InfiniBand Fabric Utilities

Table of contents

List of Figures

This section first describes common configuration, interface, and addressing for all the tools in the package.

Common Configuration, Interface and Addressing

Topology File (Optional)

An InfiniBand fabric is composed of switches and channel adapter (HCA/TCA) devices. To identify devices in a fabric (or even in one switch system), each device is given a GUID (a MAC equivalent). Since a GUID is a non-user-friendly string of characters, it is better to alias it to a meaningful, user-given name. For this objective, the IB Diagnostic Tools can be provided with a “topology file”, which is an optional configuration file specifying the IB fabric topology in user-given names.
For diagnostic tools to fully support the topology file, the user may need to provide the local system name (if the local hostname is not used in the topology file).
To specify a topology file to a diagnostic tool use one of the following two options:

1. On the command line, specify the file name using the option ‘-t <topology file name>’

2. Define the environment variable IBDIAG_TOPO_FILE

To specify the local system name to an diagnostic tool use one of the following two options:

1. On the command line, specify the system name using the option ‘-s <local system name>’

2. Define the environment variable IBDIAG_SYS_NAME

InfiniBand Interface Definition

The diagnostic tools installed on a machine connect to the IB fabric by means of an HCA port through which they send MADs. To specify this port to an IB diagnostic tool use one of the following options:

1. On the command line, specify the port number using the option ‘-p <local port number>’ (see below)

2. Define the environment variable IBDIAG_PORT_NUM

In case more than one HCA device is installed on the local machine, it is necessary to specify the device’s index to the tool as well. For this use on of the following options:

1. On the command line, specify the index of the local device using the following option: ‘-i <index of local device>’

2. Define the environment variable IBDIAG_DEV_IDX

Addressing

Note

This section applies to the ibdiagpath tool only. A tool command may require defining the destination device or port to which it applies.

The following addressing modes can be used to define the IB ports:

- Using a Directed Route to the destination: (Tool option ‘-d’)
  This option defines a directed route of output port numbers from the local port to the destination.

- Using port LIDs: (Tool option ‘-l’):
  In this mode, the source and destination ports are defined by means of their LIDs. If the fabric is configured to allow multiple LIDs per port, then using any of them is valid for defining a port.

- Using port names defined in the topology file: (Tool option ‘-n’)
  This option refers to the source and destination ports by the names defined in the topology file. (Therefore, this option is relevant only if a topology file is specified to the tool.) In this mode, the tool uses the names to extract the port LIDs from the matched topology, then the tool operates as in the ‘-l’ option.

Diagnostic Utilities

The diagnostic utilities described in this chapter provide means for debugging the connectivity and status of InfiniBand (IB) devices in a fabric.

Diagnostic Utilities

Utility	Description
dump_fts	Dumps tables for every switch found in an ibnetdiscover scan of the subnet. The dump file format is compatible with loading into OpenSM using the -R file -U /path/to/dump-file syntax. For further information, please refer to the tool's man page.
ibaddr	Can be used to show the LID and GID addresses of the specified port or the local port by default. This utility can be used as simple address resolver. For further information, please refer to the tool's man page.
ibcacheedit	Allows users to edit an ibnetdiscover cache created through the --cache option in ibnetdiscover(8). For further information, please refer to the tool's man page.
ibccconfig	Supports the configuration of congestion control settings on switches and HCAs. For further information, please refer to the tool's man page.
ibccquery	Supports the querying of settings and other information related to congestion control. For further information, please refer to the tool's man page.
ibcongest	Provides static congestion analysis. It calculates routing for a given topology (topo-mode) or uses extracted lst/fdb files (lst-mode). Additionally, it analyzes congestion for a traffic schedule provided in a "schedule-file" or uses an automatically generated schedule of all-to-all-shift. To display a help message which details the tool's options, please run "/opt/ibutils2/bin/ibcongest -h". For further information, please refer to the tool's man page.
ibdev2	Enables association between IB devices and ports and the associated net device. Additionally it reports the state of the net device link. For further information, please refer to the tool's man page.

Utility	Description
netdev	
ibdiagnet (of ibutils2)	Scans the fabric using directed route packets and extracts all the available information regarding its connectivity and devices. An ibdiagnet run performs the following stages: <ul><li>Fabric discovery</li><li>Duplicated GUIDs detection</li><li>Links in INIT state and unresponsive links detection</li><li>Counters fetch</li><li>Error counters check</li><li>Routing checks</li><li>Link width and speed checks</li><li>Alias GUIDs check</li><li>Subnet Manager check</li><li>Partition keys check</li><li>Nodes information</li></ul>**Note:** This version of ibdiagnet is included in the ibutils2 package, and it is run by default after installing NVIDIA OFED. To use this ibdiagnet version, run: ibdiagnet. For further information, either: 1. Run `ibdiagnet -H` Or 2. Refer to docs.nvidia.com/networking/display/ibdiagnetUserManualv10
ibdiagpath	Traces a path between two end-points and provides information regarding the nodes and ports traversed along the path. It utilizes device specific health queries for the different devices along the path. The way ibdiagpath operates depends on the addressing mode used in the command line. If directed route addressing is used (--dr_path flag), the local node is the source node and the route to the destination port is known apriori (for example: ibdiagpath --dr_path 0,1). On the other hand, if LID-route addressing is employed, --src_lid and --dest_lid, then the source and destination ports of a route are specified by their LIDs. In this case, the actual path from the local port to the source port, and from the source port to the destination port, is defined by means of Subnet Management Linear Forwarding Table queries of the switch nodes along that path. Therefore, the path cannot be predicted as it may change. Example: ibdiagpath --src_lid 1 --dest_lid 28

Utility	Description
	For further information, please refer to the tool's -help flag.
ibdump	Dump InfiniBand traffic that flows to and from NVIDIA's ConnectX® family adapters InfiniBand ports. Note the following: • ibdump is not supported for Virtual functions (SR-IOV) • Infiniband traffic sniffing is supported on all HCAs The dump file can be loaded by the Wireshark tool for graphical traffic analysis. The following describes a workflow for local HCA (adapter) sniffing: 1. Run ibdump with the desired options 2. Run the application that you wish its traffic to be analyzed 3. Stop ibdump (CTRL-C) or wait for the data buffer to fill (in --mem-mode) 4. Open Wireshark and load the generated file To download Wireshark for a Linux or Windows environment go to www.wireshark.org. **Notes:** • Although ibdump is a Linux application, the generated .pcap file may be analyzed on either operating system. • If one of the HCA's ports is configured as InfiniBand, ibdump requires IPoIB DMFS to be enabled. For further information, please refer to Flow Steering Configuration section. For further information, please refer to the tool’s man page.
iblinkinfo	Reports link info for each port in an InfiniBand fabric, node by node. Optionally, iblinkinfo can do partial scans and limit its output to parts of a fabric. For further information, please refer to the tool’s man page.
ibnetdiscover	Performs InfiniBand subnet discovery and outputs a human readable topology file. GUIDs, node types, and port numbers are displayed as well as port LIDs and node descriptions. All nodes (and links) are displayed (full topology). This utility can also be used to list the current connected nodes. The output is printed to the standard output unless a topology file is specified. For further information, please refer to the tool’s man page.
ibnets	Automatically groups hosts and creates scripts that can be run in order to split the network into sub-networks containing one group of hosts.

Utility	Description
plit	For further information, please refer to the tool’s man page.
ibnodes	Uses the current InfiniBand subnet topology or an already saved topology file and extracts the InfiniBand nodes (CAs and switches). For further information, please refer to the tool’s man page.
ibping	Uses vendor mads to validate connectivity between InfiniBand nodes. On exit, (IP) ping like output is show. ibping is run as client/server. The default is to run as client. Note also that a default ping server is implemented within the kernel. For further information, please refer to the tool’s man page.
ibportstate	Enables querying the logical (link) and physical port states of an InfiniBand port. It also allows adjusting the link speed that is enabled on any InfiniBand port. If the queried port is a switch port, then ibportstate can be used to: • disable, enable or reset the port • validate the port’s link width and speed against the peer port In case of multiple channel adapters (CAs) or multiple ports without a CA/ port being specified, a port is chosen by the utility according to the following criteria: • The first ACTIVE port that is found. • If not found, the first port that is UP (physical link state is LinkUp). For further information, please refer to the tool’s man page.
ibqueryerrors	The default behavior is to report the port error counters which exceed a threshold for each port in the fabric. The default threshold is zero (0). Error fields can also be suppressed entirely. In addition to reporting errors on every port, ibqueryerrors can report the port transmit and receive data as well as report full link information to the remote port if available. For further information, please refer to the tool’s man page.
ibroute	Uses SMPs to display the forwarding tables—unicast (LinearForwarding- Table or LFT) or multicast (MulticastForwardingTable or MFT)—for the specified switch LID and the optional lid (mlid) range. The default range is all valid entries in the range 1 to FDBTop. For further information, please refer to the tool’s man page.
ibstat	ibstat is a binary which displays basic information obtained from the local IB driver. Output includes LID, SMLID, port state, link width active, and port physical

Utility	Description
	state. For further information, please refer to the tool's man page.
ibstatus	Displays basic information obtained from the local InfiniBand driver. Output includes LID, SMLID, port state, port physical state, port width and port rate. For further information, please refer to the tool's man page.
ibswitches	Traces the InfiniBand subnet topology or uses an already saved topology file to extract the InfiniBand switches. For further information, please refer to the tool's man page.
ibsysstat	Uses vendor mads to validate connectivity between InfiniBand nodes and obtain other information about the InfiniBand node. ibsysstat is run as client/ server. The default is to run as client. For further information, please refer to the tool's man page.
ibtopodiff	Compares a topology file and a discovered listing ofsubnet.lst/ibdiagnet.lst and reports mismatches. Two different algorithms provided: • Using the -e option is more suitable for MANY mismatches it applies less heuristics and provide details about the match • Providing the -s, -p and -g starts a detailed heuristics that should be used when only small number of changes are expected For further information, please refer to the tool's man page.
ibtracert	Uses SMPs to trace the path from a source GID/LID to a destination GID/ LID. Each hop along the path is displayed until the destination is reached or a hop does not respond. By using the -m option, multicast path tracing can be performed between source and destination nodes. For further information, please refer to the tool's man page.
ibv_asyncwatch	Display asynchronous events forwarded to userspace for an InfiniBand device. For further information, please refer to the tool's man page.
ibv_dev	Lists InfiniBand devices available for use from userspace, including node GUIDs. For further information, please refer to the tool's man page.

Utility	Description
ices	
ibv_devinfo	Queries InfiniBand devices and prints about them information that is available for use from userspace. For further information, please refer to the tool's man page.
mstflint	Queries and burns a binary firmware-image file on non-volatile (Flash) memories of NVIDIA InfiniBand and Ethernet network adapters. The tool requires root privileges for Flash access. To run mstflint, you must know the device location on the PCI bus. **Note:** If you purchased a standard NVIDIA network adapter card, please download the firmware image from nvidia.com/en-us/networking/ Support Support Firmware Download. If you purchased a non-standard card from a vendor other than NVIDIA, please contact your vendor. For further information, please refer to the tool's man page.
perfquery	Queries InfiniBand ports' performance and error counters. Optionally, it displays aggregated counters for all ports of a node. It can also reset counters after reading them or simply reset them. For further information, please refer to the tool's man page.
saquery	Issues the selected SA query. Node records are queried by default. For further information, please refer to the tool's man page.
sminfo	Issues and dumps the output of an sminfo query in human readable format. The target SM is the one listed in the local port info or the SM specified by the optional SM LID or by the SM direct routed path. **Note:** Using sminfo for any purpose other than a simple query might result in a malfunction of the target SM. For further information, please refer to the tool's man page.
smparquery	Sends SMP query for adaptive routing and private LFT features. For further information, please refer to the tool's man page.
smpdump	A general purpose SMP utility which gets SM attributes from a specified SMA. The result is dumped in hex by default. For further information, please refer to the tool's man page.

Utility	Description
smpquery	Provides a basic subset of standard SMP queries to query Subnet management attributes such as node info, node description, switch info, and port info. For further information, please refer to the tool’s man page.

Link Level Retransmission (LLR) in FDR Links

With the introduction of FDR 56 Gbps technology, NVIDIA enabled a proprietary technology called LLR (Link Level Retransmission) to improve the reliability of FDR links.

This proprietary LLR technology adds additional CRC checking to the data stream and retransmits portions of packets with CRC errors at the local link level. Customers should be aware of the following facts associated with LLR technology:

- Traditional methods of checking the link health can be masked because the LLR technology automatically fixes errors. The traditional IB symbol error counter will show no errors when LLR is active.
- Latency of the fabric can be impacted slightly due to LLR retransmissions. Traditional IB performance utilities can be used to monitor any latency impact.
- Bandwidth of links can be reduced if cable performance degrades and LLR retransmissions become too numerous. Traditional IB bandwidth performance utilities can be used to monitor any bandwidth impact.

Due to these factors, an LLR retransmission rate counter has been added to the ibdiagnet utility that can give end users an indication of the link health.

➢ *To monitor LLR retransmission rate:*

1. Run ibdiagnet, no special flags required.
2. If the LLR retransmission rate limit is exceeded it will print to the screen.
3. The default limit is set to 500 and requires further investigation if exceeded.
4. The LLR retransmission rate is reflected in the results file /var/tmp/ibdiagnet2/ibdiagnet2.pm.

The default value of 500 retransmissions/sec has been determined by NVIDIA based on the extensive simulations and testing. Links exhibiting a lower LLR retransmission rate should not raise special concern.

Performance Utilities

The performance utilities described in this chapter are intended to be used as a performance micro-benchmark.

Utility	Description
ib_atomic_bw	Calculates the BW of RDMA Atomic transactions between a pair of machines. One acts as a server and the other as a client. The client RDMA sends atomic operation to the server and calculate the BW by sampling the CPU each time it receive a successful completion. The test supports features such as Bidirectional, in which they both RDMA atomic to each other at the same time, change of MTU size, tx size, number of iteration, message size and more. Using the "-a" flag provides results for all message sizes. For further information, please refer to the tool’s man page.
ib_atomic_lat	Calculates the latency of RDMA Atomic transaction of message_size between a pair of machines. One acts as a server and the other as a client. The client sends RDMA atomic operation and sample the CPU clock when it receives a successful completion, in order to calculate latency. For further information, please refer to the tool’s man page.
ib_read_bw	Calculates the BW of RDMA read between a pair of machines. One acts as a server and the other as a client. The client RDMA reads the server memory and calculate the BW by sampling the CPU each time it receive a successful completion. The test supports features such as Bidirectional, in which they both RDMA read from each other memory's at the same time, change of MTU size, tx size, number of iteration, message size and more. Read is available only in RC connection mode (as specified in IB spec). For further information, please refer to the tool’s man page.
ib_read_lat	Calculates the latency of RDMA read operation of message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which one side RDMA reads the memory of the other side only after the other side have read his memory. Each of the sides samples the

Utility	Description
	CPU clock each time they read the other side memory , in order to calculate latency. Read is available only in RC connection mode (as specified in IB spec). For further information, please refer to the tool’s man page.
ib_send_bw	Calculates the BW of SEND between a pair of machines. One acts as a server and the other as a client. The server receive packets from the client and they both calculate the throughput of the operation. The test supports features such as Bidirectional, on which they both send and receive at the same time, change of MTU size, tx size, number of iteration, message size and more. Using the "-a" provides results for all message sizes. For further information, please refer to the tool’s man page.
ib_send_lat	Calculates the latency of sending a packet in message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which you send packet only if you receive one. Each of the sides samples the CPU each time they receive a packet in order to calculate the latency. Using the "-a" provides results for all message sizes. For further information, please refer to the tool’s man page.
ib_write_bw	Calculates the BW of RDMA write between a pair of machines. One acts as a server and the other as a client. The client RDMA writes to the server memory and calculates the BW by sampling the CPU each time it receives a successful completion. The test supports features such as Bidirectional, in which they both RDMA write to each other at the same time, change of MTU size, tx size, number of iteration, message size and more. Using the "-a" flag provides results for all message sizes. For further information, please refer to the tool’s man page.
ib_write_lat	Calculates the latency of RDMA write operation of message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which one side RDMA writes to the other side memory only after the other side wrote on his memory. Each of the sides samples the CPU clock each time they write to the other side memory, in order to calculate latency. For further information, please refer to the tool’s man page.
raw_ethernet_bw	Calculates the BW of SEND between a pair of machines. One acts as a server and the other as a client. The server receive packets from the client and they both calculate the throughput of the operation. The test supports features such as Bidirectional, on which they both send and receive at the same time, change of MTU size, tx size, number of iteration, message size and more. Using the "-a" provides results for all message sizes.

Utility	Description
	For further information, please refer to the tool’s man page.
raw_ethernet_lat	Calculates the latency of sending a packet in message_size between a pair of machines. One acts as a server and the other as a client. They perform a ping pong benchmark on which you send packet only if you receive one. Each of the sides samples the CPU each time they receive a packet in order to calculate the latency. Using the "-a" provides results for all message sizes. For further information, please refer to the tool’s man page.