Virtualization Technologies In Modern HPC Systems

Radoslav Dimitrov, Kevin Pouget, Angelos Mouzakitis, Alvise Rigo

Virtual Open Systems

Exascale HPC Workshop, Ljubljana, Slovenia

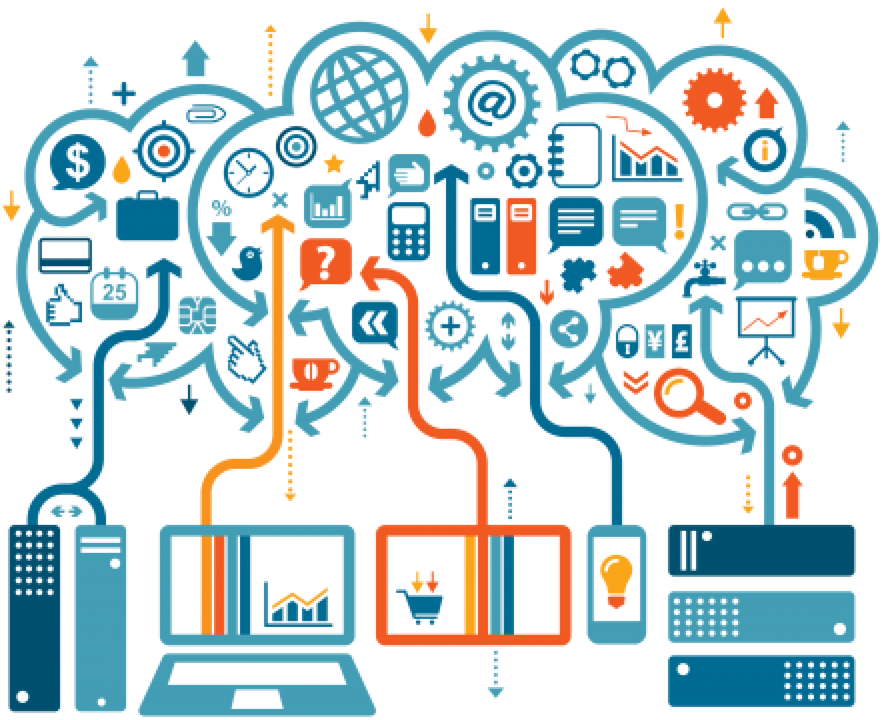
31 May 2018

Horizon 2020

Virtualization Technologies In Modern HPC Systems

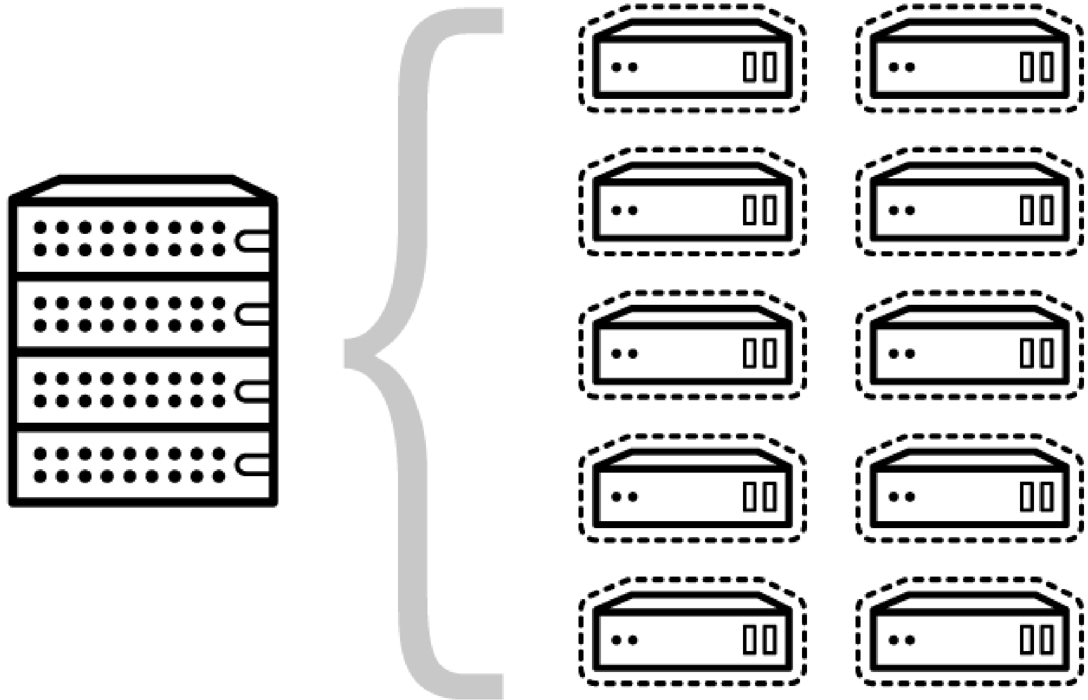
- Virtualization in HPC Systems
- Software Switches
- API Remoting
- Conclusions

Why virtualization on HPC?



- Increasing amount of data from various sources
- More industries adapting to fields like Big Data analytics
- Growing demand for HPC systems

The benefits of virtualization in HPC

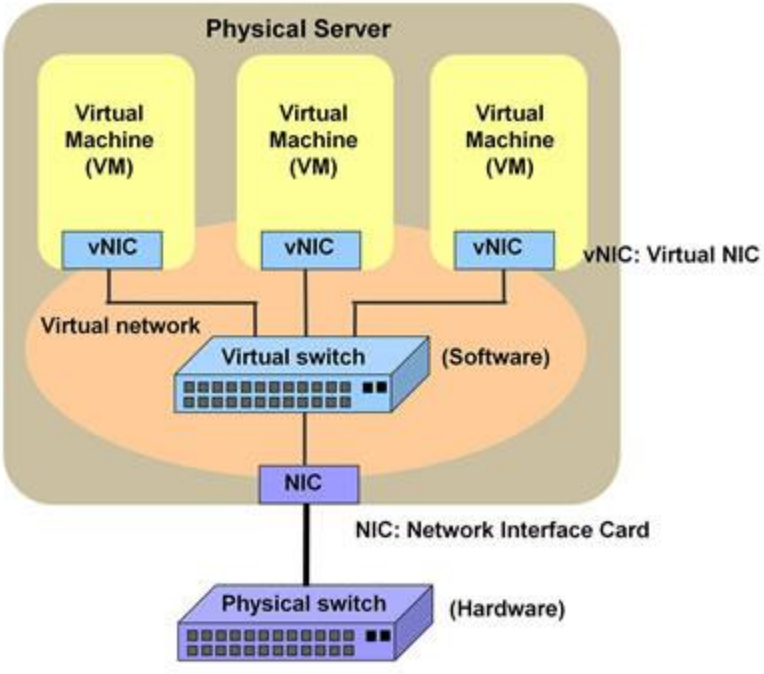


- Virtualization can help to cope with the increasing demand
- Sharing compute resources means better hardware utilization of the HPC system
- Customize and orchestrate the HPC system specifically for the workload and deliver features like auto-scaling, etc.

Virtualization Technologies In Modern HPC Systems

- Virtualization in HPC Systems
- Software Switches
- API Remoting
- Conclusions

What is a software switch?



- Integral part of a virtualized system
- Provides the networking connectivity between the virtual machines
- Implements features for security, QoS and automated control

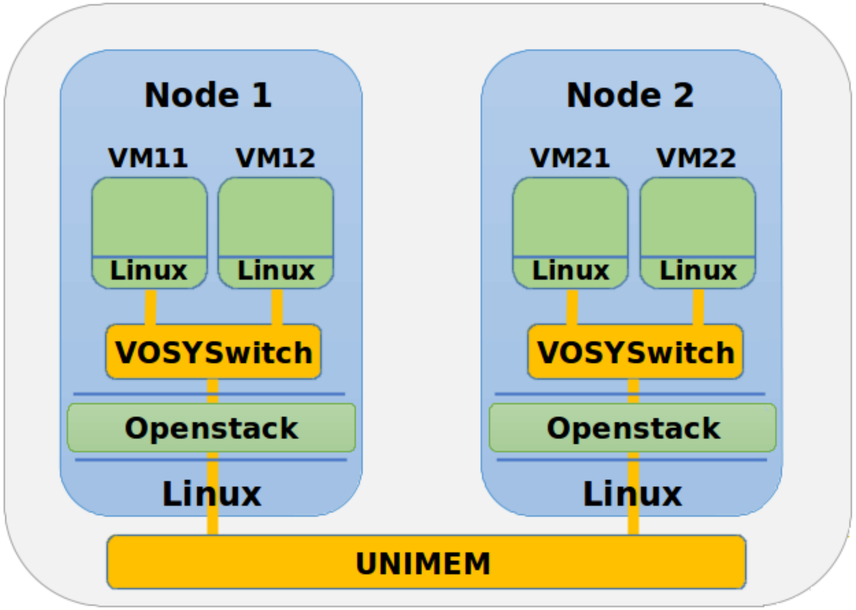
VOSYSwitch

- High-performance software switch based on a fast packet forwarding framework called Snabb
  - VOSYSwitch implements the NIC driver in user space
  - no context switches during packet handling
  - delivering near native performance
  - Supports both X86_64 and aarch64 platforms
- Fast and easily configurable through a simple JSON file

Adaptation of software switches to HPC - OpenStack

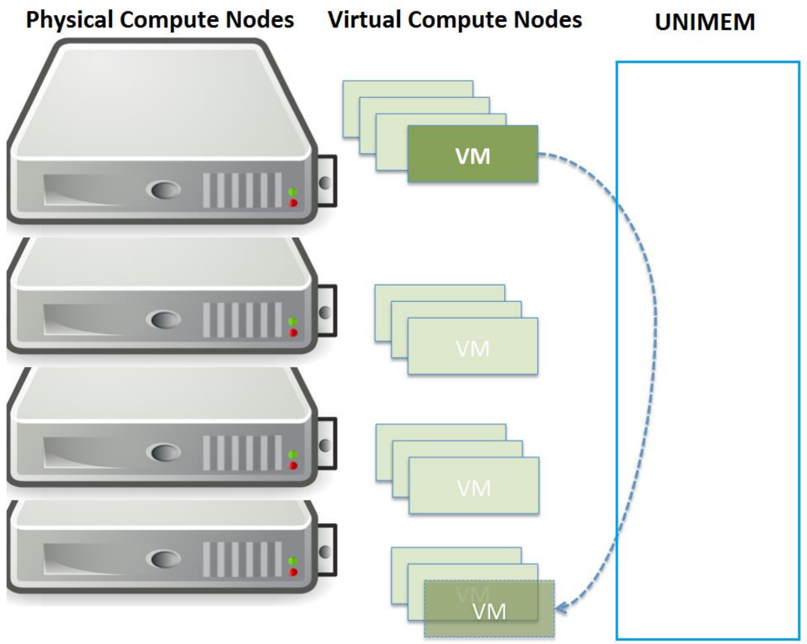
- How to manage a pool of servers and orchestrate a virtual system on top of that?
- OpenStack integration done in T2.4[ExaNeSt]
- Achieved through a dedicated ML2 plugin for Neutron
- Easy automated deployment of OpenStack w/VOSYSwitch for VM traffic
- Support for Devstack

Adaptation of software switches to HPC - UNIMEM



- UNIMEM is a scalable shared memory architecture
- VOSYSwitch support for UNIMEM
  - Achieved by developing an additional driver for UNIMEM in the core Snabb framework
- Relies on UNIMEM GSAS to exploit the high-performance shared memory of the ExaNeSt project for inter-node connectivity

Adaptation of software switches to HPC – Live Migration

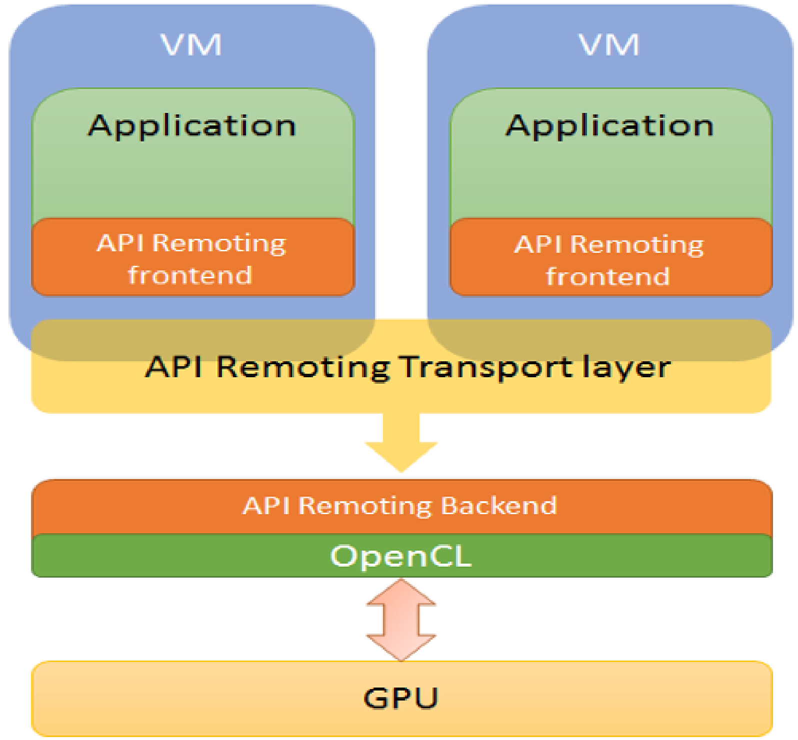


- Migration is a one of the key benefits of a virtual system
- Implemented support for Live Migration over UNIMEM:
  - Support for VHU protocol features by a separate bitmask
  - Support for broadcasting fake RARP request messages
  - VOSYSwitch establishes a connection between the two nodes

Virtualization Technologies In Modern HPC Systems

- Virtualization in HPC Systems
- Software Switches
- API Remoting
- Conclusions

Introduction to API Remoting

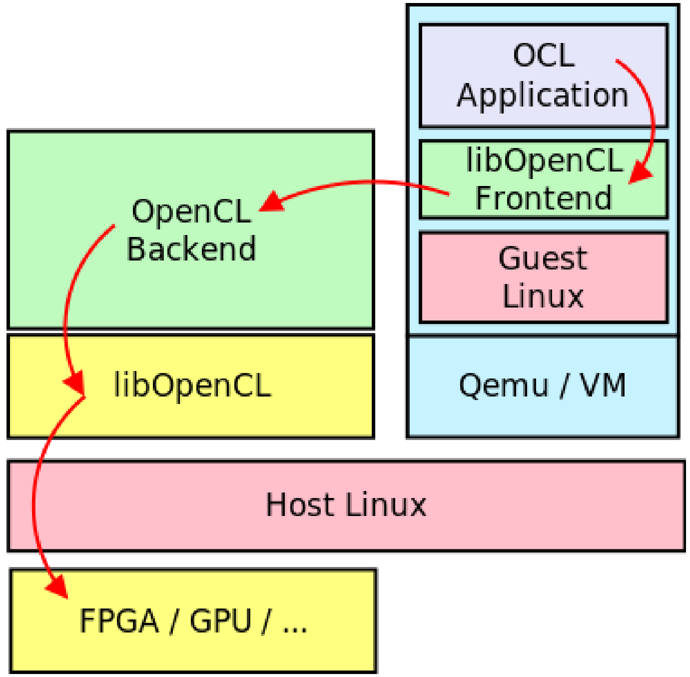


- In general, VMs cannot have access to the underlying hardware
- Virtualizing a device through its programming API interface (for ex. OpenCL, MPI, GSAS)
- Guest applications can leverage the HW through its API with no changes to their source code and no need for the hypervisor to provide a virtual HW abstraction

OpenCL in ExaNoDe and ECOSCALE

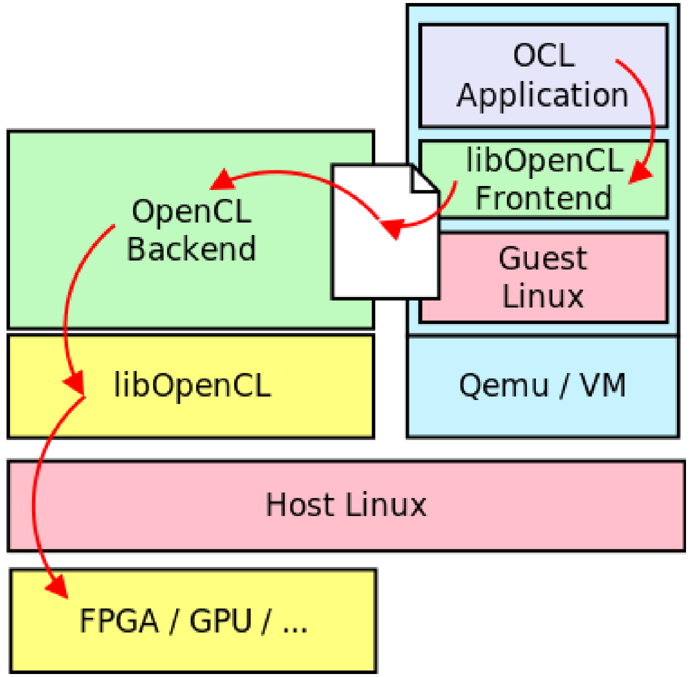
- Open standard for heterogeneous computing
  - Framework/API for controlling the computation
    - Work preparation and off-loading
    - Memory transfers
  - C-based programming language
    - Compiled on the fly
    - Executed on the accelerators
    - Task- and data-based parallelism
- Partners working at all layers of the OpenCL stack
  - ECOSCALE project brings it to the FPGA accelerator
  - ExaNeSt WP2 partners port scientific applications on top of it
  - ExaNoDe WP3 partners port the high-level data-flow programming models of OpenStream on top of it
  - ExaNoDe (VOSYS) enables using it inside virtual machines

API Remoting of OpenCL



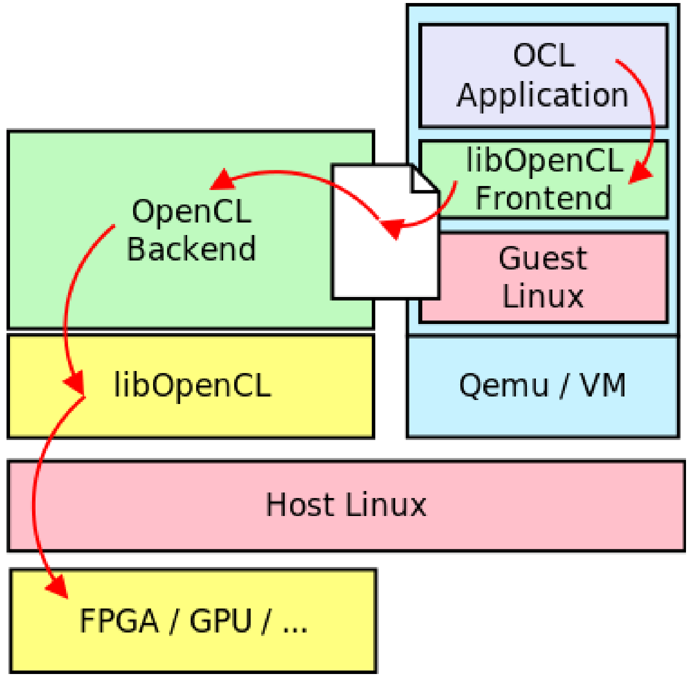
- Guest side:
  - OpenCL application is unchanged, but linked to our custom OpenCL library (called “**frontend**”) that forwards the calls to the backend
- Host side:
  - The **backend** is a host process that listens for calls from the frontends and does the actual call to the OpenCL library on the host

API Remoting of OpenCL



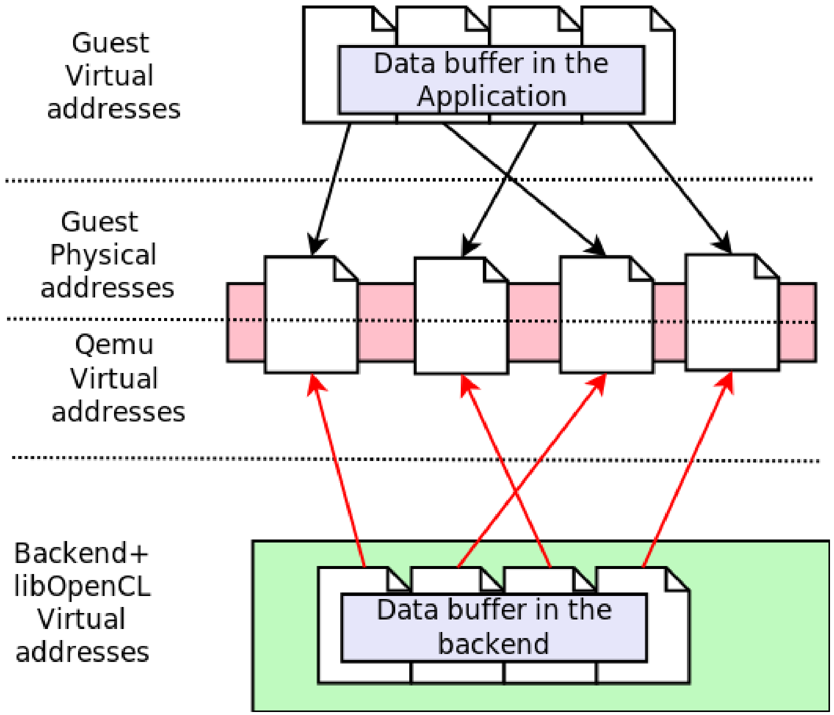
- The communication channel: **a shared page**
  - Host: Allocated in the host kernel module, mapped by the backend and then by QEMU and available in the guest address space
  - Guest: Mapped by the frontend library

API Remoting of OpenCL



- What about buffers bigger than a page?
- Naïve way:
  - Allocate more pages -> costly + many useless copies
- Better way:
  - Direct access to application buffer

API-Remoting of OpenCL – Zero-copy buffer passing



1. In the frontend:
   - Look up the guest physical addresses(GPA) making up the buffer
2. In the backend:
   - Load QEMU RAM buffer
   - Translate the GPA into valid virtual addresses
   - Remap contiguously the pages
   - Pass the buffer to OpenCL

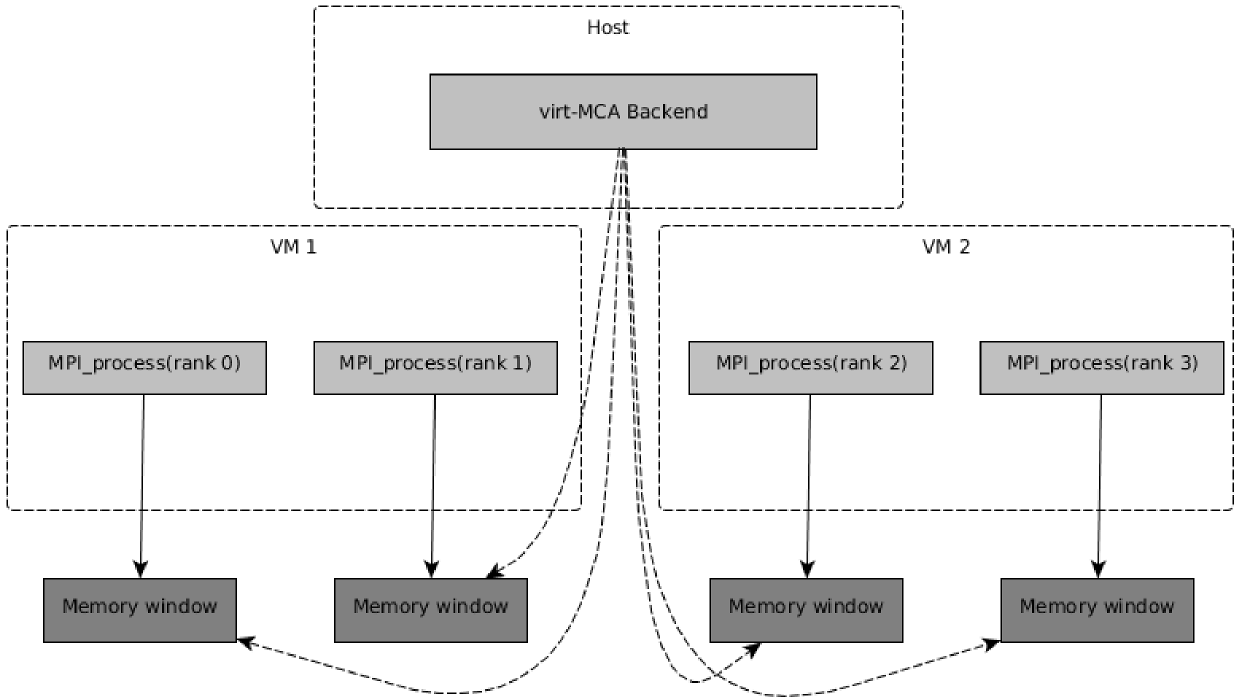
What about buffers bigger than a page?

- Naïve way:
  - Allocate more pages -> costly + many useless copies
- Better way:
  - Direct access to application buffer

API-Remoting of Intra-node MPI One-Sided Communication

- Target:
  - to enhance the performance of OSC traffic for VMs that reside in the same node
  - acceleration of the Remote Memory Access communication mechanisms
- Implementation based on OpenMPI 3.1
- An extension of the ‘pt2pt’ component for OSC
- The ‘virt’ extension uses a mix of TCP/IP and shared memory for processes communication transfers

API-Remoting of Intra-node MPI One-Sided Communication
The ‘virt’ extension of ‘pt2pt’



- Memory window allocations shared with the ‘backend’ process
- The backend has direct access on node’s memory buffers
- One-Sided transfers e.g., MPI_Put, MPI_Get use the backend to copy the data via shared memory
- Traffic out of the node fall back to the ‘pt2pt’ component

API-Remoting of MPI One-Sided Communication Preliminary Results

- Evaluated the system using Ohio's OSU-Micro-Benchmarks for One-Sided Communication
- Test cases:
  - Bandwidth and Latency tests with variable sized memory window buffers
- Up to 4x in bandwidth speedup
  - The bandwidth increases linear with the window size
  - Latency is not affected due to the preliminary TCP/IP communication for synchronization

Virtualization Technologies In Modern HPC Systems

- Virtualization in HPC Systems
- Software Switches
- API Remoting
- Conclusions

Conclusions

- Projects like ECOSCALE, ExaNeSt and ExaNoDe aim to set the ground for modern cutting-edge HPC systems through novel hardware and software solutions.
- Innovations brought by these projects introduce unexplored possibilities for the current technologies like software switches and virtualization techniques like API remoting.

Thank you!

Virtualization Technologies In Modern HPC Systems

Radoslav Dimitrov, Kevin Pouget, Angelos Mouzakitis, Alvise Rigo

Virtual Open Systems

Exascale HPC Workshop, Ljubljana, Slovenia

31 May 2018

Horizon 2020