OPENFABRICS ALLIANCE

RDMA Container Support

11TH ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP

Liran Liss

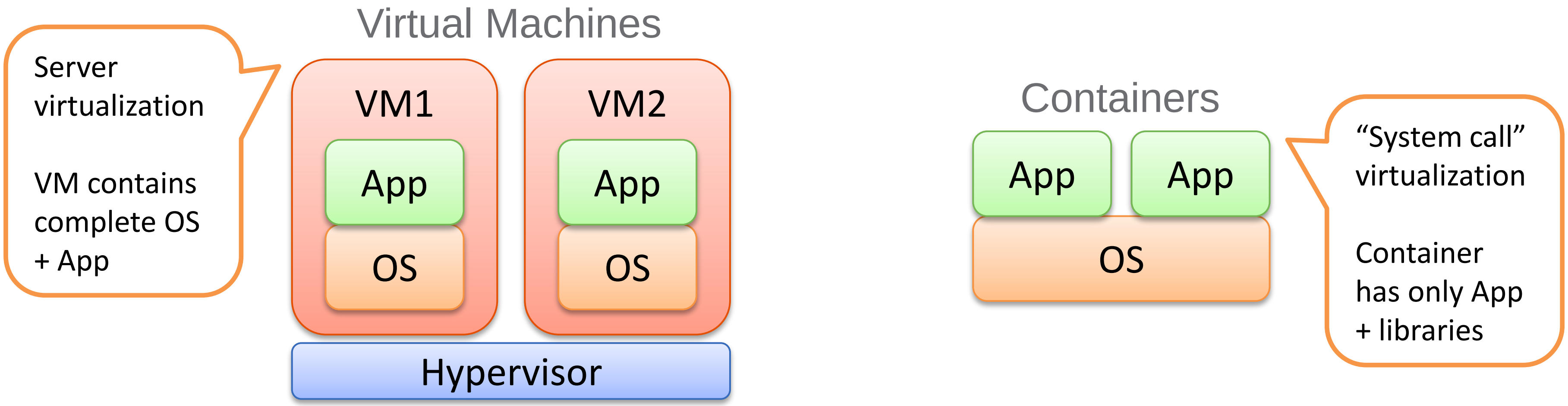
Mellanox Technologies

Agenda

- Containers 101
- RDMA isolation
- Namespace support
- Controller support
- Putting it all together
- Status
- Conclusions

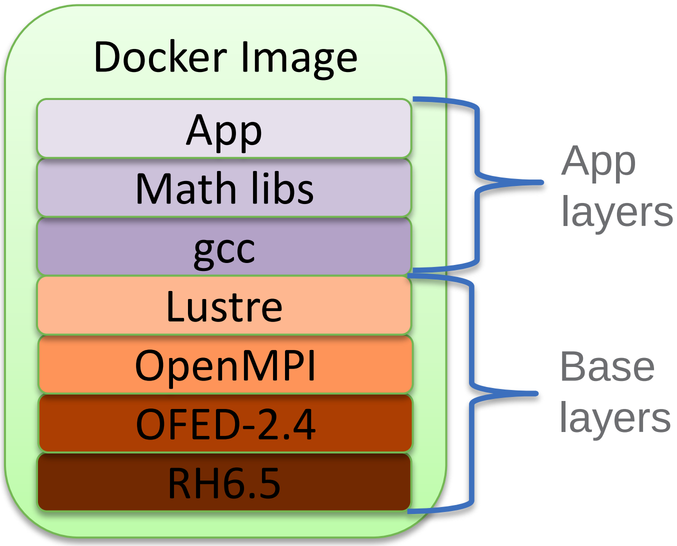
Containers 101

- A server-virtualization technology for running multiple isolated user-space instances
- Each instance
  - Has the look and feel of running over a dedicated server
  - Cannot impact the activity of other instances
- Containers and Virtual Machines (VMs) provide virtualization at different levels



Example: Docker

- Open platform to build, ship, and run distributed apps
  - Based on Linux container technology
- Main promise
  - Easily package an application and its dependencies
    - Regardless of the language, tool chain, and distribution
    - Layered images
    - Large application repository
      - Basis for further specialization
  - Deploy on any Server
    - Regardless of OS distribution
    - Regardless of underlying architecture
  - Lightweight runtime
    - Rapid scale-up/down of services



Linux Containers = Namespaces + cgroups

- Namespaces
  - Provide the illusion of running in isolation
  - Implemented for multiple OS sub-systems
- cgroups
  - Restrict resource utilization
  - Controllers for multiple resource types

Namespace examples

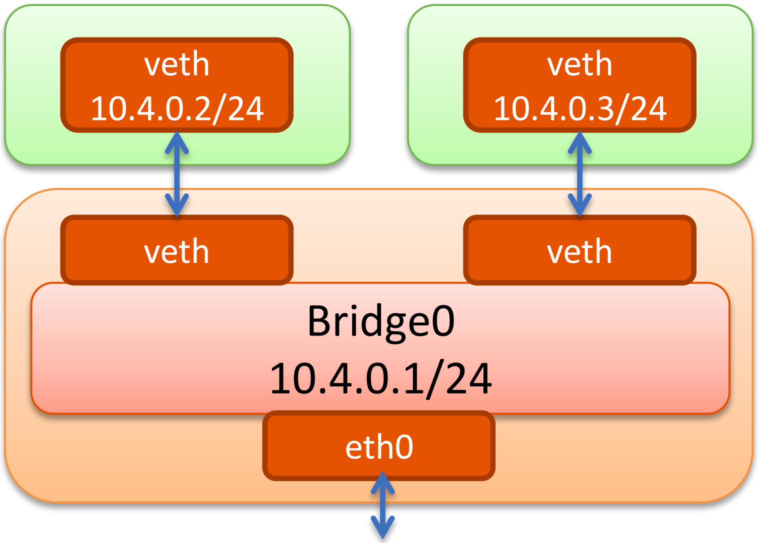
Name space	Description
pid	Process IDs
net	Network interfaces, routing tables, and netfilter
ipc	Semaphores, shared memory, and message queues
mnt	Root and file-system mounts
uts	Host name
uid	User IDs

cgroup examples

Controller	Description
blkio	Access to block devices
cpu	CPU time
cpuset	CPU cores
devices	Device access
memory	Memory usage
net_cls	Packet classification
net_prio	Packet priority

Container IP Networking

- Common models
  - Host (e.g., Mezos)
  - Physical interface / VLAN / macvlan
    - Container has global IP
  - Bridge
    - Container has global IP
  - Pod (e.g., GCE)
    - Multi-container scheduling unit
    - Global IP per POD
  - NAT (e.g., Docker)
  - Tunneling
- Building blocks
  - Network namespaces
    - Interfaces, IP tables, netfilter
  - Virtual networking
    - bridge, ovs, NAT
    - macvlan, vlan, veth



RDMA Isolation Design Goals

- Simplicity and efficiency
  - Containers share the same RDMA device instance
  - Leverage existing isolation infrastructure
    - Network namespaces and cgroups
- Focus on application APIs
  - Verbs / RDMACM
  - Exclude management and low-level APIs (e.g., umad, ucm)
    - Deny access using device controller
  - Exclude kernel ULPs (e.g., iSER, SRP)
    - Not directly exposed to applications
    - Controlled by other means (blk_io)
    - Subject for future work

Namespace Observations

OPENFABRICS ALLIANCE

- Isolating Verbs resources is not necessary
  - Only QPNs and RKeys are visible on the wire
  - Both don't have well-known names
    - Applications don't choose them
- rdmacm maps nicely to network namespaces
  - IP addresses stem from network interfaces
  - Protocols and port numbers map to ServiceID port-spaces
- RoCE requires a namespace for L3→L2 address resolution

- Namespace determined by interface
  - Physical port interfaces of PFs/VFs
  - P_Key child devices
  - Additional child devices on same P_Key
  - VLAN child devices
  - macvlan child-devices

Conclusions

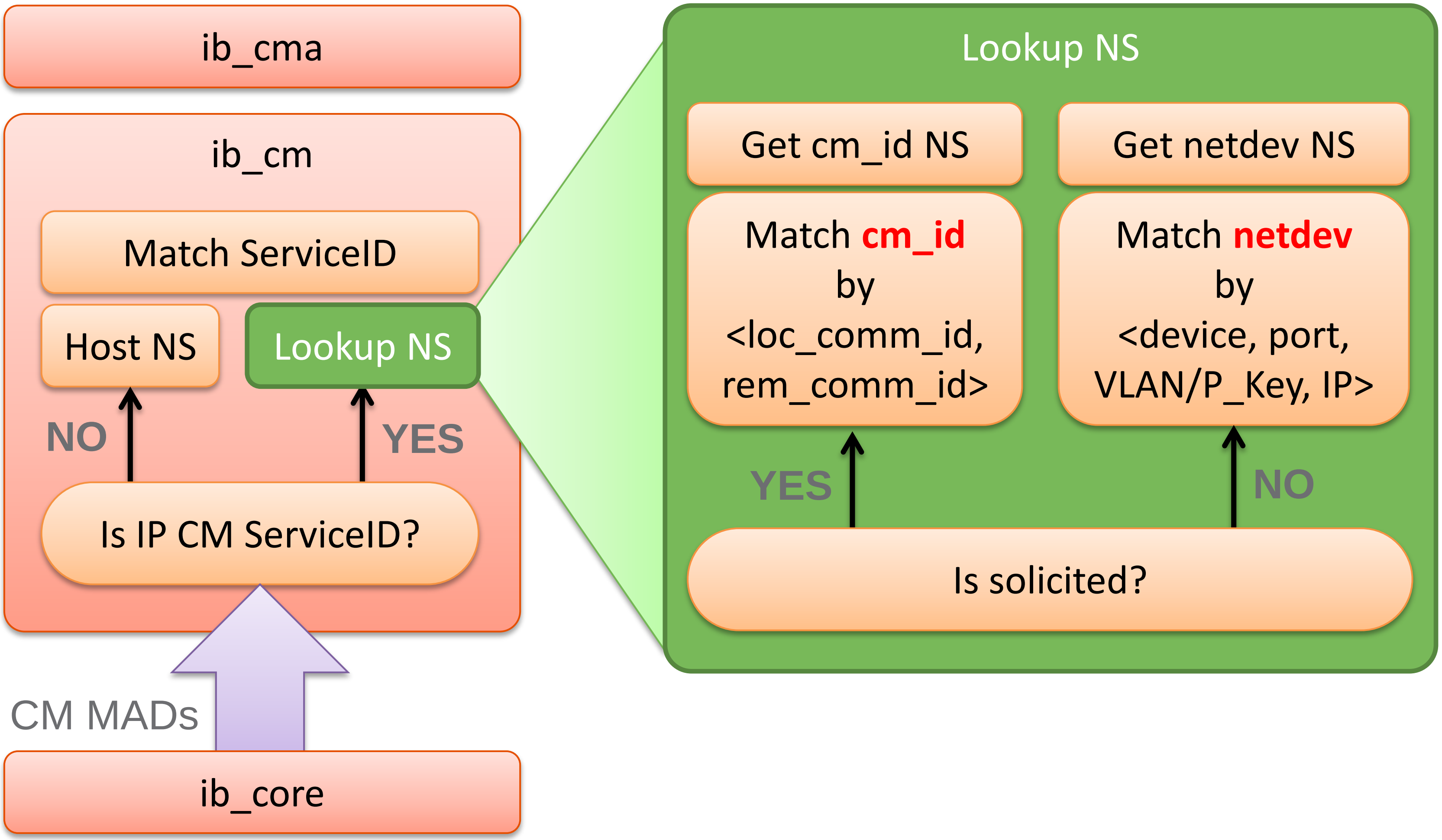
- QP and AH API calls should be processed within a namespace context
- Associate RDMA IDs with namespaces
- Maintain isolated ServiceID port-space per network namespace

Resource Namespace Association

- QP and AH namespaces
  - Used for RoCE L3→L2 address resolution
  - Determined by the process namespace during API calls
  - Default to Host namespace for kernel threads

- RDMA IDs namespaces
  - Used for Binding to ServiceIDs and solicited MAD steering (see below)
  - Determined by the process namespace upon creation
  - Matched asynchronously with incoming requests
  - Default to Host namespace for kernel threads

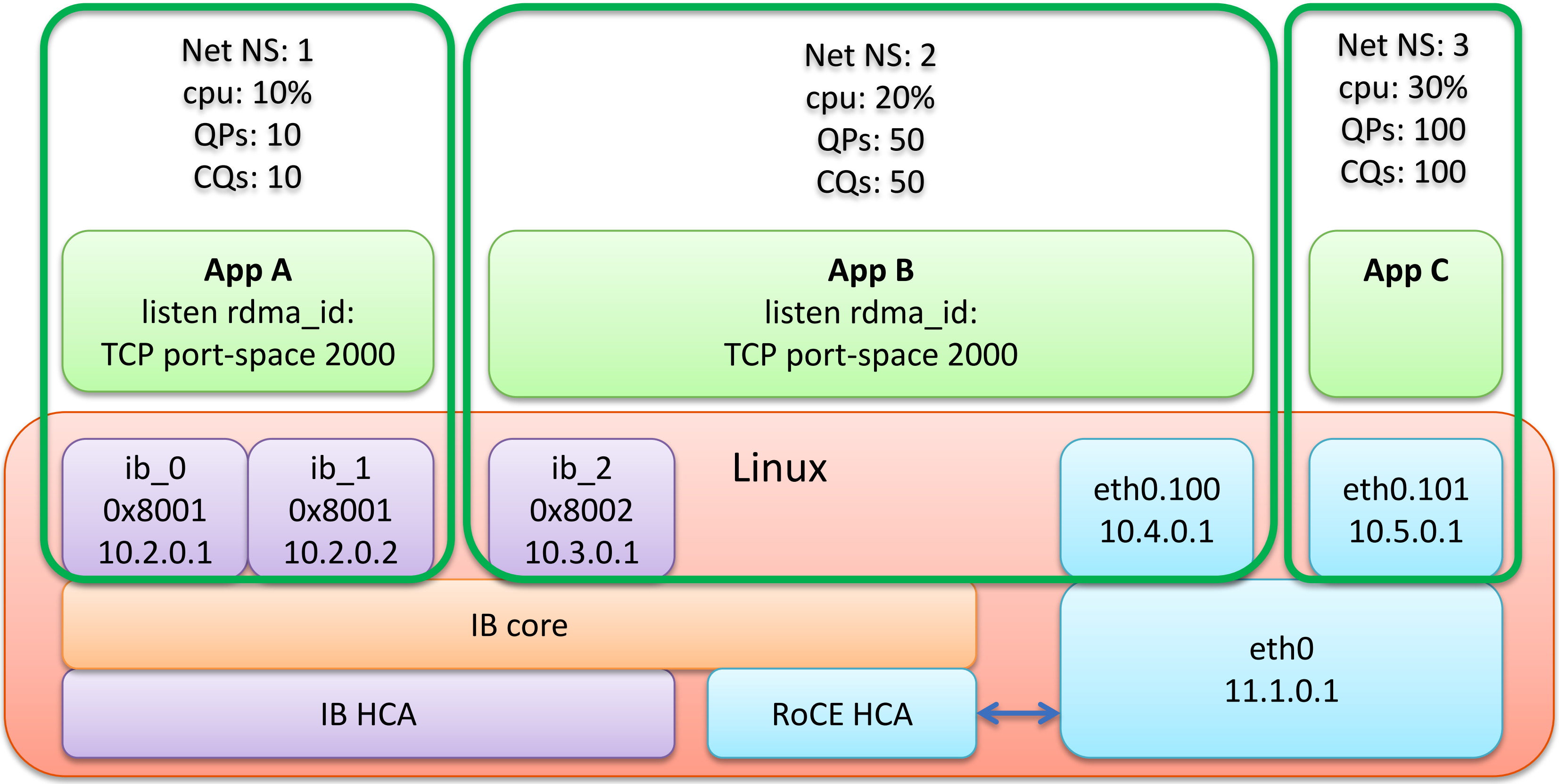
MAD ServiceID Resolution



RDMA cgroup Controller

- Governs application resource utilization per RDMA device
  - For a process or a group of processes
- Possible controlled resources
  - Opened HCA contexts
  - CQs, PDs, QPs, SRQs, MRs
  - Service Levels (SLs) and User Priorities (UPs)
    - Can’t mark individual packets in SW…

Putting it All Together



Status

- ServiceID namespace support for IB completed
  - May be used with any IPoIB interface or child interface
  - Patches sent upstream
- Coming up
  - RoCE support
  - RDMA cgroup controllers

Conclusions

- Container technology is gaining considerable traction
- The intrinsic efficiency of containers make them an attractive virtualization and deployment solution for high-performance applications
  - E.g., HPC clouds
- RDMA container support provides such applications access to high-performance networking in a secure and isolated manner

Thank You

OPENFABRICS ALLIANCE

11TH ANNUAL INTERNATIONAL OPENFABRICS SOFTWARE DEVELOPERS' WORKSHOP