SOFTWARE MANUAL

Mellanox Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™ Deployment Guide

Rev 6.0

Mellanox SHARP Software ver. 1.7.1

NOTE:
THIS HARDWARE, SOFTWARE OR TEST SUITE PRODUCT (“PRODUCT(S)”) AND ITS RELATED DOCUMENTATION ARE PROVIDED BY MELLANOX TECHNOLOGIES “AS-IS” WITH ALL FAULTS OF ANY KIND AND SOLELY FOR THE PURPOSE OF AIDING THE CUSTOMER IN TESTING APPLICATIONS THAT USE THE PRODUCTS IN DESIGNATED SOLUTIONS. THE CUSTOMER'S MANUFACTURING TEST ENVIRONMENT HAS NOT MET THE STANDARDS SET BY MELLANOX TECHNOLOGIES TO FULLY QUALIFY THE PRODUCT(S) AND/OR THE SYSTEM USING IT. THEREFORE, MELLANOX TECHNOLOGIES CANNOT AND DOES NOT GUARANTEE OR WARRANT THAT THE PRODUCTS WILL OPERATE WITH THE HIGHEST QUALITY. ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT ARE DISCLAIMED. IN NO EVENT SHALL MELLANOX BE LIABLE TO CUSTOMER OR ANY THIRD PARTIES FOR ANY DIRECT, INDIRECT, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES OF ANY KIND (INCLUDING, BUT NOT LIMITED TO, PAYMENT FOR PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY FROM THE USE OF THE PRODUCT(S) AND RELATED DOCUMENTATION EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Mellanox Technologies
350 Oakmead Parkway Suite 100
Sunnyvale, CA 94085
U.S.A.
www.mellanox.com
Tel: (408) 970-3400
Fax: (408) 970-3403

Table of Contents

Document Revision History

Table 1: Document Revision History

Revision	Date	Description

6.0	July 05, 2018	• Updated the following sections:
		  • Packages
		  • Prerequisites
		  • Running OpenMPI with Mellanox SHARP

5.0	February 28, 2018	• Updated the following sections:
		  • Packages
		  • Prerequisites
		  • Configuring Subnet Manager
		  • Configuring Aggregation Manager
		  • Running OpenMPI with Mellanox SHARP
		  • Mellanox SHARP Benchmark Script
		  • Running Mellanox SHARPD Daemon in Managed Mode

4.1	October 31, 2017	• Added the following section:
		  • sharpd Registration as a Socket Based Activated Service on all Compute Nodes
		  • Upgrading Daemons
		• Updated the following sections:
		  • Packages
		  • Prerequisites

4.0	June 13, 2017	• Added the following section:
		  • Aggregation Trees Diagnostic
		• Updated the following sections:
		  • Packages
		  • Prerequisites
		  • Configuring Subnet Manager
		  • Configuring Aggregation Manager
		  • Running Mellanox SHARP Deamons
		  • sharp_am Registration as a Service on the SM Server and its Starting
		  • sharpd Registration as Service on all Compute Nodes and its Starting

3.0	February 01st , 2017	• Added section:
		  • Job Scheduler Integration
		• Updated the following sections:
		  • Configuring Subnet Manager
		  • Configuring Aggregation Manager
		  • Running Mellanox SHARP Deamons
		  • sharpd Registration as Service on all Compute Nodes and its Starting

Revision	Date	Description

2.0	October 30th, 2016	• Added section:
		  • Setting up Mellanox SHARP Environment
		• Updated the following sections:
		  • Packages
		  • Running Mellanox SHARP Deamons
		  • Upgrading Daemons
		  • Upgrading daemons requires their removal and reregistration according to sections 6.1, 6.2, and 6.3.
		  • Running OpenMPI with Mellanox SHARP

1.0	August 15, 2016	Initial version of this document

1 Overview

Mellanox Scalable Hierarchical Aggregation and Reduction Protocol (SHARP)™ technology improves upon the performance of MPI operation, by offloading collective operations from the CPU to the switch network, and eliminating the need to send data multiple times between endpoints. This innovative approach decreases the amount of data traversing the network as aggregation nodes are reached, and dramatically reduces the MPI operations time. Implementing collective communication algorithms in the network also has additional benefits, such as freeing up valuable CPU resources for computation rather than using them to process communication.

1.1 Packages

Package	Version
MLNX OFED	4.4-x.x.x
HPC-X	2.2.x
UFM (Aggregation Manager only)	6.0.x

1.2 Prerequisites

NOTE: Mellanox SHARP requires either UFM, or a dedicated server running Subnet Manager. In the latter case, onboard Subnet Manager should be disabled in managed switches.

Name	Version
Externally managed Switch-IB 2	Firmware version: 15.1630.0210 or later
MLNX OS	3.6.8004
Subnet Manager	• 4.7 (MLNX OFED 3.3-x.x.x or UFM 5.6) or later • For Hypercube topology 4.8.1 (MLNX OFED 4.0-x.x.x or UFM 5.8). 4.7-4.8 require additional configuration in Aggregation Manager. • For virtual port support and Dragonfly+topology: 5.0.0 (MLNX OFED 4.3-x.x.x or UFM 5.10)

1.3 Supported OS and Platforms

Distro	Platform	Kernel
RHEL 6.1	x86-64	2.6.32-131.0.15
RHEL 6.2	x86-64	2.6.32-220
RHEL 6.3	x86-64	2.6.32-279
RHEL 6.4	x86-64	2.6.32-358
RHEL 6.5	x86-64	2.6.32-431
RHEL 7.0	x86-64	3.10.0-123
RHEL 7.2	x86-64	3.10.0-327
RHEL 7.2	ppcle	3.10.0-327
RHEL 7.3	ARM	4.5.0-15.el7.aarch64
RHEL 7.4	x86-64	3.10.0-693
RHEL 7.4	ARM	4.11.0-44
RHEL 7.5	x86-64	3.10.0-862
Fedora14	x86-64	2.6.35.6-45
Fedora16	x86-64	3.1.0-7
Fedora17	x86-64	3.3.4-5
Fedora18	x86-64	3.6.10-4
Fedora24	x86-64	4.5.5-300
Fedora26	x86-64	4.11.8-300
SLES 11 SP3	x86-64	3.0.76-0.11
SLES 11 SP4	x86-64	3.0.101-57
SLES 12 SP1	x86-64	3.12.49-11
SLES 12 SP2	x86-64	4.4.21-68
SLES 12 SP3	x86-64	4.4.73-5
Ubuntu14.4	x86-64	3.13.0-24
Ubuntu15.10	x86-64	4.2.0-16
Ubuntu16.10	x86-64	4.8.0-26
Ubuntu17.10	x86-64	4.13.0-17
Ubuntu18.04	x86-64	4.15.0-20
CentOS6.3	x86-64	2.6.32-279
CentOS6.0	x86-64	2.6.32-71

2 Downloading Packages

Download the HPC-X packages from the Mellanox site:
http://www.mellanox.com/page/products_dyn?product_family=189&mtag=hpc-x

3 Setting up Mellanox SHARP Environment

Mellanox SHARP binary distribution is available as part of HPC-X, MLNX_OFED or UFM packages. UFM package includes only the Aggregation Manager.

- In case of HPC-X package, please refer to HPC-X User Manual for installation and configuration procedures

  This Deployment Guide and examples refer to the following environment variables `HPCX_SHARP_DIR, OMPI_HOME` and assumes that HPC-X installation is in a shared folder accessible from all compute nodes.

- In case of MLNX_OFED distribution or custom installation, you have to set the `HPCX_SHARP_DIR` environment variable to point to the directory in which it was installed (`/opt/mellanox/sharp` is a default directory for Mellanox SHARP software in MLNX_OFED package). `OMPI_HOME` should point to the MPI installation folder.

- In case of using the Aggregation Manager from the UFM distribution, you have to enable Mellanox SHARP support in UFM. For further information, refer to the UFM User Manual.
  The rest of the Mellanox SHARP components should be installed from either the HPC-X or MLNX_OFED packages.

Make sure the following are set prior to configuring the setup:

- Mellanox SHARP configuration files must be created in the same location (please refer sections 5 and 6). Make sure that you have write permission to `HPCX_SHARP_DIR/`.

- Aggregation Manager and Mellanox SHARP daemons are trusted entities and should run by a privilege user (root user).

  Aggregation Manager is a central entity running on a dedicated server with SM. This dedicated server cannot serve as a compute node and host Mellanox SHARP daemon. Therefore, make sure onboard Subnet Manager is disabled in managed switches.

- Mellanox SHARP daemons and Aggregation Manager communicate over TCP/IP which should be configured before running Mellanox SHARP. Please check firewall settings.

- Run ibdiagnet and check the Switch-IB firmware installed. See section Prerequisites for the supported switch firmware version.

- Using UD Mcast feature requires IPoIB interface enabled in compute servers.

4 Configuring Subnet Manager

Subnet Manager should be active prior to activation of Aggregation Manager (AM). Mellanox SHARP functionality should be enabled in Switch-IB 2 switches (it is disabled by default).

- ***To enable Mellanox SHARP functionality on Switch-IB 2 based switches:***

1. Edit the opensm.conf file. Default location for MLNX_OFED opensm is /etc/opensm/opensm.conf
2. Configure the "routing_engine" parameter.

```
routing_engine updn
```

   or

```
routing_engine ftree,updn
```

3. Set the parameter "`sharp_enabled`" to "`2`".
4. Run OpenSM with the configuration file.

```
# opensm -F <opensm configuration file> -B
```

5. Verify that the Aggregation Nodes were activated by the OpenSM, run "`ibnetdiscover`".

   For example:

```
vendid=0x0
devid=0xcf09
sysimgguid=0x7cfe900300a5a2a0
caguid=0x7cfe900300a5a2a8
Ca      1 "H-7cfe900300a5a2a8"
#"Mellanox Technologies Aggregation Node" and
"[1](7cfe900300a5a2a8) "S-7cfe900300a5a2a0"
[37] # lid 256 lmc 0 "MF0;sharp2:MSB7800/U1" lid 512 4xFDR
```

- ***To configure Hypercube support in Subnet Manager:***

1. Make sure OpenSM version 4.8.1 or later is installed to support Mellanox SHARP in Hypercube fabric.
2. Configure OpenSM to use DOR routing engine.

```
routing_engine dor
```

3. Configure OpenSM to create Hypercube coordinates file.

```
dor_hyper_cube_mode TRUE
```

- ***To configure Dragonfly+support in Subnet Manager:***

1. Make sure OpenSM version 5.0.0 or later is installed to support Mellanox SHARP with Dragonfly+ topologies.
2. Use OpenSM/Adaptive Routing manager plugin manual to configure OpenSM to be able to use dragonfly+ routing.

- ***To configure virtual ports support in Subnet Manager:***

1. Make sure OpenSM version 5.0.0 or later is installed to support Mellanox SHARP with virtual ports.
2. Configure OpenSM to support virtual ports:

```
Virt_enabled 2
```

5 Configuring Aggregation Manager

Aggregation Manager (AM) is a central entity running on a dedicated server along with the Subnet Manager.

If you use AM from the UFM package, please refer to the UFM User Manual for further information.

Using OpensSM 4.9 or later does not require any special configuration in the AM for tree-based topologies.

➢ ***To configure AM using OpenSM v4.9 or later:***

Create sharp_am configuration file:

- For Hypercube topology (OpenSM ver. 4.8.1 or later):

  i. Create the `sharp_am.cfg` file:

```
# cat > $HPCX_SHARP_DIR/conf/sharp_am.cfg << EOF
topology_type hypercube
EOF
```

- For Dragonfly+ topology (OpenSM ver. 5.0.0/UFM 5.10.0 or later and Mellanox SHARP software v1.5.3 or later):

  ii. Create the `sharp_am.cfg` file:

```
# cat > $HPCX_SHARP_DIR/conf/sharp_am.cfg << EOF
topology_type dfp
EOF
```

➢ ***To configure AM with OpenSM v4.7-4.8:***

1. Create sharp_am configuration file:

   - For tree based topology:

     iii. Copy the `root_guids.conf` file if used for configuration of Subnet Manager to `$HPCX_SHARP_DIR/conf/root_guid.conf`.

     Otherwise,

     iv. Identify the root switches of the fabric and create a file with the node GUIDs of the root switches of the fabric.

     Each line in the file should contain a single node GUID in hexadecimal format. The file should be located at: `$HPCX_SHARP_DIR/conf/root_guid.conf`

     For example, if there are two root switches with node GUIDs 0x0002c90000000001 and 0x0002c90000000008, the file should be as follows:

```
0x0002c90000000001
0x0002c90000000008
```

     v. Create the `sharp_am.cfg` file:

```
# cat > $HPCX_SHARP_DIR/conf/sharp_am.cfg << EOF
root_guids_file $HPCX_SHARP_DIR/conf/root_guid.conf
ib_port_guid <PortGUID of the relevant HCA port or 0x0>
```

```
EOF
```

- For Hypercube topology (OpenSM ver. 4.8.1 or later):

  i. Create the `sharp_am.cfg` file:

```
# cat > $HPCX_SHARP_DIR/conf/sharp_am.cfg << EOF
topology_type hypercube
EOF
```

6 Running Mellanox SHARP Deamons

Mellanox SHARP software, i.e. Mellanox SHARP daemon (`sharpd`) should be executed on every compute node, and the Aggregation Manager daemon (`sharp_am`) should be executed on a dedicated server along with Subnet Manager.

NOTE: The `sharpd` and `sharp_am` commands must be executed as root user.

- `sharpd` on all compute nodes
- `sharp_am` on SM node only

➢ ***To setup the daemons the following script should be used:***

```
$HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh

Usage: sharp_daemons_setup.sh (-s | -r) [-p SHARP location dir] -d
<sharpd | sharp_am> [-m]
        -s - Setup SHARP daemon
        -r - Remove SHARP daemon
        -p - Path to alternative SHARP location dir
        -d - Daemon name (sharpd or sharp_am)
        -b - Enable socket based activation of the service
```

NOTE: Socket-based activation is only valid on systems with Systemd support.

6.1 sharp_am Registration as a Service on the SM Server and its Starting

1. Run as root the following:

```
# $HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh -s -d sharp_am
```

   Daemon's log location is: /var/log/sharp_am.log

2. Set the "run level".
3. Start sharp_am as root.

```
# service sharp_am start
```

6.2 sharpd Registration as Service on all Compute Nodes and its Starting

The procedure, described below, needs pdsh package. If you do not have pdsh, please use any other parallel execution tool and refer to the command below as an example.

1. Run as root the following:

```
# pdsh -w <hostlist> $HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh -s -d
sharpd
```

   Daemon's log location: `/var/log/sharpd.log`

2. Set the "run level"
3. Start sharpd daemons as root.

```
# pdsh -w <hostlist>  service sharpd start
```

6.3 sharpd Registration as a Socket Based Activated Service on all Compute Nodes

NOTE: Socket based activation installs sharpd as a daemon that is automatically activated when an application tries to communicate with sharpd.

Socket based activation is supported on RH 7.2 and above and requires Systemd.

The procedure, described below, needs pdsh package. If you do not have a pdsh, please use any other parallel execution tool and refer to the command below as an example.

Run as root the following:

```
# pdsh -w <hostlist> $HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh -s -d
sharpd -b
```

Daemon's log location: `/var/log/sharpd.log`

6.4 Removing Daemons

- To remove sharp_am, run on AM host:

```
# $HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh -r -d sharp_am
```

- To remove sharpd, run:

```
# pdsh -w <hostlist> $HPCX_SHARP_DIR/sbin/sharp_daemons_setup.sh -r -d
sharpd
```

6.5 Upgrading Daemons

Upgrading daemons requires their removal and reregistration according to sections 6.1, 6.2, and 6.3.

7 Running OpenMPI with Mellanox SHARP

7.1 Control Flags

The following basic flags should be used in mpirun command line to enable Mellanox SHARP protocol in HCOLL middleware. For the rest of flags please refer to Mellanox SHARP Release Notes.

FLAG	Values
HCOLL_ENABLE_SHARP	Default : 0 Possible values: • 0 – Do not use Mellanox SHARP (default) • 1 - probe Mellanox SHARP availability and use it • 2 - Force to use Mellanox SHARP • 3 - Force to use Mellanox SHARP for all MPI communicators • 4 - Force to use Mellanox SHARP for all MPI communicators and for all supported collectives(Barrier, Allreduce)
SHARP_COLL_LOG_LEVEL	Default : 2 Mellanox SHARP coll logging level. Messages with a level higher or equal to the selected will be printed. Possible values: • 0 - fatal • 1 - error • 2 - warn • 3 - info • 4 - debug • 5 - trace
SHARP_COLL_JOB_QUOTA_PAYLOAD_PER_OST	Default : 128 (Max: 256) Maximum payload per OST quota request. value 0 mean allocate default value.

For example:

```
% $OMPI_HOME/bin/mpirun --display-map --bind-to core --map-by node -H
host01,host02,host03 -np 3 -mca pml yalla -mca
btl_openib_warn_default_gid_prefix 0 -mca rmaps_dist_device mlx5_0:1 -mca
rmaps_base_mapping_policy dist:span -x MXM_RDMA_PORTS=mlx5_0:1 -x
HCOLL_MAIN_IB=mlx5_0:1 -x MXM_ASYNC_INTERVAL=1800s -x HCOLL_ENABLE_SHARP=1 -
x SHARP_COLL_JOB_QUOTA_PAYLOAD_PER_OST=128
<PATH/osu_allreduce> -i 10000 -x 1000 -f -m 256
```

The following HCOLL flags can be used when running Mellanox SHARP collective with mpirun utility:

FLAG	Values
HCOLL_SHARP_NP	Default : 2 Number of nodes(node leaders) threshold in communicator to create Mellanox SHARP group and use Mellanox SHARP collectives
HCOLL_SHARP_UPROGRESS_NUM_POLLS	Default: 999 Number of unsuccessful polling loops in libsharp coll for blocking collective wait before calling user progress (HCOLL, OMPI).
HCOLL_BCOL_P2P_ALLREDUCE_SHARP_MAX	Default : 256 Maximum allreduce size run through Mellanox SHARP. Message size greater than above will fallback to non-SHARP based algorithms (multicast based or non-multicast based)
SHARP_COLL_MAX_PAYLOAD_SIZE	Default : 256 (Max) Maximum payload size of Mellanox SHARP collective request Collective requests for larger than this size will be pipelined.
SHARP_COLL_JOB_QUOTA_PAYLOAD_PER_OST	Default : 128 (Max: 256) Maximum payload per OST quota request. value 0 mean allocate default value.
SHARP_COLL_GROUP_RESOURCE_POLICY	Default : 1 Mellanox SHARP Job resource sharing policy between the groups (communicators) Values: • 1 - equal • 2 - take_all by first group • 3 - User input percent using SHARP_COLL_USER_GROUP_QUOTA_PERCENT
SHARP_COLL_USER_GROUP_QUOTA_PERCENT	% of job quota to be allocated for each Mellanox SHARP group.
SHARP_COLL_JOB_QUOTA_OSTS	Default : 0 Maximum job (per tree) OST quota request. value 0 mean allocate default quota.
SHARP_COLL_JOB_QUOTA_MAX_GROUPS	Default: 0 Maximum no. of groups (comms) quota request. Value 0 means allocate default value.

FLAG	Values
SHARP_COLL_JOB_QUOTA_MAX_QPS_PER_PORT	Maximum QPs/port quota request. Value 0 mean allocate default value.
SHARP_COLL_PIPELINE_DEPTH	Default : 8 Size of fragmentation pipeline for larger collective payload
SHARP_COLL_STATS_FILE	Default = "" Destination to send statistics to. Possible values are: • stdout - print to standard output. • stderr - print to standard error. • file:<filename> - save to a file (%h: host, %p: pid, %t: time, %u: user, %e: exe)
SHARP_COLL_STATS_TRIGGER	Default : exit Trigger to dump statistics: • Exit - dump just before program exits. • signal:<signo> - dump when process is signaled (Not fully supported)
SHARP_COLL_STATS_DUMP_MODE	Default : 1 Stats dump modes 1 - dump per process stats 2 - dump accumulative (per job) stats **NOTE:** For accumulative mode(2), its user responsibility to call sharp_coll_dump_stats() when OOB is still active
SHARP_COLL_ENABLE_MCAST_TARGET	Default: 1 Enables MCAST target on Mellanox SHARP collective ops.
SHARP_COLL_MCAST_TARGET_GROUP_SIZE_THRESHOLD	Default: 2 Group size threshold to enable mcast targe
SHARP_COLL_POLL_BATCH	Default: 4 Defines the number of CQ completions to poll on at once. Maximum:16
SHARP_COLL_ERROR_CHECK_INTERVAL	Default: 180000 Interval, in milli second, indicates the time between the error checks.\n"

FLAG	Values
	"If you set the interval as 0, error check is not performed"
SHARP_COLL_JOB_NUM_TREES	Default: 0 Number of SHARP trees to request. 0 means to request number of trees based on number of rails and number of channels
SHARP_COLL_GROUPS_PER_COMM	Default: 1 Number of Mellanox SHARP groups per user communicator
SHARP_COLL_JOB_PRIORITY	Default: 0 Job priority
SHARP_COLL_OSTS_PER_GROUP	Default: 2 Number of OSTS per group

7.2 Running Mellanox SHARP with HCOLL - Example

```
% $OMPI_HOME/bin/mpirun --bind-to core --map-by node -hostfile /tmp/hostfile
-np 4 -mca pml yalla -mca btl_openib_warn_default_gid_prefix 0 -mca
rmaps_dist_device mlx5_0:1 -mca rmaps_base_mapping_policy dist:span -x
MXM_RDMA_PORTS=mlx5_0:1 -x HCOLL_MAIN_IB=mlx5_0:1 -x
MXM_ASYNC_INTERVAL=1800s -x MXM_LOG_LEVEL=ERROR -x HCOLL_ML_DISABLE_REDUCE=1
-x HCOLL_ENABLE_MCAST_ALL=1 -x HCOLL_MCAST_NP=1 -x
LD_LIBRARY_PATH=${LD_LIBRARY_PATH}:$HPCX_SHARP_DIR/lib -x
LD_PRELOAD=$HPCX_SHARP_DIR/lib/libsharp.so:$HPCX_SHARP_DIR/lib/libsharp_coll
.so -x HCOLL_ENABLE_SHARP=2 -x SHARP_COLL_LOG_LEVEL=3 -x
SHARP_COLL_GROUP_RESOURCE_POLICY=1 -x SHARP_COLL_MAX_PAYLOAD_SIZE=256 -x
HCOLL_SHARP_UPROGRESS_NUM_POLLS=999 -x
HCOLL_BCOL_P2P_ALLREDUCE_SHARP_MAX=4096 -x SHARP_COLL_PIPELINE_DEPTH=32 -x
SHARP_COLL_JOB_QUOTA_OSTS=32 -x SHARP_COLL_JOB_QUOTA_MAX_GROUPS=4 -x
SHARP_COLL_JOB_QUOTA_PAYLOAD_PER_OST=256 taskset -c 1 numactl --membind=0
<PATH/osu_allreduce> -i 100 -x 100 -f -m 4096:4096
```

NOTE: For the complete list of SHARP_COLL tuning options, run the `sharp_coll_dump_config` utility.

```
$HPCX_SHARP_DIR/bin/sharp_coll_dump_config -f
```

8 Sanity Testing

8.1 Aggregation Trees Diagnostic

Run ibdiagnet tool and check /var/tmp/ibdiagnet2/ibdiagnet2.sharp

```
# ibdiagnet --sharp
```

Example:

```
# This database file was automatically generated by IBDIAG


TreeID:0, Max Radix:2
(0), AN:Mellanox Technologies Aggregation Node, lid:26, port
guid:0x248a070300ea8028, Child index:0, parent QPn:0, remote parent QPn:0,
radix:2
        (1), AN:Mellanox Technologies Aggregation Node, lid:22, port
guid:0x248a070300ea8068, Child index:0, parent QPn:5246977, remote parent
QPn:5244929, radix:0
        (1), AN:Mellanox Technologies Aggregation Node, lid:21, port
guid:0x248a070300ea8048, Child index:1, parent QPn:5242881, remote parent
QPn:5246978, radix:0

TreeID:1, Max Radix:2
(0), AN:Mellanox Technologies Aggregation Node, lid:30, port
guid:0x7cfe900300bf85d8, Child index:0, parent QPn:0, remote parent QPn:0,
radix:2
        (1), AN:Mellanox Technologies Aggregation Node, lid:22, port
guid:0x248a070300ea8068, Child index:0, parent QPn:5242882, remote parent
QPn:15771649, radix:0
        (1), AN:Mellanox Technologies Aggregation Node, lid:21, port
guid:0x248a070300ea8048, Child index:1, parent QPn:5249026, remote parent
QPn:15773698, radix:0


AN:Mellanox Technologies Aggregation Node, lid:26, node
guid:0x248a070300ea8028
QPN:5244929, State:1, TS:0x00000000, G:0, SL:0, RLID:22, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5246977, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31
QPN:5246978, State:1, TS:0x00000000, G:0, SL:0, RLID:21, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5242881, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31

AN:Mellanox Technologies Aggregation Node, lid:21, node
guid:0x248a070300ea8048
QPN:5242881, State:1, TS:0x00000000, G:0, SL:0, RLID:26, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5246978, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31
QPN:5249026, State:1, TS:0x00000000, G:0, SL:0, RLID:30, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:15773698,
RNR Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31

AN:Mellanox Technologies Aggregation Node, lid:22, node
guid:0x248a070300ea8068
QPN:5242882, State:1, TS:0x00000000, G:0, SL:0, RLID:30, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:15771649,
RNR Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31
```

```
QPN:5246977, State:1, TS:0x00000000, G:0, SL:0, RLID:26, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5244929, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31

AN:Mellanox Technologies Aggregation Node, lid:30, node
guid:0x7cfe900300bf85d8
QPN:15771649, State:1, TS:0x00000000, G:0, SL:0, RLID:22, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5242882, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31
QPN:15773698, State:1, TS:0x00000000, G:0, SL:0, RLID:21, Traffic Class:0,
Hop Limit:0, RGID:::, RQ PSN:0, SQ PSN:0, PKey:0x0000ffff, RQPN:5249026, RNR
Mode:0, RNR Retry Limit:0x00000007, Timeout Retry Limit:7, Local Ack
Timeout:31
```

8.2 Mellanox SHARP Benchmark Script

Mellanox SHARP distribution provides a test script which executes OSU (allreduce, barrier) benchmark running with and without Mellanox SHARP. To run the Mellanox SHARP benchmark script the following prerequisites are required:

- ssh
- pdsh
- environment-modules.x86_64

You can find this script at `$HPCX_SHARP_DIR/sbin/sharp_benchmark.sh`. This script should be launched from a host running SM and Aggregation Manager. It receives a list of compute host from SLURM allocation or from “hostlist” environment variable. “hostlist” is comma separated list. Also it requires hca environment variables to be supplied. It runs OSU all reduce and OSU barrier benchmarks with and without Mellanox SHARP.

Help:

```
This script includes OSU benchmarks for MPI_Allreduce and MPI_Barrier
blocking collective operations.
Both benchmarks run with and without using SHARP technology.

Usage: sharp_benchmark.sh [-t] [-d] [-h] [-f]
        -t - tests list (e.g. sharp:barrier)
        -d - dry run
        -h - display this help and exit
        -f - supress error in prerequsites checking

Configuration:
 Runtime:
  sharp_ppn - number of processes per compute node (default 1)
  sharp_ib_dev - Infiniband device used for communication. Format
<device_name>:<port_number>.
                 For example: sharp_ib_dev="mlx5_0:1"
                 This is a mandatory parameter. If it's absent,
sharp_benchmark.sh tries to use the first active device on local machine
  sharp_groups_num - number of groups per communicator. (default is the
number of devices in sharp_ib_dev)
  sharp_num_trees - number of trees to request. (default num tress based on
the #rails and #channels)
  sharp_job_members_type - type of sharp job members list. (default is
SHARP_MEMBER_LIST_PROCESSES_DATA)
  sharp_hostlist - hostnames of compute nodes used in the benchmark. The
list may include normal host names,
                   a range of hosts in hostlist format. Under SLURM
allocation, SLURM_NODELIST is used as a default
```

```
  sharp_test_iters - number of test iterations (default 10000)
  sharp_test_skip_iters - number of test iterations (default 1000)
  sharp_test_max_data - max data size used for testing (default and maximum
4096)
 Environment:
  SHARP_INI_FILE - takes configuration from given file instead of
/labhome/danielk/.sharp_benchmark.ini
  SHARP_TMP_DIR - store temporary files here instead of /tmp
  HCOLL_INSTALL - use specified hcoll install instead from hpcx

Examples:
  sharp_ib_dev="mlx5_0:1" sharp_benchmark.sh  # run using "mlx5_0:1" IB
port. Rest parameters are loaded from /labhome/danielk/.sharp_benchmark.ini
or default
  SHARP_INI_FILE=~/benchmark.ini  sharp_benchmark.sh # Override default
configuration file
  SHARP_INI_FILE=~/benchmark.ini  sharp_hostlist=ajna0[2-3]
sharp_ib_dev="mlx5_0:1" sharp_benchmark.sh # Use specific host list
  sharp_ppn=1 sharp_hostlist=ajna0[1-8] sharp_ib_dev="mlx5_0:1"
sharp_benchmark.sh  -d # Print commands without actual run

Dependencies:
  This script uses "python-hostlist" package. Visit
https://www.nsc.liu.se/~kent/python-hostlist/ for details
```

9 Job Scheduler Integration

9.1 Running Mellanox SHARPD Daemon in Managed Mode

When running the daemon in a managed mode, it expects communication from the `prolog/epilog` scripts of the Job Scheduler (JS). The prolog/epilog scripts should invoke the "`sharp_job_quota`" executable to communicate with Mellanox SHARP.

To run SHARPD in managed mode, use the "`mgmt_mode`" option (default: 0 – run in "unmanaged" mode).

JS can set/unset upper limit for Mellanox SHARP resources (e.g OSTs, groups and etc.) allowed for a particular user/job via `sharp_job_quota` using the "`set`" and "`remove`" commands.

Usage

```
sharp_job_quota [OPTIONS]
```

sharp_job_quota option

	Required/ Optional	Arguments	Description
`-t, --operation`	Required	set / remove	Sets or removes quota
`-i, --allocation-id`	Required	Unique numeric 64-bit ID	This is the scheduler id for the job. No other job in the system at the same time can have the same id
`-u, --uid`	Optional	Numeric	UID of the user allowed to run the job
`-n, --user_name`	Optional	string	Name of the user allowed to run the job
`--coll_job_quota_max_groups`	Optional	Numeric value: 0..256	Maximum number of Mellanox SHARP groups (communicators) allowed. Default value: 0. 0 means there is not limit for the job. It can ask for any number.
`--coll_job_quota_max_qps_per_port`	Optional	Numeric value: 0..256	Maximum QPs/port allowed. Default value: 0. 0 means there is not

	Required/ Optional	Arguments	Description
			limit for the job. It can ask for any number.
--coll_job_quota_max_payload_per_ost	Optional	Numeric value: 0..256	Maximum payload per OST allowed. Default value: 256
--coll_job_quota_max_osts	Optional	Numeric value: 0..512	Indicates the maximum number of OSTs allowed for job per collective operation. Default value: 0. 0 means there is not limit for the job. It can ask for any number.
-- coll_job_quota_max_num_trees	Optional	Numeric Value: 0..4	Indicates the maximum number of trees allowed for the job.
--job_priority	Optional	Numeric value 0..9	Indicates priority of the job.
-- coll_job_quota_percentage	Optional	Number value 0..100	Indicates percentage of resources to request for the job.

Important Notes:

- The executable needs to run with the same user as the SD (root).
- When using the "set" operation either the uid or the user_name must be provided
- Regardless of the job quota set in prolog, the AM can allocate less resources than requested or decline the request

Examples

```
# sharp_job_quota --operation set --user_name jobrunner --allocation_id 2017
--coll_job_quota_max_groups 10
# sharp_job_quota --operation remove --allocation_id 2017
```

SLURM Examples

```
#sharp_job_quota --operation set --uid $SLURM_JOB_UID --allocation_id
$SLURM_JOB_ID
#sharp_job_quota --operation remove --allocation_id $SLURM_JOB_ID
```