



RDMA Virtualization



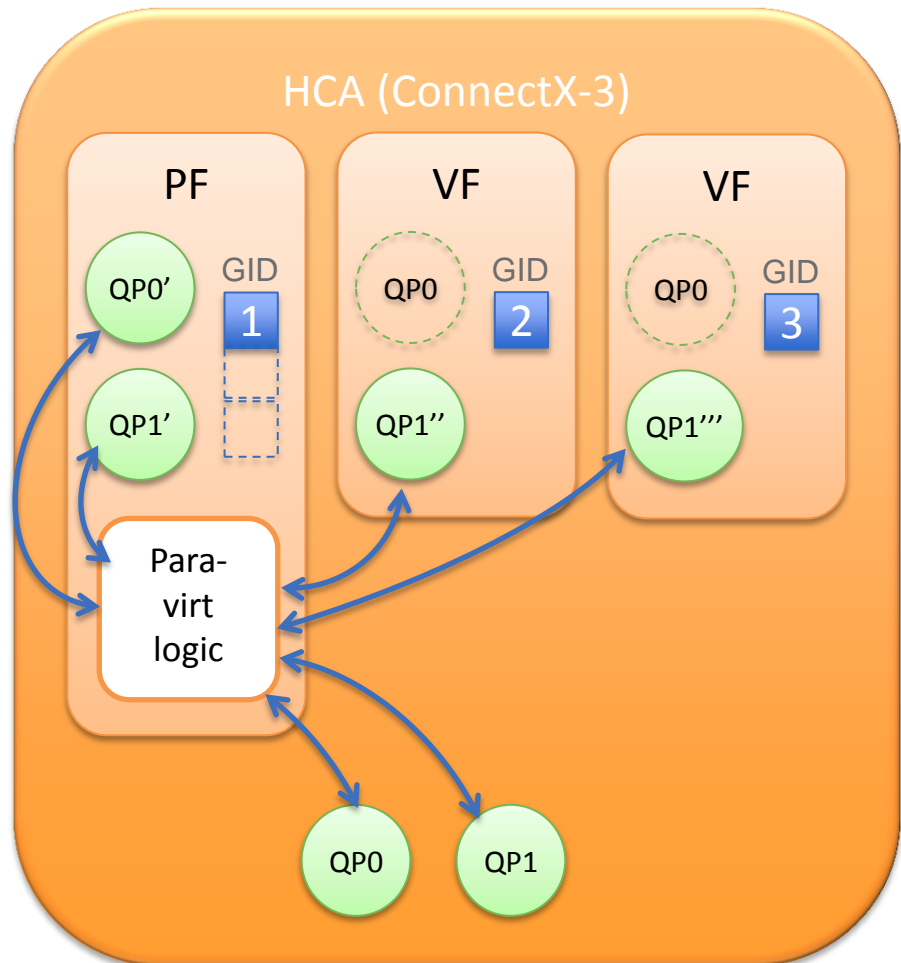
Alex Netes and Liran Liss
Mellanox Technologies

Agenda

- RDMA virtualization today
- Requirements and solution space
- Introducing Virtual Ports (VPorts)
- Virtualization software
 - Model
 - Subnet management
 - Subnet administration
 - Host stack
 - Backward compatibility
- Hypervisor APIs
- Conclusions

RDMA Virtualization Today

- Shared port model
- vHCAs are emulated by PF driver
 - Transaction ID mapping
 - GMP multiplexing
 - Multicast proxy
 - InformInfo proxy
 - Connection management proxy
- Not visible to Subnet Management
- Per-device sysfs APIs

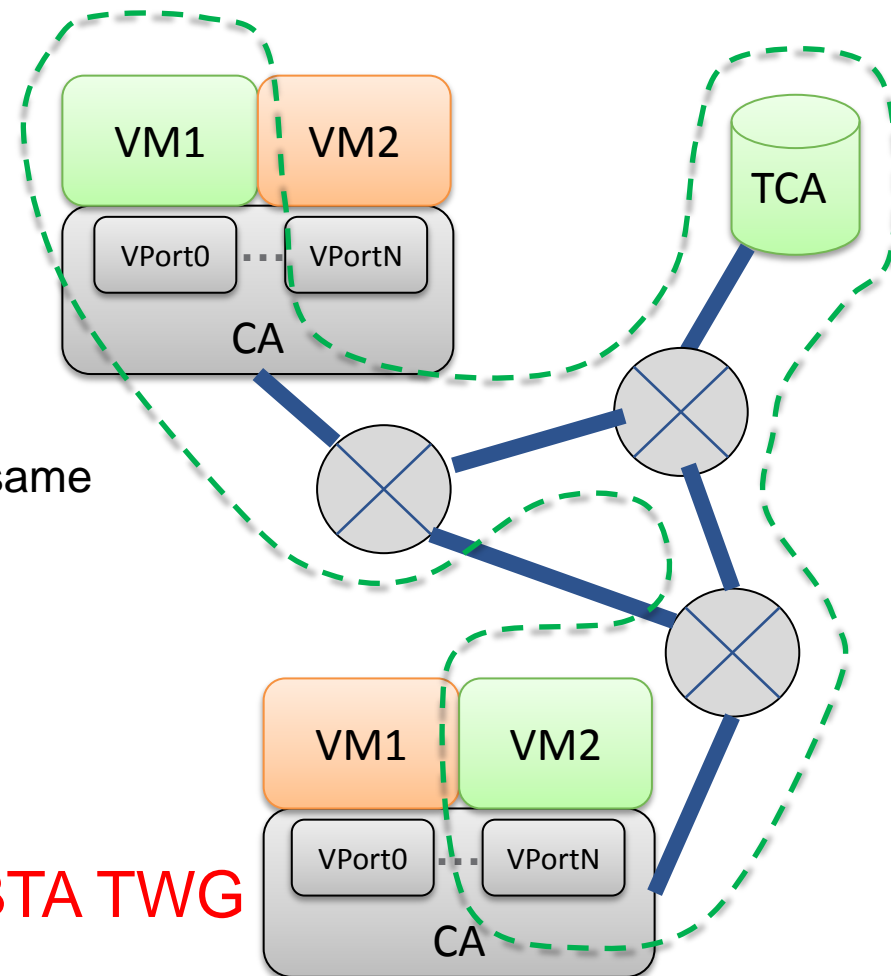


Virtualization Requirements

- Scalable
- Explicit Subnet Management
 - Virtual endpoints get full representation
- Simplicity
 - No PF-VF driver communication
 - No para-virtualization
- Backward compatibility
 - Legacy SM
 - Legacy nodes

Virtualization Approach

- Solution space
 - (Emulated) IB switch
 - (Emulated) extended IB switch
 - Virtual Port (VPort) array
- Proposal: VPort array
 - Similar to SRIOV
 - Multiple PFs and VFs share the same function space on the PCI
 - Similar to NPIV
 - vHCAs have a unique ID
 - But share the physical port
 - Recognized by the fabric
 - LUN masking, zoning, etc.
- Initial presentation given to IBTA TWG

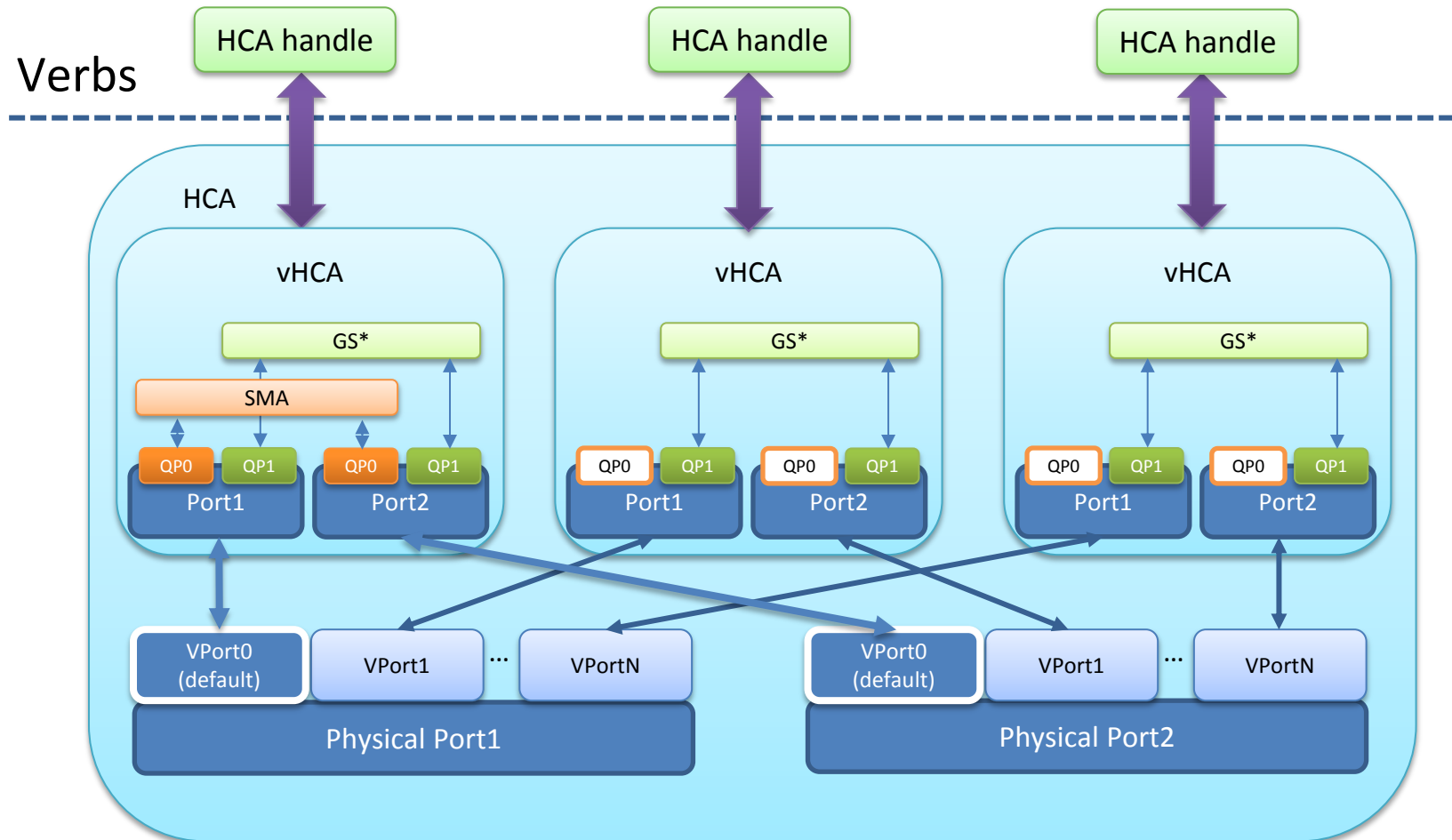


Virtualization Software



- Implications of virtual transport endpoints (VPort array)
 - Device model
 - VPort properties
 - OpenSM
 - Subnet management
 - Subnet administration
 - Host stack
 - Backward compatibility
- Track IBTA progress
 - Model, packet relay, Verbs semantics, management, MAD formats, etc.

Device Model



VPort Properties

- Independent transport attributes
 - Gid Table
 - P_KeyTable
 - (Logical) LinkState
 - Capability Mask
 - P_KeyViolations counter
 - Q_KeyViolations counter
 - ClientReregister
- L2 attributes are shared
 - LID, LMC, SL2VL, VL arbitration, etc.

Subnet Management

- Virtualization is an extended CA capability
 - Enabled by a virtualization-aware SM
- Virtual ports discovered and configured just like physical ports
 - Partitioning
 - PortState
- MAD processing
 - Virtualization discovery and management
 - Target physical port
 - Virtual ports properties
 - PortState, P_Keys, etc.
 - Target specific VPort
 - Dynamic Virtual Port monitoring
 - Aggregate VPortState
 - VPortState Change Trap

OpenSM Operation

- Fabric initialization
 - Physical subnet discovery and initialization
 - Fabric sweep, routing configuration, port initialization
 - Virtual Ports discovery and initialization
 - Discover enabled VPorts, configure partitioning
- Fabric maintenance
 - Physical fabric changes
 - Periodic sweep or PortState Change trap
 - Virtual Ports changes
 - VPort State Change trap from a physical port

Subnet Administration

- Process GMPs with GRH
 - Assuming that VPorts are identified by GIDs
- Partition checks apply to VPort P_Key tables
- SA query subsystem VPort support for
 - PathRecord
 - MCMemberRecord
 - InformInfoRecord
 - ServiceRecord
 - MultiPathRecord

Host Stack

- Verbs
 - **ibv_query_device()** returns VHCA properties
 - **ibv_query_port()** returns VPort transport properties
 - Transport APIs refer to VHCA resources
- Unaffiliated asynchronous errors and events
 - Transport events
 - Refer to VHCA transport resources
 - **IBV_EVENT_PORT_ACTIVE/_ERR**
 - Refers to VPortState
 - **IBV_EVENT_CLIENT_REREGISTER**
 - Reported for both physical and virtual ports
 - **IBV_EVENT_GID_CHANGE** and **IBV_EVENT_PKEY_CHANGE**
 - Refers to changes in VPort GID and P_Key tables

Host Stack (cont.)

- Virtualization is not transparent to software
 - Concise with explicit IB management
 - Mostly informational
 - Used only by OpenSM or other management utilities

Essentially *no* changes to OFED applications that work with GIDs

Backward Compatibility

- Possible approach: nominate one VPort as special
 - E.g., VPort0 on each physical port
 - Typically would be assigned to the PF in SRIOV
- Mirrors physical port transport attributes
 - GID table, P_Key Table, Capabilities, etc.
- Default steering target
 - Traffic with no GRH
 - Miss on DGID match
- Privileged
 - SMPs
 - Raw Ethertype
 - Raw IPv6

Provides **full** backward compatibility with

- Legacy SM
- Legacy OFED Stack running on PF
- Legacy OFED stack running on peers

Hypervisor APIs

- Control VF
 - Identity
 - Port state
 - QoS (e.g., rate limit)
 - Resource quotas
- Exposed by PF IPoIB interfaces (ndo_* ops)
 - Reuse existing functions for IB ports

```
int (*ndo_set_vf_rate)(struct net_device *dev, int vf, int min_tx_rate, int max_tx_rate);  
int (*ndo_set_vf_link_state)(struct net_device *dev, int vf, int link_state);
```

- Extend rtnetlink + ndo_* ops for IB specific operations

```
int (*ndo_set_vf_node_guid)(struct net_device *dev, int vf, u64 guid);  
int (*ndo_set_vf_port_guid)(struct net_device *dev, int vf, u64 guid); /* port implied by netdev */  
int (*ndo_set_vf_hca_resources)(struct net_device *dev, int vf, struct nlattr *resources[]);
```

Conclusions

- The time is ripe for RDMA virtualization
- Virtualization should be an equal citizen in RDMA fabrics
 - Discovery
 - Network services
- IBTA requested to take on standardization
 - Virtual transport endpoints (VPorts) look like a promising direction
- Virtualization software required!
 - Host stack, OpenSM, management tools



Thank You



#OFADevWorkshop