Building HPC Cloud with InfiniBand: Efficient Support in MVAPICH2 for KVM, Docker, Singularity, OpenStack, and SLURM

A Tutorial at MUG 2018

by

Xiaoyi Lu

The Ohio State University

E-mail: luxi@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~luxi

HPC Meets Cloud Computing

- Cloud Computing widely adopted in industry computing environment
- Cloud Computing provides high resource utilization and flexibility
- Virtualization is the key technology to enable Cloud Computing
- Intersect360 study shows cloud is the fastest growing class of HPC
- HPC Meets Cloud: The convergence of Cloud Computing and HPC

HPC Cloud - Combining HPC with Cloud

- IDC expects that by 2019, HPC ecosystem revenue will jump to a record $30.2 billion. IDC foresees public clouds, and especially custom public clouds, supporting an increasing proportion of the aggregate HPC workload as these cloud facilities grow more capable and mature (Courtesy: http://www.idc.com/getdoc.jsp?containerId=247846)
- Combining HPC with Cloud is still facing challenges because of the performance overhead associated virtualization support
  - **Lower performance of virtualized I/O devices**
- HPC Cloud Examples
  - **Amazon EC2 with Enhanced Networking**
    - Using Single Root I/O Virtualization (SR-IOV)
    - Higher performance (packets per second), lower latency, and lower jitter
    - 10 GigE
  - **NSF Chameleon Cloud**

Outline

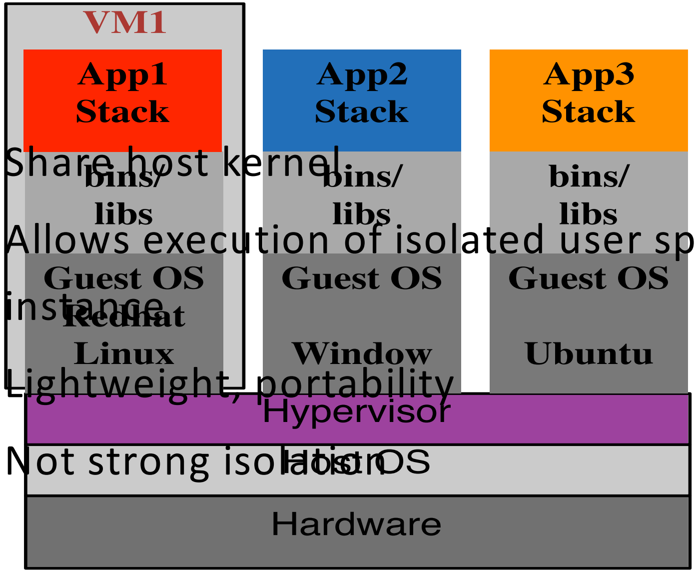
- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

Virtualization Technology (Hypervisor vs. Container)

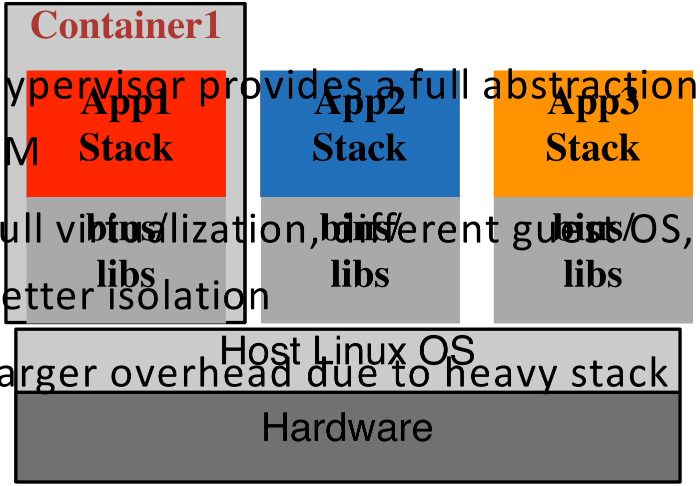
- Provides abstractions of multiple virtual resources by utilizing an intermediate software layer on top of the underlying system

- Share host kernel
- Allows execution of isolated user space instance
- Lightweight, portability
- Not strong isolation

- Hypervisor provides a full abstraction of VM
- Full virtualization, different guest OS, better isolation
- Larger overhead due to heavy stack



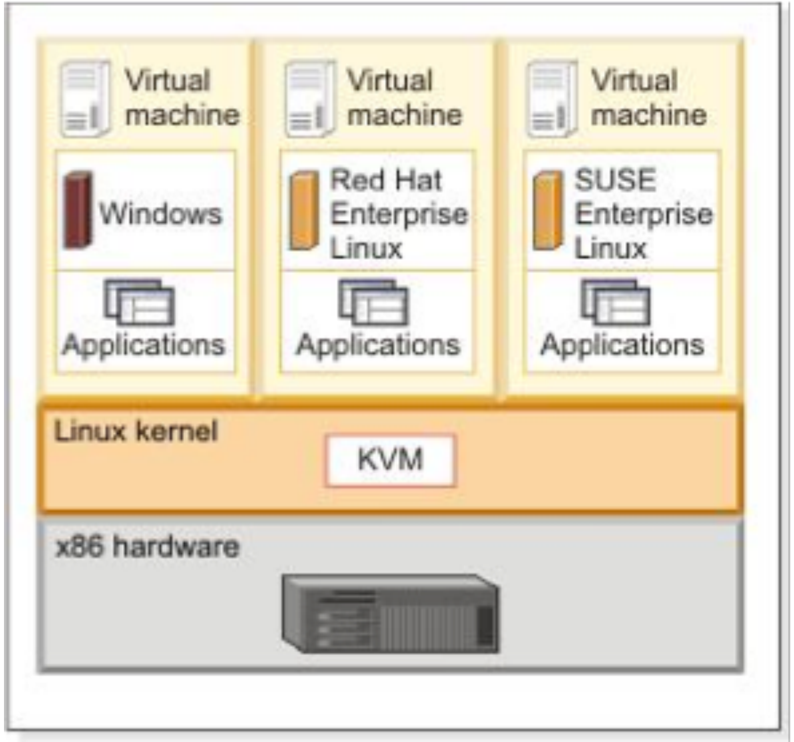
Hypervisor-based Virtualization



Container-based Virtualization

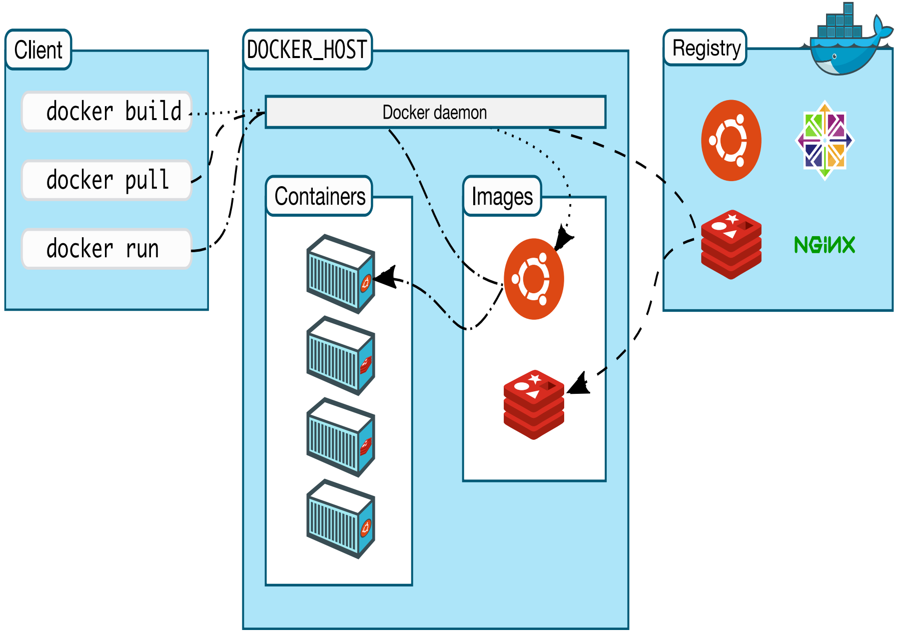
Overview of Kernel-based Virtual Machine (KVM)

- A full virtualization solution for Linux on x86 hardware that contains virtualization extensions (Intel VT or AMD-V)
- The KVM module creates a bare metal hypervisor on the Linux kernel
- KVM hosts the virtual machine images as regular Linux processes
- Each virtual machine image can use all of the features of the Linux kernel, including hardware, security, storage, etc.



https://www.ibm.com/support/knowledgecenter/en/linuxonibm/liaat/liaatkvmover.htm

Container Technology - Docker



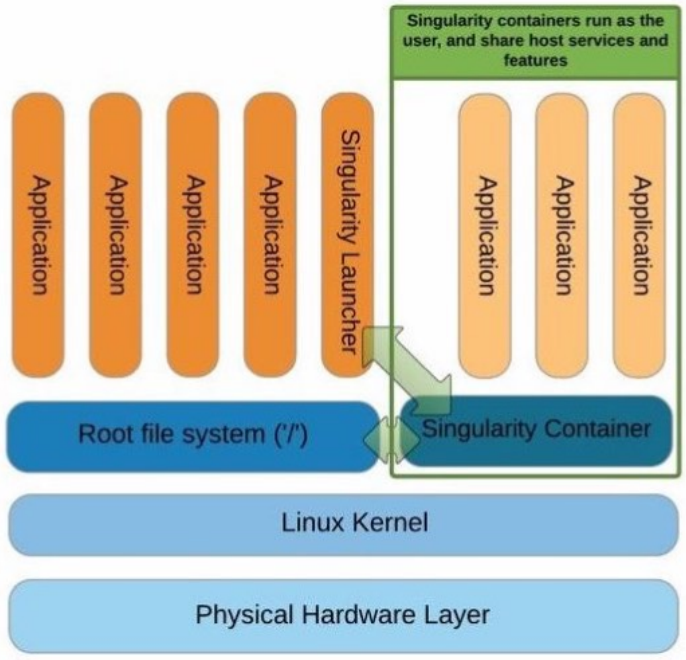
- Inherit advantages of container technique
- Active community contribution
- Root owned daemon process
- Root escalation in Docker container
- Non-negligible performance overhead

Singularity Overview

- Reproducible software stacks
  - Easily verify via checksum or cryptographic signature
- Mobility of compute
  - Able to transfer (and store) containers via standard data mobility tools
- Compatibility with complicated architectures
  - Runtime immediately compatible with existing HPC architecture
- Security model
  - Support untrusted users running untrusted containers

http://singularity.lbl.gov/about

Container Technology (Docker vs. Singularity)



- Singularity aims to provide reproducible and mobile environments across HPC centers
- NO root owned daemon
- NO root escalation
- mpirun_rsh –np 2 –hostfile htfiles singualrity exec /tmp/Centos-7.img /usr/bin/osu_latency

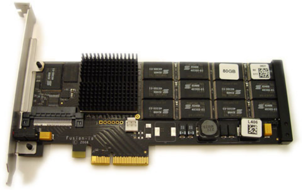
Outline

- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

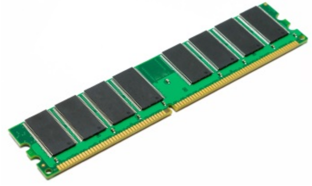
Drivers of Modern HPC Cluster and Cloud Architecture

Multi-/Many-core Processors

High Performance Interconnects – InfiniBand (with SR-IOV)
<1usec latency, 200Gbps Bandwidth>



SSDs, Object Storage Clusters



Large memory nodes (Upto 2 TB)

- Multi-core/many-core technologies, Accelerators
- Large memory nodes
- Solid State Drives (SSDs), NVM, Parallel Filesystems, Object Storage Clusters
- Remote Direct Memory Access (RDMA)-enabled networking (InfiniBand and RoCE)
- Single Root I/O Virtualization (SR-IOV)

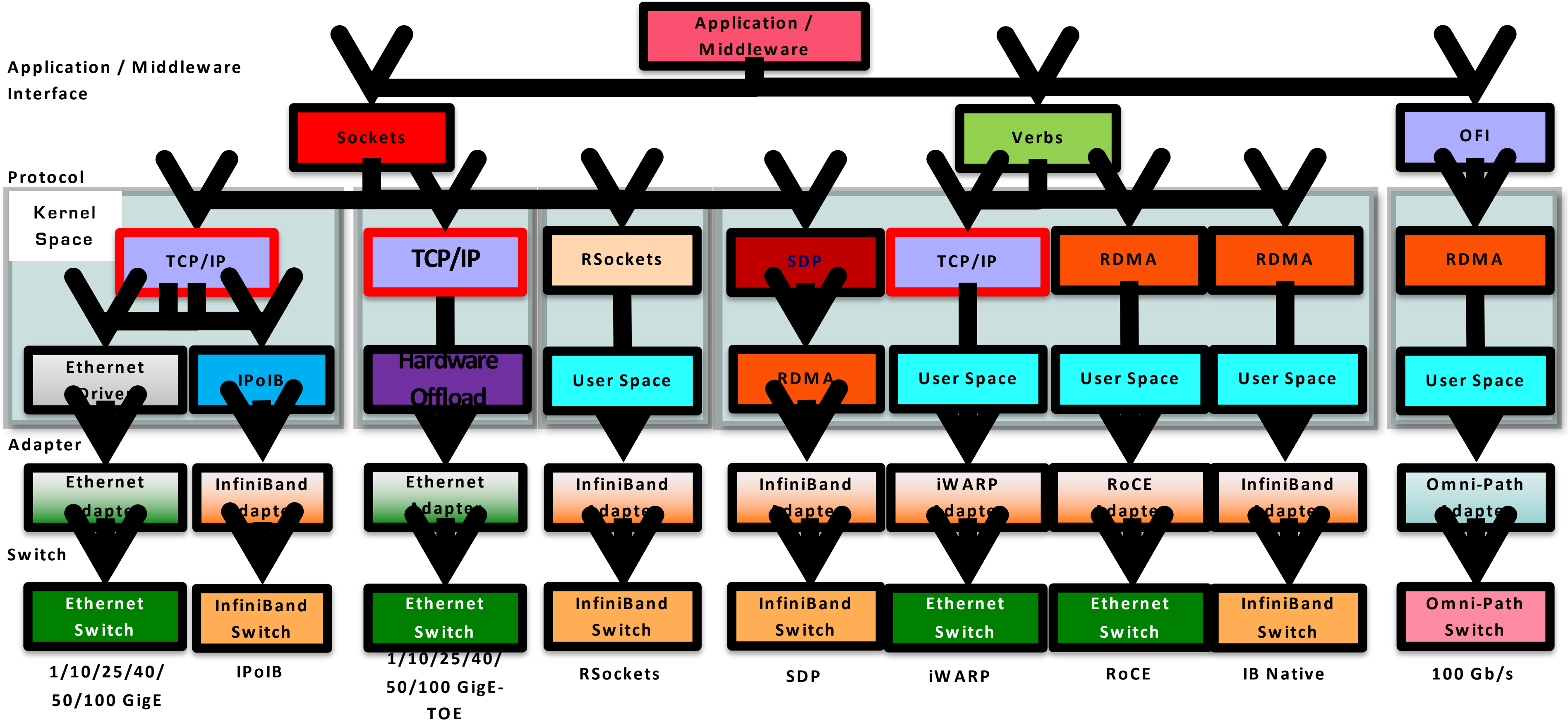
SDSC Comet

TACC Stampede

Trends in High-Performance Networking Technologies

- Advanced Interconnects and RDMA protocols
  - InfiniBand (up to 200 Gbps, HDR)
  - 10/40/100 Gigabit Ethernet/iWARP
  - RDMA over Converged Enhanced Ethernet (RoCE)
- Omni-Path
- Delivering excellent performance (Latency, Bandwidth and CPU Utilization)
- Has influenced re-designs of enhanced HPC middleware
  - Message Passing Interface (MPI) and PGAS
  - Parallel File Systems (Lustre, GPFS, ..)
- Paving the way to the wide utilization in HPC Cloud with virtualization support (SR-IOV)

Available Interconnects and Protocols for Data Centers



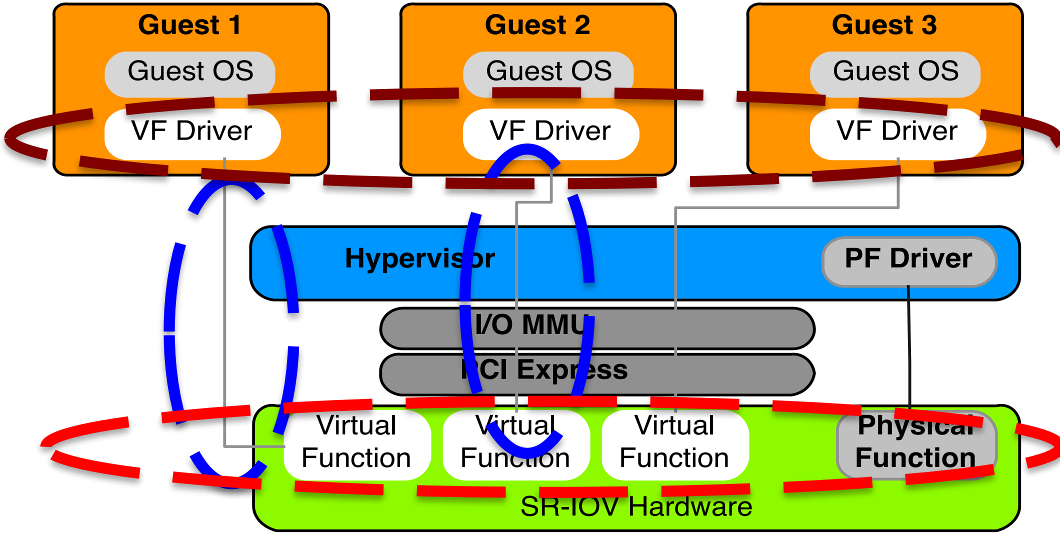
Open Standard InfiniBand Networking Technology

- Introduced in Oct 2000
- High Performance Data Transfer
  - Interprocessor communication and I/O
  - Low latency (<1.0 microsec), High bandwidth (up to 25 GigaBytes/sec -> 200Gbps), and low CPU utilization (5-10%)
- Flexibility for LAN and WAN communication
- Multiple Transport Services
  - Reliable Connection (RC), Unreliable Connection (UC), Reliable Datagram (RD), Unreliable Datagram (UD), and Raw Datagram
  - Provides flexibility to develop upper layers
- Multiple Operations
  - Send/Recv
  - RDMA Read/Write
  - Atomic Operations (very unique)
    - high performance and scalable implementations of distributed locks, semaphores, collective communication operations
- Leading to big changes in designing HPC clusters, file systems, cloud computing systems, grid computing systems, ….

Single Root I/O Virtualization (SR-IOV)

- Single Root I/O Virtualization (SR-IOV) is providing new opportunities to design HPC cloud with very little low overhead

- Allows a single physical device, or a Physical Function (PF), to present itself as multiple virtual devices, or Virtual Functions (VFs)
- VFs are designed based on the existing non-virtualized PFs, no need for driver change
- Each VF can be dedicated to a single VM through PCI pass-through
- Work with 10/40 GigE and InfiniBand



Overview of MVAPICH2 Project

- High Performance open-source MPI Library for InfiniBand, Omni-Path, Ethernet/iWARP, and RDMA over Converged Ethernet (RoCE)
  - MVAPICH (MPI-1), MVAPICH2 (MPI-2.2 and MPI-3.0), Started in 2001, First version available in 2002
  - MVAPICH2-X (MPI + PGAS), Available since 2011
  - Support for GPGPUs (MVAPICH2-GDR) and MIC (MVAPICH2-MIC), Available since 2014
  - **Support for Virtualization (MVAPICH2-Virt), Available since 2015**
  - Support for Energy-Awareness (MVAPICH2-EA), Available since 2015
  - Support for InfiniBand Network Analysis and Monitoring (OSU INAM) since 2015
  - **Used by more than 2,925 organizations in 86 countries**
  - **More than 484,000 (> 0.4 million) downloads from the OSU site directly**
  - Empowering many TOP500 clusters (Jul '18 ranking)
    - **2nd ranked 10,649,640-core cluster (Sunway TaihuLight) at NSC, Wuxi, China**
    - 12th, 556,104 cores (Oakforest-PACS) in Japan
    - 15th, 367,024 cores (Stampede2) at TACC
    - 24th, 241,108-core (Pleiades) at NASA and many others
  - Available with software stacks of many vendors and Linux Distros (RedHat and SuSE)
  - http://mvapich.cse.ohio-state.edu

Architecture of MVAPICH2 Software Family

High Performance Parallel Programming Models

- Message Passing Interface (MPI)
- PGAS (UPC, OpenSHMEM, CAF, UPC++)
- Hybrid --- MPI + X (MPI + PGAS + OpenMP/Cilk)

High Performance and Scalable Communication Runtime

Diverse APIs and Mechanisms

- Point-to-point Primitives
- Collectives Algorithms
- Job Startup
- Energy-Awareness
- Remote Memory Access
- I/O and File Systems
- Fault Tolerance
- Virtualization
- Active Messages
- Introspection & Analysis

Support for Modern Networking Technology
(InfiniBand, iWARP, RoCE, Omni-Path)

Transport Protocols

- RC
- XRC
- UD
- DC

Modern Features

- UMR
- ODP
- SR-IOV
- Multi Rail

Support for Modern Multi-/Many-core Architectures
(Intel-Xeon, OpenPower, Xeon-Phi (MIC, KNL), NVIDIA GPGPU)

Transport Mechanisms

- Shared Memory
- CMA
- IVSHMEM

Modern Features

- MCDRAM*
- NVLink*
- CAPI*

* Upcoming

Outline

- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

Building HPC Cloud with SR-IOV and InfiniBand

- High-Performance Computing (HPC) has adopted advanced interconnects and protocols
  - InfiniBand
  - 10/40/100 Gigabit Ethernet/iWARP
  - RDMA over Converged Enhanced Ethernet (RoCE)
- Very Good Performance
  - Low latency (few micro seconds)
  - High Bandwidth (200 Gb/s with HDR InfiniBand)
  - Low CPU overhead (5-10%)
- OpenFabrics software stack with IB, iWARP and RoCE interfaces are driving HPC systems
- How to Build HPC Clouds with SR-IOV and InfiniBand for delivering optimal performance?

HPC, Big Data, and Deep Learning on Cloud Computing Systems: Challenges

Applications

HPC, Big Data, Deep Learning Middleware

- HPC (MPI, PGAS, MPI+PGAS, MPI+OpenMP, etc.)
- Big Data (HDFS, MapReduce, Spark, HBase, Memcached, etc.)
- Deep Learning (Caffe, TensorFlow, CNTK, BigDL, etc.)

Resource Management and Scheduling Systems for Cloud Computing
(OpenStack Nova, Swift, Heat; Slurm, etc.)

Communication and I/O Library

- Communication Channels (SR-IOV, IVShmem, IPC-Shm, CMA)
- Locality- and NUMA-aware Communication
- Virtualization (Hypervisor and Container)
- Data Placement & Task Scheduling
- Fault-Tolerance (Migration, Replication, etc.)
- QoS-aware, etc.

- Networking Technologies (InfiniBand, Omni-Path, 1/10/40/100 GigE and Intelligent NICs)
- Commodity Computing System Architectures (Multi- and Many-core architectures and accelerators)
- Storage Technologies (HDD, SSD, NVRAM, and NVMe-SSD)

Broad Challenges in Designing Communication and I/O Middleware for HPC on Clouds

- Virtualization Support with Virtual Machines and Containers
  - KVM, Docker, Singularity, etc.
- Communication coordination among optimized communication channels on Clouds
  - SR-IOV, IVShmem, IPC-Shm, CMA, etc.
- Locality-aware communication
- Scalability for million to billion processors
  - Support for highly-efficient inter-node and intra-node communication (both two-sided and one-sided)
- Scalable Collective communication
  - Offload; Non-blocking; Topology-aware
- Balancing intra-node and inter-node communication for next generation nodes (128-1024 cores)
  - Multiple end-points per node
- NUMA-aware communication for nested virtualization
- Integrated Support for GPGPUs and Accelerators
- Fault-tolerance/resiliency
  - Migration support with virtual machines
- QoS support for communication and I/O
- Support for Hybrid MPI+PGAS programming (MPI + OpenMP, MPI + UPC, MPI + OpenSHMEM, MPI+UPC++, CAF, …)
- Energy-Awareness
- Co-design with resource management and scheduling systems on Clouds
  - OpenStack, Slurm, etc.

Outline

- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

High-Performance MPI Library on HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)

HPC on Cloud Computing Systems: Challenges Addressed by OSU So Far

Applications

HPC and Big Data Middleware

HPC (MPI, PGAS, MPI+PGAS, MPI+OpenMP, etc.)

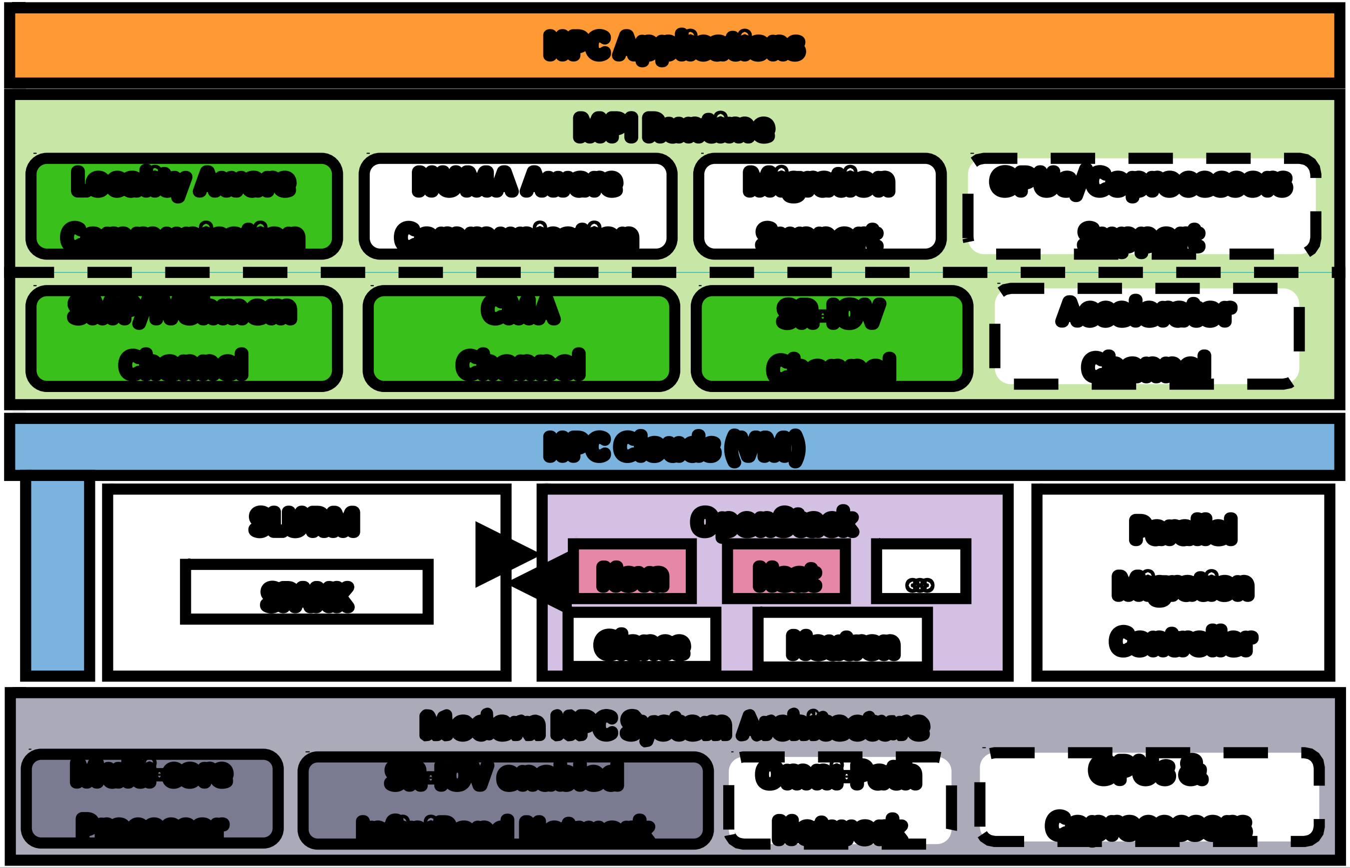
Resource Management and Scheduling Systems for Cloud Computing (OpenStack Nova, Heat; Slurm)

Communication and I/O Library

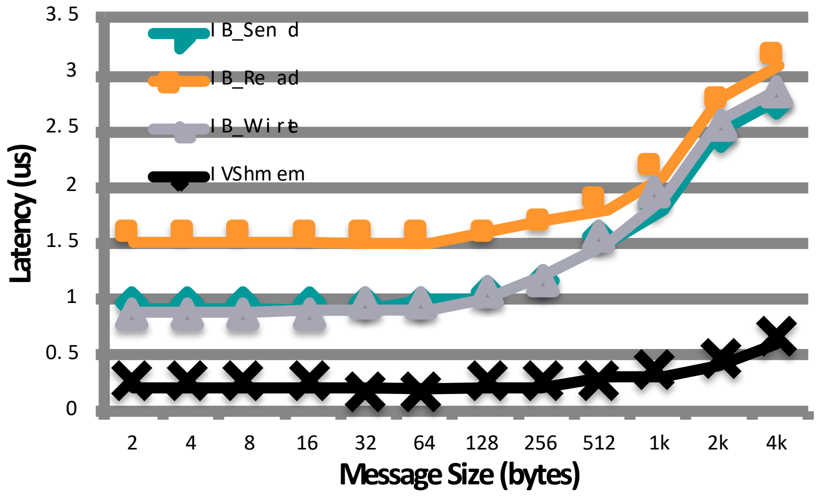
- Communication Channels (SR-IOV, IVShmem, IPC-Shm, CMA)
- Locality- and NUMA-aware Communication
- Virtualization (Hypervisor and Container)
- Fault-Tolerance & Consolidation (Migration)
- QoS-aware
- Future Studies

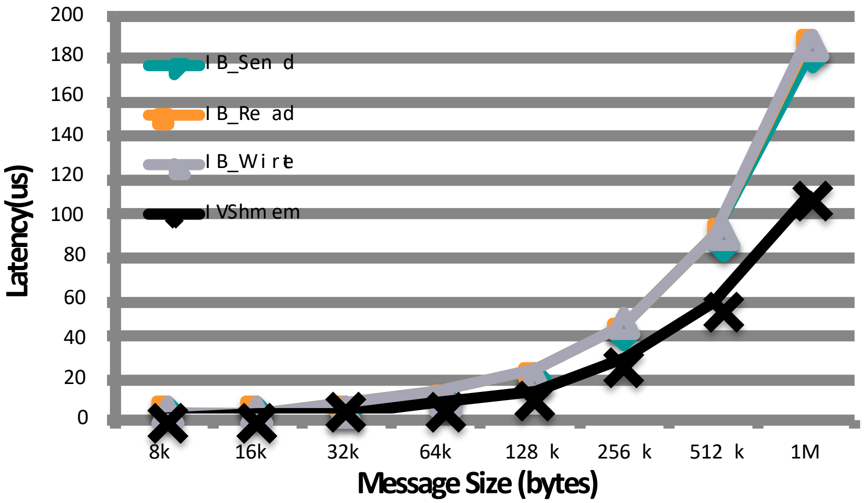
Networking Technologies (InfiniBand, Omni-Path, 1/10/40/100 GigE and Intelligent NICs)	Commodity Computing System Architectures (Multi- and Many-core architectures and accelerators)	Storage Technologies (HDD, SSD, NVRAM, and NVMe-SSD)

MVAPICH2-Virt with SR-IOV and IVSHMEM



Intra-Node Inter-VM Performance



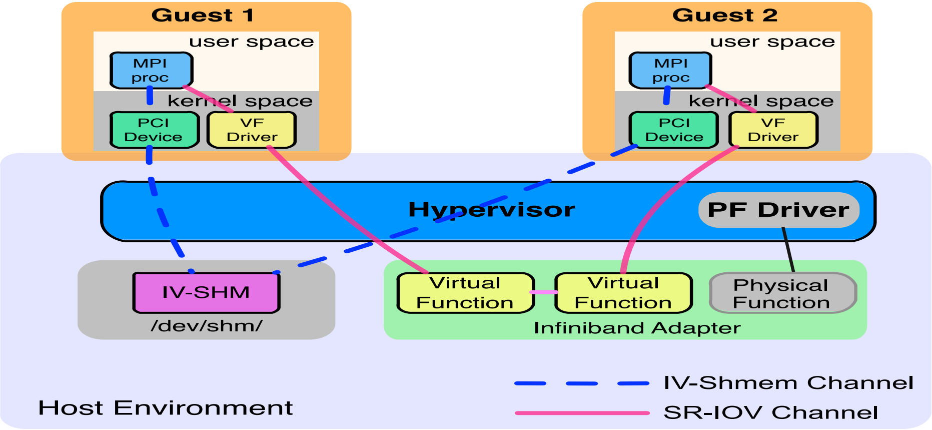


Latency at 64 bytes message size: SR-IOV(IB_Send) - 0.96µs, IVShmem - 0.2µs

Can IVShmem scheme benefit MPI communication within a node on SR-IOV enabled InfiniBand clusters?

Overview of MVAPICH2-Virt with SR-IOV and IVSHMEM

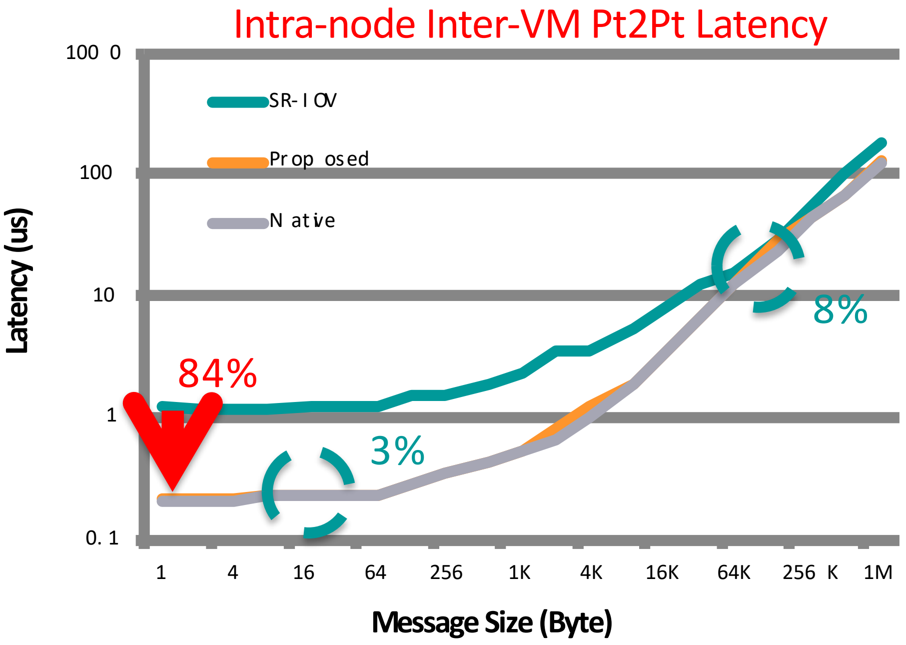
- Redesign MVAPICH2 to make it virtual machine aware
  - SR-IOV shows near to native performance for inter-node point to point communication
  - IVSHMEM offers shared memory based data access across co-resident VMs
  - Locality Detector: maintains the locality information of co-resident virtual machines
  - Communication Coordinator: selects the communication channel (SR-IOV, IVSHMEM) adaptively

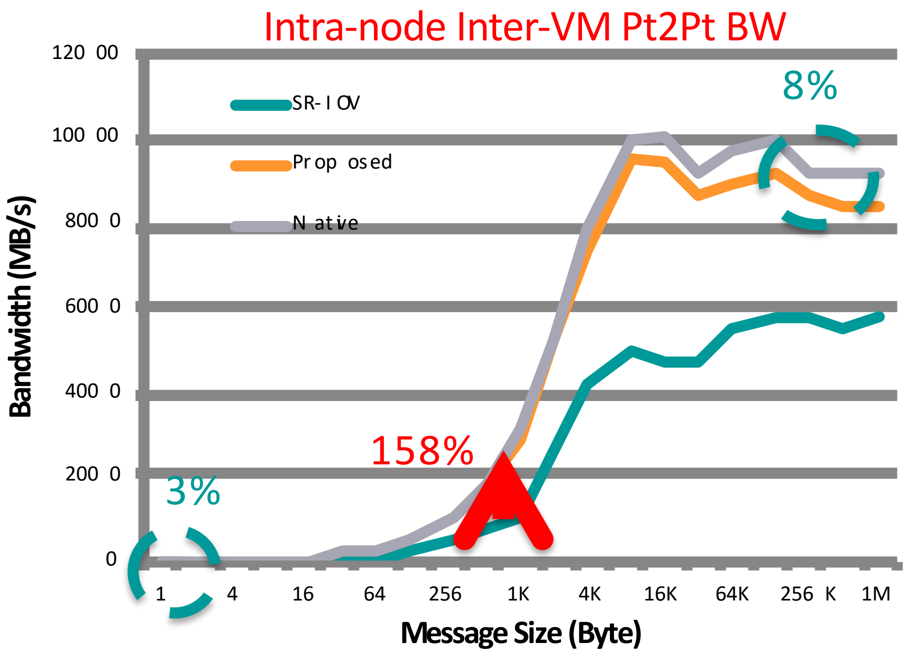


J. Zhang, X. Lu, J. Jose, R. Shi, D. K. Panda. Can Inter-VM Shmem Benefit MPI Applications on SR-IOV based Virtualized InfiniBand Clusters? Euro-Par, 2014

J. Zhang, X. Lu, J. Jose, R. Shi, M. Li, D. K. Panda. High Performance MPI Library over SR-IOV Enabled InfiniBand Clusters. HiPC, 2014

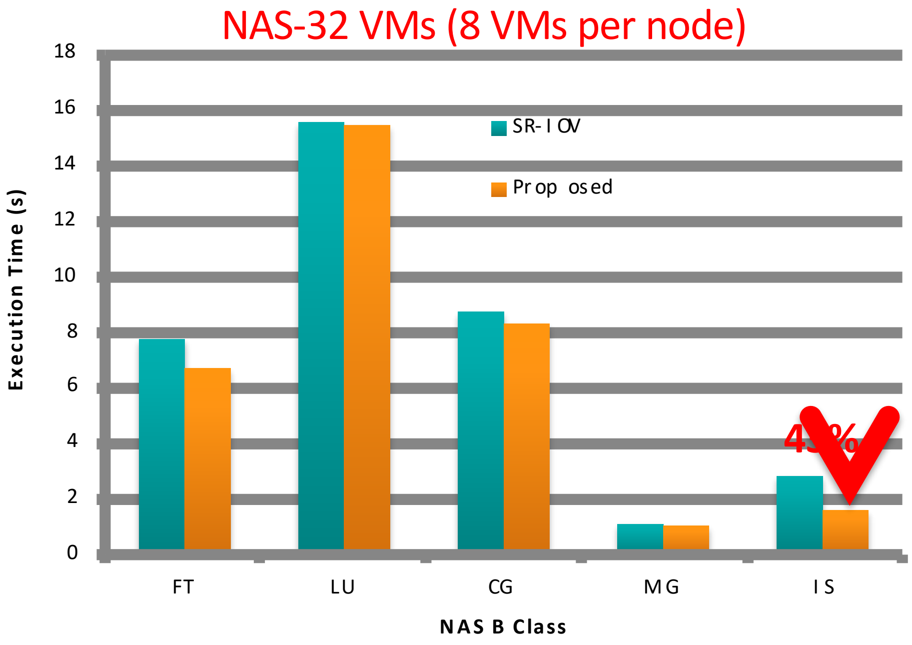
Intra-node Inter-VM Point-to-Point Performance

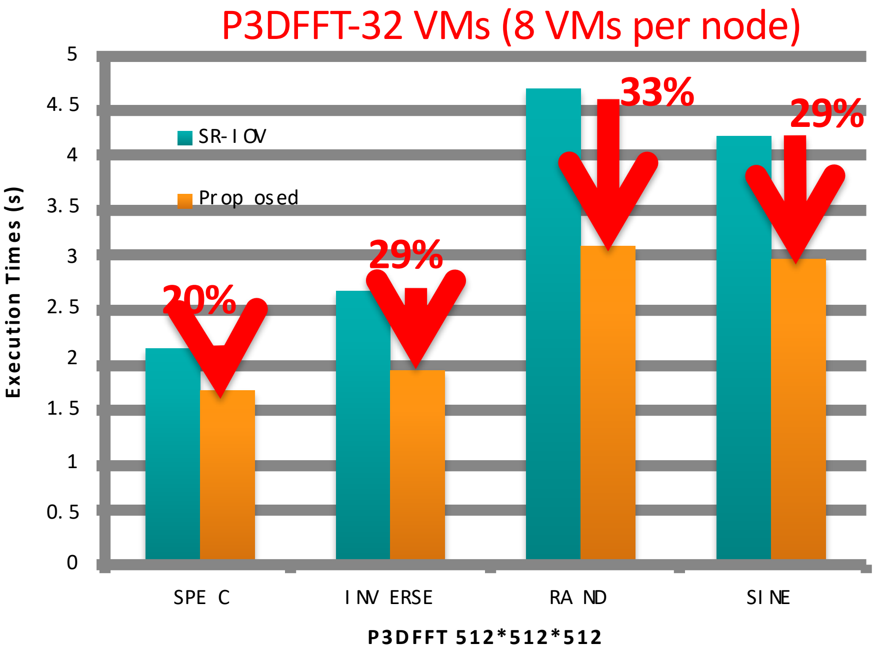




- Compared to SR-IOV, up to 84% and 158% improvement on Latency & Bandwidth
- Compared to Native, 3%-8% overhead on both Latency & Bandwidth

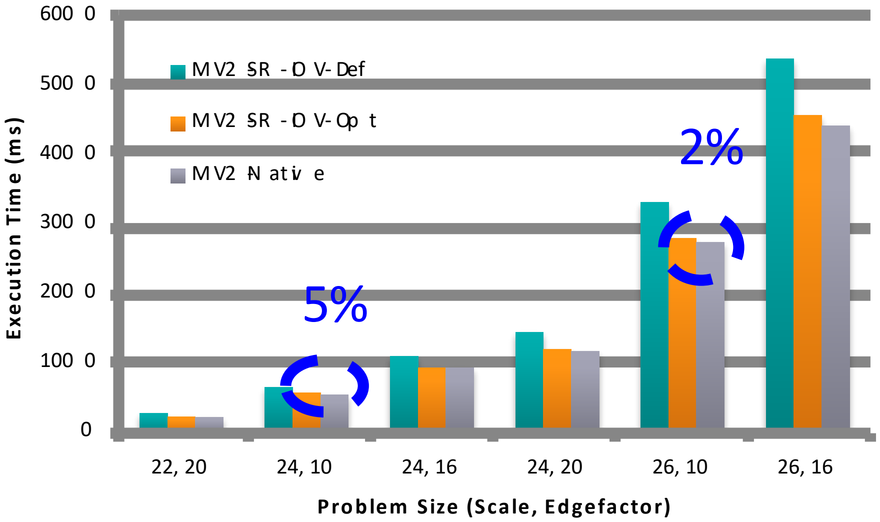
Application Performance (NAS & P3DFFT)



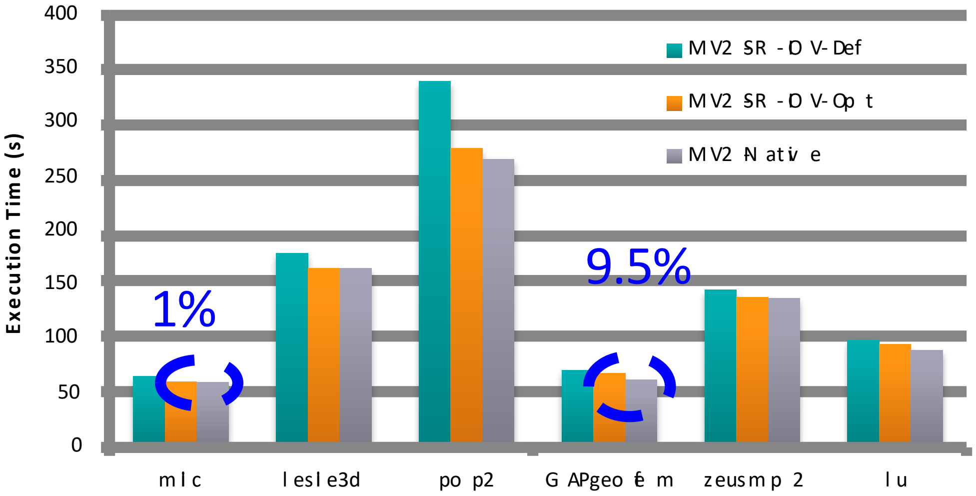


- Proposed design delivers up to 43% (IS) improvement for NAS
- Proposed design brings 29%, 33%, 29% and 20% improvement for INVERSE, RAND, SINE and SPEC

Application-Level Performance on Chameleon



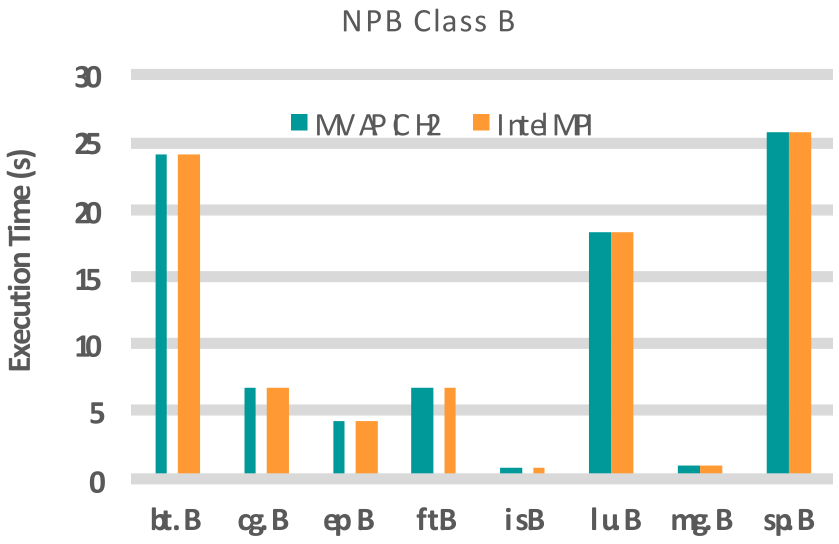
Graph500

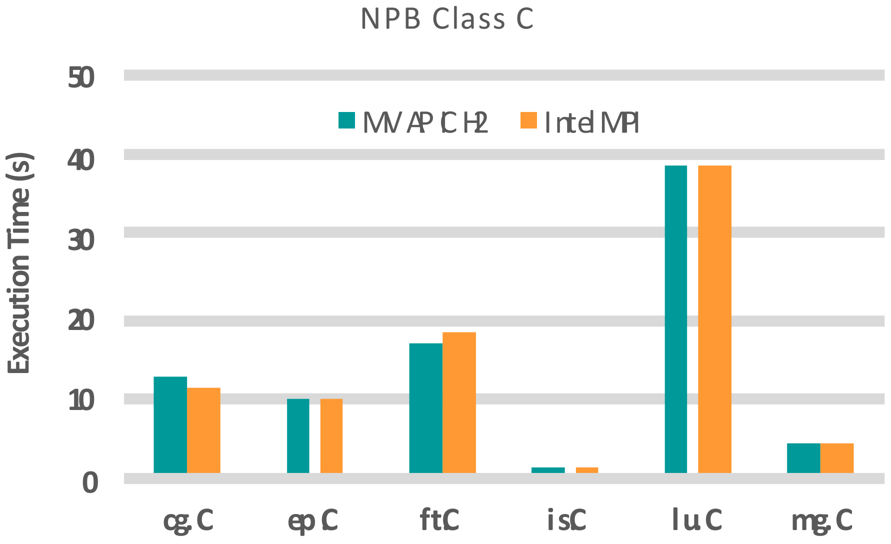


SPEC MPI2007

- 32 VMs, 6 Core/VM
- Compared to Native, 2-5% overhead for Graph500 with 128 Procs
- Compared to Native, 1-9.5% overhead for SPEC MPI2007 with 128 Procs

Application-Level Performance on Azure



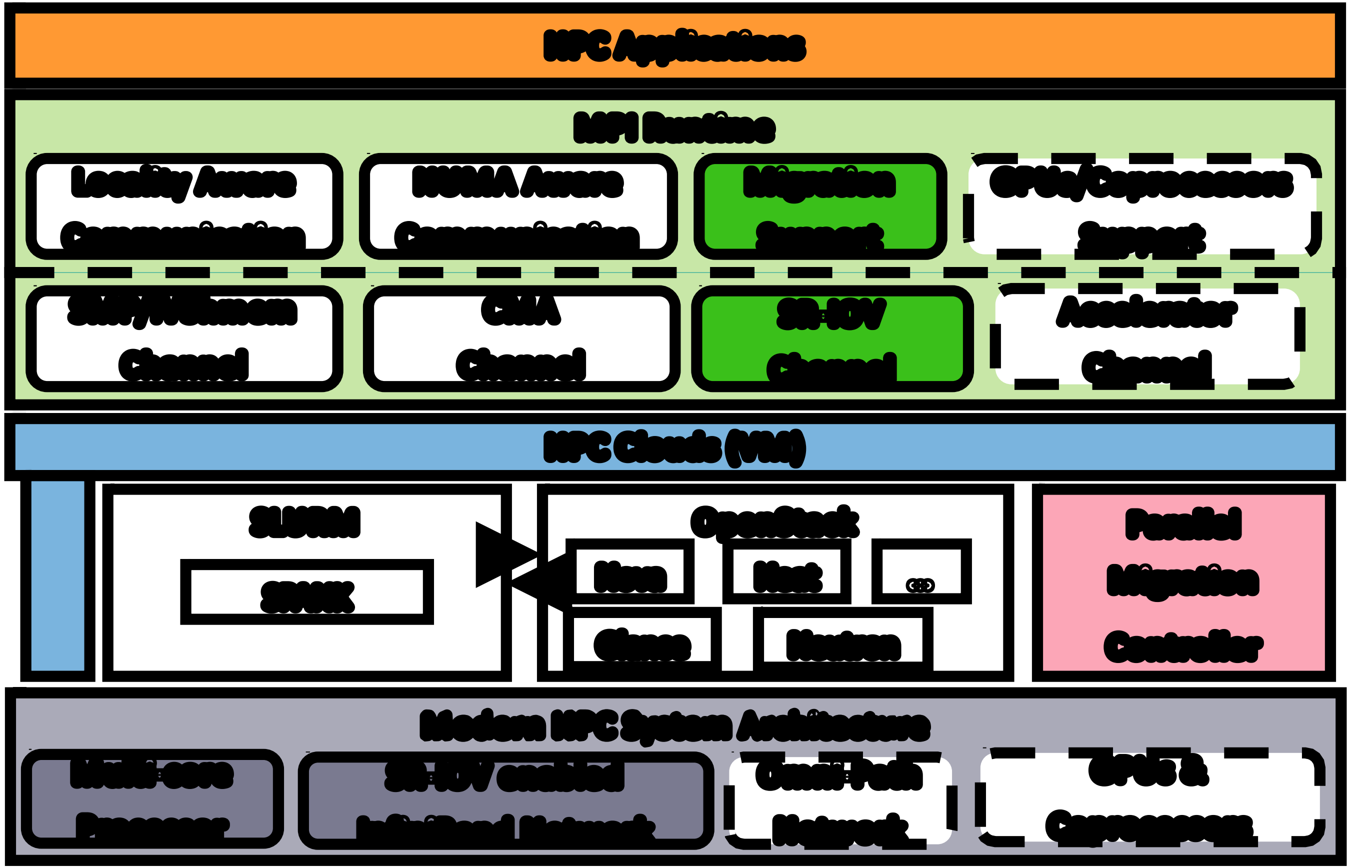


- NPB Class B with 16 Processes on 2 Azure A8 instances
- NPB Class C with 32 Processes on 4 Azure A8 instances
- Comparable performance between MVAPICH2 and IntelMPI

High-Performance MPI Library on HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)

SR-IOV-enabled VM Migration Support in MVAPICH2



Execute Live Migration with SR-IOV Device

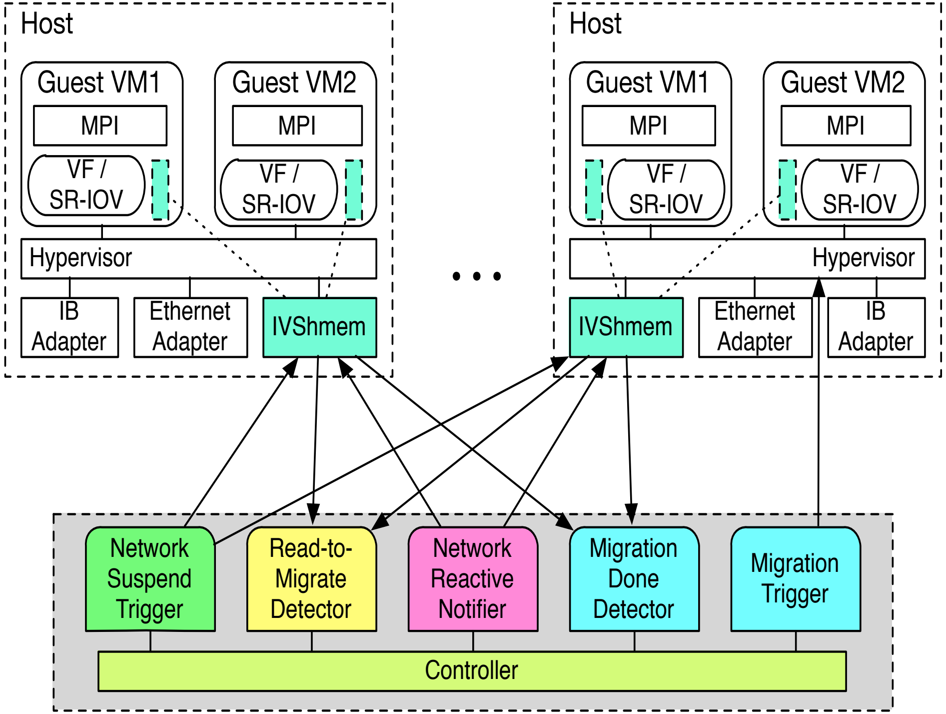
```
[root@sandy1:migration]$
[root@sandy1:migration]$ssh sandy3-vm1 lspci
root@sandy3-vm1's password:
00:00.0 Host bridge: Intel Corporation 440FX - 82441FX PMC [Natoma] (rev 02)
00:01.0 ISA bridge: Intel Corporation 82371SB PIIX3 ISA [Natoma/Triton II]
00:01.1 IDE interface: Intel Corporation 82371SB PIIX3 IDE [Natoma/Triton II]
00:01.2 USB controller: Intel Corporation 82371SB PIIX3 USB [Natoma/Triton II] (rev 01)
00:01.3 Bridge: Intel Corporation 82371AB/EB/MB PIIX4 ACPI (rev 03)
00:02.0 VGA compatible controller: Cirrus Logic GD 5446
00:04.0 Infiniband controller: Mellanox Technologies MT27700 Family [ConnectX-4 Virtual Function]
[root@sandy1:migration]$
[root@sandy1:migration]$
[root@sandy1:migration]$
[root@sandy1:migration]$
[root@sandy1:migration]$
[root@sandy1:migration]$virsh migrate --live --rdma-pin-all --migrateuri rdma://sandy3-ib sandy1-vm1 qemu://sandy3-ib/system
error: Requested operation is not valid: domain has assigned non-USB host devices

[root@sandy1:migration]$
```

Overview of Existing Migration Solutions for SR-IOV

	Platform	NIC	No Guest OS Modification	Device Driver Independent	Hypervisor Independent
Zhai, etc (Linux bonding driver)	Ethernet	N/A	Yes	No	Maybe (Xen)
Kadav, etc (shadow driver)	Ethernet	Intel Pro/1000 gigabit NIC	No	Yes	No (Xen)
Pan, etc (CompSC)	Ethernet	Intel 82576, Intel 82599	Yes	No	No (Xen)
Guay, etc	InfiniBand	Mellanox ConnectX2 QDR HCA	Yes	No	Yes (Oracle VM Server (OVS) 3.0.)
Han	Ethernet	Huawei smart NIC	Yes	No	No (QEMU+KVM)
Xu, etc (SRVM)	Ethernet	Intel 82599	Yes	Yes	No (VMware EXSi)

Can we have a hypervisor-independent and device driver-independent solution for InfiniBand based HPC Clouds with SR-IOV?

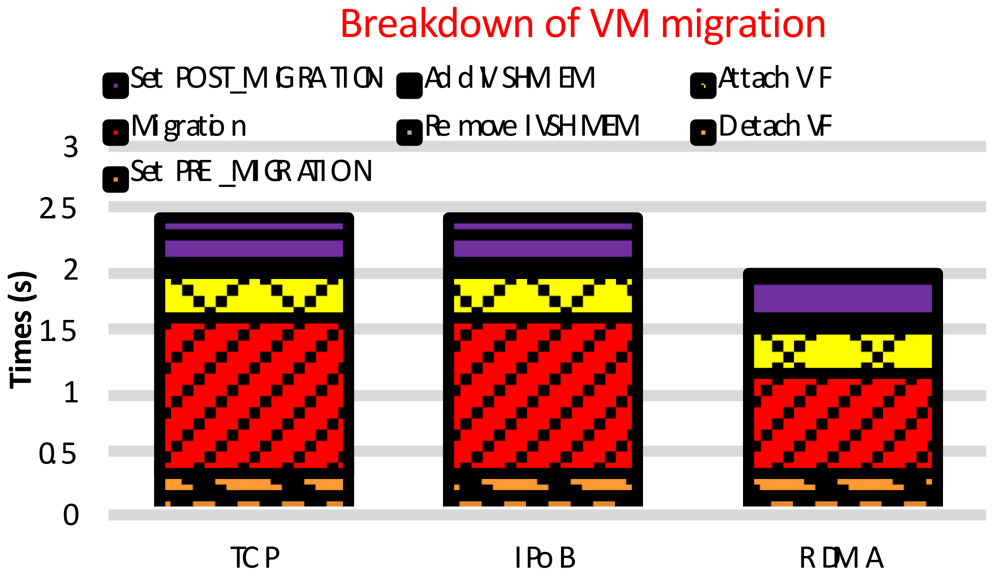
High Performance SR-IOV enabled VM Migration Framework

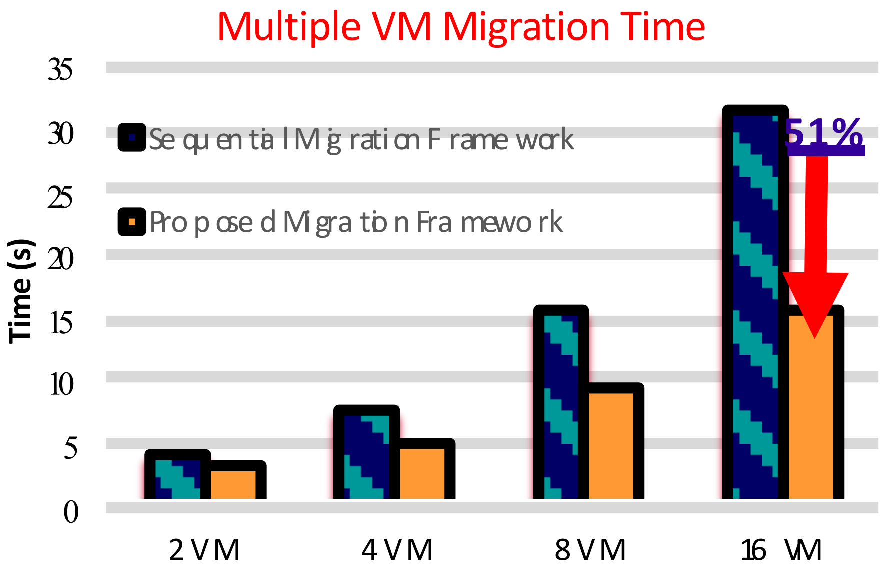


- Consist of SR-IOV enabled IB Cluster and External Migration Controller
- **Detachment/Re-attachment of virtualized devices:** Multiple parallel libraries to coordinate VM during migration (detach/reattach SR-IOV/IVShmem, migrate VMs, migration status)
- **IB Connection**: MPI runtime handles IB connection suspending and reactivating
- Propose Progress Engine (**PE**) and Migration Thread based (**MT**) design to optimize VM migration and MPI application performance

J. Zhang, X. Lu, D. K. Panda. High-Performance Virtual Machine Migration Framework for MPI Applications on SR-IOV enabled InfiniBand Clusters. IPDPS, 2017

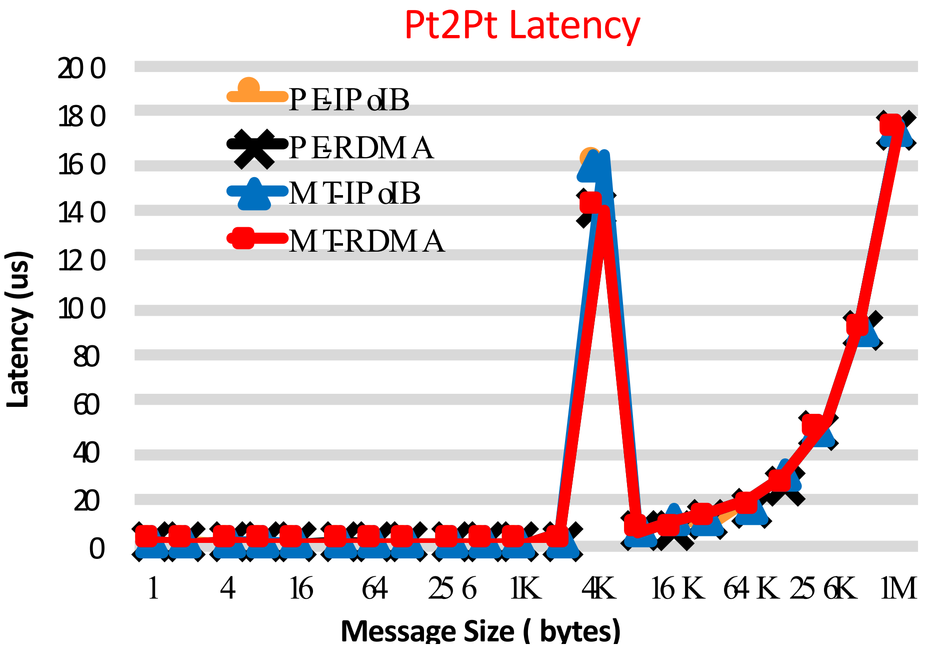
Performance Evaluation of VM Migration Framework

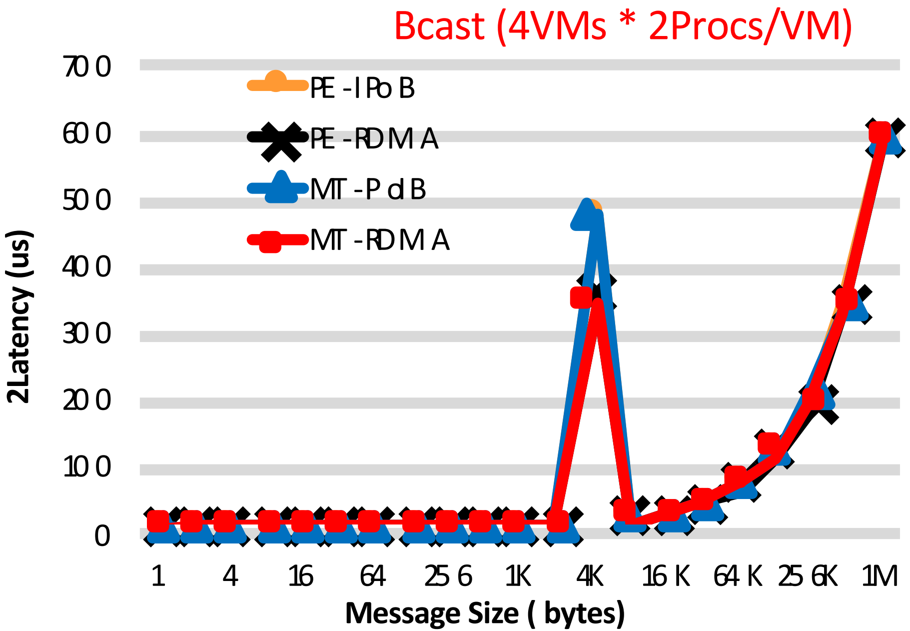




- Compared with the TCP, the RDMA scheme reduces the total migration time by 20%
- Total time is dominated by `Migration' time; Times on other steps are similar across different schemes
- Proposed migration framework could reduce up to 51% migration time

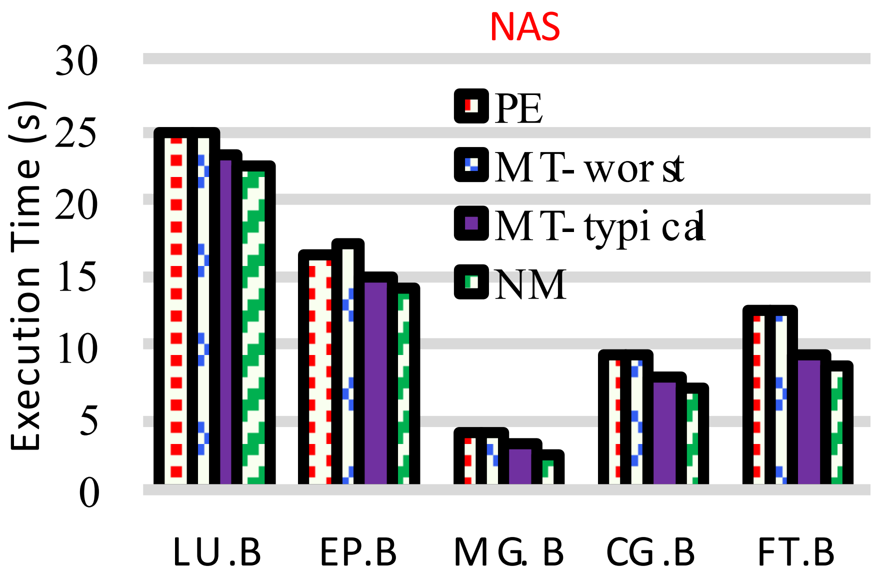
Performance Evaluation of VM Migration Framework

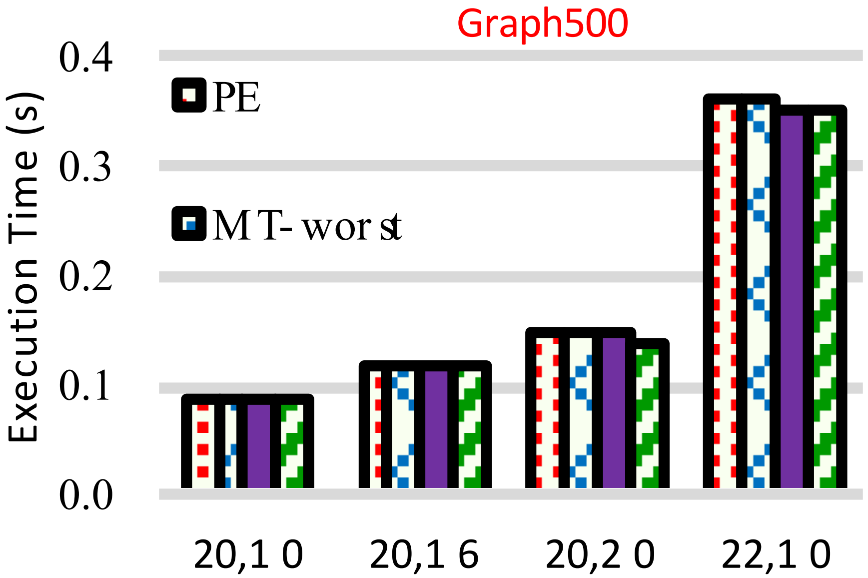




- Migrate a VM from one machine to another while benchmark is running inside
- Proposed MT-based designs perform slightly worse than PE-based designs because of lock/unlock
- No benefit from MT because of NO computation involved

Performance Evaluation with Applications



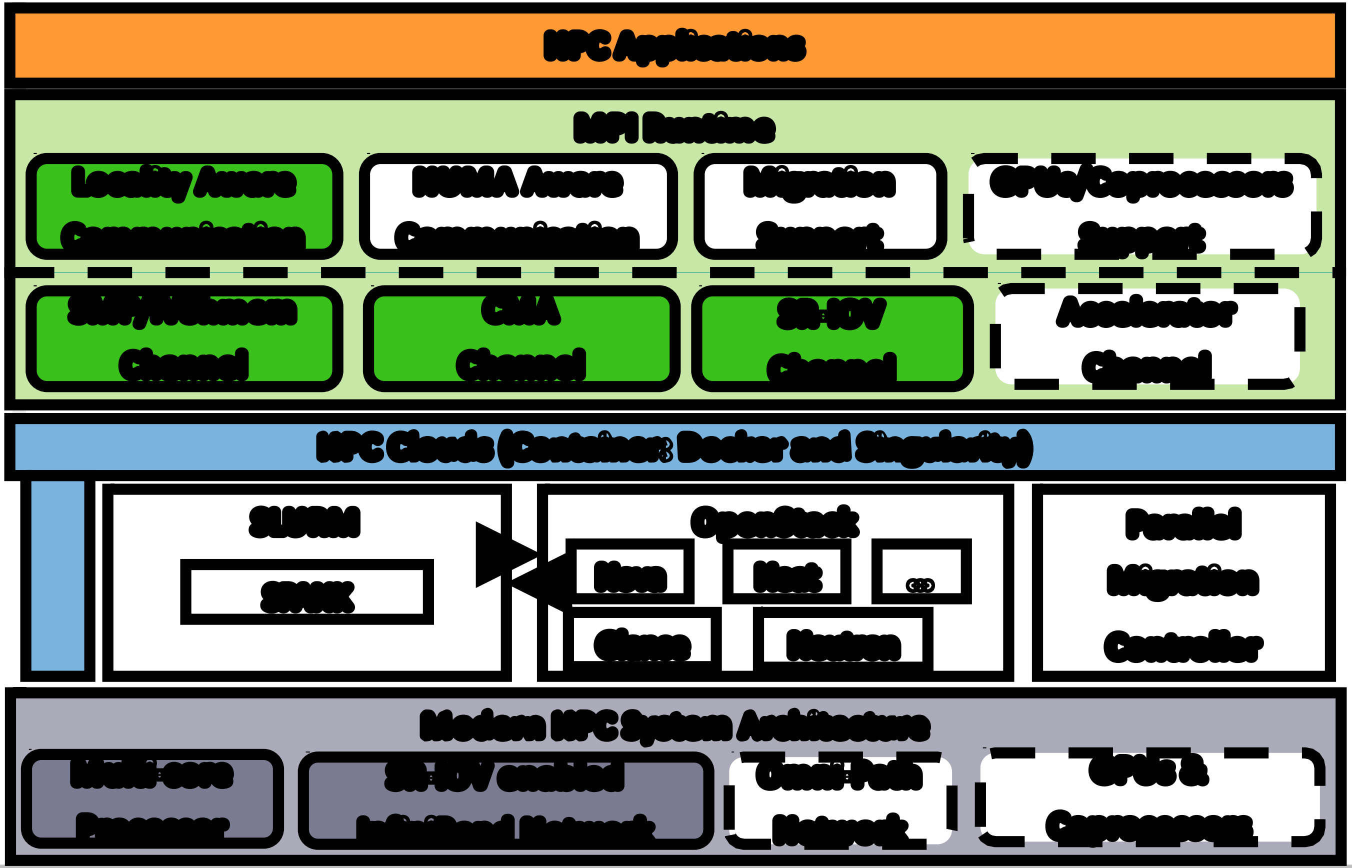


- 8 VMs in total and 1 VM carries out migration during application running
- Compared with NM, MT- worst and PE incur some overhead
- MT-typical allows migration to be completely overlapped with computation

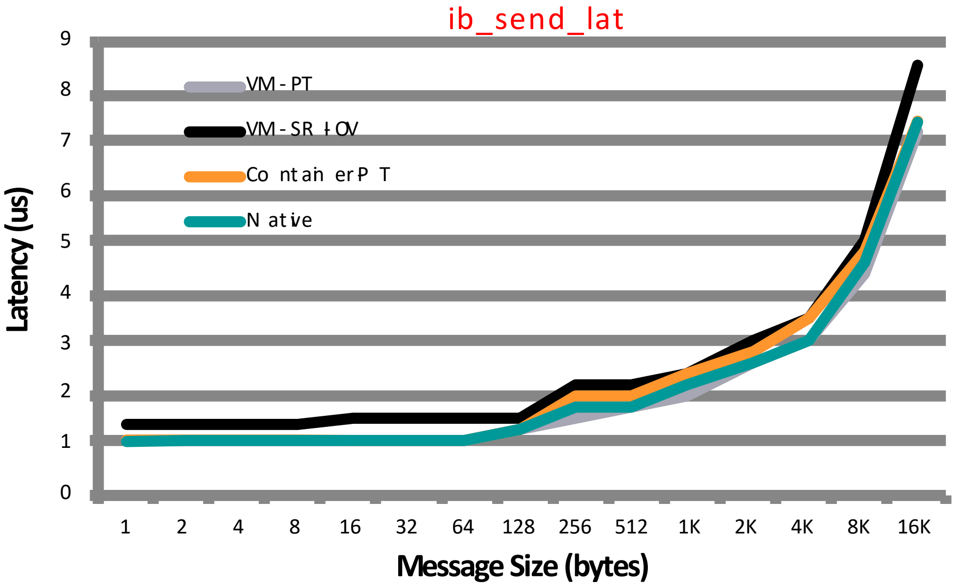
High-Performance MPI Library on HPC Clouds

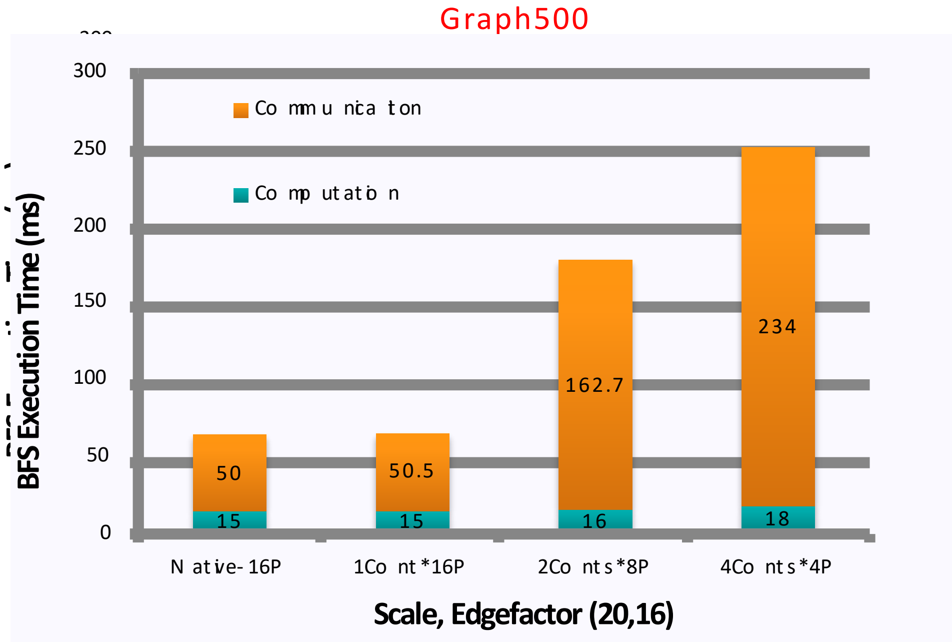
- MVAPICH2-Virt with SR-IOV and IVSHMEM
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)

MVAPICH2 with Containers (Docker and Singularity)



Benefits of Containers-based Virtualization for HPC on Cloud



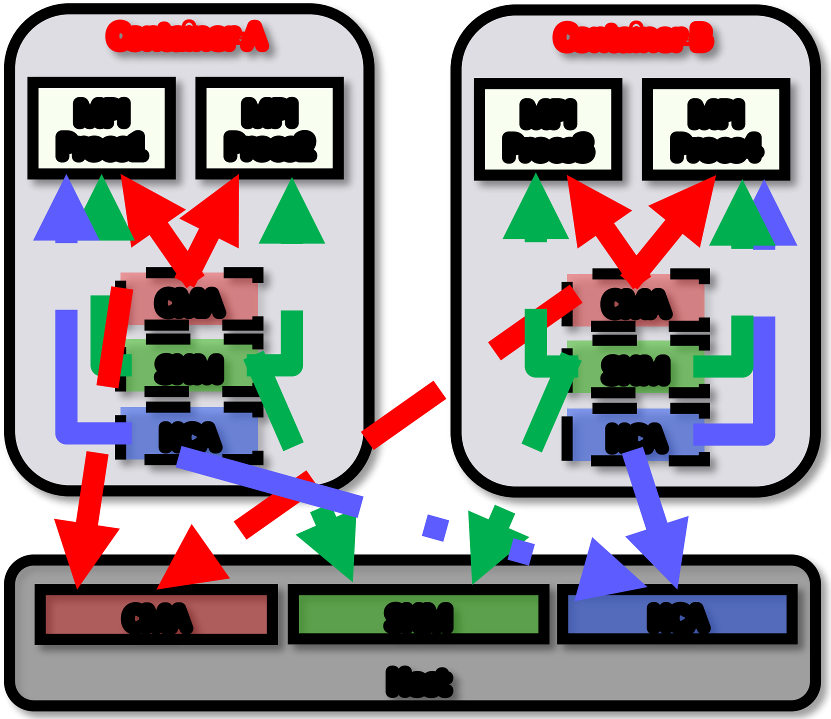


- Experiment on NFS Chameleon Cloud
- Container has less overhead than VM
- BFS time in Graph 500 significantly increases as the number of container increases on one host. Why?

J. Zhang, X. Lu, D. K. Panda. Performance Characterization of Hypervisor- and Container-Based Virtualization for HPC on SR-IOV Enabled InfiniBand Clusters. IPDRM, IPDPS Workshop, 2016

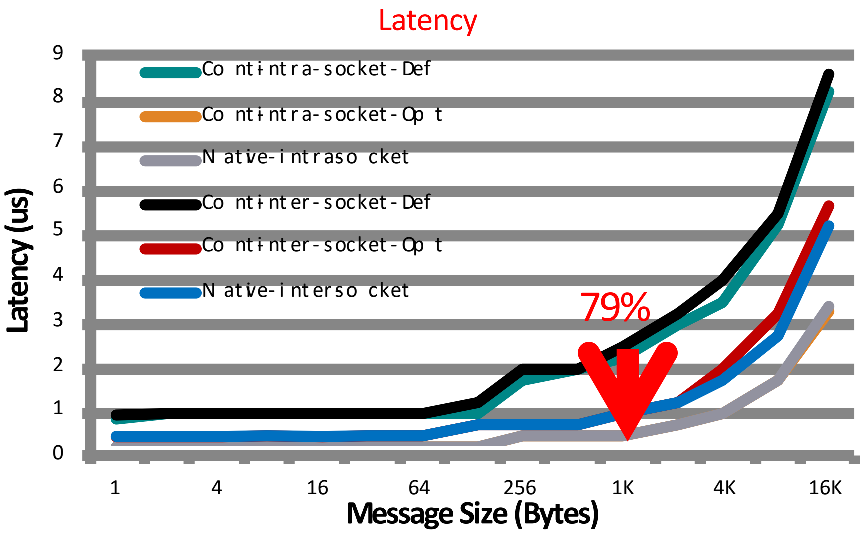
Containers-based Design: Issues, Challenges, and Approaches

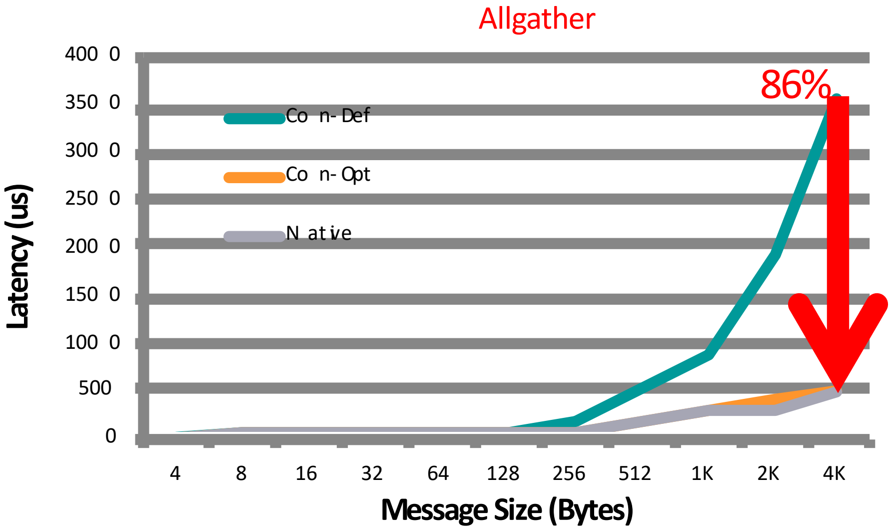
- What are the performance bottlenecks when running MPI applications on multiple containers per host in HPC cloud?
- Can we propose a new design to overcome the bottleneck on such container-based HPC cloud?
- Can optimized design deliver near-native performance for different container deployment scenarios?
- Locality-aware based design to enable CMA and Shared memory channels for MPI communication across co-resident containers



J. Zhang, X. Lu, D. K. Panda. High Performance MPI Library for Container-based HPC Cloud on InfiniBand Clusters. ICPP, 2016

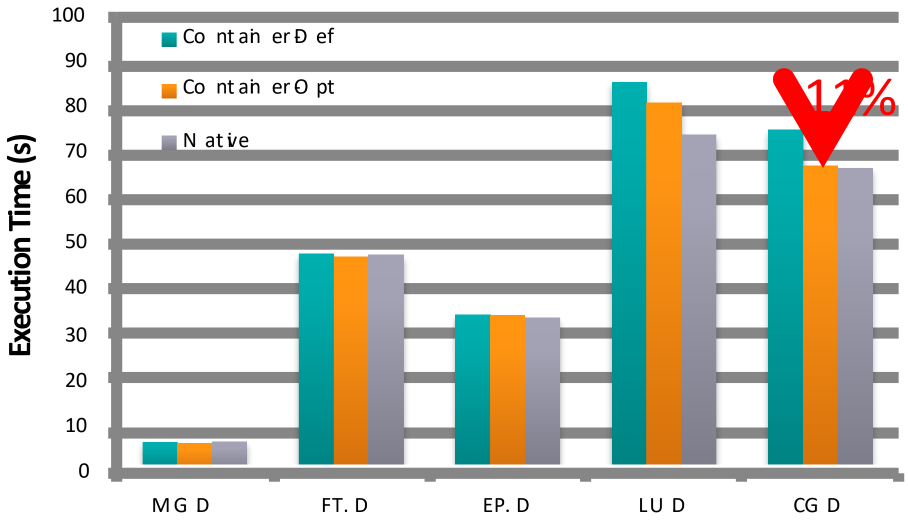
MPI Point-to-Point and Collective Performance



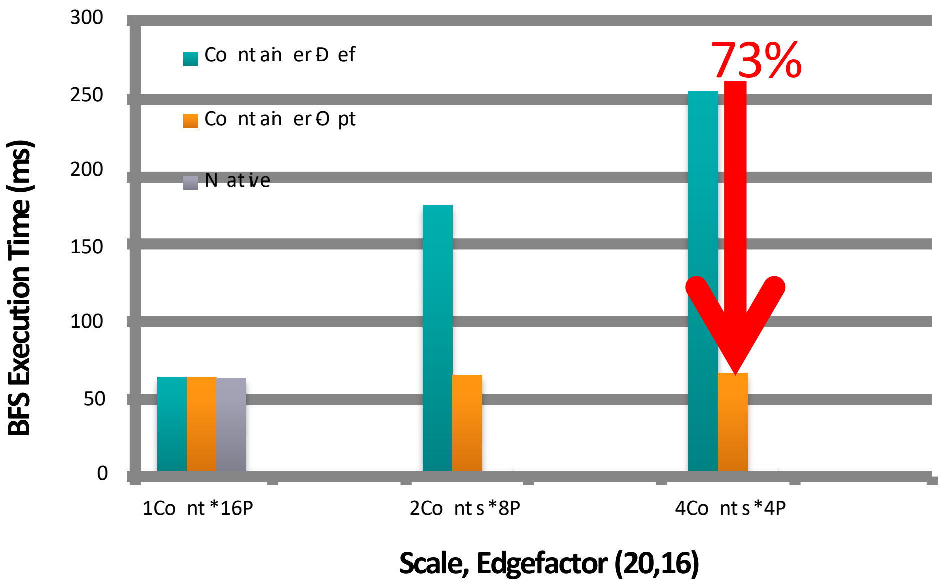


- Two containers are deployed on the same socket and different socket
- 256 procs totally (4 pros/container, 64 containers across 16 nodes evenly)
- Up to 79% and 86% improvement for Point-to-Point and MPI_Allgather, respectively (Cont-Opt vs. Cont-Def)
- Minor overhead, compared with Native performance (Cont-*-Opt vs. Native-*)

Application-Level Performance on Docker with MVAPICH2



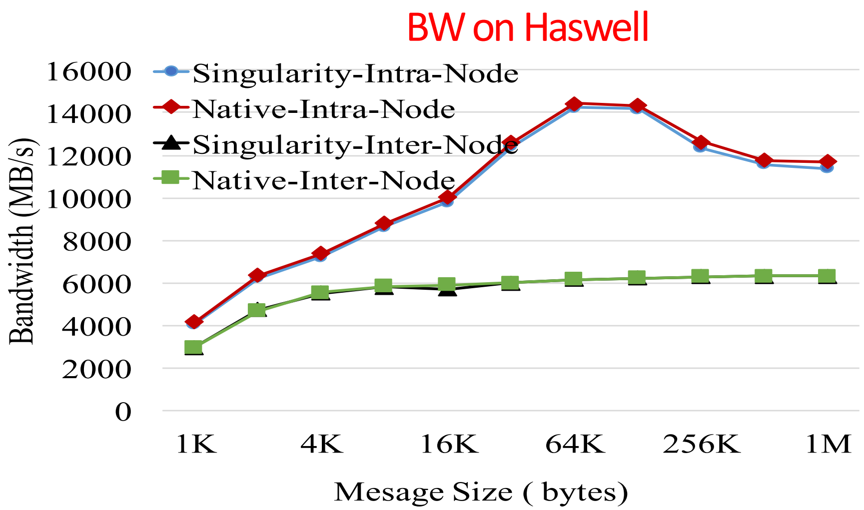
NAS

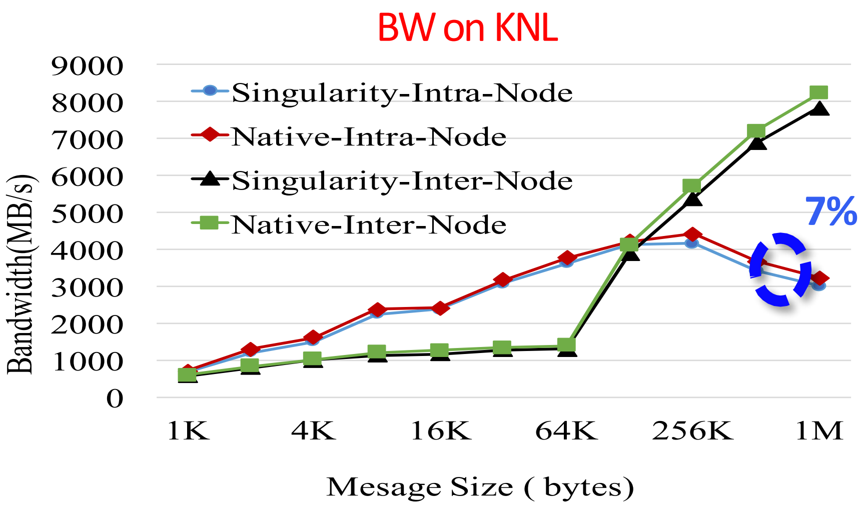


Graph 500

- 64 Containers across 16 nodes, pining 4 Cores per Container
- Compared to Container-Def, up to 11% and 73% of execution time reduction for NAS and Graph 500
- Compared to Native, less than 9 % and 5% overhead for NAS and Graph 500

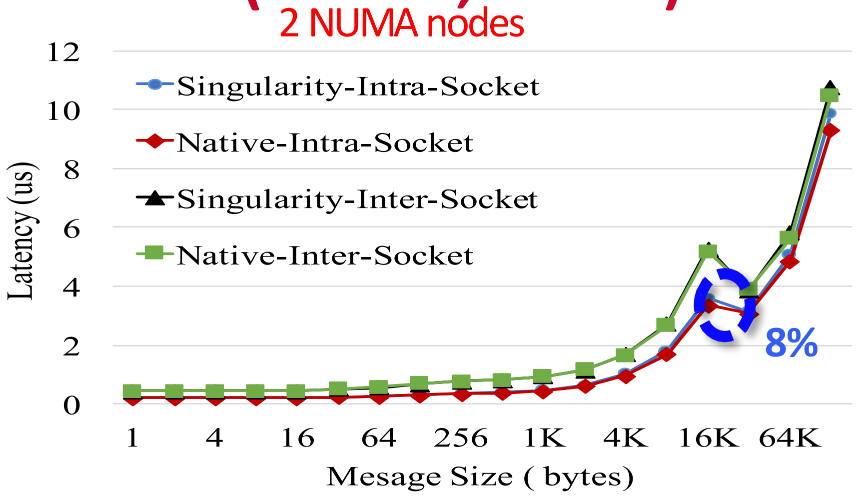
Singularity Performance on Different Processor Architectures

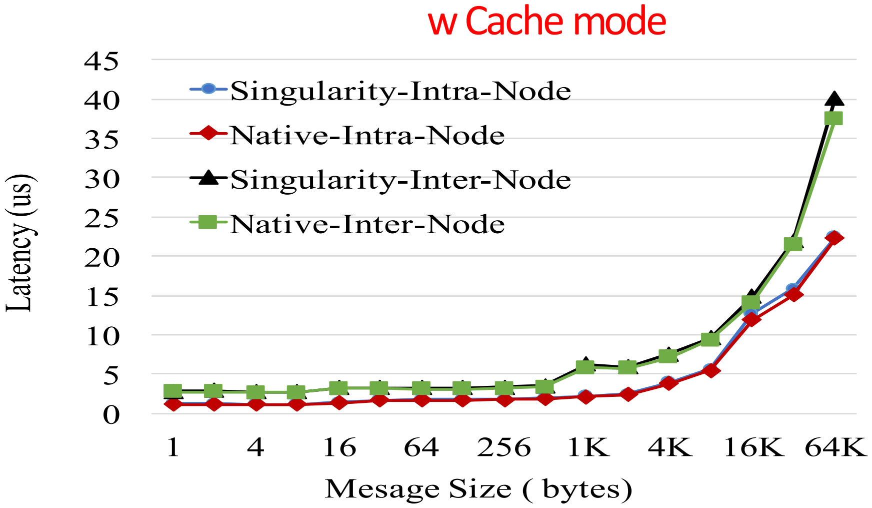




- MPI point-to-point Bandwidth
- On both Haswell and KNL, less than 7% overhead for Singularity solution
- Worse intra-node performance than Haswell because low CPU frequency, complex cluster mode, and cost maintaining cache coherence
- KNL - Inter-node performs better than intra-node case after around 256 Kbytes, as Omni-Path interconnect outperforms shared memory-based transfer for large message size

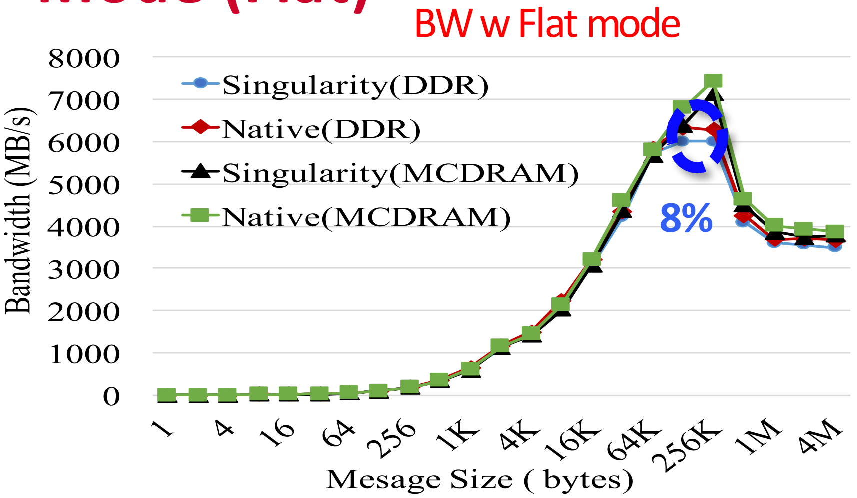
Singularity Performance on Different Memory Access Mode (NUMA, Cache)

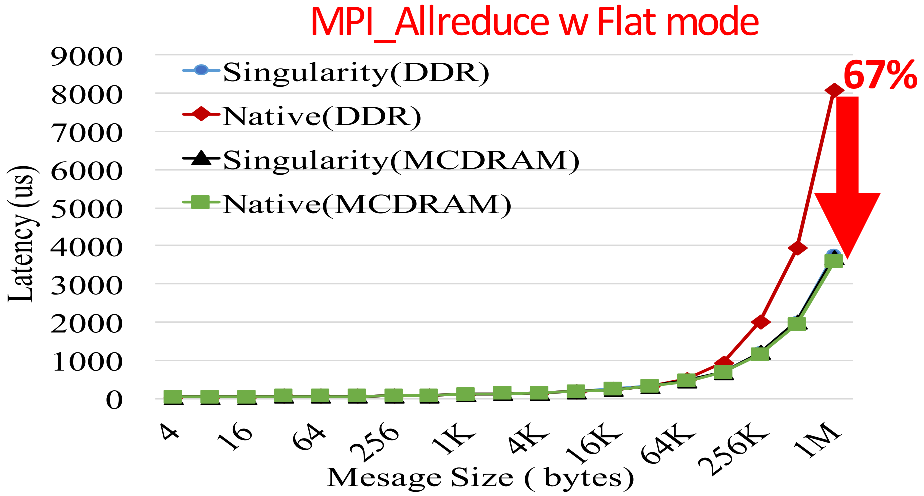




- MPI point-to-point Latency
- NUMA
  - Intra-socket performs better than inter-socket case, as the QPI bottleneck between NUMA nodes
  - Performance difference is gradually decreased, as the message size increases
- Overall, less than 8% overhead for Singularity solution in both cases, compared with Native

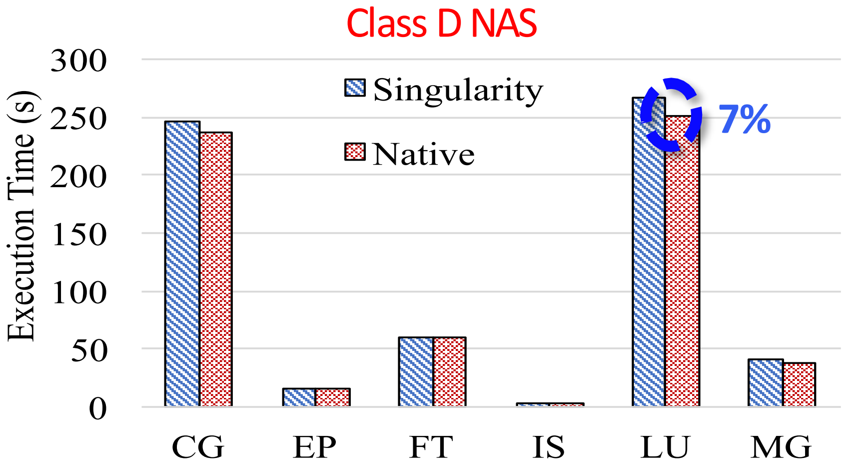
Singularity Performance on Different Memory Access Mode (Flat)

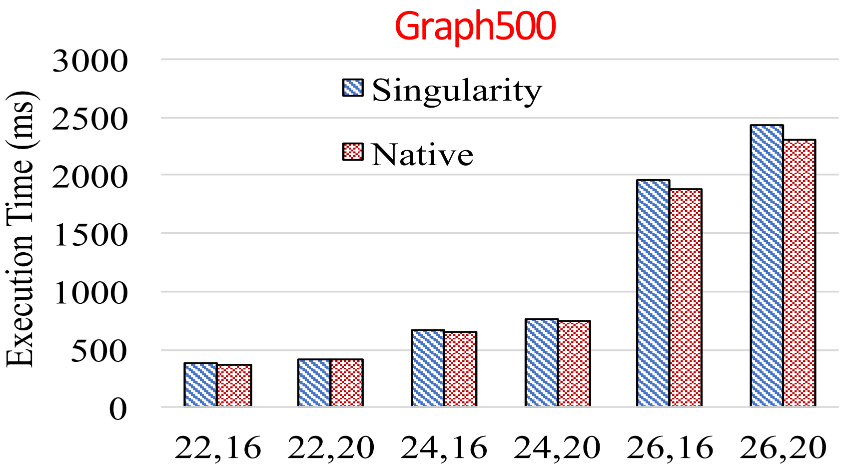




- Explicitly specify DDR or MCDRAM for memory allocation
- MPI point-to-point BW: No significant performance difference
- MPI collective Allreduce: Clear benefits (up to 67%) with MCDRAM after around 256 KB message, compared with DDR
- More parallel processes increase data access, which can NOT fit in L2 cache, higher BW in MCDRAM
- Near-native performance for Singularity (less than 8% overhead)

Singularity Performance on Haswell with InfiniBand





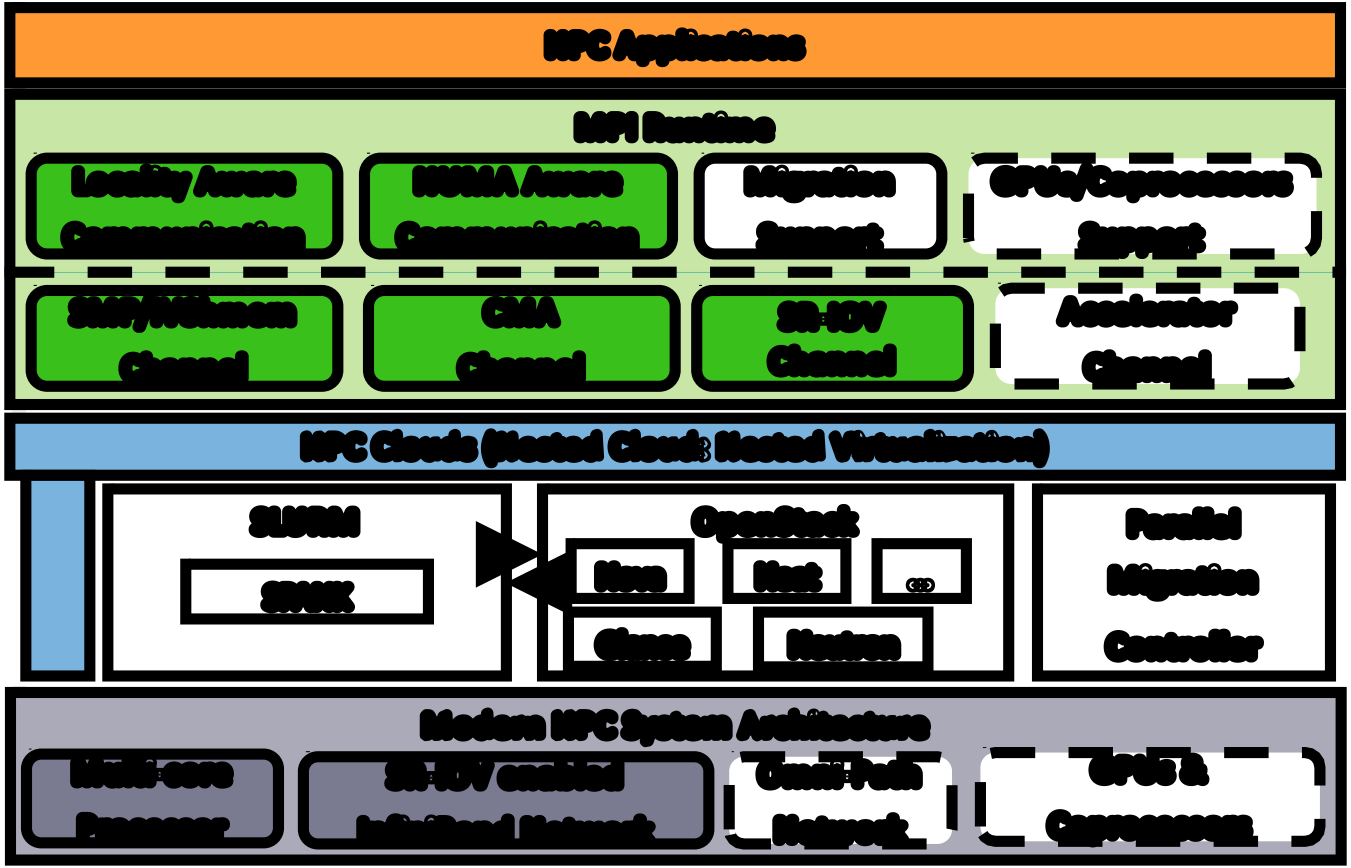
- 512 processors across 32 Haswell nodes
- Singularity delivers near-native performance, less than 7% overhead on Haswell with InfiniBand

J. Zhang, X. Lu, D. K. Panda. Is Singularity-based Container Technology Ready for Running MPI Applications on HPC Clouds? UCC 2017. (Best Student Paper Award)

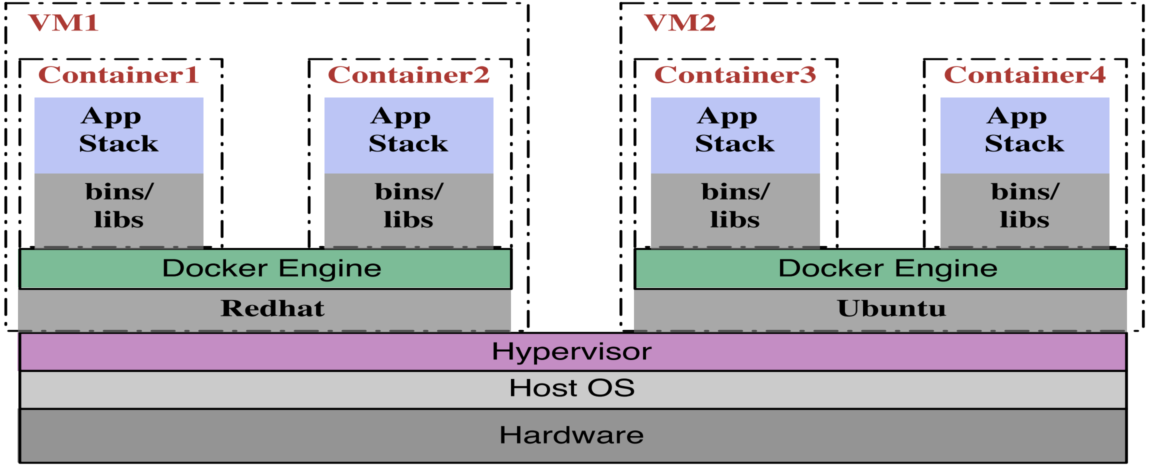
High-Performance MPI Library on HPC Clouds

- MVAPICH2-Virt with SR-IOV and IVSHMEM
- SR-IOV-enabled VM Migration Support in MVAPICH2
- MVAPICH2 with Containers (Docker and Singularity)
- MVAPICH2 with Nested Virtualization (Container over VM)

MVAPICH2 with Nested Virtualization

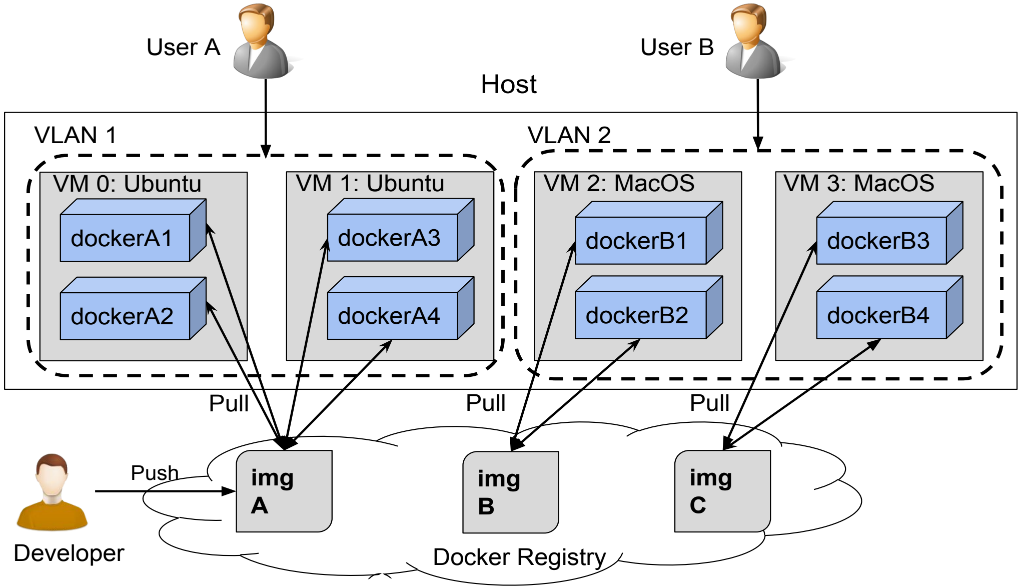


Nested Virtualization: Containers over Virtual Machines



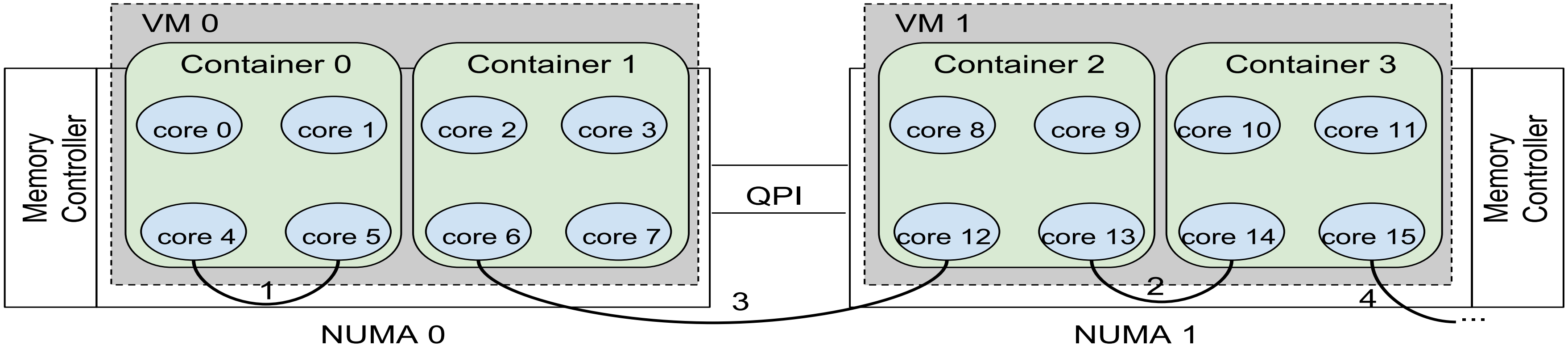
- Useful for live migration, sandbox application, legacy system integration, software deployment, etc.
- Performance issues because of the redundant call stacks (two-layer virtualization) and isolated physical resources

Usage Scenario of Nested Virtualization



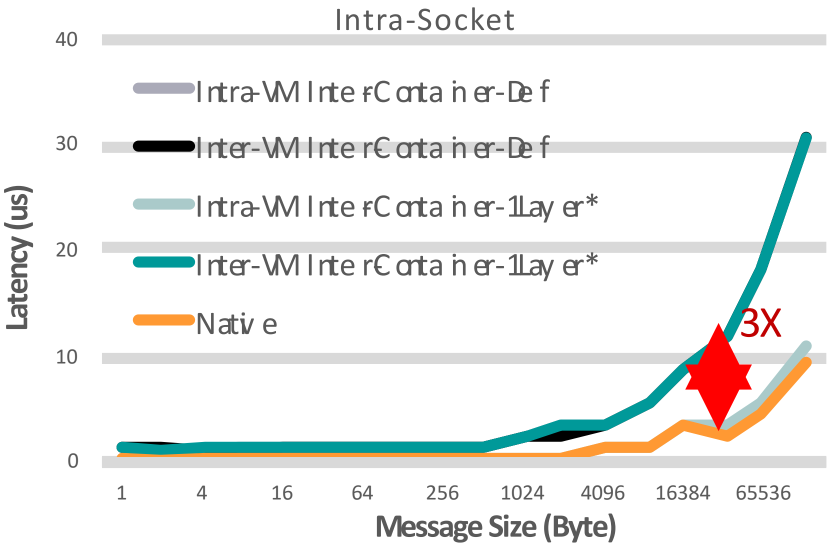
- VM provides good isolation and security so that the applications and workloads of users A and B will not interfere with each other
- Root permission of VM can be given to do special configuration
- Docker brings an effective, standardized and repeatable way to port and distribute the applications and workloads

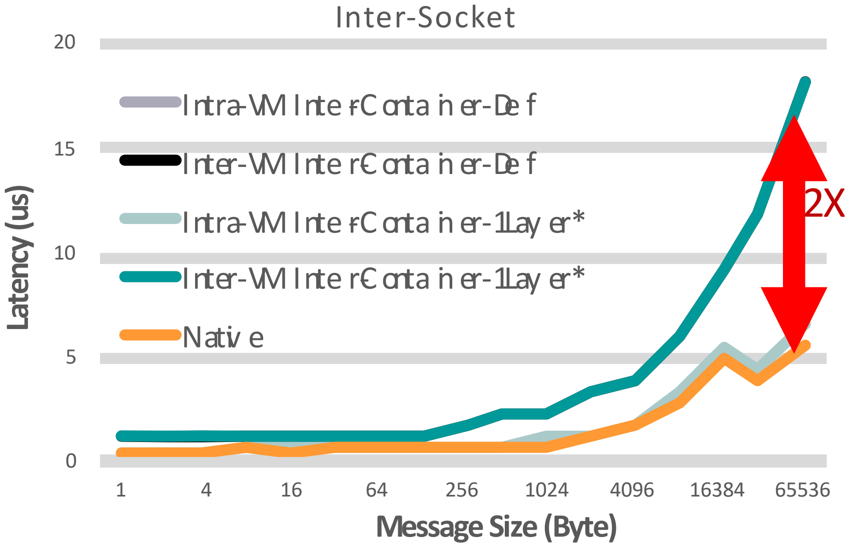
Multiple Communication Paths in Nested Virtualization



- Different VM placements introduce multiple communication paths on container level
  1. Intra-VM Intra-Container (across core 4 and core 5)
  2. Intra-VM Inter-Container (across core 13 and core 14)
  3. Inter-VM Inter-Container (across core 6 and core 12)
  4. Inter-Node Inter-Container (across core 15 and the core on remote node)

Performance Characteristics on Communication Paths





- Two VMs are deployed on the same and different socket, respectively
- *-Def and Inter-VM Inter-Container-1Layer have similar performance
- Still large gap compared to native performance with just 1layer design

1Layer* - J. Zhang, X. Lu, D. K. Panda. High Performance MPI Library for Container-based HPC Cloud on InfiniBand, ICPP, 2016

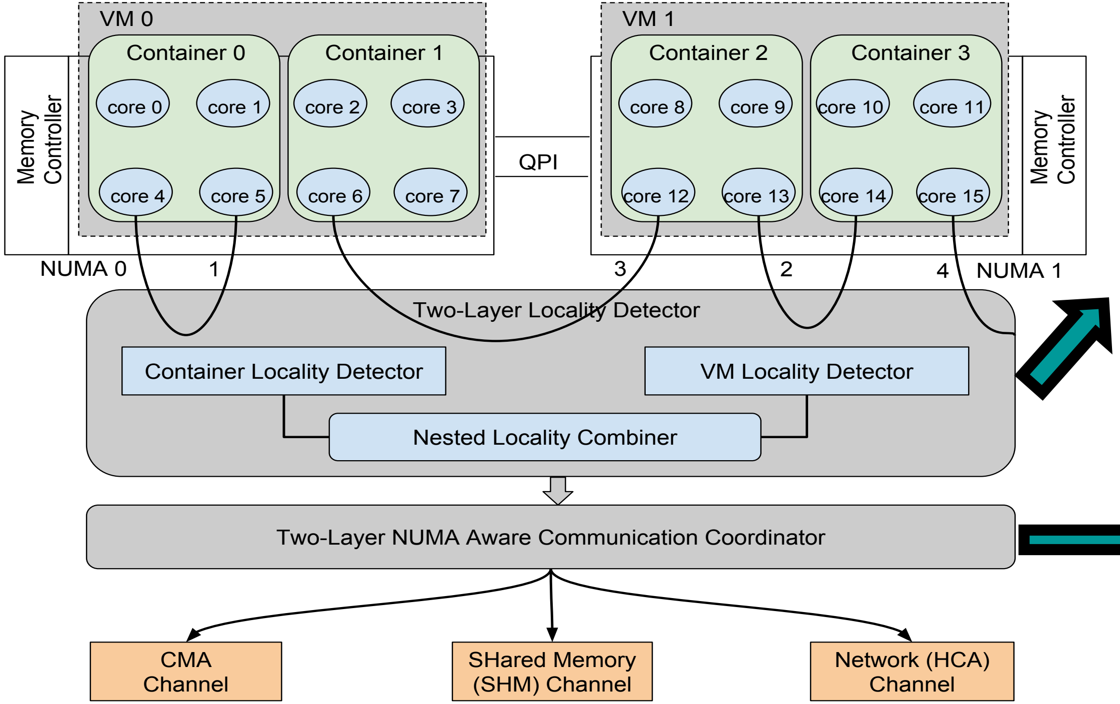
Challenges of Nested Virtualization

- How to further reduce the performance overhead of running applications on the nested virtualization environment?

- What are the impacts of the different VM/container placement schemes for the communication on the container level?

- Can we propose a design which can adapt these different VM/container placement schemes and deliver near-native performance for nested virtualization environments?

Overview of Proposed Design in MVAPICH2

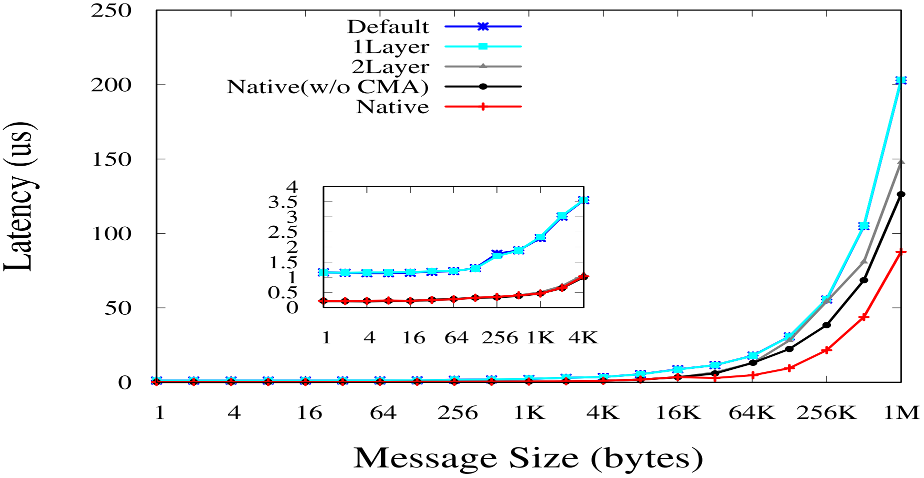


Two-Layer Locality Detector: Dynamically detecting MPI processes in the co-resident containers inside one VM as well as the ones in the co-resident VMs

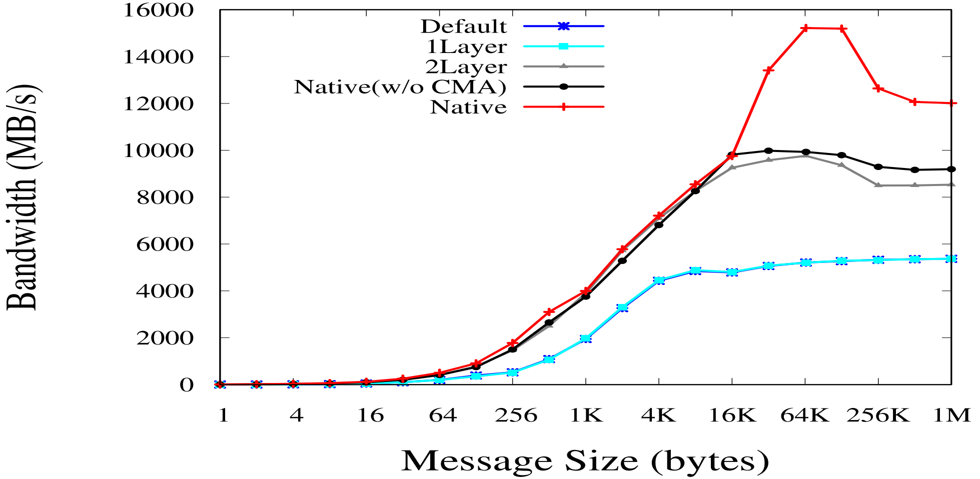
Two-Layer NUMA Aware Communication Coordinator: Leverage nested locality info, NUMA architecture info and message to select appropriate communication channel

J. Zhang, X. Lu, D. K. Panda. Designing Locality and NUMA Aware MPI Runtime for Nested Virtualization based HPC Cloud with SR-IOV Enabled InfiniBand, VEE, 2017

Inter-VM Inter-Container Pt2Pt (Intra-Socket)



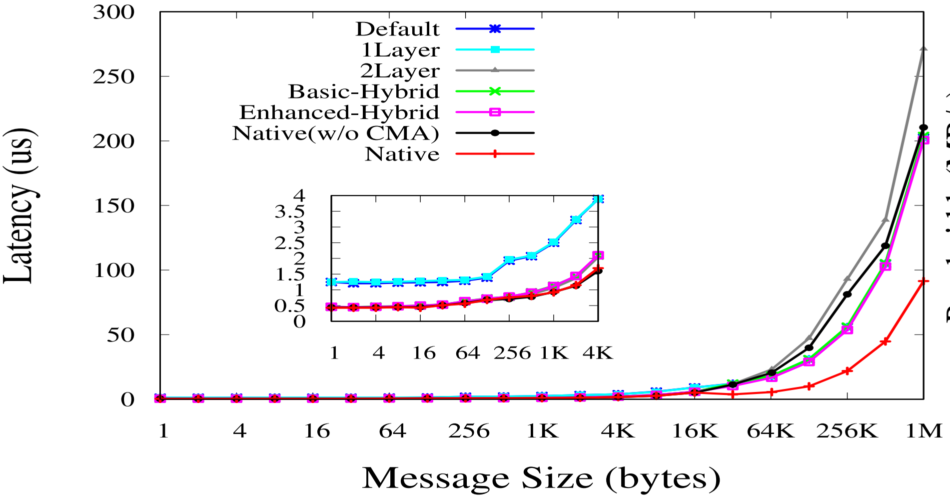
Latency



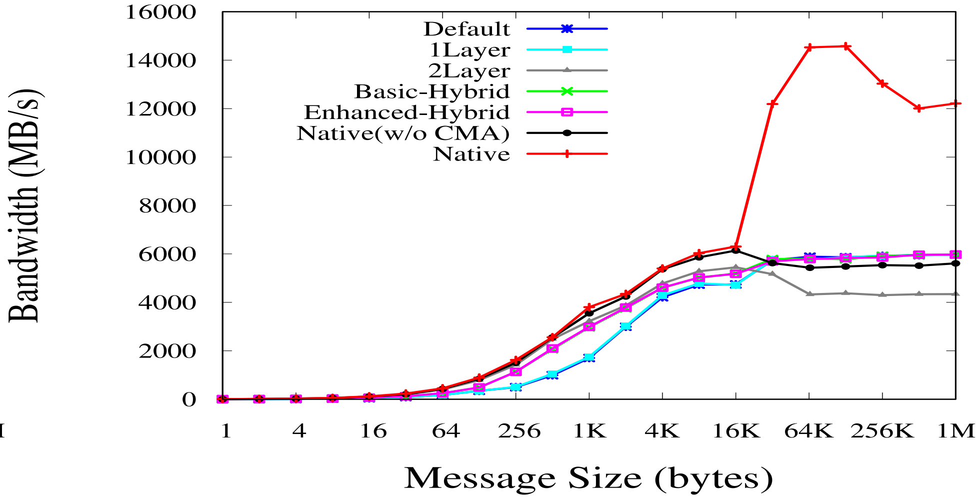
BW

- 1Layer has similar performance to the Default
- Compared with 1Layer, 2Layer delivers up to 84% and 184% improvement for latency and BW

Inter-VM Inter-Container Pt2Pt (Inter-Socket)



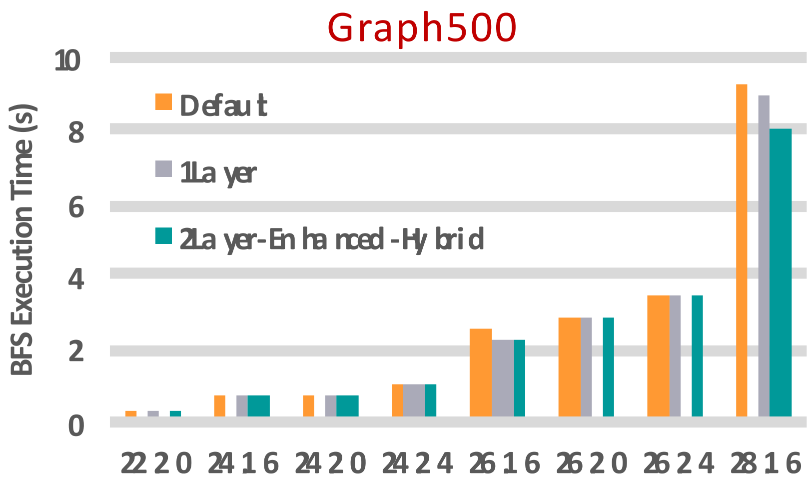
Latency

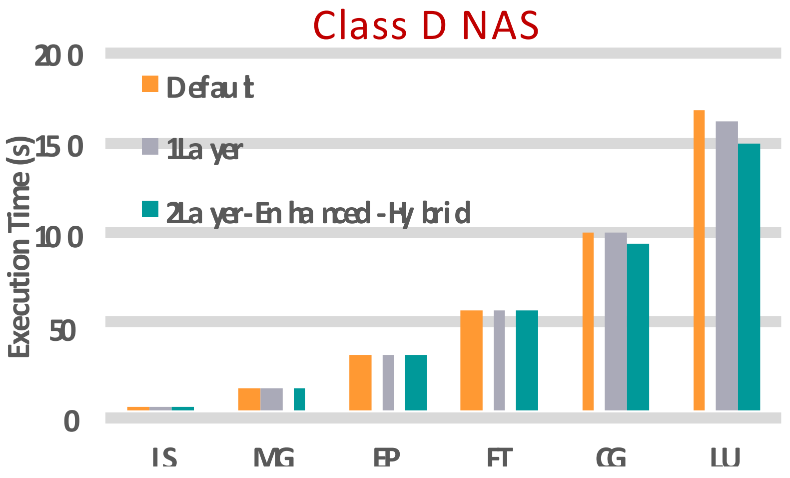


BW

- 1-Layer has similar performance to the Default
- 2-Layer has near-native performance for small msg, but clear overhead on large msg
- Compared to 2-Layer, Hybrid design brings up to 42% and 25% improvement for latency and BW, respectively

Applications Performance



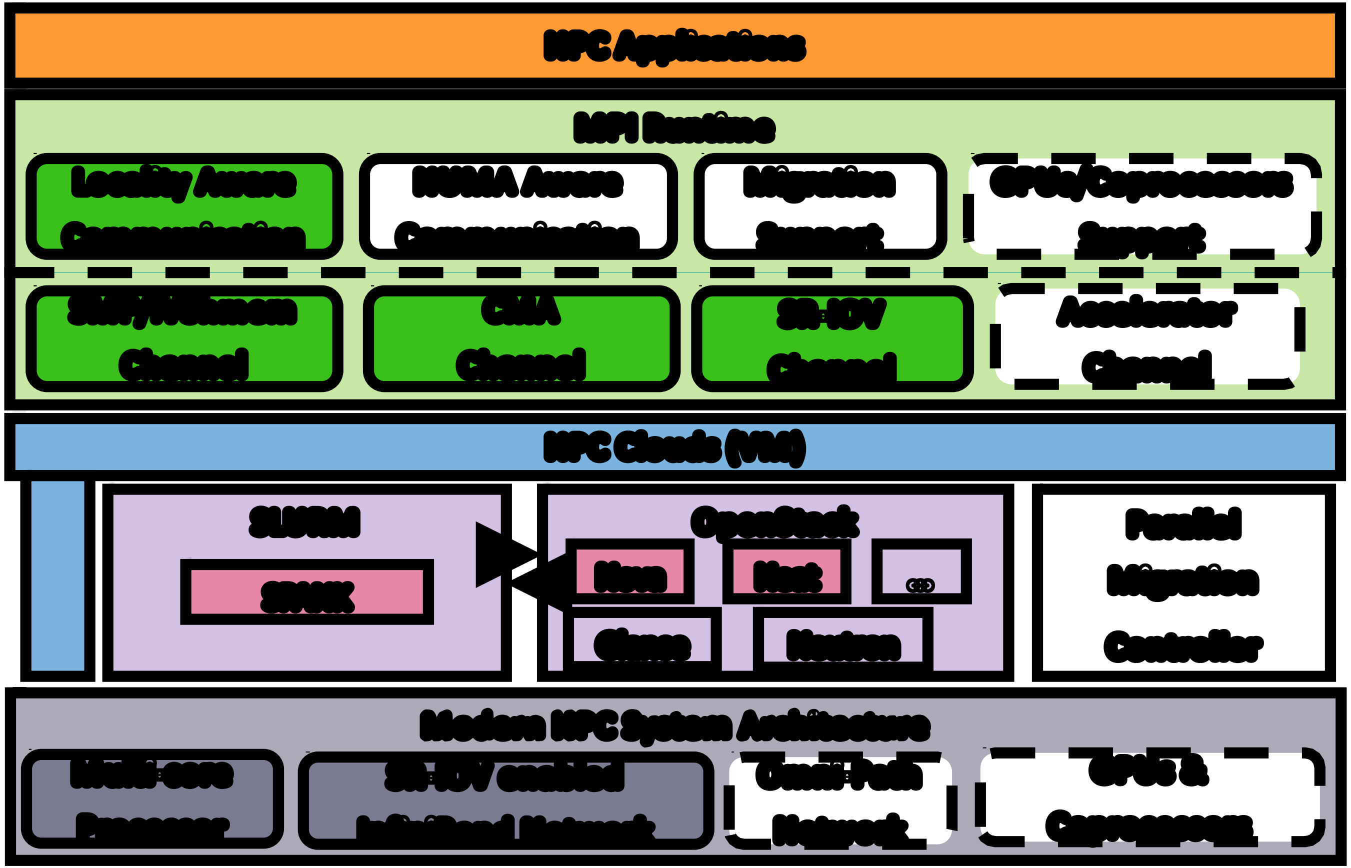


- 256 processes across 64 containers on 16 nodes
- Compared with Default, enhanced-hybrid design reduces up to 16% (28,16) and 10% (LU) of execution time for Graph 500 and NAS, respectively
- Compared with the 1Layer case, enhanced-hybrid design also brings up to 12% (28,16) and 6% (LU) performance benefit.

Outline

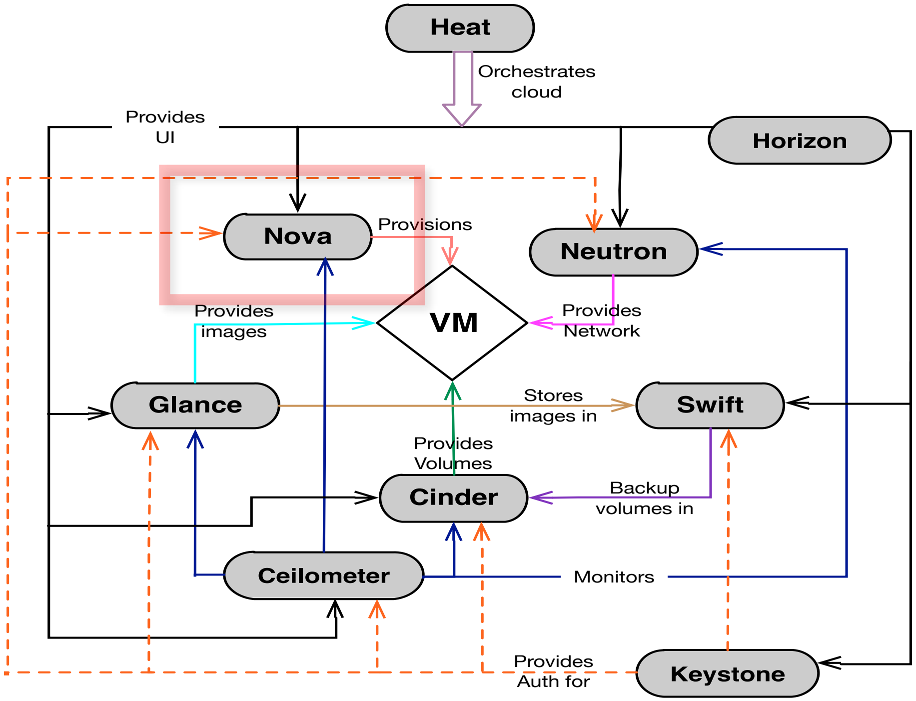
- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

Integrated Design with SLURM and OpenStack



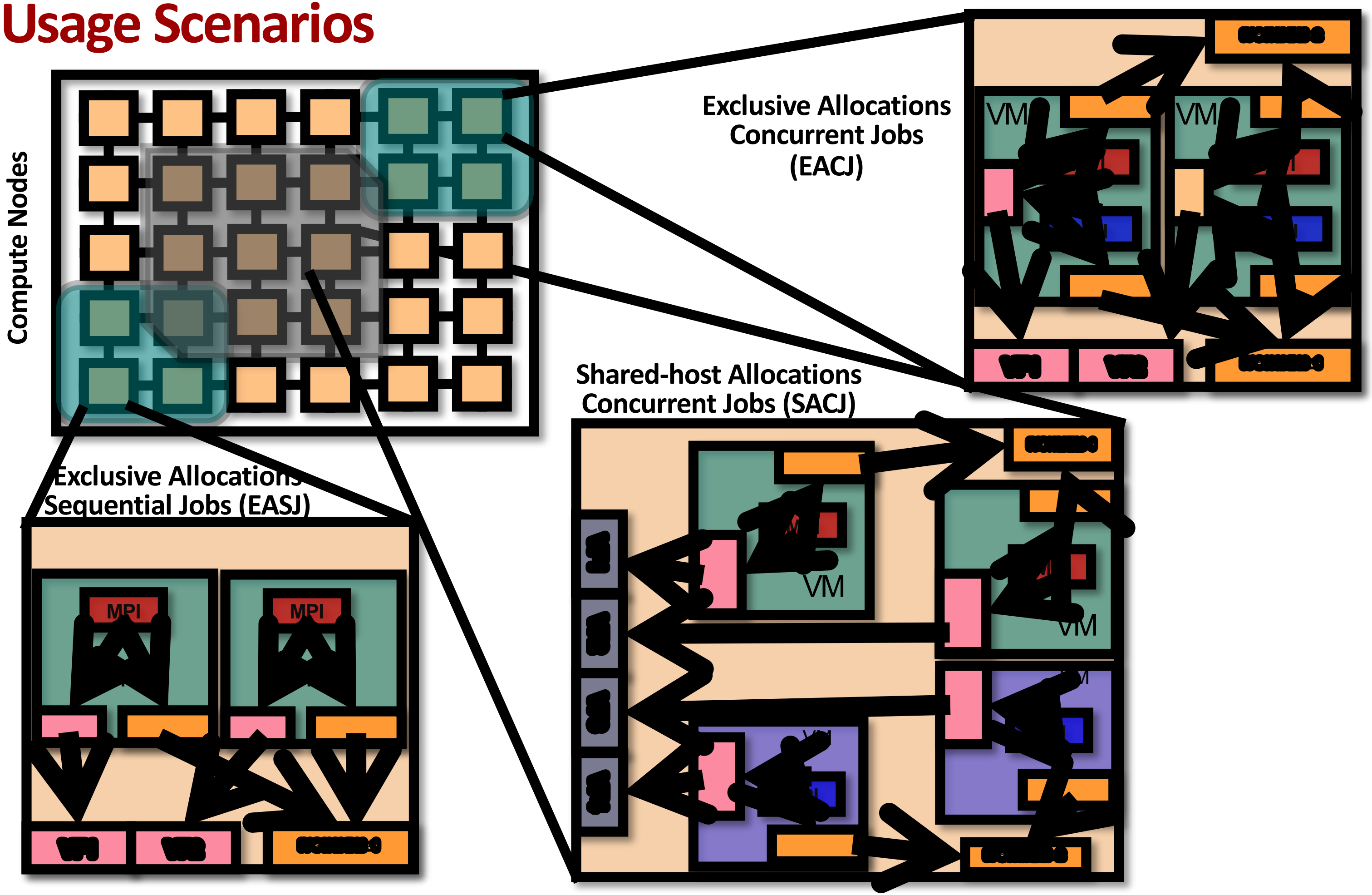
MVAPICH2-Virt with SR-IOV and IVSHMEM over OpenStack

- OpenStack is one of the most popular open-source solutions to build clouds and manage virtual machines
- Deployment with OpenStack
  - Supporting SR-IOV configuration
  - Supporting IVSHMEM configuration
  - Virtual Machine aware design of MVAPICH2 with SR-IOV
- An efficient approach to build HPC Clouds with MVAPICH2-Virt and OpenStack



J. Zhang, X. Lu, M. Arnold, D. K. Panda. MVAPICH2 over OpenStack with SR-IOV: An Efficient Approach to Build HPC Clouds. CCGrid, 2015

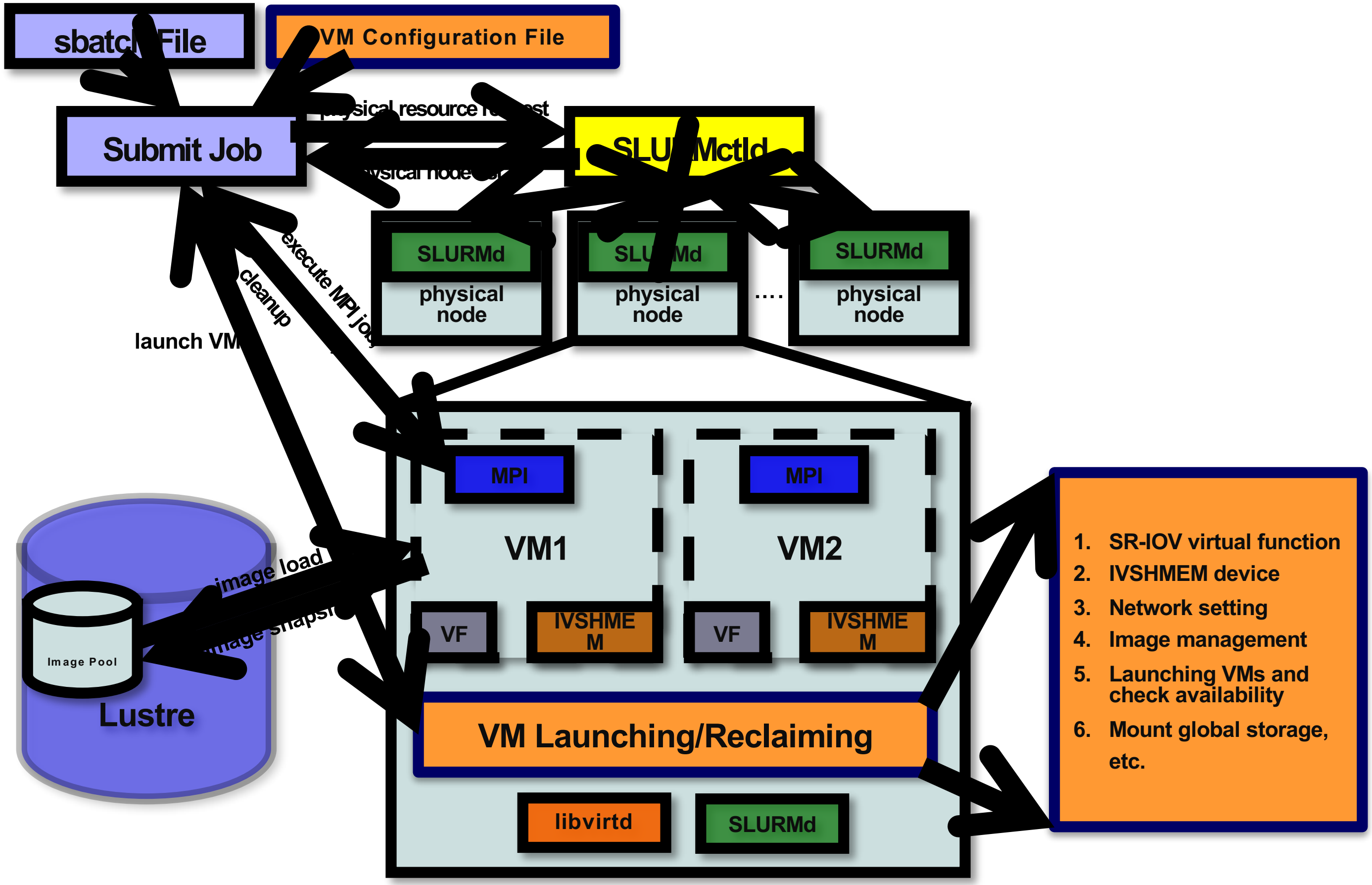
Typical Usage Scenarios



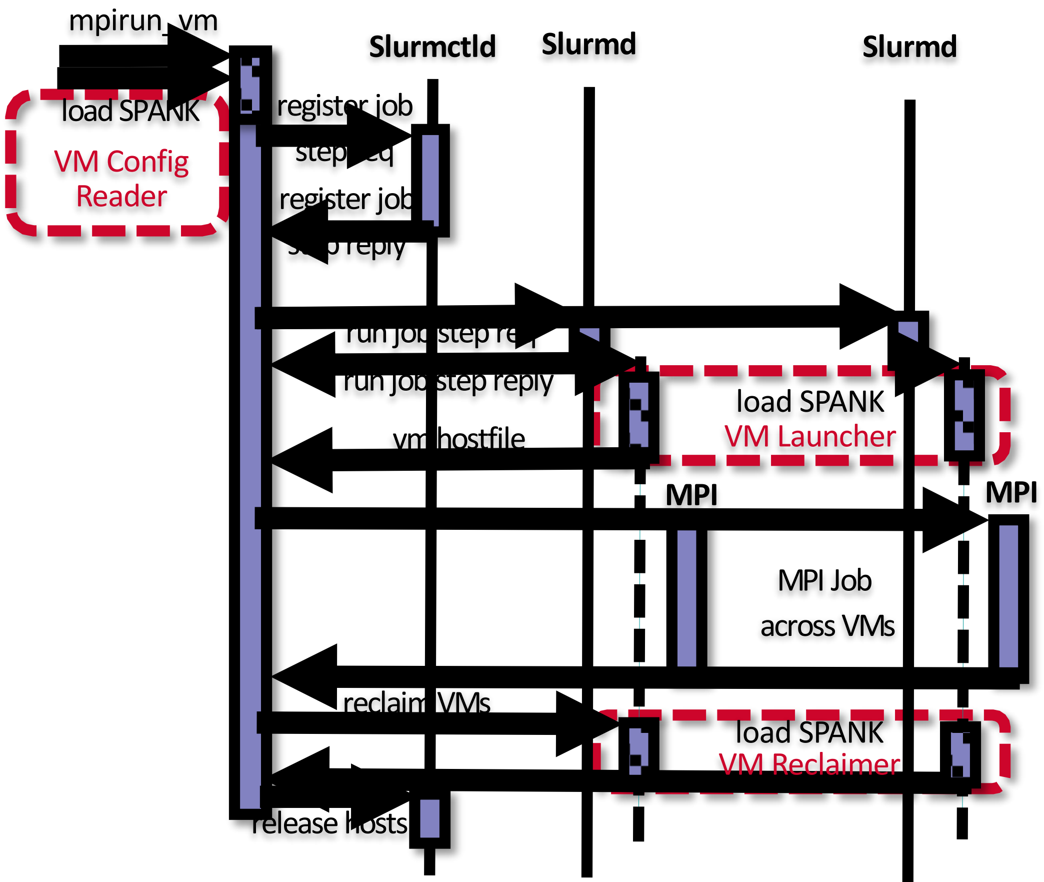
Need for Supporting SR-IOV and IVSHMEM in SLURM

- Requirement of managing and isolating virtualized resources of SR-IOV and IVSHMEM
- Such kind of management and isolation is hard to be achieved by MPI library alone, but much easier with SLURM
- Efficient running MPI applications on HPC Clouds needs SLURM to support managing SR-IOV and IVSHMEM
  - Can critical HPC resources be efficiently shared among users by extending SLURM with support for SR-IOV and IVSHMEM based virtualization?
  - Can SR-IOV and IVSHMEM enabled SLURM and MPI library provide bare-metal performance for end applications on HPC Clouds?

Architecture Overview

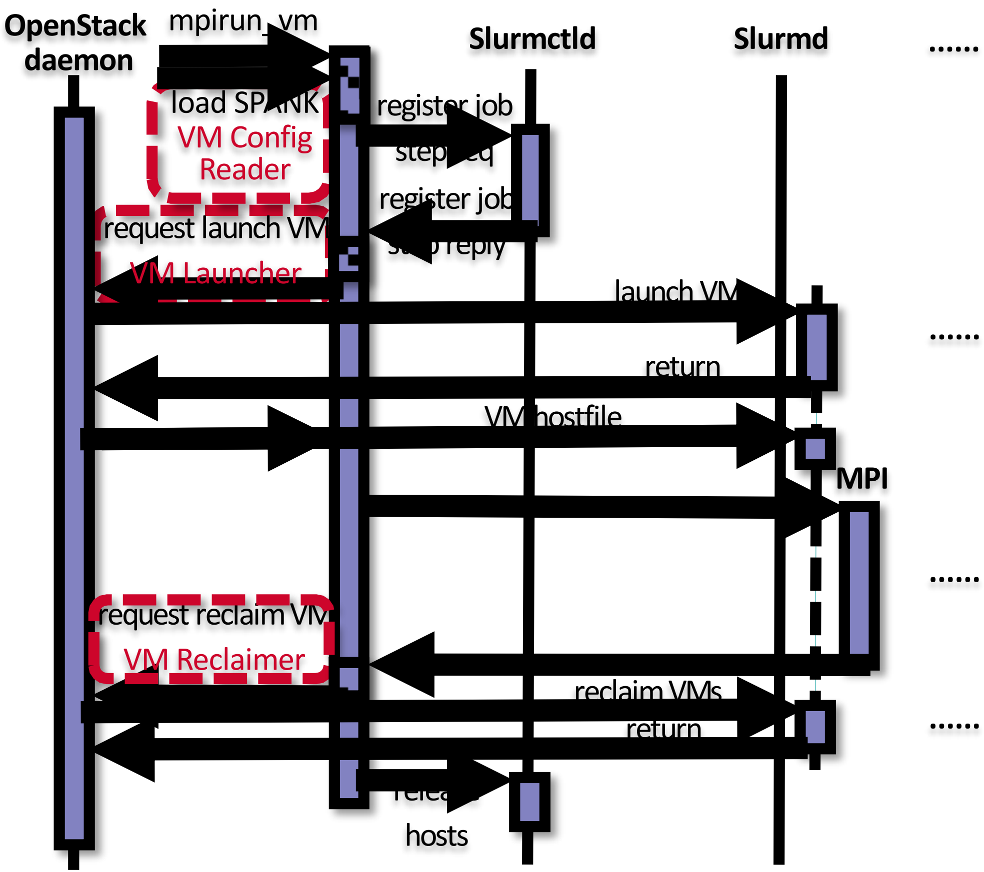


SLURM SPANK Plugin based Design



- **VM Configuration Reader –** Register all VM configuration options, set in the job control environment so that they are visible to all allocated nodes.
- **VM Launcher –** Setup VMs on each allocated nodes.
  - File based lock to detect occupied VF and exclusively allocate free VF
  - Assign a unique ID to each IVSHMEM and dynamically attach to each VM
- **VM Reclaimer –** Tear down VMs and reclaim resources

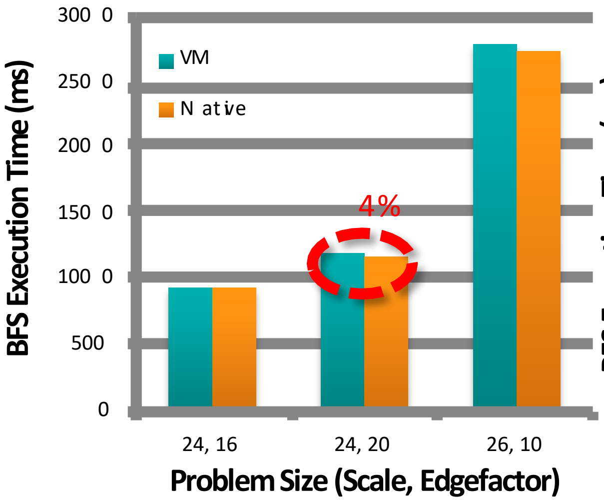
SLURM SPANK Plugin with OpenStack based Design



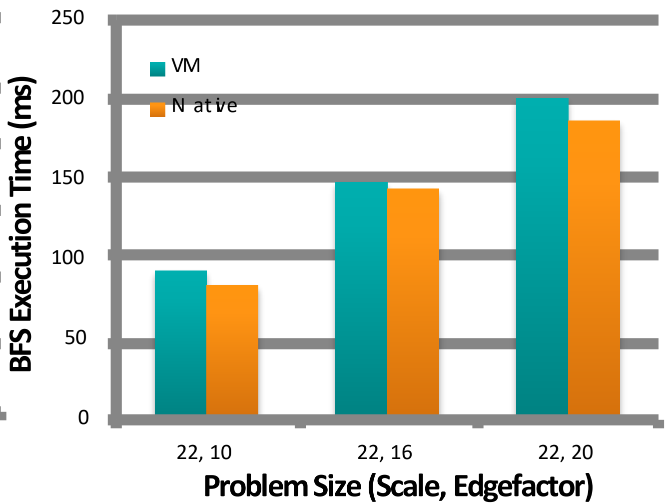
- VM Configuration Reader – VM options register
- VM Launcher, VM Reclaimer – Offload to underlying OpenStack infrastructure
  - PCI Whitelist to passthrough free VF to VM
  - Extend Nova to enable IVSHMEM when launching VM

J. Zhang, X. Lu, S. Chakraborty, D. K. Panda. SLURM-V: Extending SLURM for Building Efficient HPC Cloud with SR-IOV and IVShmem. Euro-Par, 2016

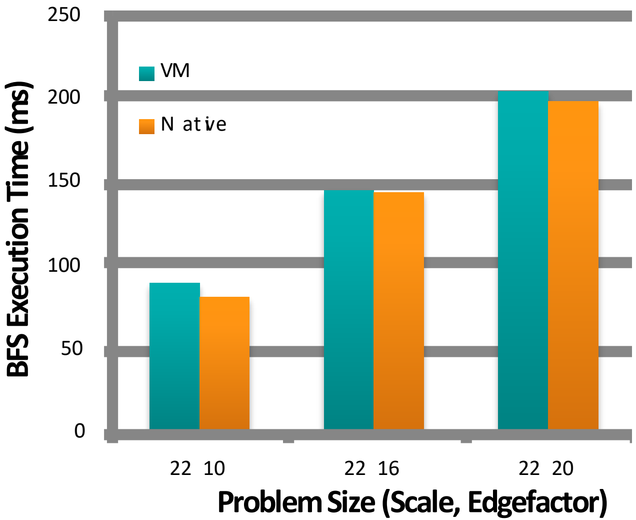
Application-Level Performance on Chameleon (Graph500)



Exclusive Allocations
Sequential Jobs



Shared-host Allocations
Concurrent Jobs



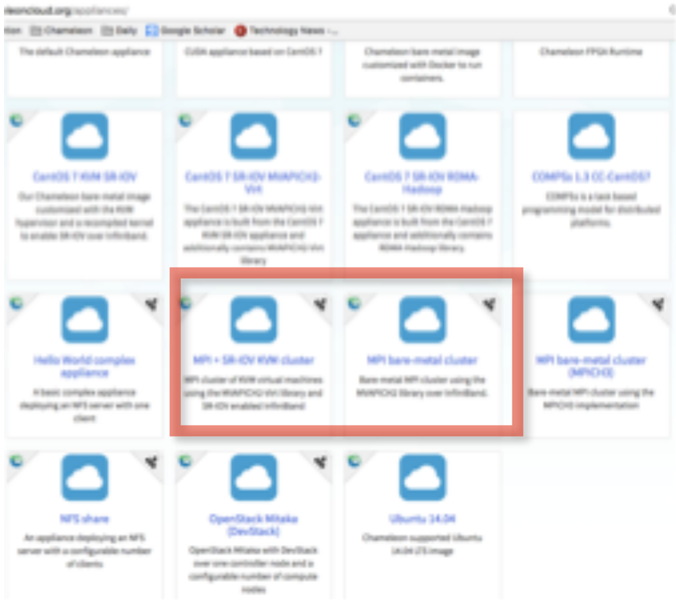
Exclusive Allocations
Concurrent Jobs

- 32 VMs across 8 nodes, 6 Core/VM
- EASJ - Compared to Native, less than 4% overhead with 128 Procs
- SACJ, EACJ – Also minor overhead, when running NAS as concurrent job with 64 Procs

Outline

- Overview of Cloud Computing System Software
- Overview of Modern HPC Cloud Architecture
- Challenges of Building HPC Clouds
- High-Performance MPI Library on HPC Clouds
- Integrated Designs with Cloud Resource Manager
- Appliances and Demos on Chameleon Cloud
- Conclusion and Q&A

Available Appliances on Chameleon Cloud*

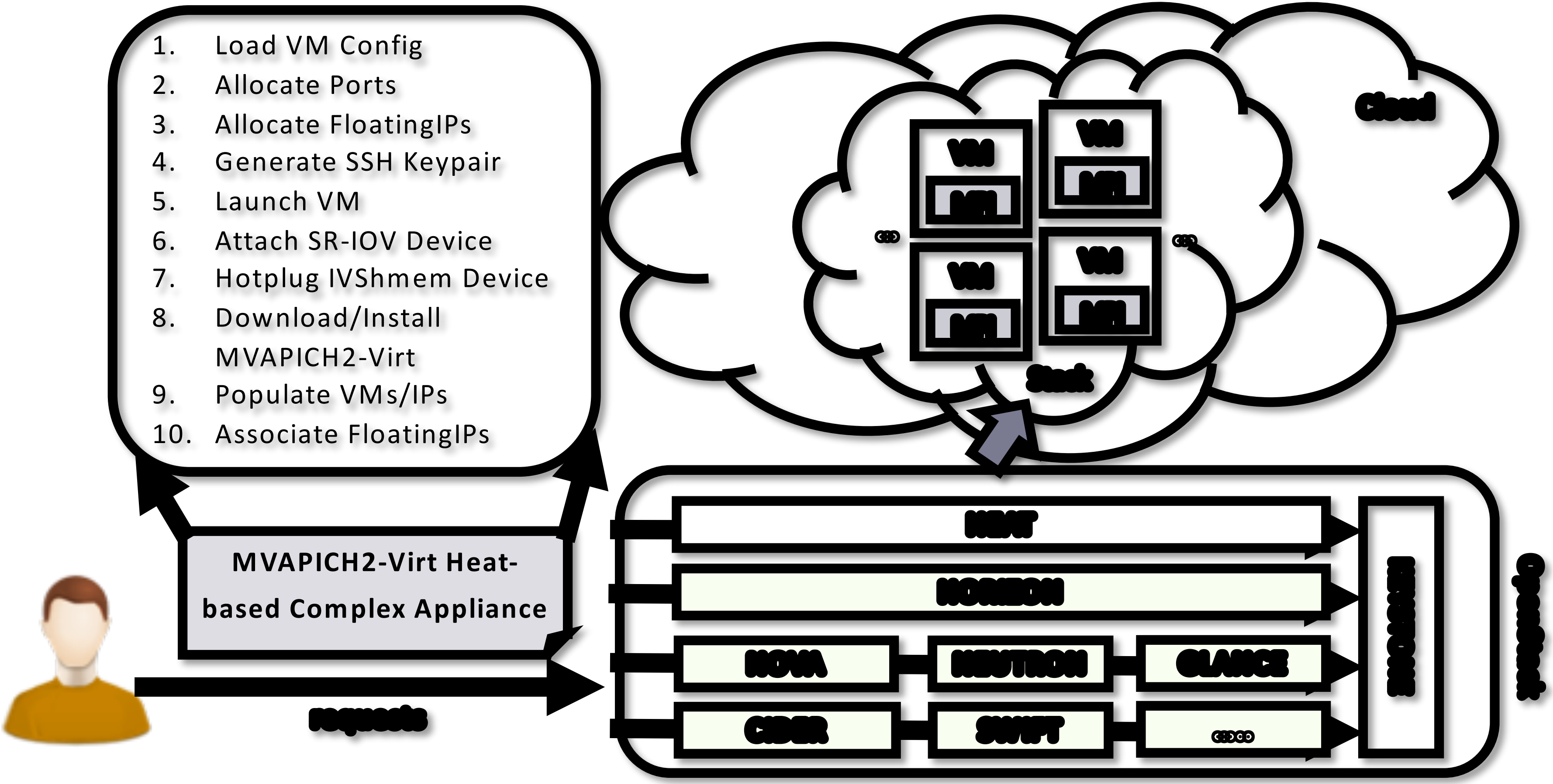


Appliance	Description
CentOS 7 KVM SR-IOV	Chameleon bare-metal image customized with the KVM hypervisor and a recompiled kernel to enable SR-IOV over InfiniBand. https://www.chameleoncloud.org/appliances/3/
MPI bare-metal cluster complex appliance (Based on Heat)	This appliance deploys an MPI cluster composed of bare metal instances using the MVAPICH2 library over InfiniBand. https://www.chameleoncloud.org/appliances/29/
MPI + SR-IOV KVM cluster (Based on Heat)	This appliance deploys an MPI cluster of KVM virtual machines using the MVAPICH2-Virt implementation and configured with SR-IOV for high-performance communication over InfiniBand. https://www.chameleoncloud.org/appliances/28/
CentOS 7 SR-IOV RDMA-Hadoop	The CentOS 7 SR-IOV RDMA-Hadoop appliance is built from the CentOS 7 appliance and additionally contains RDMA-Hadoop library with SR-IOV. https://www.chameleoncloud.org/appliances/17/

- Through these available appliances, users and researchers can easily deploy HPC clouds to perform experiments and run jobs with
  - High-Performance SR-IOV + InfiniBand
  - High-Performance MVAPICH2 Library over bare-metal InfiniBand clusters
  - High-Performance MVAPICH2 Library with Virtualization Support over SR-IOV enabled KVM clusters
  - High-Performance Hadoop with RDMA-based Enhancements Support

[*] Only include appliances contributed by OSU NowLab

MPI Complex Appliances based on MVAPICH2 on Chameleon



Demos on Chameleon Cloud

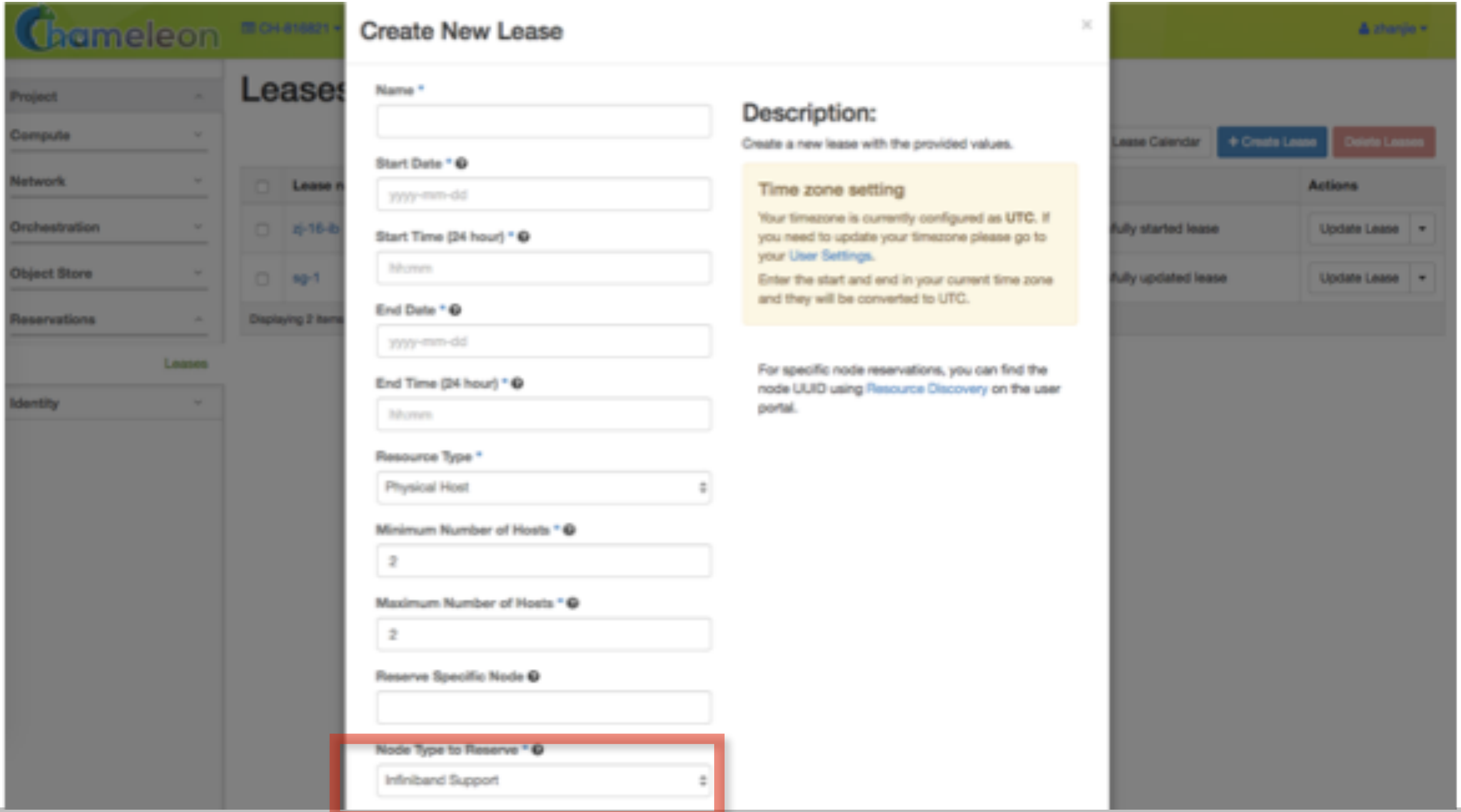
- A Demo of Deploying MPI Bare-Metal Cluster with InfiniBand
- A Demo of Deploying MPI KVM Cluster with SR-IOV enabled InfiniBand
- Running MPI Programs on Chameleon

Login to Chameleon Cloud

https://chi.tacc.chameleoncloud.org/dashboard/auth/login/

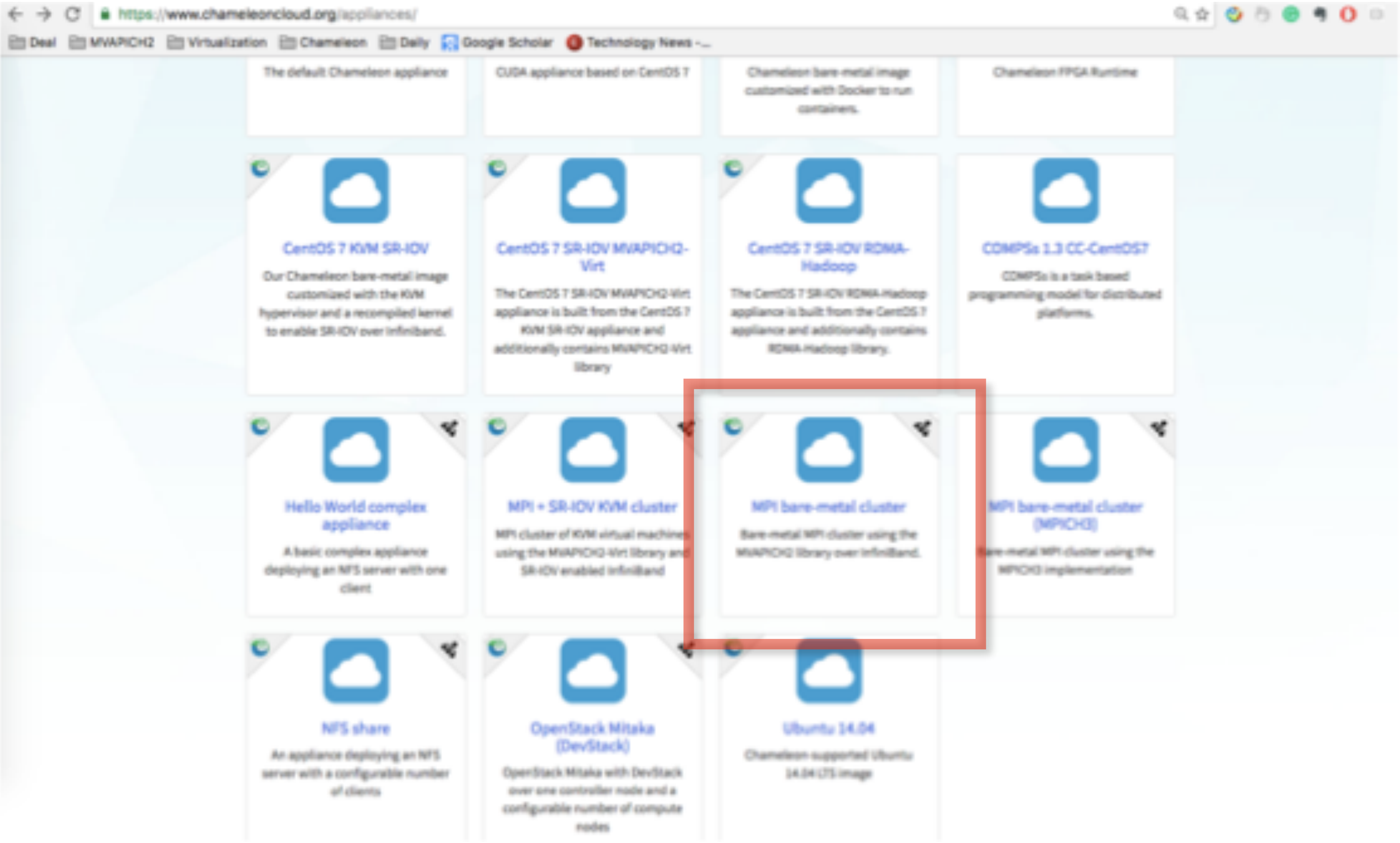
Create a Lease

Create a Lease

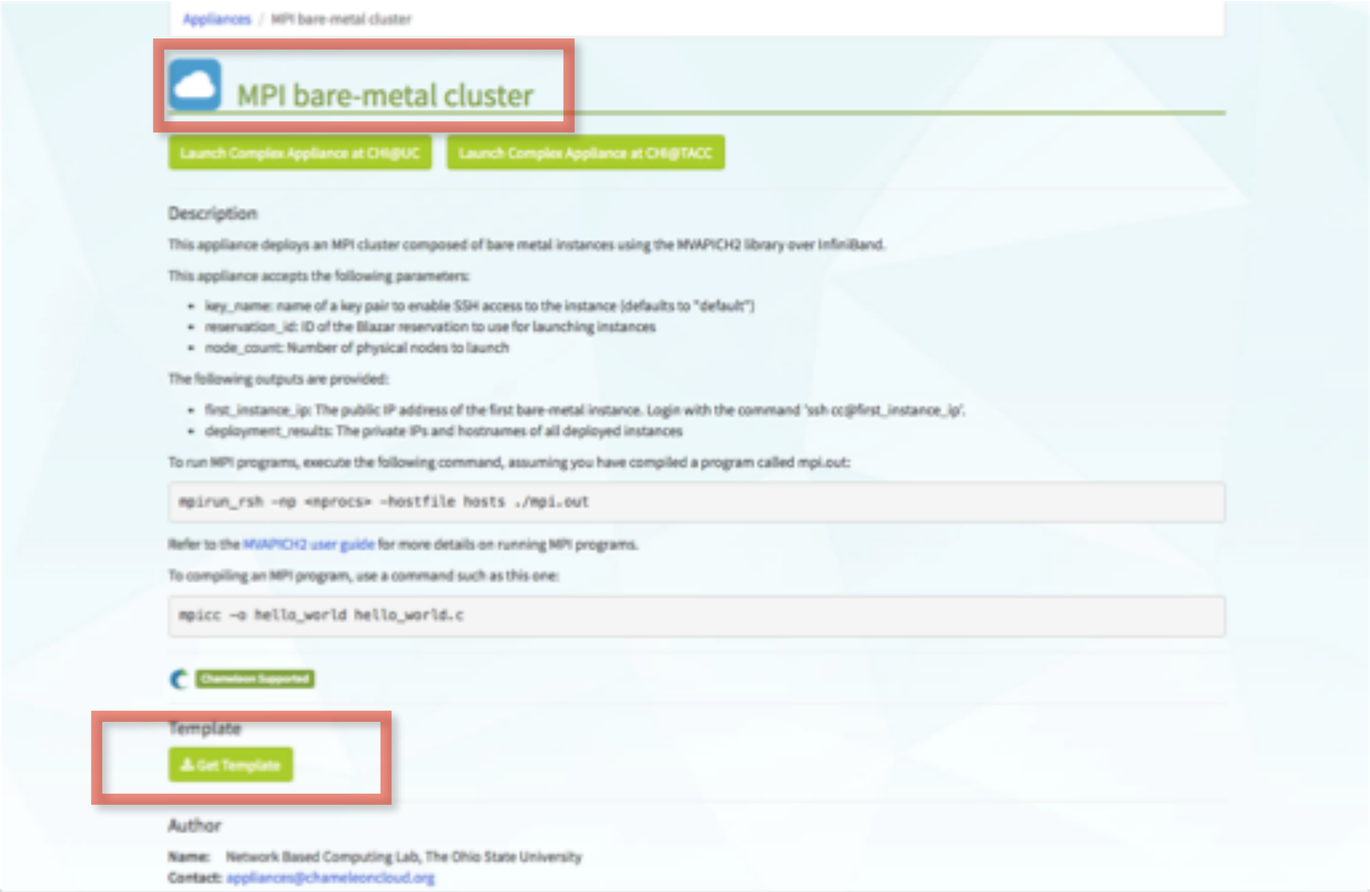


Lease Starts

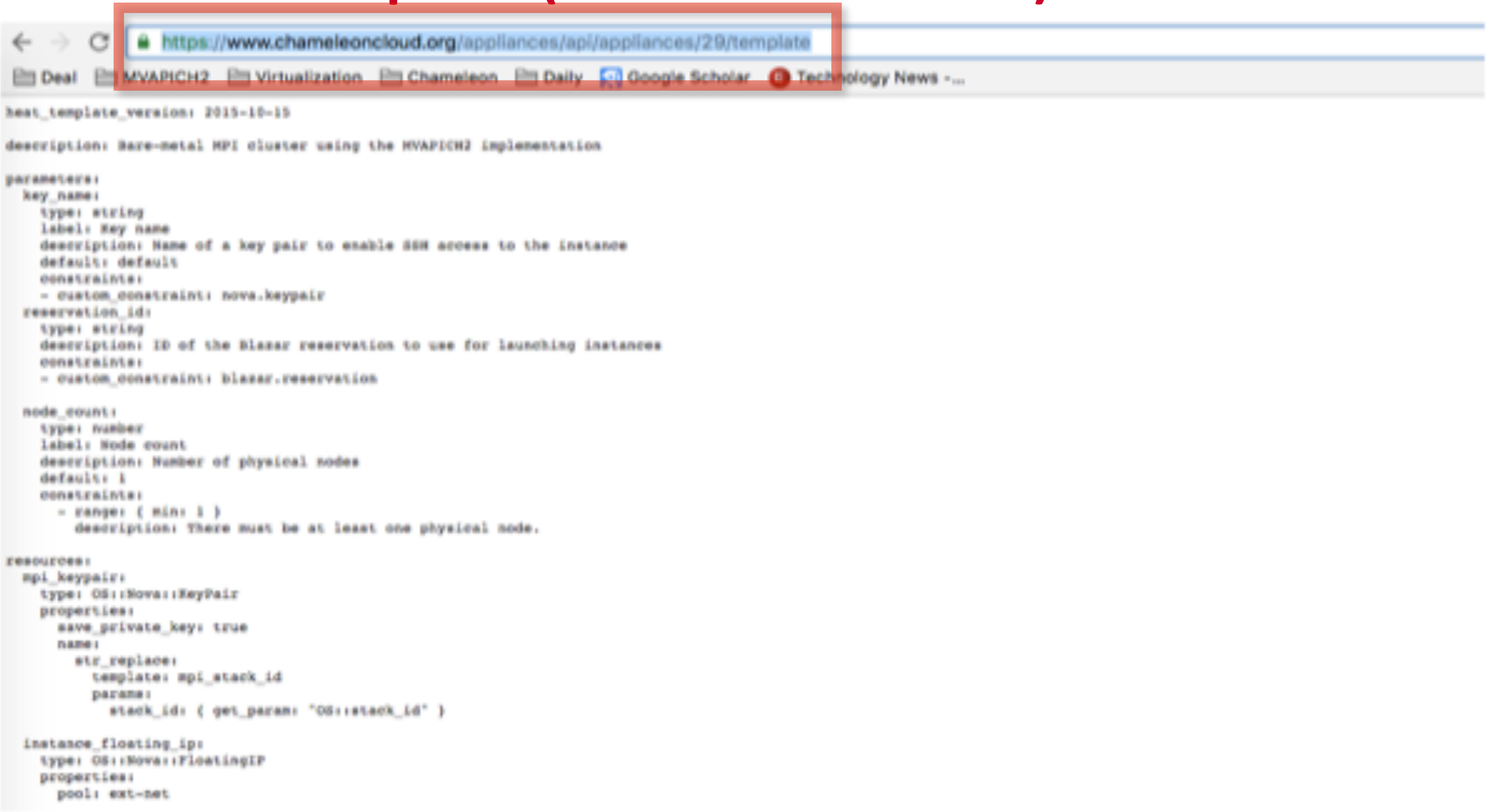
Select Complex Appliance - MPI Bare-Metal Cluster



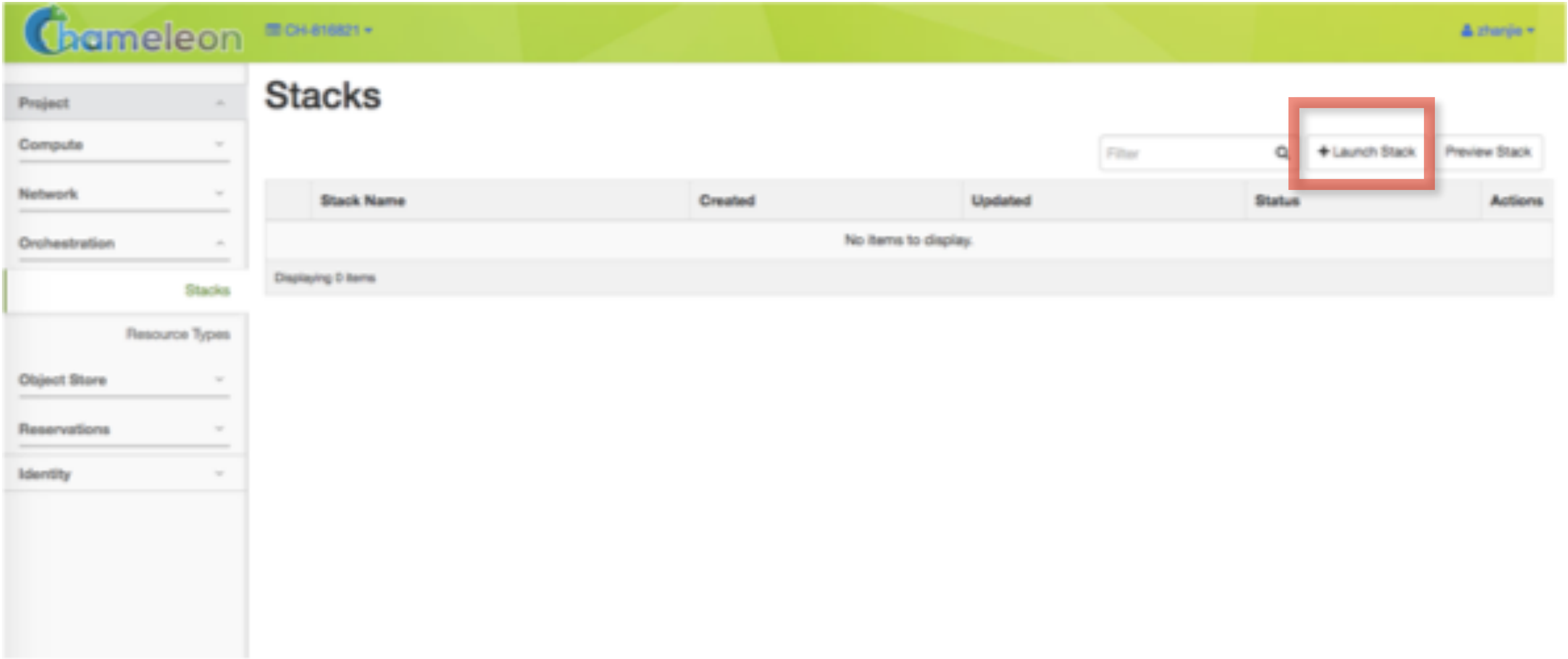
Get Template of MPI Bare-Metal Appliance



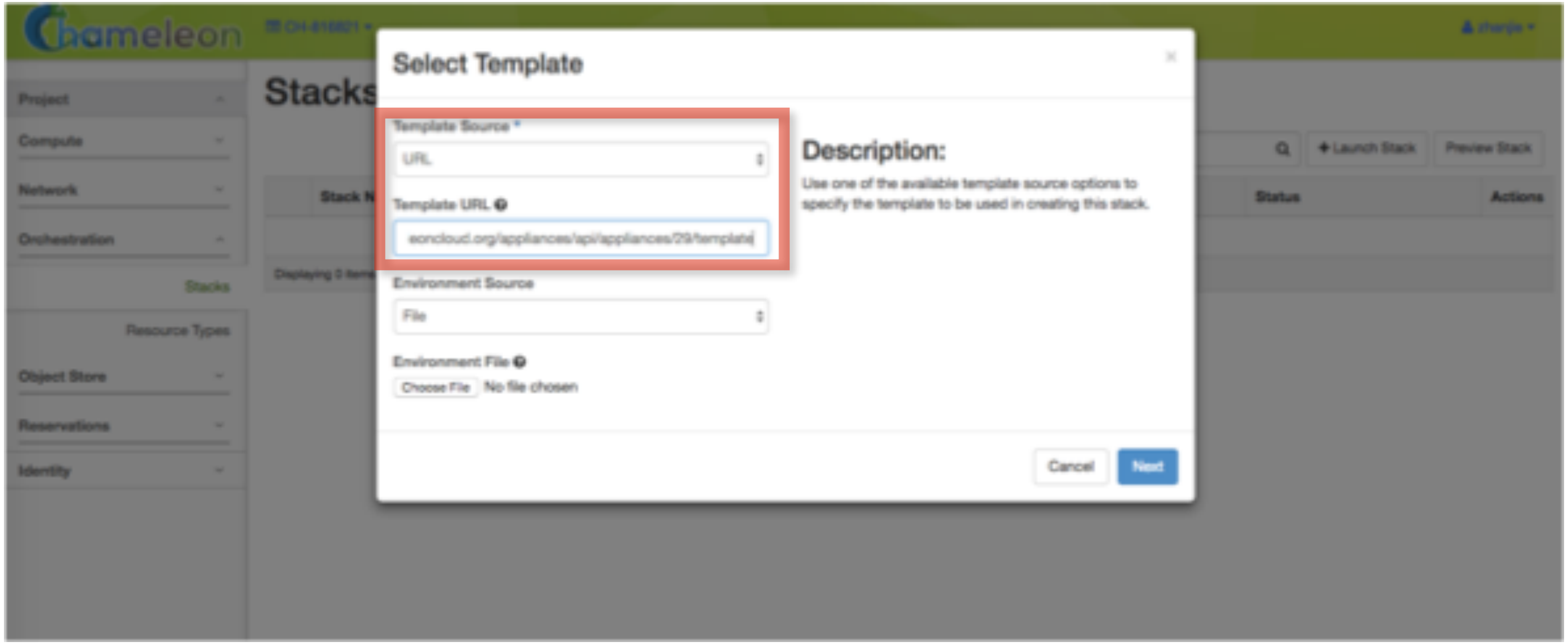
Save URL of Template (Will be used later)



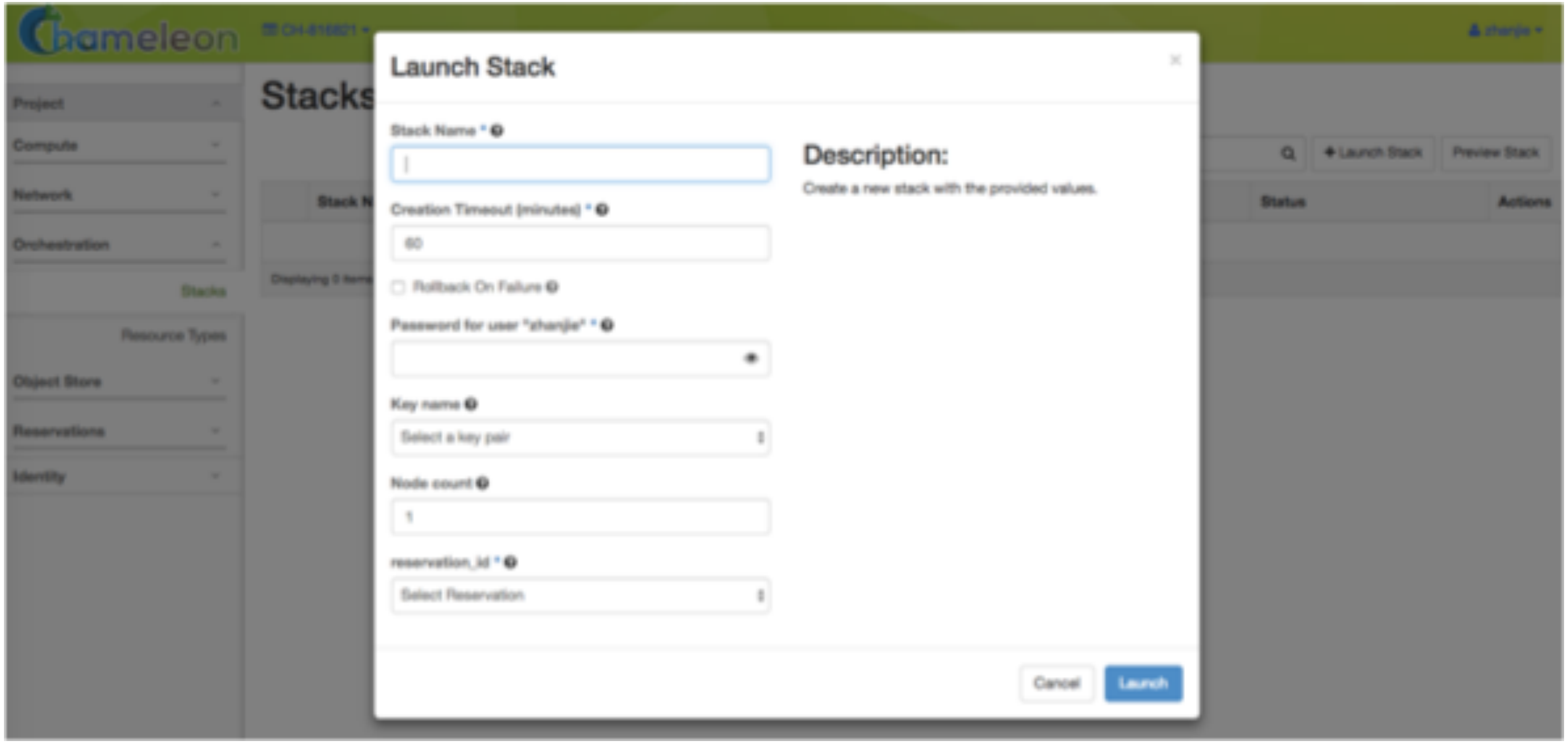
Launch Stack



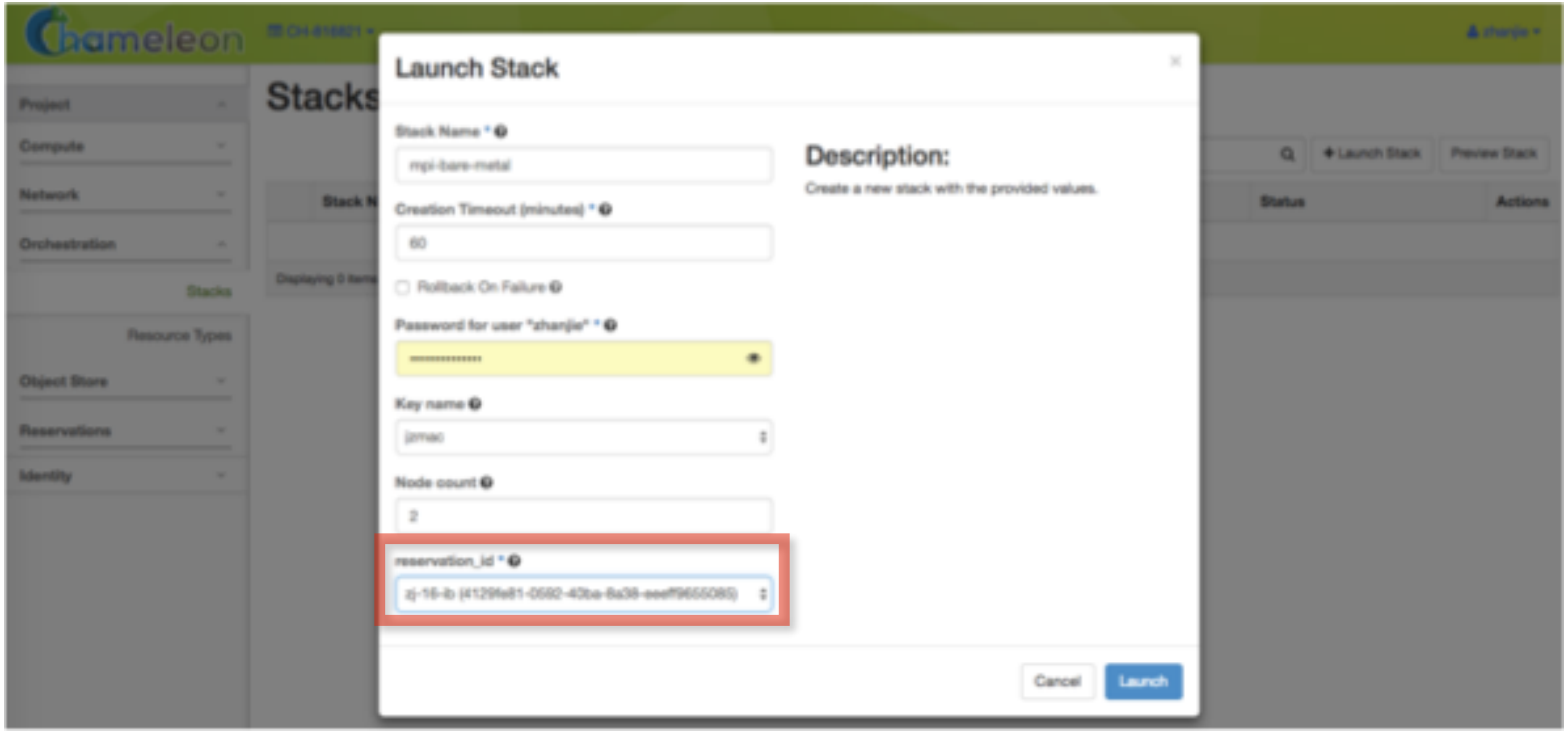
Use Saved Template URL as Source



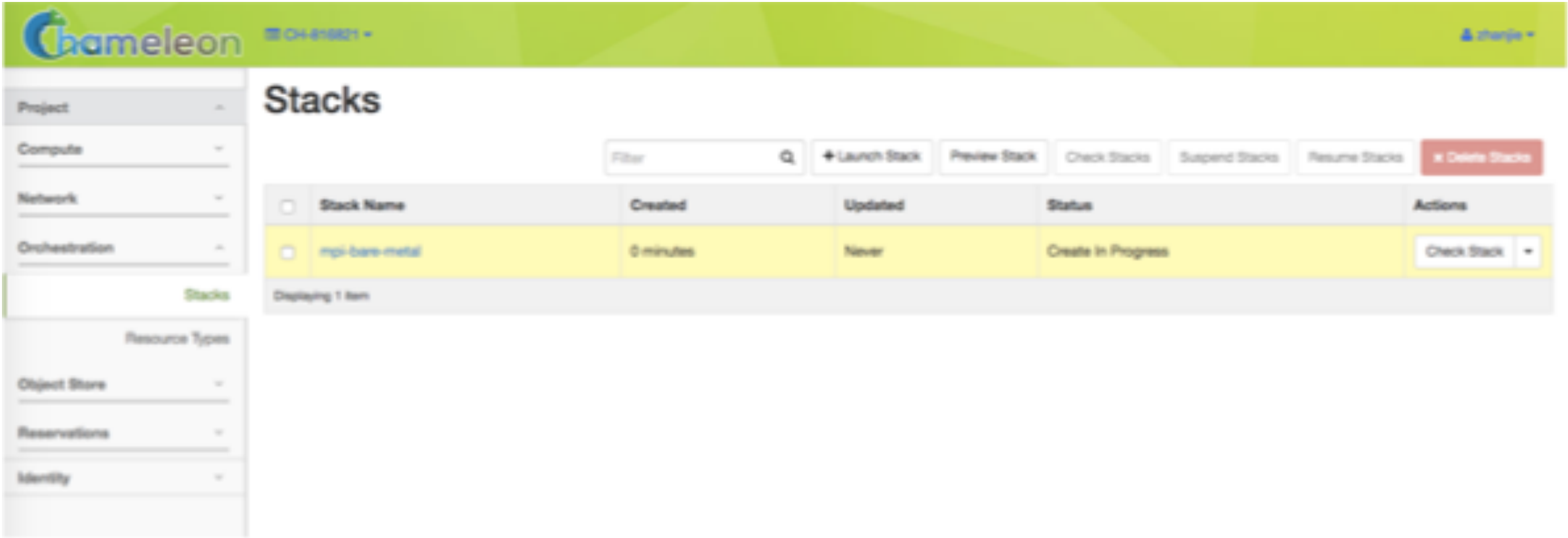
Input Stack Information



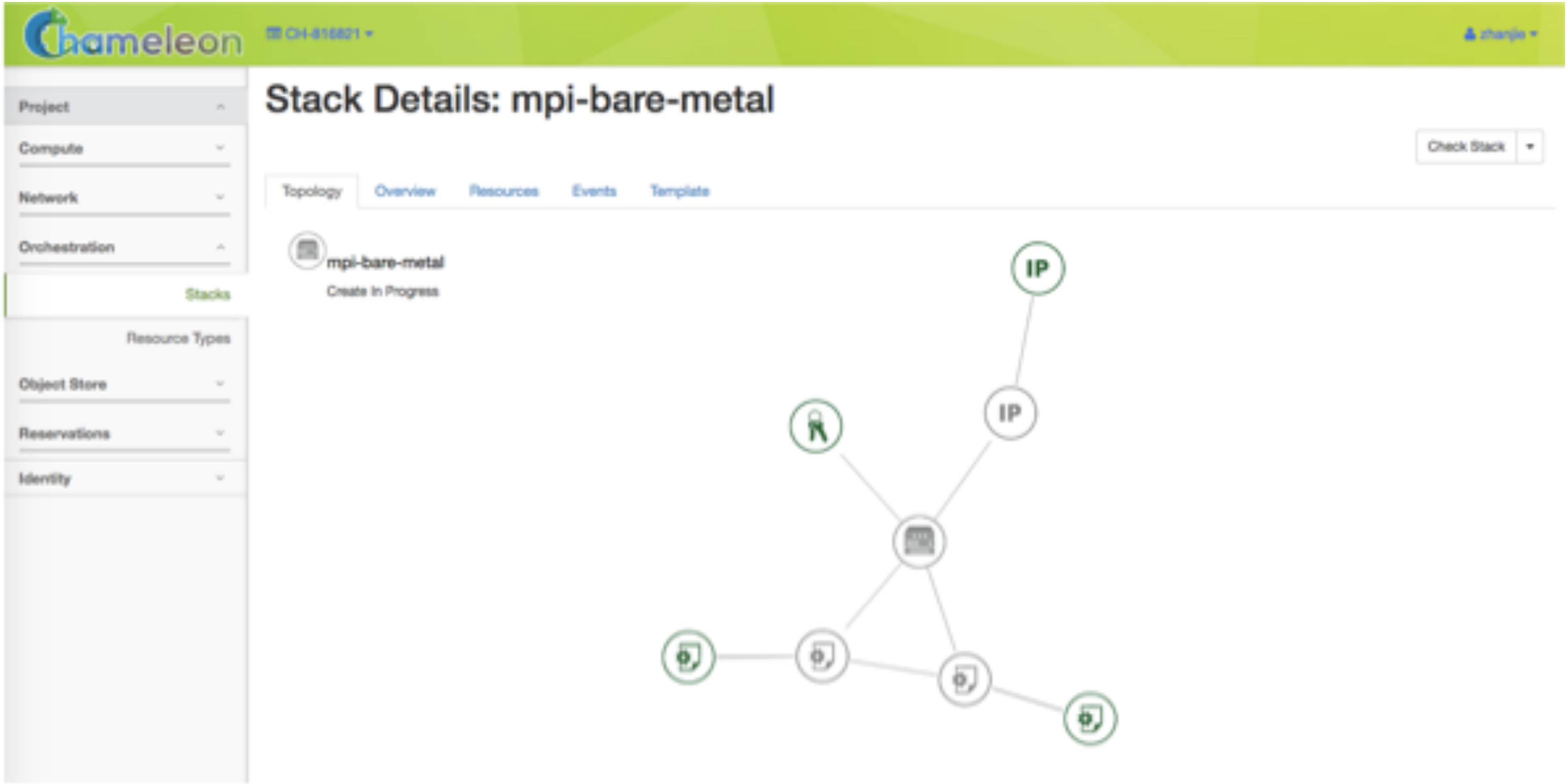
Use Created Lease



Stack Creation In Progress



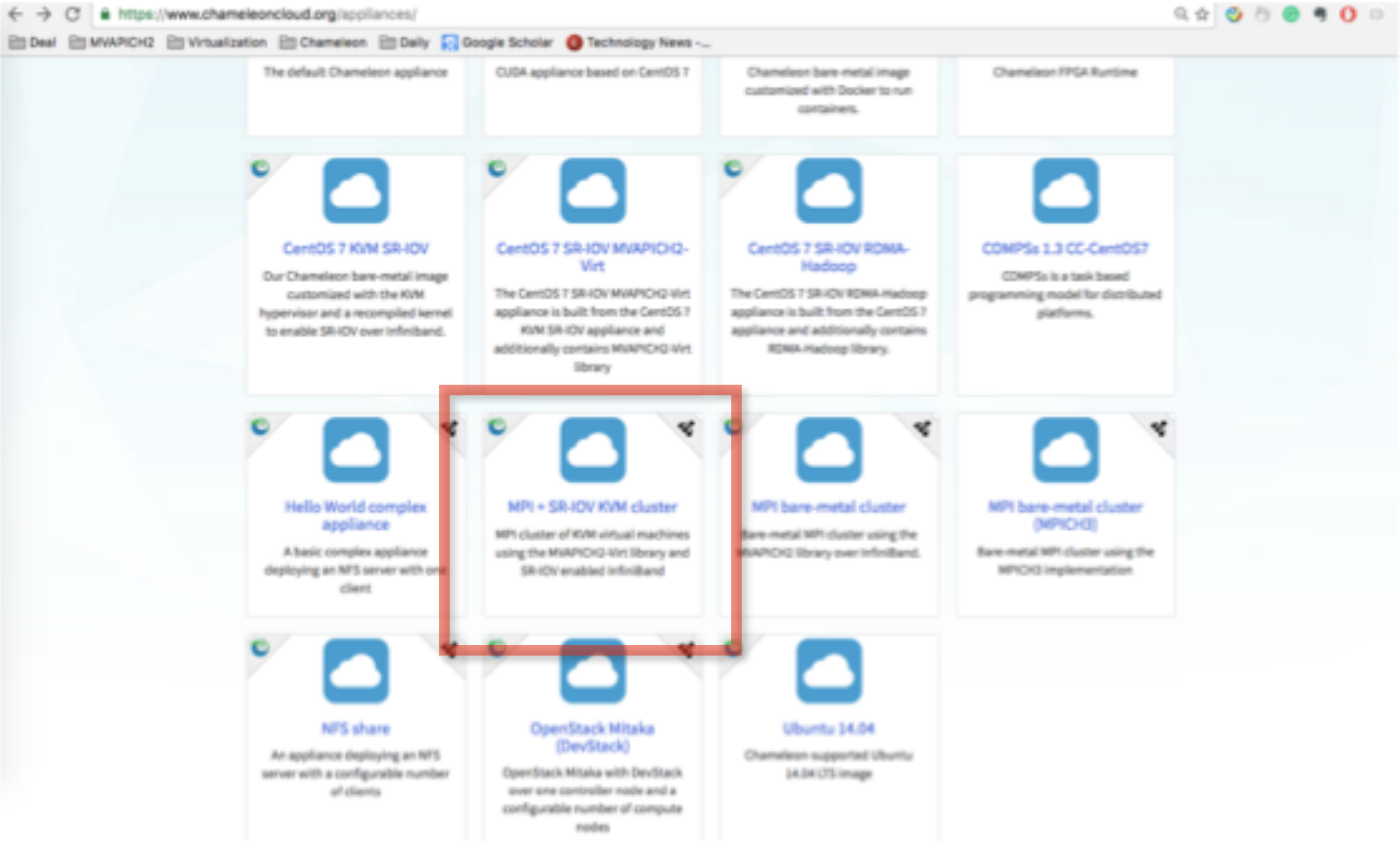
Stack Details



Demos on Chameleon Cloud

- A Demo of Deploying MPI Bare-Metal Cluster with InfiniBand
- A Demo of Deploying MPI KVM Cluster with SR-IOV enabled InfiniBand
- Running MPI Programs on Chameleon

Select Complex Appliance - MPI KVM with SR-IOV Cluster



Get Template of MPI KVM with SR-IOV Appliance

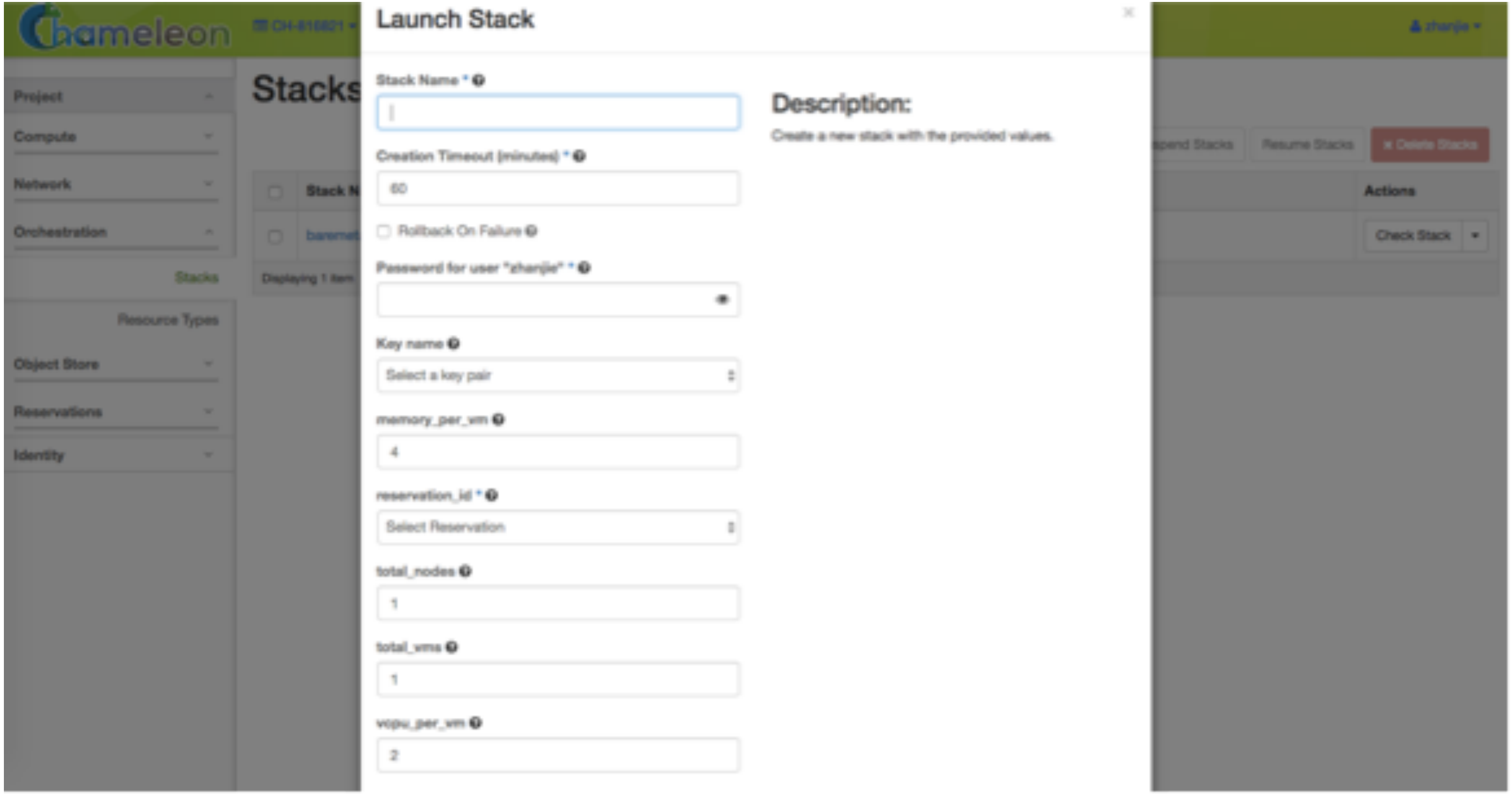
Save URL of Template (Will be used later)

https://www.chameleoncloud.org/appliances/api/appliances/28/template

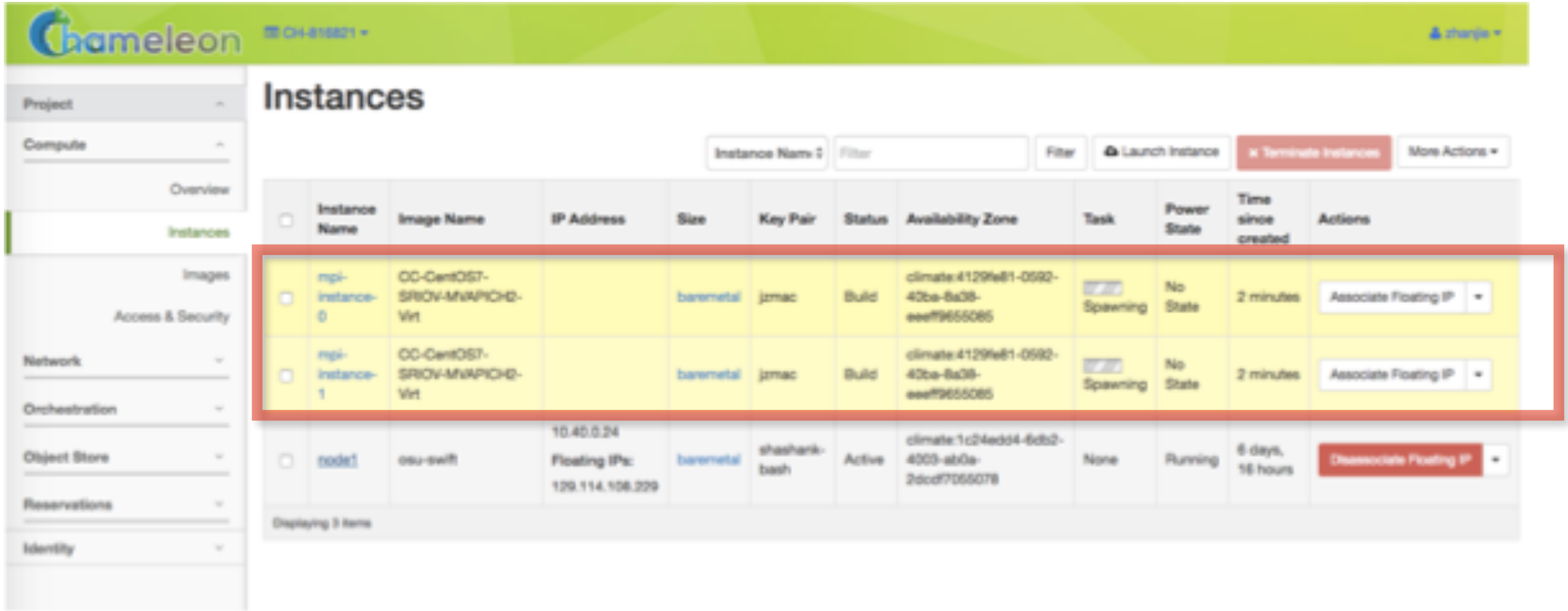
Deal | MVAPICH2 | Virtualization | Chameleon | Daily | Google Scholar | Technology News -...

```
heat_template_version: 2015-10-15

description: MPI cluster using KVM + SR-IOV and the MVAPICH2-Virt implementation

parameters:
  key_name:
    type: string
    label: Key name
    description: Name of a key pair to enable SSH access to the instance
    default: default
    constraints:
    - custom_constraint: nova.keypair
  reservation_id:
    type: string
    description: ID of the Blazar reservation to use for launching instances
    constraints:
    - custom_constraint: blazar.reservation

  total_nodes:
    type: number
    description: Number of physical nodes to launch
    default: 1
    constraints:
      - range: { min: 1 }
        description: There must be at least one physical node.
  total_vms:
    type: number
    description: Number of virtual machines to create
    default: 1
    constraints:
      - range: { min: 1 }
        description: There must be at least one virtual machine.
  vcpu_per_vm:
    type: number
    description: Number of VCPUs per virtual machine
    default: 2
    constraints:
      - range: { min: 1 }
        description: There must be at least one VCPU per virtual machine.
  memory_per_vm:
    type: number
    description: Amount of memory size per virtual machine (in GiB)
    default: 4
    constraints:
      - range: { min: 1 }
        description: There must be at least one VCPU per virtual machine.

resources:
```

Launch Stack



Instances in Stack (Spawning …)

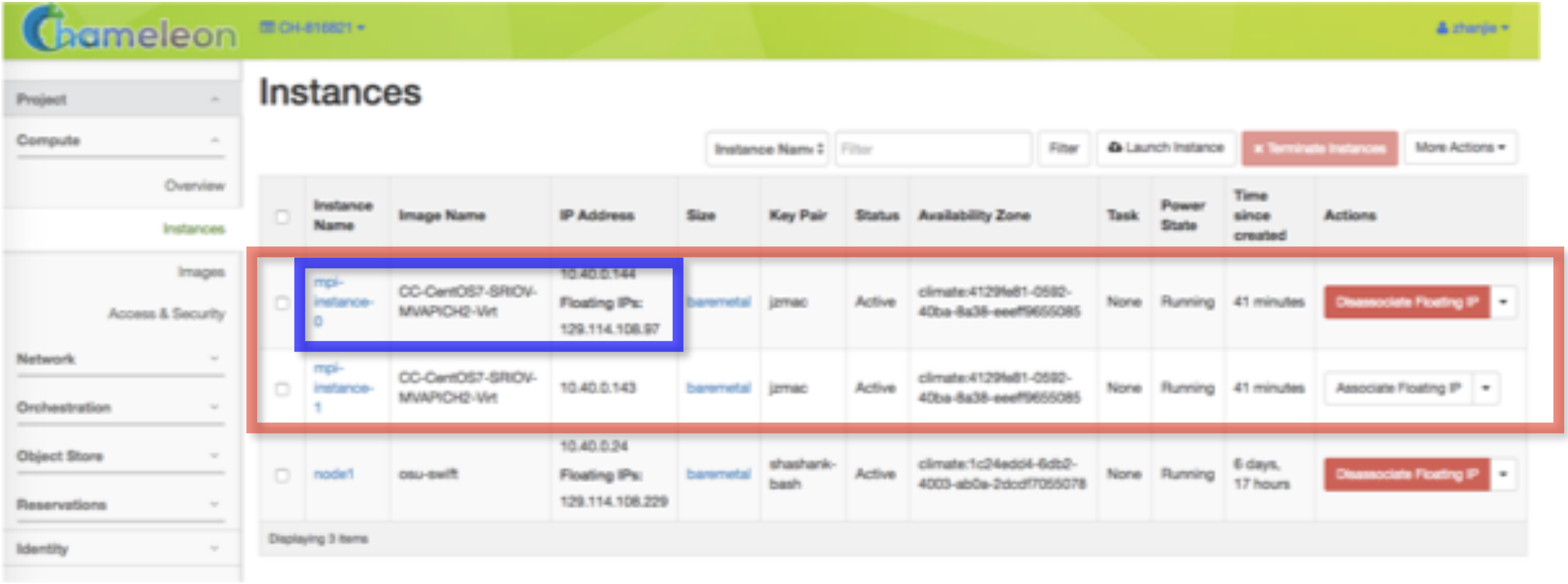


Demos on Chameleon Cloud

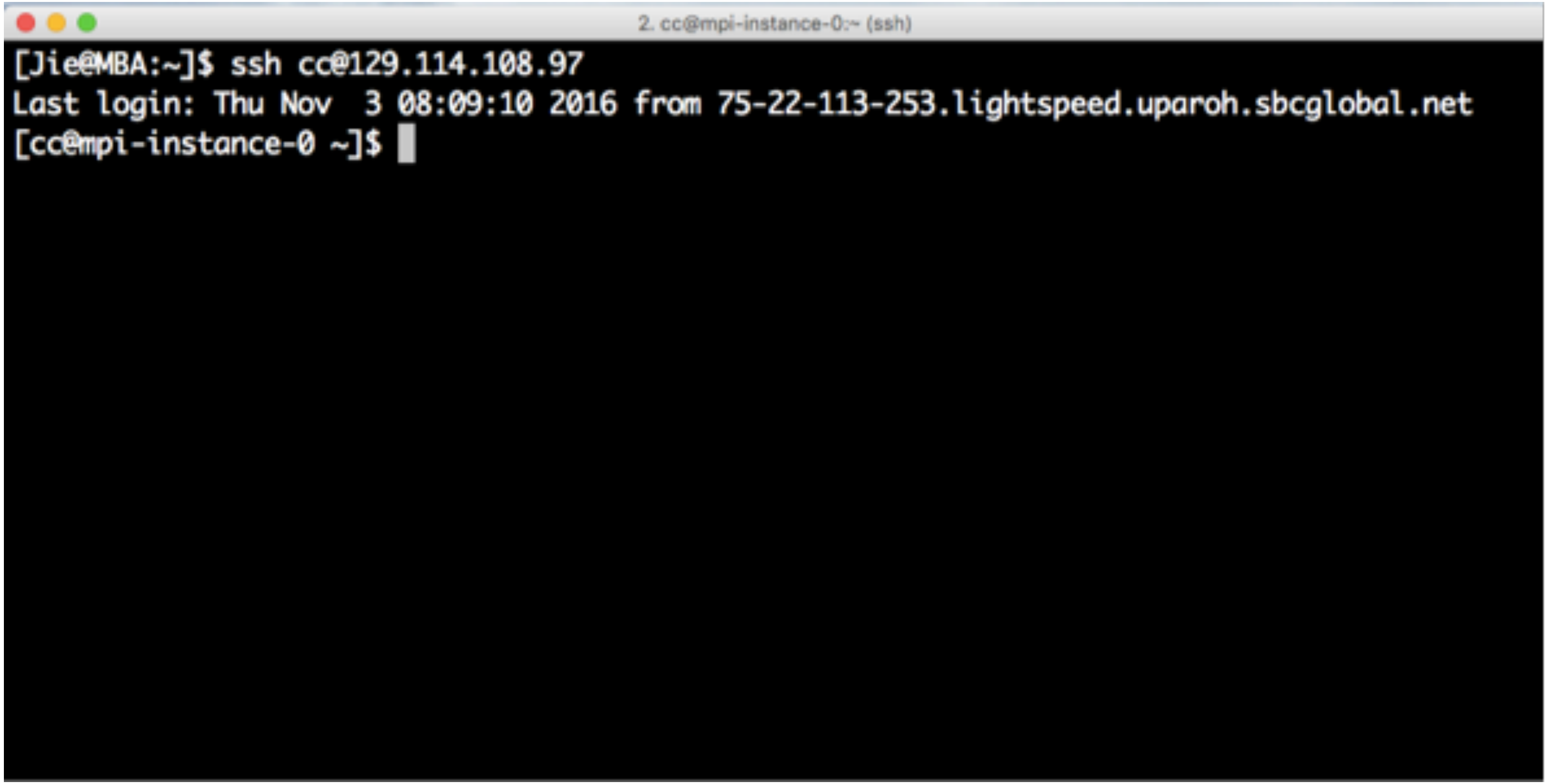
- A Demo of Deploying MPI Bare-Metal Cluster with InfiniBand
- A Demo of Deploying MPI KVM Cluster with SR-IOV enabled InfiniBand
- Running MPI Programs on Chameleon

Create Stack Successfully

Login to the first instance with Floating IP



Login Instance with Floating IP



SSH to Other Instances

2. cc@mpi-instance-1:~ (ssh)

```
[Jie@MBA:~]$ ssh cc@129.114.108.97
Last login: Thu Nov  3 08:09:10 2016 from 75-22-113-253.lightspeed.uparoh.sbcglobal.net
[cc@mpi-instance-0 ~]$ cat /etc/hosts
127.0.0.1   localhost localhost.localdomain localhost4 localhost4.localdomain4
::1         localhost localhost.localdomain localhost6 localhost6.localdomain6
10.40.0.143 mpi-instance-1
10.40.0.144 mpi-instance-0
[cc@mpi-instance-0 ~]$ ssh mpi-instance-1
Last login: Thu Nov  3 08:03:47 2016 from mpi-instance-0
[cc@mpi-instance-1 ~]$
```

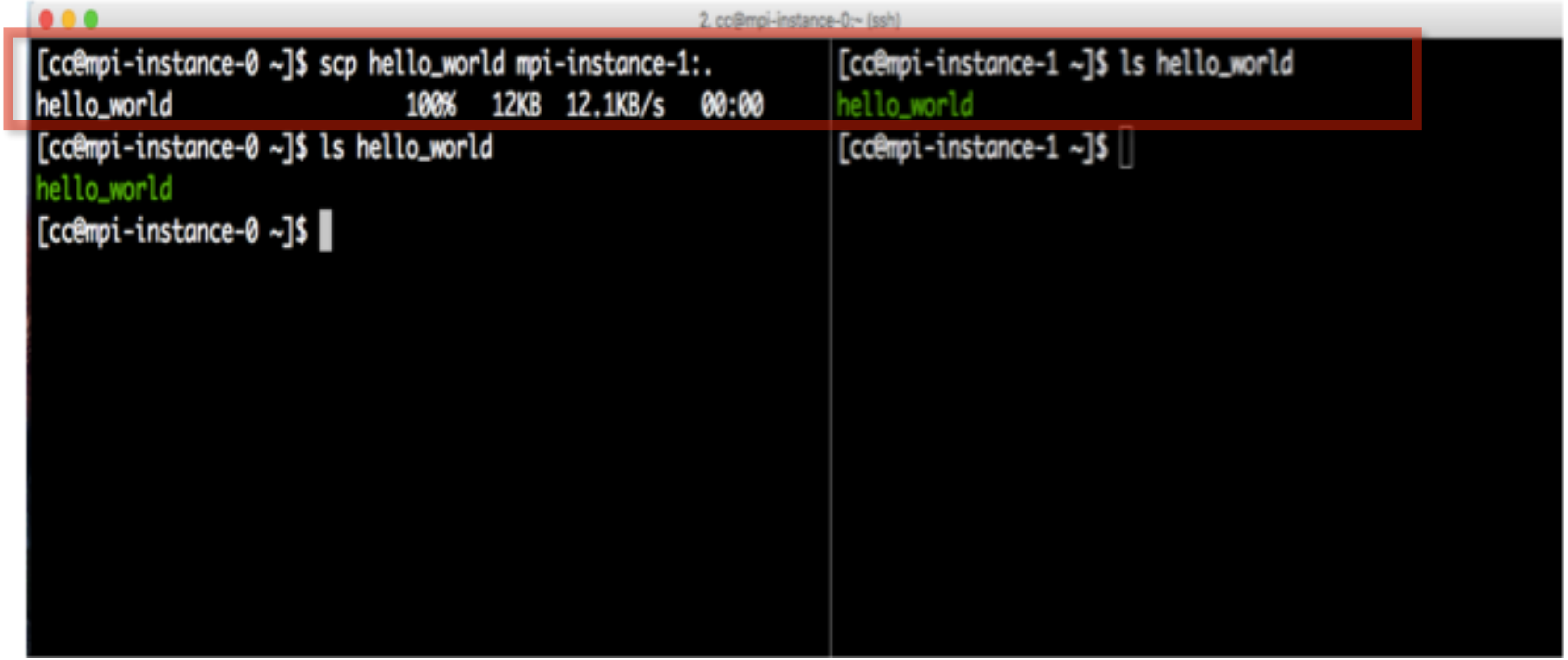
Compile MPI Program – Hello World

2. cc@mpi-instance-0:~ (ssh)

```c
#include <stdio.h>
#include <mpi.h>


int main (int argc, char** argv)
{
    int rank, size;

    MPI_Init (&argc, &argv);
    MPI_Comm_rank (MPI_COMM_WORLD, &rank);
    MPI_Comm_size (MPI_COMM_WORLD, &size);
    printf( "Hello world from process %d of %d\n", rank, size );
    MPI_Finalize();

    return 0;
}
~
~
~
~
```

```
[cc@mpi-instance-0 ~]$ mpicc -o hello_world hello_world.c
[cc@mpi-instance-0 ~]$ ls hello_world
hello_world
[cc@mpi-instance-0 ~]$
```

Distribute Executable to Other Instances



Run MPI Program – Hello World

2. cc@mpi-instance-0:~ (ssh)

```
[cc@mpi-instance-0 ~]$ mpirun_rsh -np 2 mpi-instance-0 mpi-instance-1 hello_world
Hello world from process 0 of 2
Hello world from process 1 of 2
[cc@mpi-instance-0 ~]$
```

Overview of OSU MPI Micro-Benchmarks (OMB) Suite

- A comprehensive suite of benchmarks to
  - Compare performance of different MPI libraries on various networks and systems
  - Validate low-level functionalities
  - Provide insights to the underlying MPI-level designs
- Started with basic send-recv (MPI-1) micro-benchmarks for latency, bandwidth and bi-directional bandwidth
- Extended later to
  - MPI one-sided
  - Collectives
  - GPU-aware data movement
  - OpenSHMEM (point-to-point and collectives)
  - UPC
- Has become an industry standard
- Extensively used for design/development of MPI libraries, performance comparison of MPI libraries and even in procurement of large-scale systems
- Available from http://mvapich.cse.ohio-state.edu/benchmarks
- Available in an integrated manner with MVAPICH2 stack

Examples of OMB Benchmarks

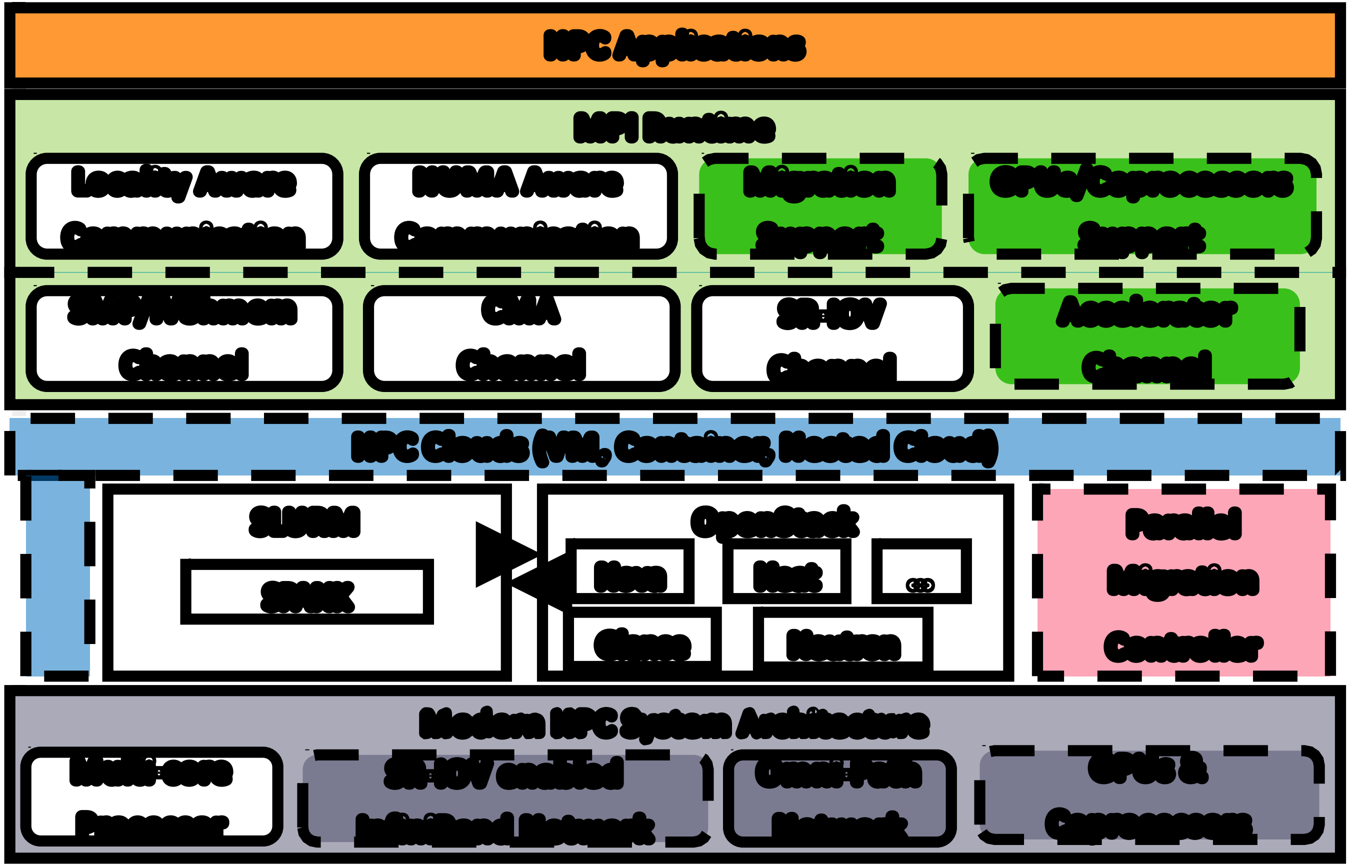
MPI Two-sided Pt2Pt	MPI One-sided Pt2Pt	MPI Collectives (64 Procs = 4 nodes * 16 Procs/node)
MPI_Latency (Inter-node, Intra-node)	MPI_Put_Latency	MPI_Bcast
MPI_BW (Inter-node, Intra-node)		MPI_Alltoall

Run OSU MPI Benchmarks – Latency and Bandwidth

```
[cc@mpi-instance-0 ~]$ mpirun_rsh -np 2 -hostfile hosts /opt/mvapich2/l
ibexec/osu-micro-benchmarks/mpi/pt2pt/osu_latency
# OSU MPI Latency Test v5.3
# Size          Latency (us)
0                       1.26
1                       1.29
2                       1.30
4                       1.31
8                       1.30
16                      1.32
32                      1.34
64                      1.38
128                     1.47
256                     1.92
512                     2.03
1024                    2.35
2048                    2.86
4096                    3.33
8192                    4.76
16384                   7.02
32768                  10.71
65536                  16.04
131072                 26.50
262144                 47.28
524288                 89.24
1048576               172.01
2097152               340.85
4194304               677.87
[cc@mpi-instance-0 ~]$
```

```
[cc@mpi-instance-0 ~]$ mpirun_rsh -np 2 -hostfile hosts /opt/mvapich2/
libexec/osu-micro-benchmarks/mpi/pt2pt/osu_bw
# OSU MPI Bandwidth Test v5.3
# Size      Bandwidth (MB/s)
1                       3.50
2                       7.12
4                      14.59
8                      28.68
16                     57.70
32                    113.70
64                    225.90
128                   439.72
256                   844.32
512                  1628.04
1024                 2938.93
2048                 4602.78
4096                 5481.66
8192                 5780.06
16384                5905.00
32768                5912.54
65536                6150.79
131072               6233.94
262144               6298.78
524288               6319.60
1048576              6321.27
2097152              6268.38
4194304              6264.26
[cc@mpi-instance-0 ~]$
```

Next Steps of MVAPICH2-Virt



Conclusions

- MVAPICH2-Virt over SR-IOV-enabled InfiniBand is an efficient approach to build HPC Clouds
  - Standalone, OpenStack, Slurm, and Slurm + OpenStack
  - Support Virtual Machine Migration with SR-IOV InfiniBand devices
  - Support Virtual Machine, Container (Docker and Singularity), and Nested Virtualization
- Very little overhead with virtualization, near native performance at application level
- Much better performance than Amazon EC2
- **MVAPICH2-Virt** is available for building HPC Clouds
  - SR-IOV, IVSHMEM, Docker and Singularity, OpenStack
- Appliances for MVAPICH2-Virt are available for building HPC Clouds
- Demos on NSF Chameleon Cloud
- Future releases for supporting running MPI jobs in VMs/Containers with SLURM, etc.
- SR-IOV/container support and appliances for other MVAPICH2 libraries (MVAPICH2-X, MVAPICH2-GDR, ...)

Thank You!

luxi@cse.ohio-state.edu

http://www.cse.ohio-state.edu/~luxi

Network-Based Computing Laboratory

http://nowlab.cse.ohio-state.edu/

MVAPICH: MPI over InfiniBand, Omni-Path, Ethernet/iWARP, and RoCE

http://mvapich.cse.ohio-state.edu/