PARAVIRTUAL RDMA DEVICE
Aditya Sarwade, Adit Ranadive, Jorgen Hansen, Bhavesh Davda, George Zhang, Shelley Gong

VMware, Inc.

[ April 5th, 2016 ]
MOTIVATION

RDMA Enables
- OS bypass
- Zero-copy
- Low Latency (<1µs)
- High Bandwidth

Why not PCI Passthrough?
- No live migration support
- Transport dependent
- Needs an HCA
- Cannot share non-SRIOV HCA
INTRODUCTION

• Paravirtual RDMA (PVRDMA) is a new PCIe virtual NIC
• Supports standard Verbs API
• Uses HCA for performance, but works without it
• Multiple virtual devices can share an HCA without SR-IOV
• Supports vMotion (live migration)!
ARCHITECTURE

- Exposes a dual function PCIe device to the guest
  - VMXNET3
  - RDMA (RoCE)
- RDMA component reuses Ethernet properties from the paired NIC
- Plugs into the OFED stack in the VM
- Provides verbs-level emulation
  - Guest kernel driver
  - User level library
- Operates over ESX RDMA stack (VMkernel)
- GIDs generated by guest kernel registered with HCA
• Virtualize some hardware resources (like QPs and MRs)
  • Required for vMotion
  • Create corresponding physical resources on the HCA
ARCHITECTURE (CONT.)

- Guest MR registered directly with the HCA
  - Guest PA converted to machine addresses
  - Zero-copy

- Diagram:
  - Application buffer
  - Guest VA, length
  - Guest PA list
  - MA (host PA) list
  - Guest VA -> MA list
  - HCA
  - Guest kernel
  - Device emulation
  - Guest userspace
CONTROL AND DATA PATH

- **Guest OS 1**
  - RDMA App
  - Buffers
  - libvrdma
  - libibverbs
  - PVRDMA Driver
  - PVRDMA NIC
  - PVRDMA Device Emulation
  - ESXi RDMA Stack
  - HCA Device Driver

- **Guest OS 2**
  - RDMA App
  - Buffers
  - libvrdma
  - libibverbs
  - PVRDMA Driver
  - PVRDMA Device Emulation
  - ESXi RDMA Stack
  - HCA Device Driver

- **RoCE** (RDMA over Converged Ethernet)
RDMA TRANSPORT SELECTION

- **PVRDMA Transport Selection**
  - **Memcpy** – RDMA between peers on same host
  - **TCP** – RDMA between peers without HCAs (slow path)
  - **RDMA** – Fast Path RDMA between peers with HCAs

- **PVRDMA vMotion**
  - Leverage transport selection to support vMotion of RDMA VMs
**Challenge:**

- Lots of RDMA state within hardware
- Physical resource IDs (like QPNs/MR keys) may change after migration
- Peers will not be aware of the new IDs
- Currently, no support to create resources with specified IDs
Current (partial) solution:-

- Emulation layer can get virtual to physical translations from peer
- Notify peer about vMotion and pause QP/CQ processing
- After vMotion resume QPs with the new translations
- Invisible to guest
- Can only work when both endpoints are VMs
vMOTION (FUTURE WORK)

- Support vMotion when one of endpoints is native (non-VM)
- Need hardware support
- Recreate specific QPNs and MR keys
- Ability to pause and resume QP state on the hardware
  - Save/Restore intermediate QP states
- Provide isolated resource space to each PVRDMA device
  - Guarantee that specified resources can be recreated
  - Avoid collisions with existing resources
- Expose hardware resources directly to guest
  - Lower virtualization overhead
PERFORMANCE

- **Testbed**
  - 2 x Dell T320 Hosts E5-2440 @ 2.40GHz, 24 GiB, Mellanox ConnectX - 3
  - VMs: Ubuntu 12.04, 3.5.0.45, x86_64, 2 vCPUs, 2 GiB
  - OFED Send Latency Test
    - Half RTT for 10K iterations

![Graph showing latency vs. message size for different transport methods: vRDMA over RDMA, vRDMA over Memcpy, and Native. The latency increases with message size and is higher for vRDMA over RDMA compared to the other two methods.](image-url)
CURRENT LIMITATIONS

• Communication between VM and native endpoints not supported
  • Need a way to create resources with specified IDs
  • May need additional hardware support from vendors
  • Formalize vMotion support on hardware

• Currently only supports RoCEv1 in the guest
  • Can still operate over underlying RoCEv2-only HCA
  • No InfiniBand/iWARP support (future work)

• No remote READ/WRITE support on DMA MRs

• No SRQ/Atomics support yet
  • SRQs not currently supported on host ESX

• Only supports Linux guests currently

• No failover support for PVRDMA
THANK YOU

Aditya Sarwade [asarwade@vmware.com]

VMware, Inc.