A Case for High Performance Computing with Virtual Machines

Wei Huang*, Jiuxing Liu+, Bulent Abali+, and Dhabaleswar K. Panda*

***The Ohio State University**

+IBM T. J. Waston Research Center

ICS'06 -- June 28th, 2006

Presentation Outline

- Virtual Machine environment and HPC
- Background -- VMM-bypass I/O
- A framework for HPC with virtual machines
- A prototype implementation
- Performance evaluation
- Conclusion

What is Virtual Machine Environment?

- A Virtual Machine environment provides virtualized hardware interface to VMs through Virtual Machine Monitor (VMM)
- A physical node may host several VMs, with each running separate OSes
- Benefits: ease of management, performance isolation, system security, checkpoint/restart, live migration …

Why HPC with Virtual Machines?

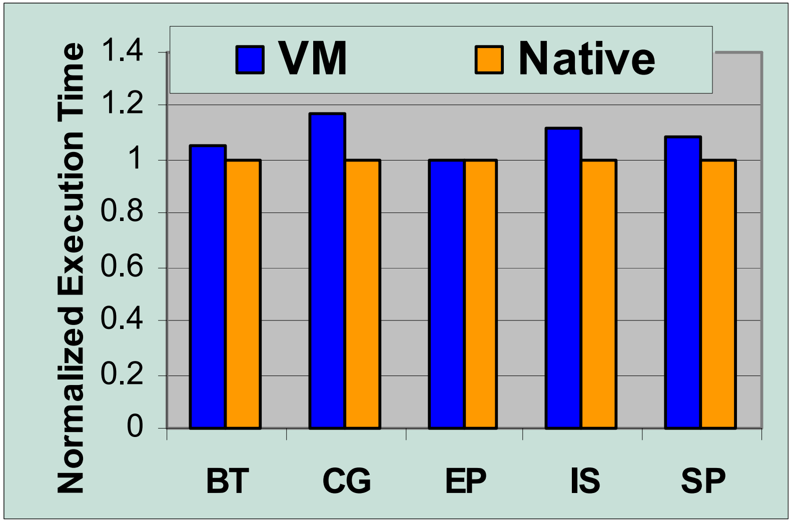
- Ease of management
- Customized OS
  - Light-weight OSes customized for applications can potentially gain performance benefits [FastOS]
  - No widely adoption due to management difficulties
  - VM makes it possible
- System security

[FastOS]: Forum to Address Scalable Technology for Runtime and Operating Systems

Why HPC with Virtual Machines?

- Ease of management
- Customized OS
- System security
  - Currently, most HPC environment disallow users to performance privileged operations (e.g. loading customized kernel modules)
  - Limit productivities and convenience
  - Users can do ‘anything’ in VM, in the worst case crash an VM, not the whole system

But Performance?



	Dom0	VMM	DomU
CG	16.6%	10.7%	72.7%
IS	18.1%	13.1%	68.8%
EP	00.6%	00.3%	99.0%
BT	06.1%	04.0%	89.9%
SP	09.7%	06.5%	83.8%

- NAS Parallel Benchmarks (MPICH over TCP) in Xen VM environment
  - Communication intensive benchmarks show bad results
- Time Profiling using Xenoprof
  - Many CPU cycles are spent in VMM and the device domain to process network IO requests

Challenges

- I/O virtualization overhead
- A framework to virtualize the cluster environment
  - Jobs require multiple processes distributed across multiple physical nodes
  - Typically requires all nodes have the same setup
  - How to allow customized OS?
  - How to reduce other virtualization overheads (memory, storage, etc …)
  - How to reconfigure nodes and start jobs efficiently?

Challenges

- I/O virtualization overhead [USENIX '06]
- A framework to virtualize the cluster environment
  - Jobs requires multiple processes distributed across multiple physical nodes
  - Typically requires all nodes have the same setup
  - How to allow customized OS?
  - How to reduce other virtualization overheads (memory, storage, etc …)
  - How to reconfigure nodes and start jobs efficiently?

[USENIX '06]: J. Liu, W. Huang, B. Abali, D. K. Panda. High Performance VMM-bypass I/O in Virtual Machines

Challenges

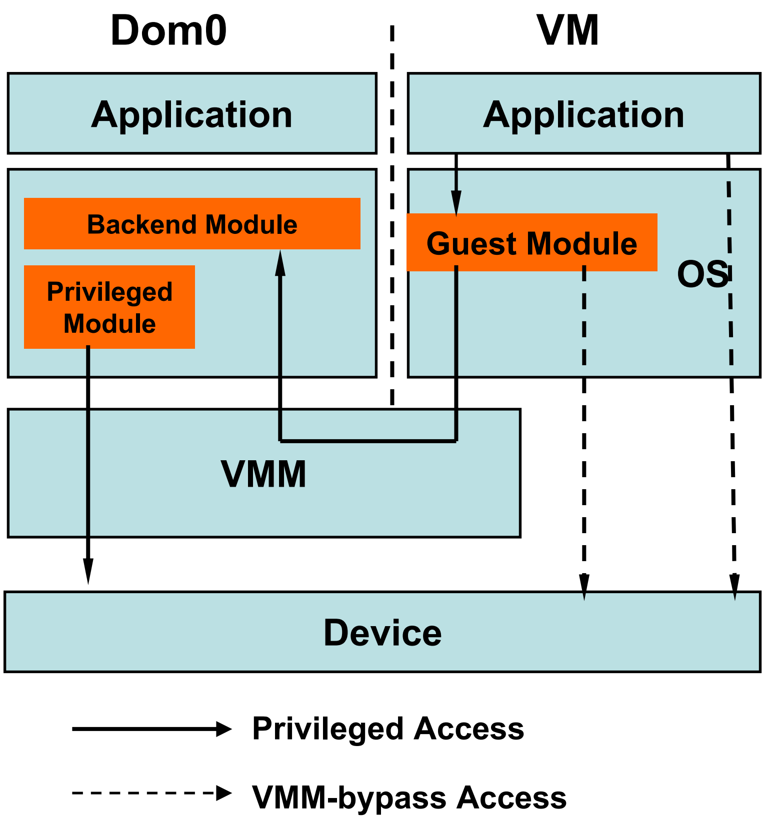
- I/O virtualization overhead [USENIX '06]
  - Evaluation of VMM-bypass I/O with HPC benchmarks
- A framework to virtualize the cluster environment
  - Jobs requires multiple processes distributed across multiple physical nodes
  - Typically requires all nodes have the same setup
  - How to allow customized OS?
  - How to reduce other virtualization overheads (memory, storage, etc …)
  - How to reconfigure nodes and start jobs efficiently?

[USENIX '06]: J. Liu, W. Huang, B. Abali, D. K. Panda. High Performance VMM-bypass I/O in Virtual Machines

Presentation Outline

- Virtual Machines and HPC
- Background -- VMM-bypass I/O
- A framework for HPC with virtual machines
- A prototype implementation
- Performance evaluation
- Conclusion

VMM-Bypass I/O

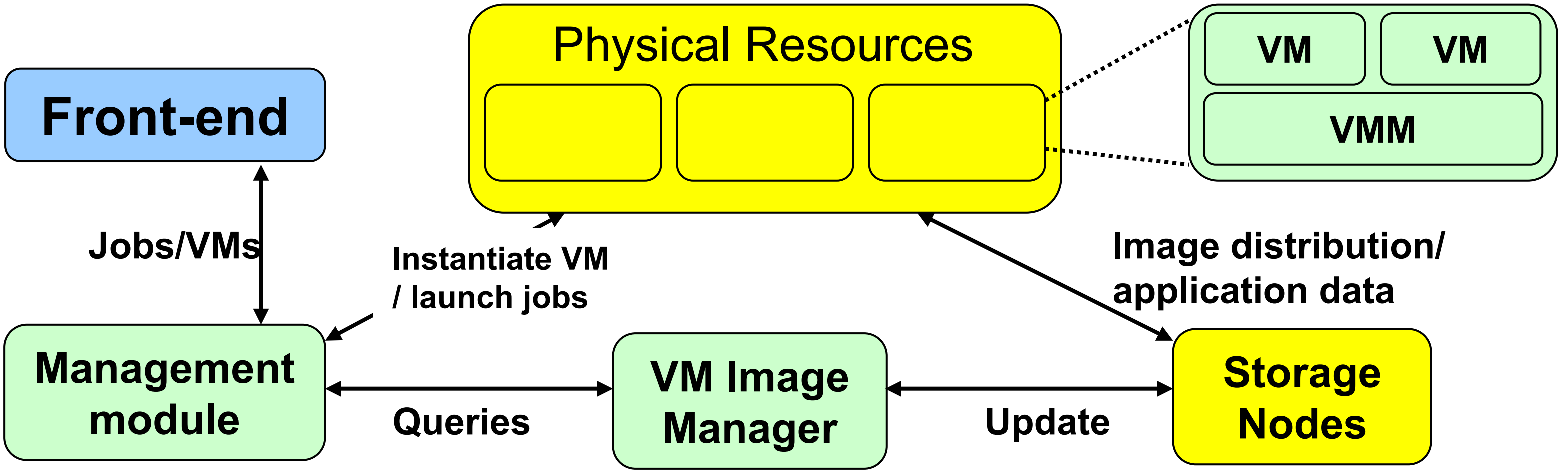


- **Original Scheme:** Guest module contact with privileged domain to complete I/O
  - Packets are sent to backend module, which are sent out through the privileged module (e.g. drivers)
  - Extra communication, domain switch, is very costly
- **VMM-Bypass I/O:** Guest modules in guest VMs handle setup and management operations (privileged access).
  - Once things are setup properly, devices can be accessed directly from guest VMs (VMM-bypass access).
  - Requires the device to have OS-bypass feature, e.g. InfiniBand
  - Can achieve native level performance

Presentation Outline

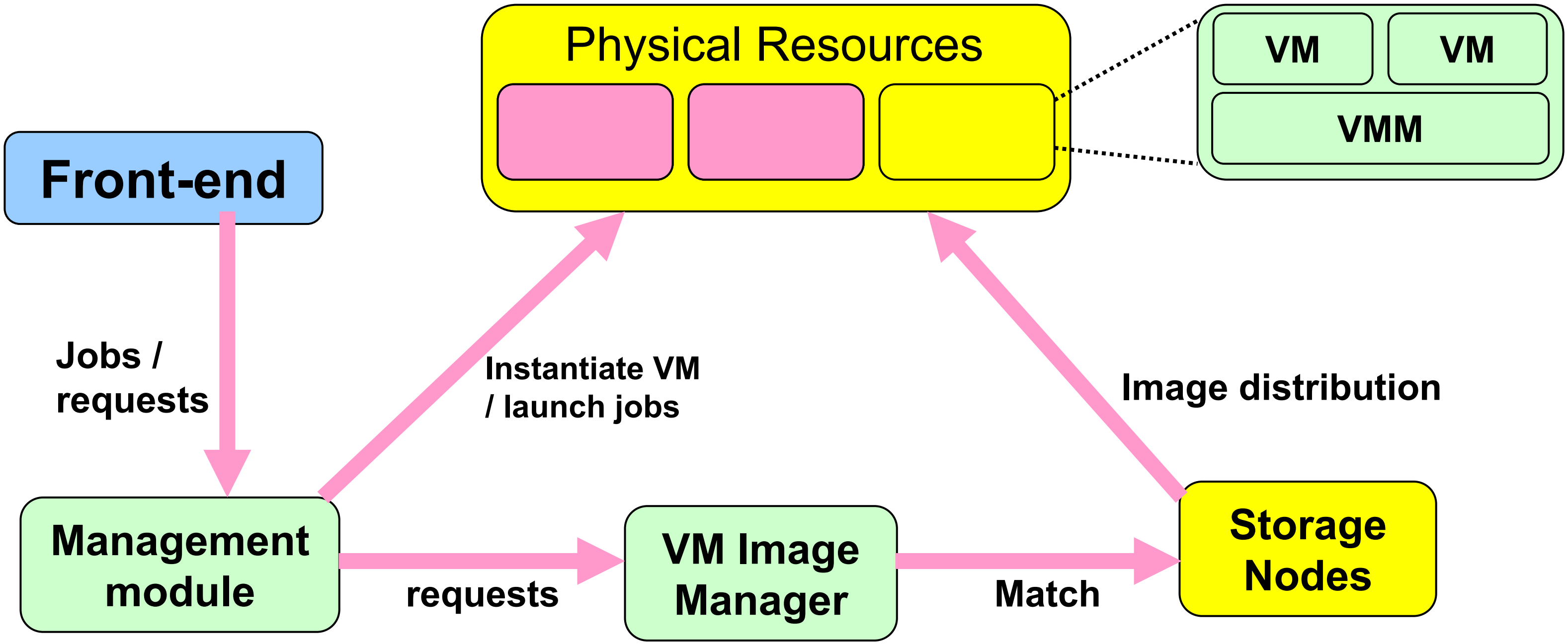
- Virtual Machines and HPC
- Background -- VMM-bypass I/O
- A framework for HPC with virtual machines
- A prototype implementation
- Performance evaluation
- Conclusion

Framework for VM-based Computing



- Physical Nodes: each running VM environment
  - typically no more VM instances than number of physical CPUs
  - Customized OS is achived through different versions images used to instantiate VMs
- Front-end node: user submit jobs / customized versions of VMs
- Management: batch job processing, instantiate VMs/ lauch jobs
- VM image manager: update user VMs, match user request with VM image versions
- Storage: Store different versions of VM images and application generated data, fast distribution of VM images

How it works?



- User requests: number of VMs, number of VCPUs per VM, operating systems, kernels, libraries, etc.
  - Or: previously submitted versions of VM image
- Matching requests: many algorithms have been studied in grid environment, e.g. *Matchmaker* in *Condor*

Challenges

- I/O virtualization overhead [USENIX '06]
  - Evaluation of VMM-bypass I/O with HPC benchmarks
- A framework to virtualize the cluster environment
  - Jobs requires multiple processes distributed across multiple physical nodes
  - Typically requires all nodes have the same setup
  - How to allow customized OS?
  - How to reduce other virtualization overheads (memory, storage, etc …)
  - How to reconfigure nodes and start jobs efficiently?

[USENIX '06]: J. Liu, W. Huang, B. Abali, D. K. Panda. High Performance VMM-bypass I/O in Virtual Machines

Prototype – Setup

- A Xen-based VM environment on an eight-node SMP cluster with InfiniBand
  - Node with dual Intel Xeon 3.0GHz
  - 2 GB memory
- Xen-3.0.1: an open-source high performance VMM originally developed at the University of Cambridge
- InfiniBand: a high performance Interconnect with OS-bypass features

Prototype Implementation

- Reducing virtualization overhead:
  - I/O overhead
    - Xen-IB, the VMM-bypass I/O implementation for InfiniBand in Xen environment
  - Memory overhead: Including the memory footprints of VMM and the OS in VMs:
    - VMM: can be as small as 20KB per extra domain
    - Guest OSes: specific tuned for HPC, we reduce it to 23MB at fresh boot-up in our prototype

Prototype Implementation

- Reducing the VM image management cost
  - VM images must be as small as possible to be efficiently stored and distributed
    - Images created based on ttylinux can be as small as 30MB
    - Basic system calls
    - MPI libraries
    - Communication libraries
    - Any user specific libraries
  - Image distribution: distributed through a binomial tree
  - VM image caching: VM image cached at the physical nodes as long as there is enough local storage
- Things left to future work:
  - VM-awareness storage to further reduce the storage overhead
  - Matching and scheduling

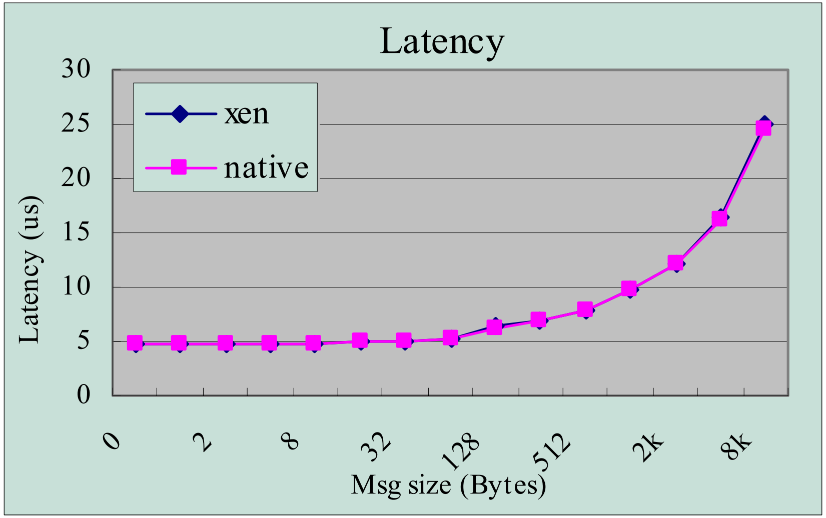
Presentation Outline

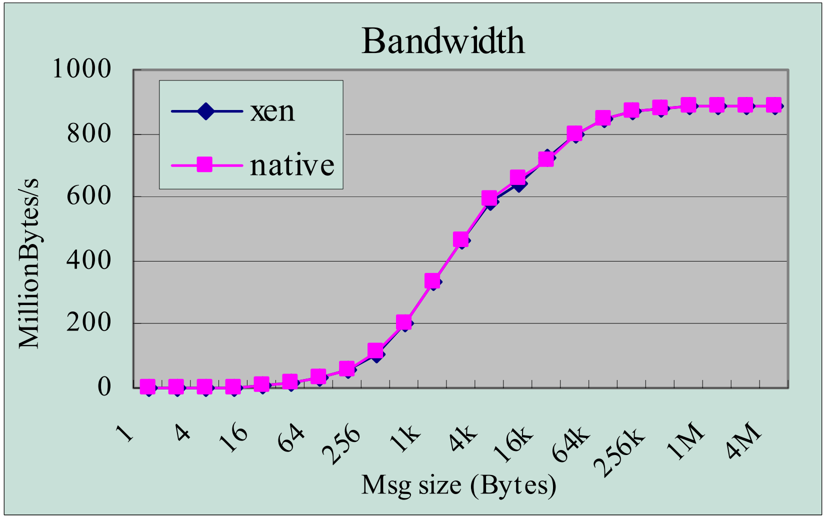
- Virtual Machines and HPC
- Background -- VMM-bypass I/O
- A framework for HPC with virtual machines
- A prototype implementation
- Performance evaluation
- Conclusion

Performance Evaluation Outline

- Focused on MPI applications
  - MVAPICH: high performance MPI implementation over InfiniBand, from the Ohio State University. Current used by over 370 organizations across 30 countries
- Micro-benchmarks
- Application-level benchmarks (NAS & HPL)
- Other virtualization overhead (memory overhead, startup time, image distribution, etc.)

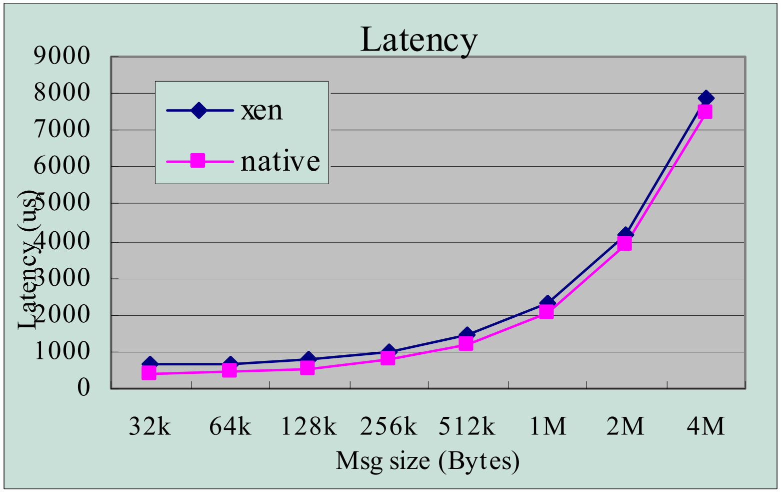
Micro-benchmarks

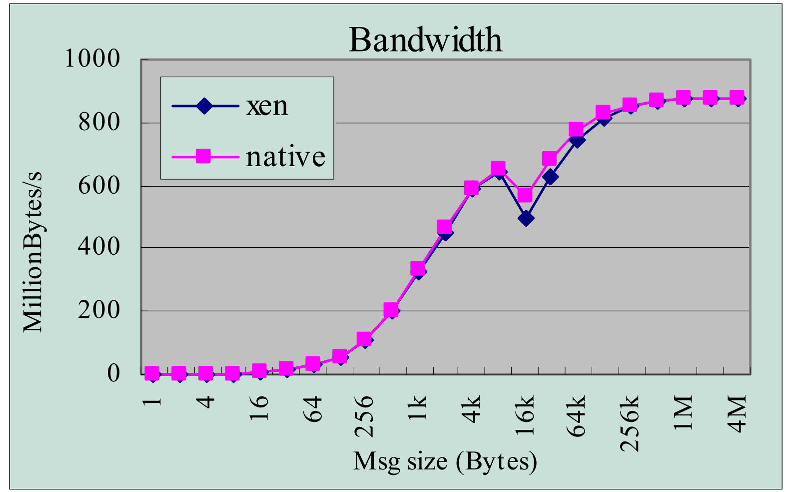




- Latency/bandwidth:
  - between 2 VMs on 2 different nodes
  - Performance in VM environment matches with native ones
- Registration cache in effect:
  - data are sent from the same user buffer multiple times
  - InfiniBand requires registration, tests are benefited from registration cache
  - Registration cost (privileged operations) in VM environment is higher

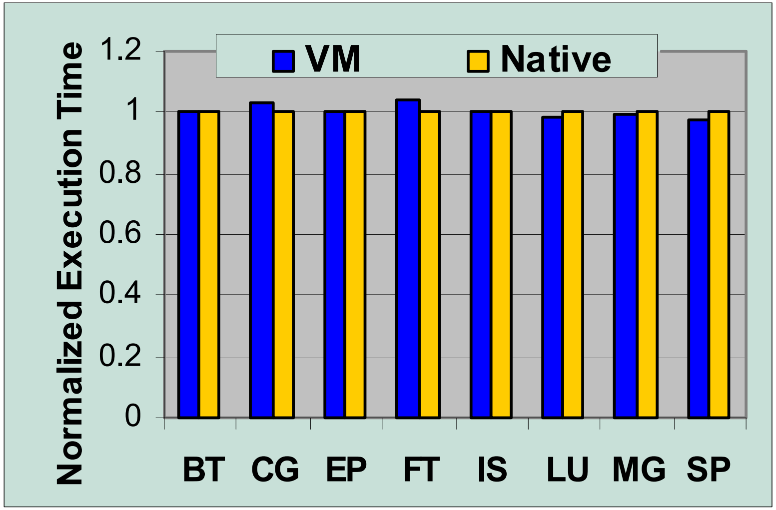
Micro-benchmarks (2)





- The set of results are taken without registration cache
- For MVAPICH, small messages are sent through pre-registered buffer, so only for medium to large messages (>16k) we see the difference
- Latency: a consistent around 200us higher in VM environment
- Bandwidth: difference is smaller due to potential overlap of registration and communication
- The worst case scenario is shown: many applications show good buffer re-use.

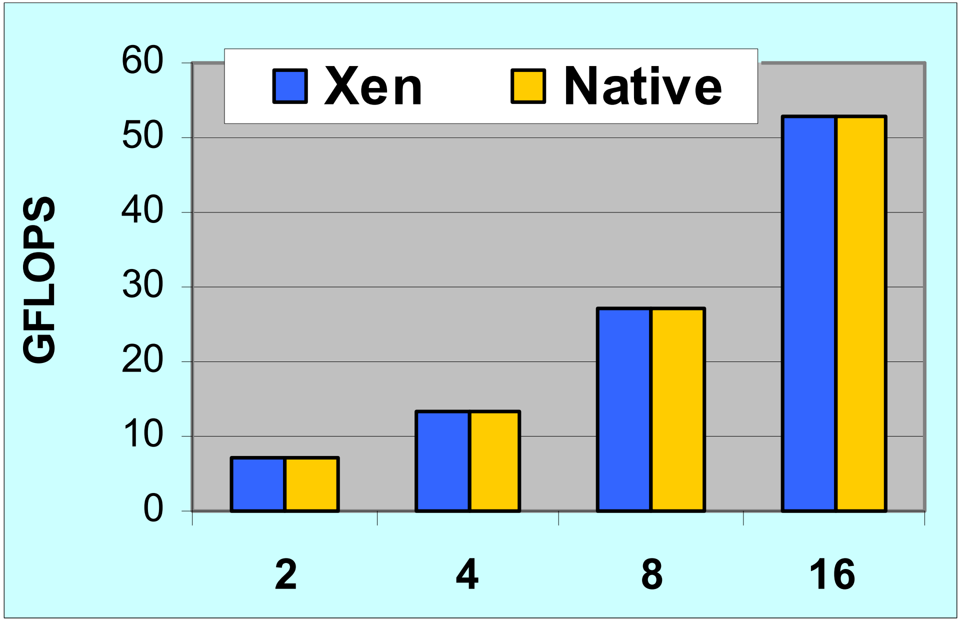
HPC Benchmarks (NAS)



	Dom0	VMM	DomU
BT	0.4%	0.2%	99.4%
CG	0.6%	0.3%	99.0%
EP	0.6%	0.3%	99.3%
FT	1.6%	0.5%	97.9%
IS	3.6%	1.9%	94.5%
LU	0.6%	0.3%	99.0%
MG	1.8%	1.0%	97.3%
SP	0.3%	0.1%	99.6%

- NAS Parallel Benchmarks achieves similar performance in VM and native environment
- Time Profiling using Xenoprof
  - It is clear that most time is spent in effective computation in DomUs

HPC Benchmarks (HPL)



- HPL: the achievable GFLOPS in VM and Native environment is within 1% difference

Management Overhead

	Startup	Shutdown	Memory
ttylinux-domU	5.3s	5.0s	23.6MB
AS4-domu	24.1s	13.2s	77.1MB
AS4-native	58.9s	18.4s	90.0MB

Scheme	1	2	4	8
Binomial tree	1.3s	2.8s	3.7s	5.0s
NFS	4.1s	6.2s	12.1s	16.1s

- VM image size: ~30MB
- Reduced services allows VM to be started very efficiently
- Small image size and the binomial tree distribution make the image distribution fast

Conclusion

- We proposed a framework to use VM-based computing environment for HPC applications
- We explained how the disadvantages of virtual machines can be addressed using current technologies with our framework using a prototype implementation
- We carried out detailed performance evaluations on the overhead of VM-based computing for HPC applications, where we show the virtualization cost is marginal
- Our case study held promises to bring the benefits of VMs to the area of HPC

Future work

- Migration support for VM-based computing environment with VMM-bypass I/O
- Investigate scheduling and resource management schemes
- More detailed evaluations of VM-based computing environments

Acknowledgements

Our research at the Ohio State University is supported by the following organizations:

- Current Funding support by
  - Office of Science, U.S. Department of Energy
  - NSF
  - Mellanox Technologies
  - Cisco Systems — Empowering the Internet Generation
  - intel
  - Linux Networx — proven
  - NetApp
  - Sun microsystems
- Current Equipment support by
  - Mellanox Technologies
  - intel
  - AMD
  - Sun microsystems
  - APPRO — HPC Cluster Solutions
  - Apple
  - advanced clustering technologies, inc.
  - IBM
  - Microway
  - PathScale
  - SilverStorm

Thank you!

Network-Based Computing Laboratory

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