge-1 temperature 44, maxtemperature 45
Temperature test returned OK
Starting FAN test:
Fan 0 not present
Fan 1 running at rpm 12426
Fan 2 running at rpm 12317
Fan 3 running at rpm 12317
Fan 4 not present
FAN test returned OK
Starting Connector test:
Connector test returned OK
Starting Onboard ibdevice test:
Switch OK
Bridge-0 OK
Bridge-1 OK
All Internal ibdevices OK
Onboard ibdevice test returned OK
Environment test FAILED
```

Listlinkup – Displays connector label to switch logical port information and link state.

```
[root@scae01sw-ib02 ~]# listlinkup
Connector  0A Present <-> Switch Port 20 up (Enabled)
Connector  1A Present <-> Switch Port 22 up (Enabled)
Connector  2A Not present
Connector  3A Not present
Connector  4A Not present
Connector  5A Not present
Connector  6A Present <-> Switch Port 35 up (Enabled)
Connector  7A Present <-> Switch Port 33 up (Enabled)
Connector  8A Present <-> Switch Port 31 up (Enabled)
Connector  9A Present <-> Switch Port 14 down (Enabled)
Connector 10A Present <-> Switch Port 16 up (Enabled)
Connector 11A Present <-> Switch Port 12 up (Enabled)
Connector 12A Present <-> Switch Port 18 down (Enabled)
```

```
Connector 13A Present <-> Switch Port 9 up (Enabled)
Connector 14A Present <-> Switch Port 7 up (Enabled)
Connector 15A Present <-> Switch Port 5 up (Enabled)
Connector 0A-ETH Present
  Bridge-0 Port 0A-ETH-1 (Bridge-0-2) up (Enabled)
  Bridge-0 Port 0A-ETH-2 (Bridge-0-2) up (Enabled)
  Bridge-0 Port 0A-ETH-3 (Bridge-0-1) down (Enabled)
  Bridge-0 Port 0A-ETH-4 (Bridge-0-1) down (Enabled)
Connector 1A-ETH Present
  Bridge-1 Port 1A-ETH-1 (Bridge-1-2) up (Enabled)
  Bridge-1 Port 1A-ETH-2 (Bridge-1-2) up (Enabled)
  Bridge-1 Port 1A-ETH-3 (Bridge-1-1) down (Enabled)
  Bridge-1 Port 1A-ETH-4 (Bridge-1-1) down (Enabled)
Connector  0B Present <-> Switch Port 19 up (Enabled)
Connector  1B Present <-> Switch Port 21 up (Enabled)
Connector  2B Not present
Connector  3B Not present
Connector  4B Not present
Connector  5B Not present
Connector  6B Present <-> Switch Port 36 up (Enabled)
Connector  7B Present <-> Switch Port 34 up (Enabled)
Connector  8B Present <-> Switch Port 32 up (Enabled)
Connector  9B Present <-> Switch Port 13 up (Enabled)
Connector 10B Present <-> Switch Port 15 up (Enabled)
Connector 11B Present <-> Switch Port 17 up (Enabled)
Connector 12B Present <-> Switch Port 11 up (Enabled)
Connector 13B Present <-> Switch Port 10 up (Enabled)
Connector 14B Present <-> Switch Port 8 up (Enabled)
Connector 15B Present <-> Switch Port 6 up (Enabled)
Connector 0B-FC Not present
Connector 1B-FC Not present
```

Version – Displays switch firmware version, serial number.

```
[root@scae01sw-ib02 ~]# version
SUN DCS gw version: 2.0.6-1
Build time: Jan 17 2012 14:29:13
FPGA version: 0x33
SP board info:
Manufacturing Date: 2010.05.05
Serial Number: "NCD4J0943"
Hardware Revision: 0x0006
Firmware Revision: 0x0102
BIOS version: NOW1R112
BIOS date: 04/24/2009
```

smpartition list active – Displays configured partition information.

```
[root@scae01sw-ib02 ~]# smpartition list active
# Sun DCS IB partition config file
# This file is generated, do not edit
#! version_number : 333
Default=0x7fff, ipoib :
ALL_CAS=full,
ALL_SWITCHES=full,
SELF=full;
SUN_DCS=0x0001, ipoib :
ALL_SWITCHES=full;
  = 0x8001,ipoib:
0x0021280001a0a5c6=both,
0x0021280001a0a5c5=both,
0x0021280001a0a36e=both,
0x0021280001a0a36d=both,
0x00212800013ea434=full,
0x00212800013ea433=full,
0x00212800013ea3ec=full,
0x00212800013ea3eb=full,
0x0021280001a0a30e=both,
0x0021280001a0a30d=both,
0x0021280001a0a392=both,
0x0021280001a0a391=both,
0x0002c903000a7776=both,
0x0002c903000a7775=both;
  = 0x8002,ipoib:
0x0021280001a0a5c6=both,
0x0021280001a0a5c5=both,
0x0021280001a0a36e=both,
0x0021280001a0a36d=both,
0x00212800013ea434=full,
0x00212800013ea433=full,
0x00212800013ea3ec=full,
0x00212800013ea3eb=full,
0x0021280001a0a30e=both,
0x0021280001a0a30d=both,
0x0021280001a0a392=both,
0x0021280001a0a391=both,
0x0002c903000a7776=both,
0x0002c903000a7775=both;
```

setsmpriority list – Displays SM priority, handover state, subnet prefix and M_Key value.

```
[root@scae01sw-ib02 ~]# setsmpriority list
Current SM settings:
smpriority 5
controlled_handover TRUE
subnet_prefix 0xfe80000000000000
M_Key None
```

Showsmlog – Displays subnet manager logs.

```
------------------------------------------------
OpenSM 3.2.6_20120116 - Oracle patch 10.5.2

Jul 13 04:47:22 649004 [B7EFB8D0] 0x80 -> OpenSM 3.2.6_20120116 - Oracle patch 10.5.2
Entering DISCOVERING state

Jul 13 04:47:22 684996 [B7EFB8D0] 0x80 -> Entering DISCOVERING state
Using default GUID 0x2128547f22c0a0
Entering STANDBY state

Jul 13 04:47:22 695994 [B7EFB8D0] 0x02 -> osm_vendor_bind: Binding to port 0x2128547f22c0a0,
class 129 version 1
Jul 13 04:47:22 722987 [B7EFB8D0] 0x02 -> osm_vendor_bind: Binding to port 0x2128547f22c0a0,
class 3 version 2
Jul 13 04:47:22 738984 [B7EFB8D0] 0x02 -> osm_console_init: Console listening on port 10000
Jul 13 04:47:22 965932 [B66F7B90] 0x80 -> Entering STANDBY state
Jul 14 00:00:47 841754 [B6EF8B90] 0x01 -> osm_vendor_send: ERR 5430: Send p_madw = 0x8244b48
of size 256 TID 0x455c540004639e failed -5
Jul 14 00:00:47 841754 [B6EF8B90] 0x01 -> vl15_send_mad: ERR 3E03: MAD send failed
(IB_UNKNOWN_ERROR)
Jul 20 19:26:18 802645 [B6EF8B90] 0x01 -> osm_vendor_send: ERR 5430: Send p_madw = 0x824cff8
of size 256 TID 0x455c540018e943 failed -5
Jul 20 19:26:18 803645 [B6EF8B90] 0x01 -> vl15_send_mad: ERR 3E03: MAD send failed
(IB_UNKNOWN_ERROR)
Jul 23 05:57:46 940382 [B6EF8B90] 0x01 -> osm_vendor_send: ERR 5430: Send p_madw = 0x824cff8
of size 256 TID 0x455c54001ef85f failed -5
Jul 23 05:57:46 941382 [B6EF8B90] 0x01 -> vl15_send_mad: ERR 3E03: MAD send failed
(IB_UNKNOWN_ERROR)
OpenSM: Got signal 15 - exiting...
Exiting SM

Jul 27 18:08:45 154750 [B7EFB8D0] 0x80 -> Exiting SM
```

Getmaster – Displays subnet manager master role information.

```
[root@scae01sw-ib02 ~]# getmaster
Local SM enabled and running
20120727 18:09:10 Master SubnetManager on sm lid 65 sm guid 0x212856d0a2c0a0 : SUN IB QDR GW
switch scae01sw-ib04 10.133.42.177 leaf:3
```

Global InfiniBand Commands

Global InfiniBand commands are those which can be run from any system participating in the network. These are usually provided through common software stack APIs. For example, OFED software package delivers a set of tools and commands for Linux-based compute nodes as well as Sun DCS QDR switches. Solaris-based hosts also provision a similar set of commands, but they are exactly the same in terms of usage and outputs.

ibdiagnet - ibdiagnet scans the fabric using directed route packets and extracts all the available information regarding its connectivity and devices.

There are multiple switches that you can use to query various type of information. Summaries are usually printed on the terminal itself, but the detailed logs are generated in the `/tmp` directory.

```
[root@scae01cn01 ~]# ibdiagnet -ls 10 -lw 4x
Loading IBDIAGNET from: /usr/lib64/ibdiagnet1.5.4
-W- Topology file is not specified.
    Reports regarding cluster links will use direct routes.
Loading IBDM from: /usr/lib64/ibdm1.5.4
-W- A few ports of local device are up.
    Since port-num was not specified (-p option), port 1 of device 1 will be
    used as the local port.
-I- Discovering ... 65 nodes (8 Switches & 57 CA-s) discovered.


-I---------------------------------------------------
-I- Bad Guids/LIDs Info
-I---------------------------------------------------
-I- No bad Guids were found

-I---------------------------------------------------
-I- Links With Logical State = INIT
-I---------------------------------------------------
-I- No bad Links (with logical state = INIT) were found

-I---------------------------------------------------
-I- General Device Info
-I---------------------------------------------------

-I---------------------------------------------------
-I- PM Counters Info
-I---------------------------------------------------
-I- No illegal PM counters values were found

-I---------------------------------------------------
-I- Links With links width != 4x (as set by -lw option)
-I---------------------------------------------------
-I- No unmatched Links (with width != 4x) were found

-I---------------------------------------------------
-I- Links With links speed != 10 (as set by -ls option)
-I---------------------------------------------------
-I- No unmatched Links (with speed != 10) were found

-I---------------------------------------------------
-I- Fabric Partitions Report (see ibdiagnet.pkey for a full hosts list)
-I---------------------------------------------------
-I-    PKey:0x7fff Hosts:113 full:113 limited:0

-I---------------------------------------------------
-I- IPoIB Subnets Check
-I---------------------------------------------------
-I- Subnet: IPv4 PKey:0x7fff QKey:0x00000b1b MTU:2048Byte rate:10Gbps SL:0x00
-W- Suboptimal rate for group. Lowest member rate:40Gbps > group-rate:10Gbps
```

```
-I---------------------------------------------------
-I- Bad Links Info
-I- No bad link were found
-I---------------------------------------------------
----------------------------------------------------------------
-I- Stages Status Report:
    STAGE                                  Errors Warnings
    Bad GUIDs/LIDs Check                   0      0
    Link State Active Check                0      0
    General Devices Info Report            0      0
    Performance Counters Report            0      0
    Specific Link Width Check              0      0
    Specific Link Speed Check              0      0
    Partitions Check                       0      0
    IPoIB Subnets Check                    0      1

Please see /tmp/ibdiagnet.log for complete log
----------------------------------------------------------------

-I- Done. Run time was 10 seconds.
[root@scae01cn01 ~]# ls –l /tmp/ibdiagnet*
-rw-r--r-- 1 root root 317258 Apr  6 17:02 ibdiagnet.db
-rw-r--r-- 1 root root  63784 Apr  6 17:02 ibdiagnet.lst
-rw-r--r-- 1 root root   7768 Apr  6 17:02 ibdiagnet.pkey
-rw-r--r-- 1 root root  33752 Apr  6 17:02 ibdiagnet.fdbs
-rw-r--r-- 1 root root    398 Apr  6 17:02 ibdiagnet.sm
-rw-r--r-- 1 root root   6161 Apr  6 17:02 ibdiagnet.mcfdbs
-rw-r--r-- 1 root root 716912 Apr  6 17:02 ibdiagnet.slvl
-rw-r--r-- 1 root root 353078 Apr  6 17:03 ibdiagnet.psl
-rw-r--r-- 1 root root   1931 Apr  6 17:03 ibdiagnet.log
```

Analysis from ibdagnet logs

/tmp/ibdiagnet.sm – Contains all instances of opensm running in the network.

```
[root@scae01cn01 ~]# cat /tmp/ibdiagnet.sm
ibdiagnet fabric SM report

  SM - master
    Port=13 lid=0x003d guid=0x0021286cc8aca0a0 dev=48438 priority:14

  SM - standby
    Port=6 lid=0x0001 guid=0x002128547f22c0a0 dev=48438 priority:5
    Port=21 lid=0x0040 guid=0x00212856d162c0a0 dev=48438 priority:5
    Port=20 lid=0x0041 guid=0x00212856d0a2c0a0 dev=48438 priority:5
    Port=22 lid=0x003f guid=0x002128548042c0a0 dev=48438 priority:5
```

/tmp/ibdiagnet.pkey – list of all partitions in use and associated hosts with their membership

```
[root@scae01cn01 ~]# cat /tmp/ibdiagnet.pkey
GROUP PKey:0x7fff Hosts:113
   Full scac01cel08/U/P1 lid=0x006e guid=0x0002c903000a7c41 dev=26428
   Full scac01cel08/U/P2 lid=0x006f guid=0x0002c903000a7c42 dev=26428
   Full scac01cel14/U/P1 lid=0x0017 guid=0x0002c903000a740d dev=26428
   Full scac01cel14/U/P2 lid=0x0003 guid=0x0002c903000a740e dev=26428
```

```
   Full scae01cn14/U/P1 lid=0x002f guid=0x0002c903000a7b6d dev=26428
   Full scae01cn14/U/P2 lid=0x0037 guid=0x0002c903000a7b6e dev=26428
   Full MT25408/P1 lid=0x006d guid=0x0002c903000a7429 dev=26428
   Full MT25408/P2 lid=0x0007 guid=0x0002c903000a742a dev=26428
<.. snip ..>
   Full scac01db05/U/P1 lid=0x0018 guid=0x0002c903000a7711 dev=26428
   Full scae01cn02/U/P1 lid=0x0030 guid=0x0021280001a0a35d dev=26428
   Full scae01cn02/U/P2 lid=0x0031 guid=0x0021280001a0a35e dev=26428
------------------------------------------------------------------------------

/tmp/ibdiagnet.fdbs - dump of the unicast forwarding tables of the fabric switches
```

This file may have a lot of entries. Some of them may say "UNREACHABLE". This means that those LIDs were known to the switches before and are no longer available.

Last column is expected to have keyword 'yes' for the optimal path calculated by the subnet manager routing algorithm.

ibdiagpath - Traces a path between two endpoints and provides information regarding the nodes and ports traversed along the path.

```
[root@scae01cn06 ~]# ibdiagpath -l 65,1
Loading IBDIAGPATH from: /usr/lib64/ibdiagpath1.5.4
-W- Topology file is not specified.
    Reports regarding cluster links will use direct routes.
Loading IBDM from: /usr/lib64/ibdm1.5.4
-W- A few ports of local device are up.
    Since port-num was not specified (-p option), port 1 of device 1 will be
    used as the local port.

-I---------------------------------------------------
-I- Traversing the path from local to source
-I---------------------------------------------------
-I- From: lid=0x0072 guid=0x0021280001a0a3fd dev=26428 scae01cn06/U/P1
-I- To:   lid=0x0001 guid=0x002128547f22c0a0 dev=48438 Port=9

-I- From: lid=0x0001 guid=0x002128547f22c0a0 dev=48438 Port=33
-I- To:   lid=0x000f guid=0x002128468d27a0a0 dev=48438 Port=25

-I- From: lid=0x000f guid=0x002128468d27a0a0 dev=48438 Port=24
-I- To:   lid=0x0041 guid=0x00212856d0a2c0a0 dev=48438 Port=35


-I---------------------------------------------------
-I- Traversing the path from source to destination
-I---------------------------------------------------
-I- From: lid=0x0041 guid=0x00212856d0a2c0a0 dev=48438 Port=20
-I- To:   lid=0x003d guid=0x0021286cc8aca0a0 dev=48438 Port=9

-I- From: lid=0x003d guid=0x0021286cc8aca0a0 dev=48438 Port=32
-I- To:   lid=0x0001 guid=0x002128547f22c0a0 dev=48438 Port=21


-I---------------------------------------------------
-I- PM Counters Info
-I---------------------------------------------------
```

```
-I- No illegal PM counters values were found

-I---------------------------------------------------
-I- Path Partitions Report
-I---------------------------------------------------
-I- Source Port=35 lid=0x0041 guid=0x00212856d0a2c0a0 dev=48438 Port 35
    PKeys:Not-Enforced
-I- Destination  lid=0x0001 guid=0x002128547f22c0a0 dev=48438 PKeys:Not-Enforced

-E- No shared PKeys found on Path! Nodes can not communicate!
    Aborting route tracing.
```

Network Packet Captures

During a network troubleshooting and diagnosis phase, you may need to capture packets for analysis. The information in these packet captures can be very useful and lead you to a root cause of the problem or confirm some other findings. You can view various things like sequence of messages, handshake flow, latencies, packet formats, missing messages, etc.

To capture the packets in Linux, you can use `tcpdump`, which has several options. Often it is useful to save the captures for post analysis. The flow being captured can be very fast and it is advisable to save it in a tmpfs file system in various incremental files. Be sure to clean up the tmpfs location after saving the files elsewhere. In Solaris, you can snoop to perform the same activity. The captured format of both is the same and is called *pcap*.

In InfiniBand, we also have a similar tool called `ibdump`. This tool is provided by Mellanox and can be downloaded from their website:

http://www.mellanox.com/content/pages.php?pg=products_dyn&product_family=110&menu_section=34

As this is relatively new, not all features compared to tcpdump /snoop are available at this time. However, the output format is the same as pcap. `ibdump` captures can also provide very useful information in debugging InfiniBand messaging on the interface.

All pcap files can be played back in a graphical tool called Wireshark. In earlier times, this used to be ethereal. It can be downloaded from following website:

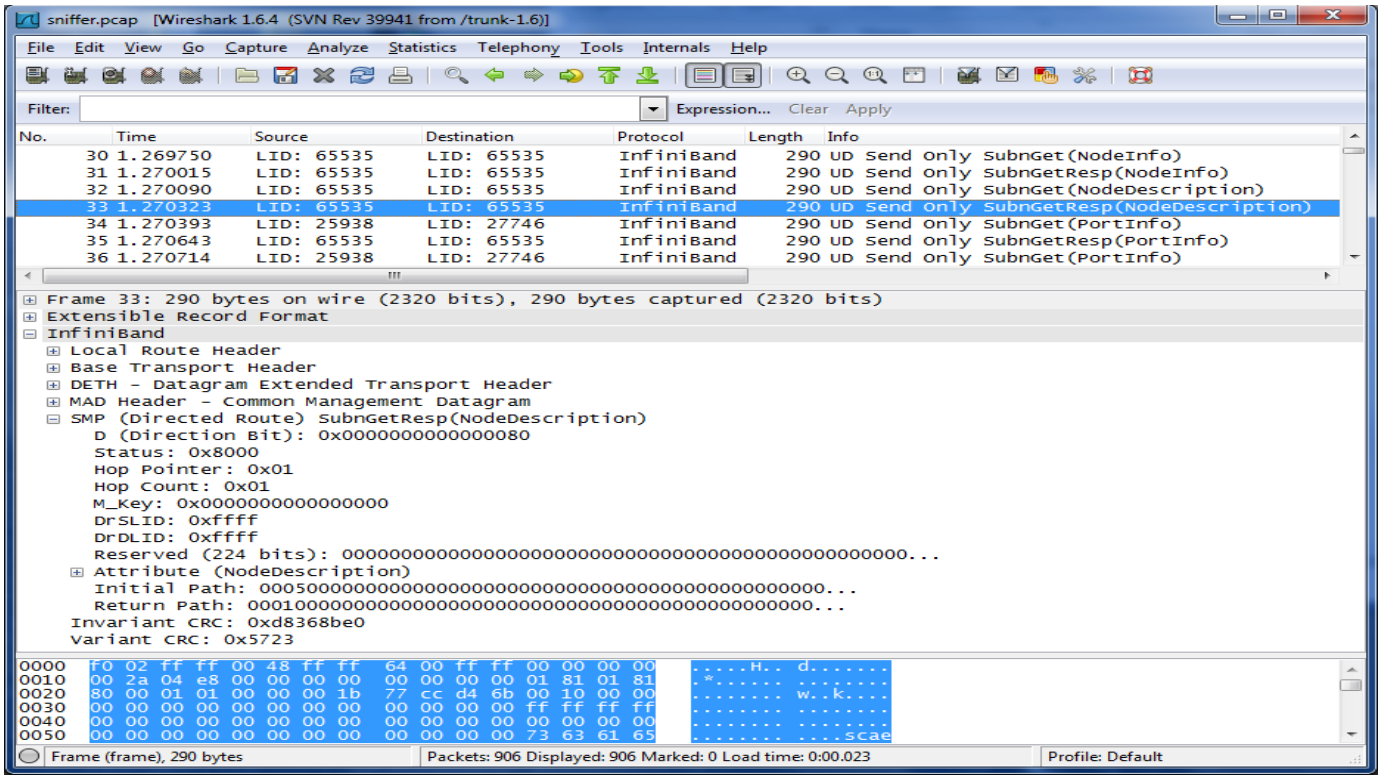
http://www.wireshark.org/download.html

This interface is very powerful, and you can apply various types of filters for easy analysis. It also has capabilities to parse and display various protocols, which helps in understanding the handshakes and their flow sequences.

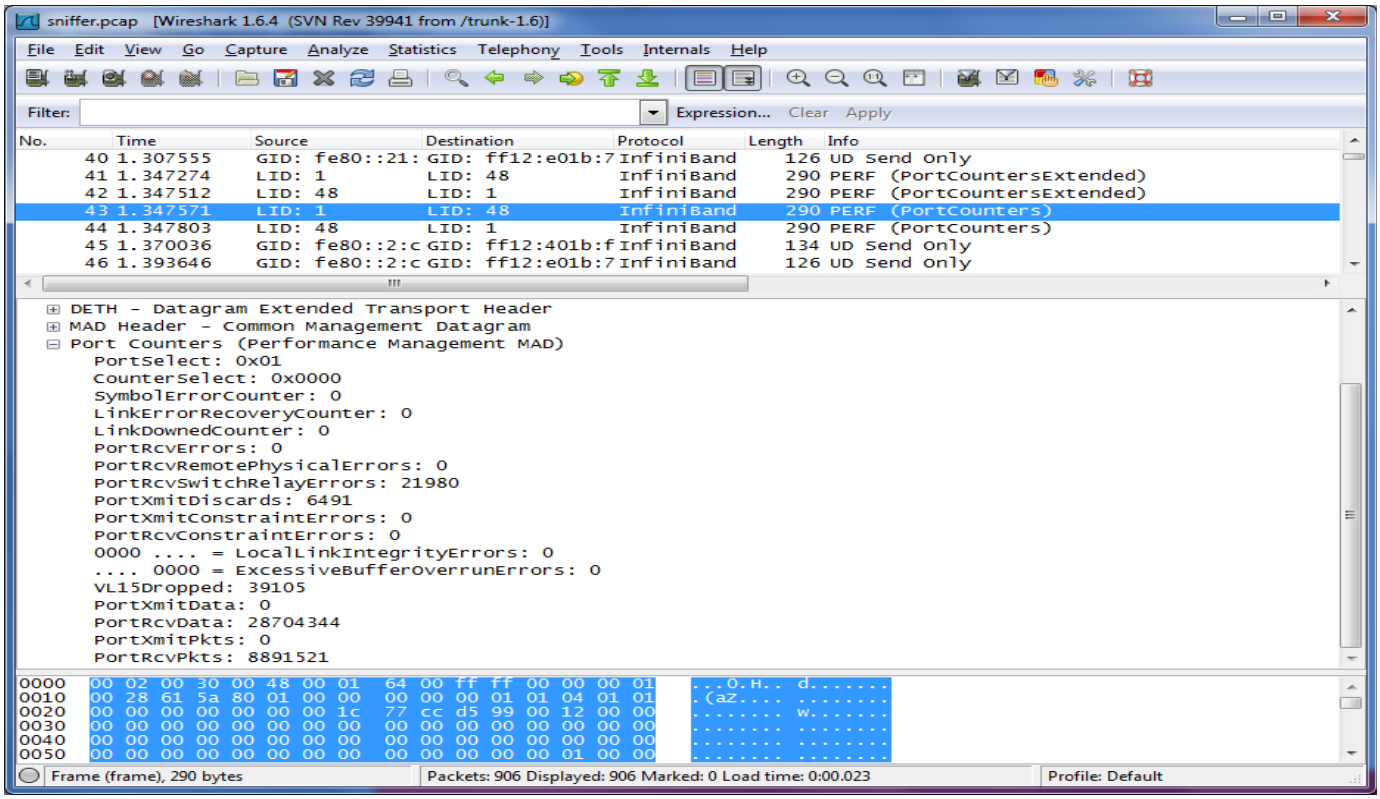
Screenshots of ibdump pcaps in Wireshark

This section includes some screenshots taken during InfiniBand packet analysis. Wireshark version 1.6 was used on a Windows 7 laptop to play back the captured pcap files. It is recommended to use latest version, as the InfiniBand protocol parsing is recently introduced. If you intend to use Wireshark on a Linux machine, then it may need to be compiled from source if a pre-built RPM is not available.

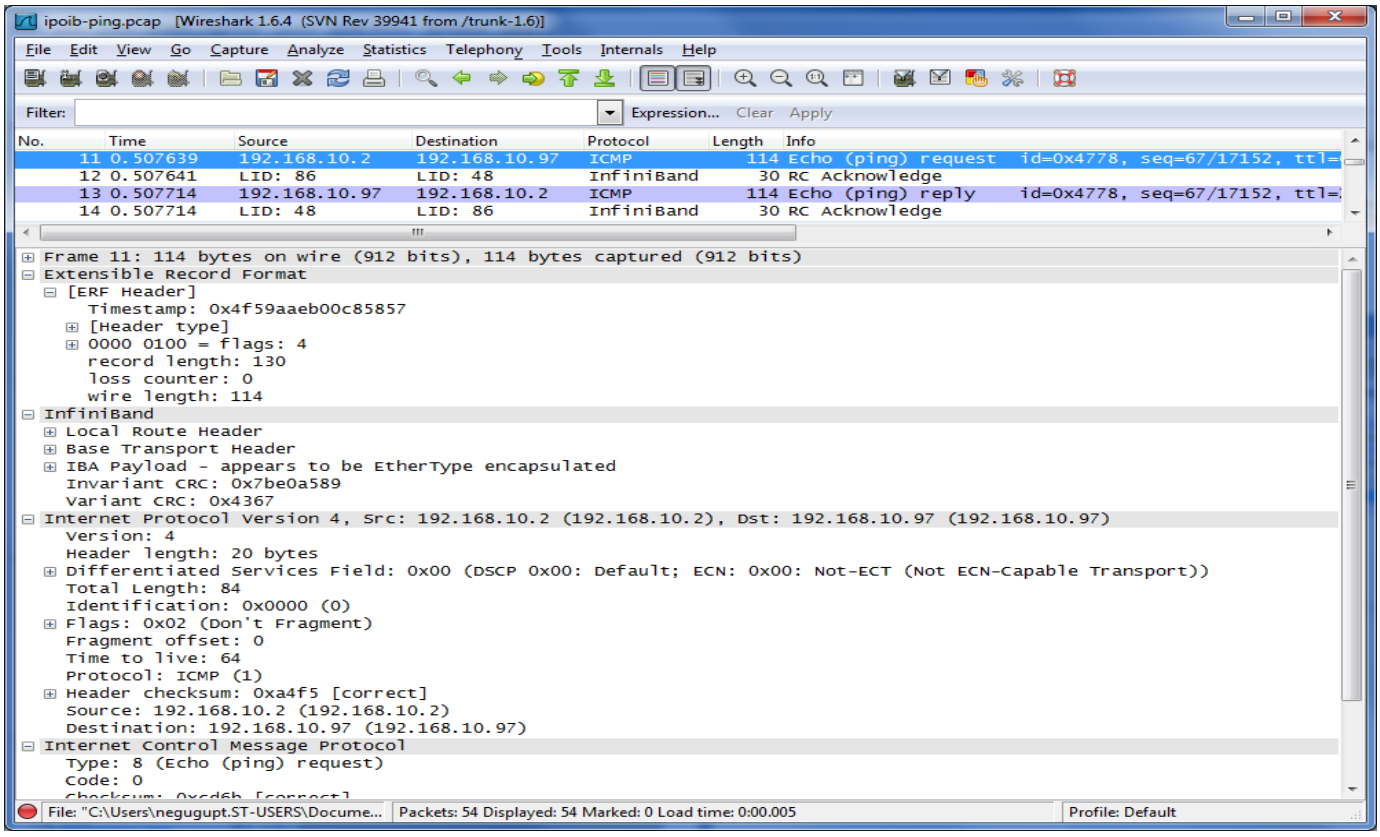
Node Description Query



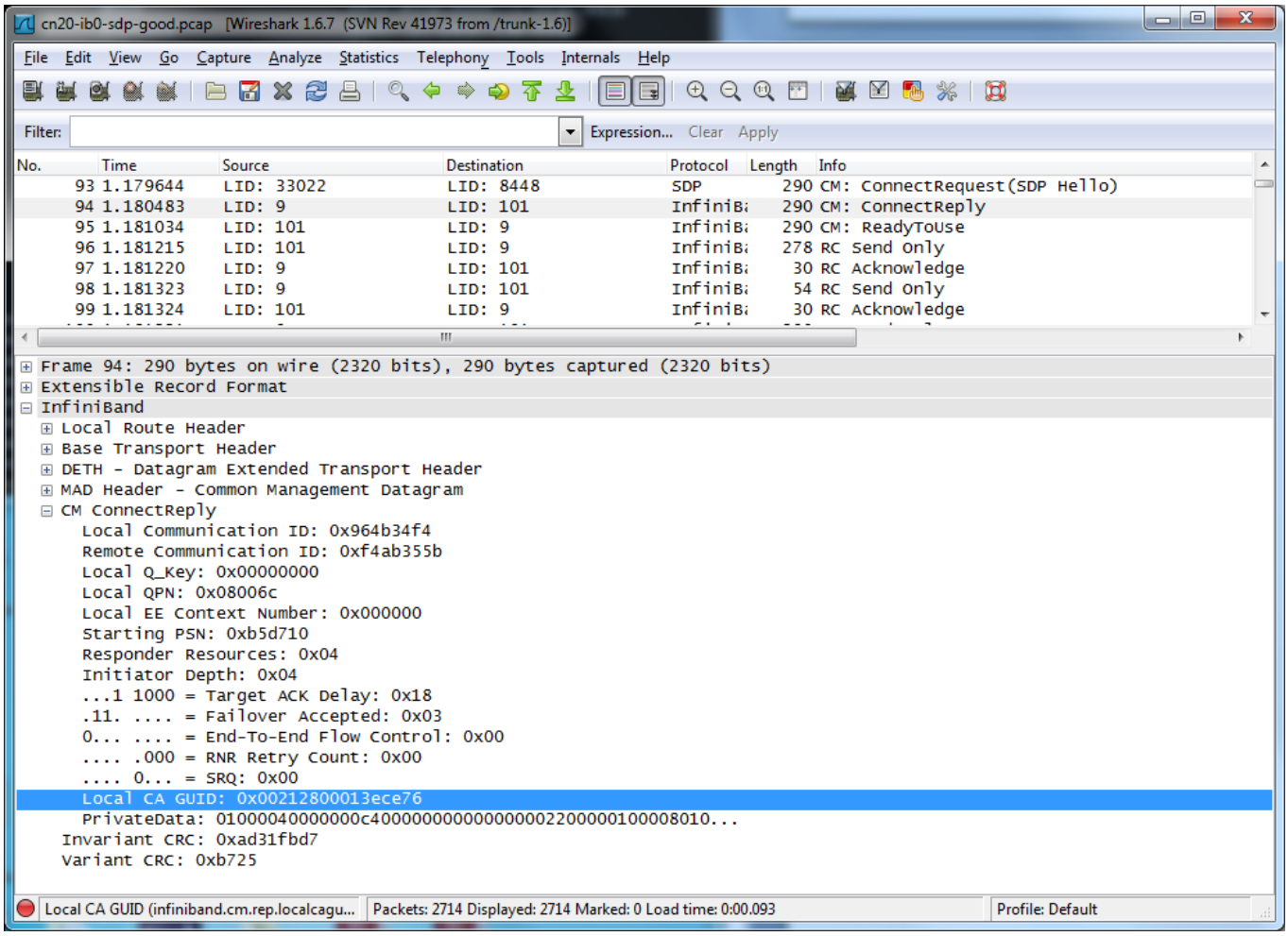
Performance Counter Query



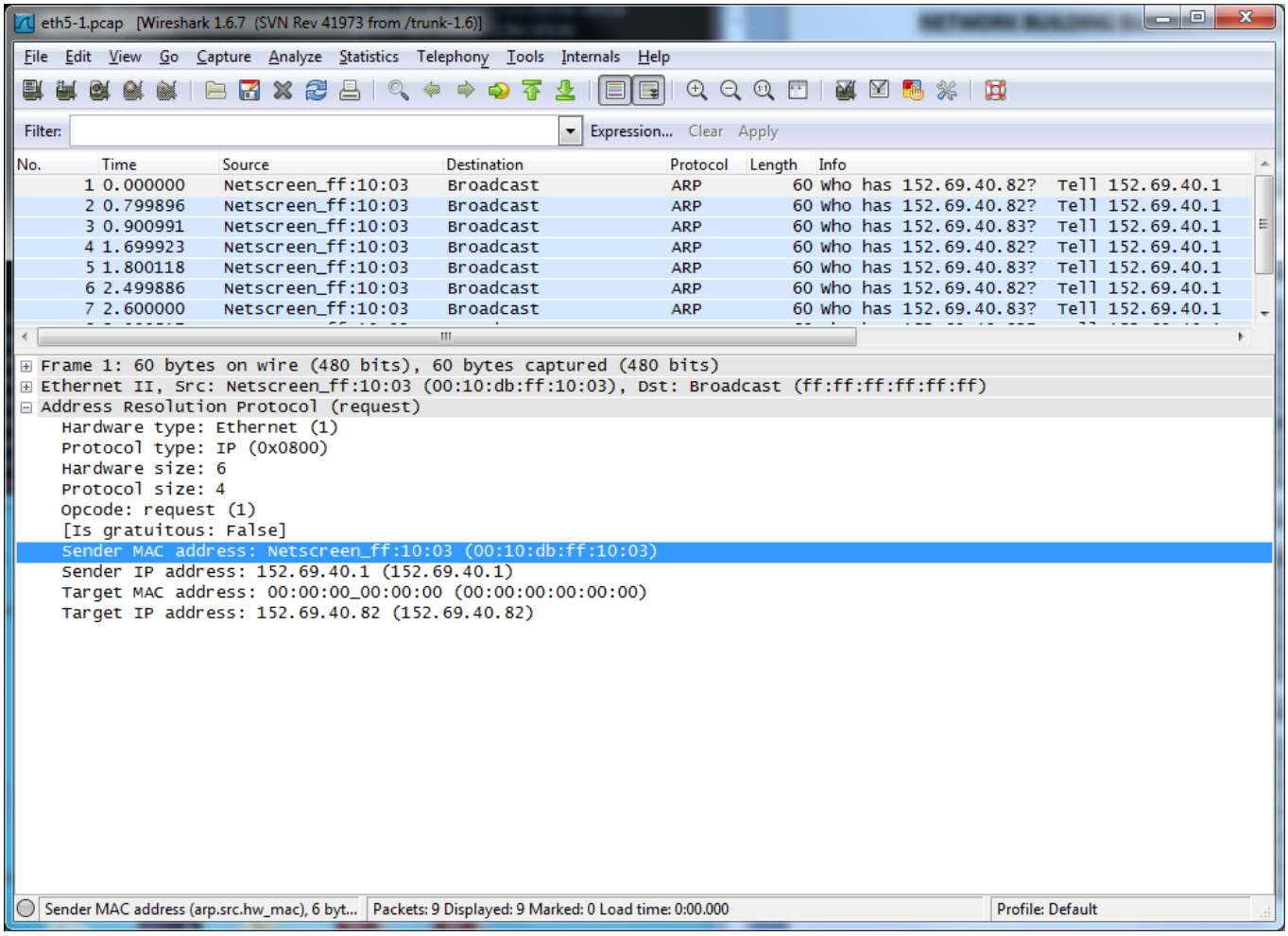
IPv4 ICMP (ping) on IPoIB



SDP Handshakes



ARP Request Failures



Troubleshooting Scenarios

Use Case Analysis #1

Periodic data collection process is indicating symbol errors or link recovers in perfquery outputs.

InfiniBand Link does not come in 4X QDR.

Brief background

Symbol error counters in perfquery indicate a problem at electrical level on the specific link. Such errors are confined to a specific port. They are not indicative of any problem outside of this particular link and associated hardware components.

Link recoveries in `perfquery` indicate that the link is not stable at its desired electrical levels and it is retrying auto-negotiations.

An IB link showing width 1X or its speed as DDR or SDR indicates that it is not able to negotiate the desired parameters due to some problems at hardware layers.

Troubleshooting Guidelines

Reset port counters and observe if the error symptoms are coming back again within observation period.

Reset the port on both sides of the link one at a time and observe the status again.

Re-seat each end of the link physically one at a time and observe the status again.

Use a spare port on switch side if possible to see if the error state goes away.

Use a different cable temporarily to see if the fault was on cable itself.

After performing these tasks, if the problem is resolved then you can conclude that the hardware does not have any non-recoverable faults. The error states could have been caused due to improper connector latch or link negotiation. However if the issue is not resolved then you can conclude which component is at fault based on exit status from each of the steps you perform.

Use Case Analysis #2

No master subnet manager instance in InfiniBand fabric.

LID values are 0.

Brief Background

Subnet manager is the most important software required for InfiniBand network functionality. This runs on NM2 switches, and if multiple instances are running in same IB subnet then one of them is elected to be master and others become standby. An OFED tool – sminfo or NM2 switch command – `getmaster` can be used to query currently master subnet manager. To query all instances of subnet manager, you can use OFED tool `ibdiagnet` which generates `/tmp/ibdiagnet.sm`.

Troubleshooting Guidelines

Check log files on all NM2 switches where opensm instance is running. `/var/log/messages` and `/var/log/opensm.log.`

Verify that each NM2 switch can resolve localhost to 127.0.0.1 through `/etc/hosts` file.

Check partitions configuration for correctness and any syntax issues.

Check `/conf/configvalid` file to make sure it has a single line with numeric value '1'.

Check output of `ibdiagnet` to make sure there are no duplicate GUIDs or errored links in fabric.

Check system stats of each NM2 switch to make sure they don't have a full file system or are running low on free memory.

Verify that none of the hosts is running a subnet manager instance.

Use Case Analysis #3

Unable to ping some nodes within an IPoIB network.

Brief Background

If a set of hosts are configured in a valid IPoIB subnet inside an IB fabric, then they can communicate with each other.

Troubleshooting Guidelines

Check that there are no problems in the IPoIB subnet configuration, i.e. all nodes have matching subnet mask.

Check that all nodes are in same IB partition.

Check that nodes under troubleshooting can communicate with each other based on partition memberships.

Check that there is a master subnet manager in network.

Check log subnet manager log file `/var/log/opensm.log` for any visible problems.

Check that the nodes and switches are visible in '`ibhosts`, `ibswitches`' command outputs.

Check if there is any InfiniBand multicast group with MTU code > 0x84.

Check for correct path between hosts under analysis.

Check for bonding driver status on the host to make sure there is an active interface.

Check the log files of the host to see if the IPoIB interface is reporting any transmit timeouts.

Use Case Analysis #4

Cannot login to NM2 InfiniBand Switch.

Brief Background

NM2 switch has a gigabit Ethernet port for management access. If needed, the switch can also be accessed over serial via USB converter.

Troubleshooting Guidelines

Run a ping test to the NM2 switch. If this is successful then IP connectivity is OK.

Check if the switch can be accessed via its browser interface.

Check to see if the switch is visible in `ibswitches` output from same IB fabric.

Connect to the switch using its USB-serial port and attempt to login.

If the switch is still not accessible then a power cycle may be required.

If login through USB-serial is working, then attempt to restart SSH service via ILOM CLI.

If the switch was not reachable over IP ping, then check if the link to Ethernet is active or not. Use the command `ethtool` for this.

Once the switch is recovered, check for file system status and log files for any visible issues.

Use Case Analysis #5

Unable to get EoIB VNICs in working state.

Brief Background

This case is applicable to NM2GW but its interaction with NM2-36P should also be analyzed during troubleshooting. There could be many reasons for vnics not in UP state and you need to take a step-by-step approach to diagnose.

Troubleshooting Guidelines

Check for correct GUID, MAC addresses in created VNICs.

Check for the 10GbE uplink status on NM2GW.

Check for `/conf/configvalid` file and make sure it has a single line with '1'.

Check for presence of a master subnet manager in IB fabric.

Check for correct Data SL=1 and Control SL=2 by running `showgwconfig` command.

Check if correct PKEY and VLAN are using if applicable.

Check to see if Bridge-X GUIDs are added to the appropriate IB partitions also.

Check for correct OFED version on corresponding hosts.

Check that the drivers are properly installed and running on host. `/etc/inifiniband/openib.conf` should have VNIC parameter set to 'yes'. lsmod should have mlx4_vnic driver loaded.

If the host is running Solaris, then make sure that the device instances have been configured using `dladm` command for the first time.

If possible, restart BXM service to see if VNICs come up.

Check to see if NM2GW switch is not running out of system resources such as memory and disk space.

Use Case Analysis #6

VNICs cannot communicate to external Ethernet network.

Brief Background

In some cases, VNICs may be functioning among themselves but not with external networks. This situation needs some analysis on the external network components and their configurations.

Troubleshooting Guidelines

Check the 10GbE uplink status and make sure its active.

Check to make sure that the uplink is connected to the correct external LAN.

Check to see if a router is needed on VNIC hosts to reach the desired networks. Router IP should be reachable from VNICs.

If bonding over VNICs is used then make sure both paths are properly configured with external network switches.

Under active/standby bonding, we can test to see if both interfaces are not working or if the problem is just with one interface.

If VLANs are being used then check to make sure external switches are also configured properly with VLANs.

If the target test host is several hops away, then check if a different host can be tested for communication with fewer hops to eliminate the issues.

Use Case Analysis #7

Cannot establish connections using SDP over IB.

Brief Background

When SDP support is enabled in InfiniBand drivers, qualified applications can use SDP for socket based connections instead of TCP.

Troubleshooting Guidelines

Check if the applications on same host pair can use TCP or not. If successful, then problem is SDP specific.

Check if another client can connect to this service on SDP.

Check if the client can connect to another server on SDP.

Based on these results, the problem may be on server or client side.

Capture packets using `ibdump` utility for offline analysis.

If no anomalies are seen, then try to reboot the problem node if possible. A ticket may need to be filed.

Use Case analysis #8

EoIB bonding does not failover when 10GbE uplink is failed.

Brief Background

EoIB VNICs depend on the NM2GW's 10GbE uplinks. If you are setup in 1+1 redundant paths using Linux bonding, then VNICs should failover when uplink fails.

Troubleshooting Guidelines

Check to see if `eport_state_enforce` option is set to '1' or not. This is required.

Check the OFED and NM2GW versions for compatibility.

It may be desired to use `arp_ip_target` option in Linux bonding depending on network architecture.

Use Case analysis #9

`rds-ping` shows high latency for the first return.

Brief Background

`rds-ping` is a basic connectivity test tool specifically meant to verify rds connection between two Exadata DB or CELL nodes. It does not work for other types of nodes or storage appliances. Due to some fabric-wide issues, `rds-ping` can take longer to establish communication channels, so the first response can take longer than subsequent responses. If the first response takes longer than normal then there may be some problem in fabric.

Troubleshooting Guidelines

Check the fabric health using `ibdiagnet –ls 10 –lw 4x`.

Check the count of IB hosts and switches. This should match the expected number as per our fabric blueprint.

Check the health of all IPoIB targets by using `ping` command.

Check to see if there is any unresponsive IB node in fabric. This usually causes delays in IB queries.

Use Case analysis #10

NFS mount points hang or stall.

Brief Background

Exalogic machine has built-in ZFSSA which serves file system over NFS. Often times Exadata machine may also use an external ZFSSA for backup and other purposes. These NFS mounts are usually over IPoIB channel. If you see a hang or stall on the mount points during file system operations, then there may be some problem either on IPoIB or the NFS server/client itself.

Troubleshooting Guidelines

Check the fabric health using `ibdiagnet –ls 10 –lw 4x`.

Check the count of IB hosts and switches. This should match the expected number as per our fabric blueprint.

Check the health of all IPoIB storage targets by using `ping` command. Usually if `ping` between compute node and storage succeeds that confirms IPoIB is healthy and problem may be elsewhere.

Check to see if the mount is hanging on a specific node or all NFS clients. This will help us identify if NFS client has an issue or the NFS server.

If NFS v4 is being used, then check for associated critical services like NIS or LDAP used for domain mapping.

It is also advisable to check the ZFSSA software version and if needed upgrade to the latest released versions to avoid known issues.

Use Case analysis #11

Unable to reach NFS service over a partition based IPoIB path.

Brief Background

InfiniBand partition provides a virtualized network path over same physical medium. A set of configurations is required to get successful connectivity. If the default network works, but not a partition based network, then you may need to analyze this state.

Troubleshooting Guidelines

Check to see if the partition is created on IB switch using `smpartition list active` command. You can also use `ibdiagnet` log files in `/tmp` to query the same information.

Check to see if the storage GUIDs are added as full members to the partition. Storage cannot be a limited member by design.

Check if the partition configuration has been committed on the IB switches or not. This must be performed on the master subnet manager. If needed, you can commit it again to ensure.

If the partitioned interfaces are configured in IPMP, you can attempt to see if swapping of active and standby has any effect. If it does then there may be a problem with only one interface and not both.

Check if the same partitioned interface can communicate with another node member of same partition. This check should also be done on NFS clients.

You should also check for IPoIB subnet to make sure there is no error at layer-3 configuration.

Use Case analysis #12

InfiniBand interfaces on compute nodes do not come up as active.

Brief Background

InfiniBand interfaces have a physical state and a logical state. The physical state is controlled at the electrical level using firmware. The logical state is controlled by device drivers and their interaction with the subnet manager.

Troubleshooting Guidelines

Check the local IB device health using `ibstat` command. If the physical state is down or polling, it indicates some problem at the hardware level.

Check to see if the corresponding port on IB switch is disabled or inactive. Use `listlinkup` command for this.

Check and ensure that the cable is connected properly on both ends. If needed, test with a spare cable or swap to a different switch port.

Once the physical state is changed from down or polling, it will become active.

The logical state could be 'Initializing'. If it stays like this for a long time, then there is a problem communicating with the subnet manager.

Check to ensure that subnet manager is active and assigning LIDs to end points.

Use Case analysis #13

`netstat` output shows dropped packets for EoIB interface.

Brief Background

EoIB interfaces have statistic counters similar to regular network interfaces. A mis-configured network can lead to increase in dropped RX packets. There could be multiple reasons for such scenarios.

Troubleshooting Guidelines

Check and monitor other non-zero counters in `netstat` output to see any co-relation.

Check to see if VLAN related error counters are also incrementing. If yes, then it means there is a vlan misconfiguration.

Take a packet capture using `tcpdump` and analyze in Wireshark.

In multi-homed environments where multiple ports are connected to same Ethernet switch, a routing error can result in RX dropped packets.

InfiniBand Network Troubleshooting Guidelines
August 2012
Author: Neeraj Gupta

Oracle Corporation
World Headquarters
500 Oracle Parkway
Redwood Shores, CA 94065
U.S.A.

Worldwide Inquiries:
Phone: +1.650.506.7000
Fax: +1.650.506.7200

oracle.com