XEN and KVM in
INFN production systems and a comparison between them

Riccardo Veraldi
Andrea Chierici
INFN - CNAF
HEPiX Spring 2009

Outline

- xen
- kvm
- Test description
- Benchmarks
- Conclusions

What we have now: Xen

- Open source “industry standard” for virtualization
- Included in all recent linux distributions
- Supports para and hvm approaches
- Widely used in HEP community
- Won’t go into details during presentation

What could be the future: KVM

- Kernel-based Virtual Machine
- Open source
- included in latest linux kernels: implemented as a module
  - A user space program uses /dev/kvm interface to set up VMs (qemu-kvm)
- Supports hvm approach
- Rather new to HEP community
- Qumranet now owned by Red Hat
  - Foreseen boost in development

KVM virtio

- Direct access to hardware
  - Hard disk (/dev/vdx)
  - Ethernet interface
- Support from kernel 2.6.25
- Supported since RHEL 5.3 (virtio backport)
  - kernel 2.6.18-128.1.1
  - Virtio drivers must be on the VM, not on the host
  - EGEE sw can’t currently benefit from this

KVM seen by system admin

- rpms: basically the kernel module and a modified version of qemu (*qemu-kvm*)
- Network configuration to be done by hand in order to get public IP
  - bridge-utils and tunctl rpms were used to set-up “tap” interface
    - Script developed to automate the process
  - A modified init.d script to configure software bridge has been developed at CNAF
- No VM configuration file direct support
  - VMs are launched via a standard UNIX command + command line options
  - libvirt

KVM seen by system admin

- No direct interface to manage VMs by command line (e.g. “xm”)
  - Libvirt support
- Virt-manager working
- Monitor interface
  - Very powerful, lots of options available
- qemu already well documented, support available on-line

KVM introduced at CNAF

- We use **quattor** to configure and install grid nodes and virtual machines
  - Profiles describe machine configuration
  - Network boot for installation working
    - Keeps booting from network, need to restart with *"-boot c"*
  - No need for any modification, just like xen-hvm

Qualitative test

- CNAF is running LHCb tier2 site entirely on xen VMs (2 CEs, 1 SE)
- Changed one Computing Element with a KVM machine
  - Has been working for more that 3 weeks flawlessly
  - Quattor machine profile unmodified, no effort for sysadmin
- CMS secondary squid server installed on the same host with same result (2 weeks ago)
- Running a KVM in production since 2 months for INFN-CA backups. Works flawlessly
- KVM executed (and live migrated!) a win7 VM
- Hardware used: 1 node, dual E5420, 16GB ram, sata disks via Areca controller

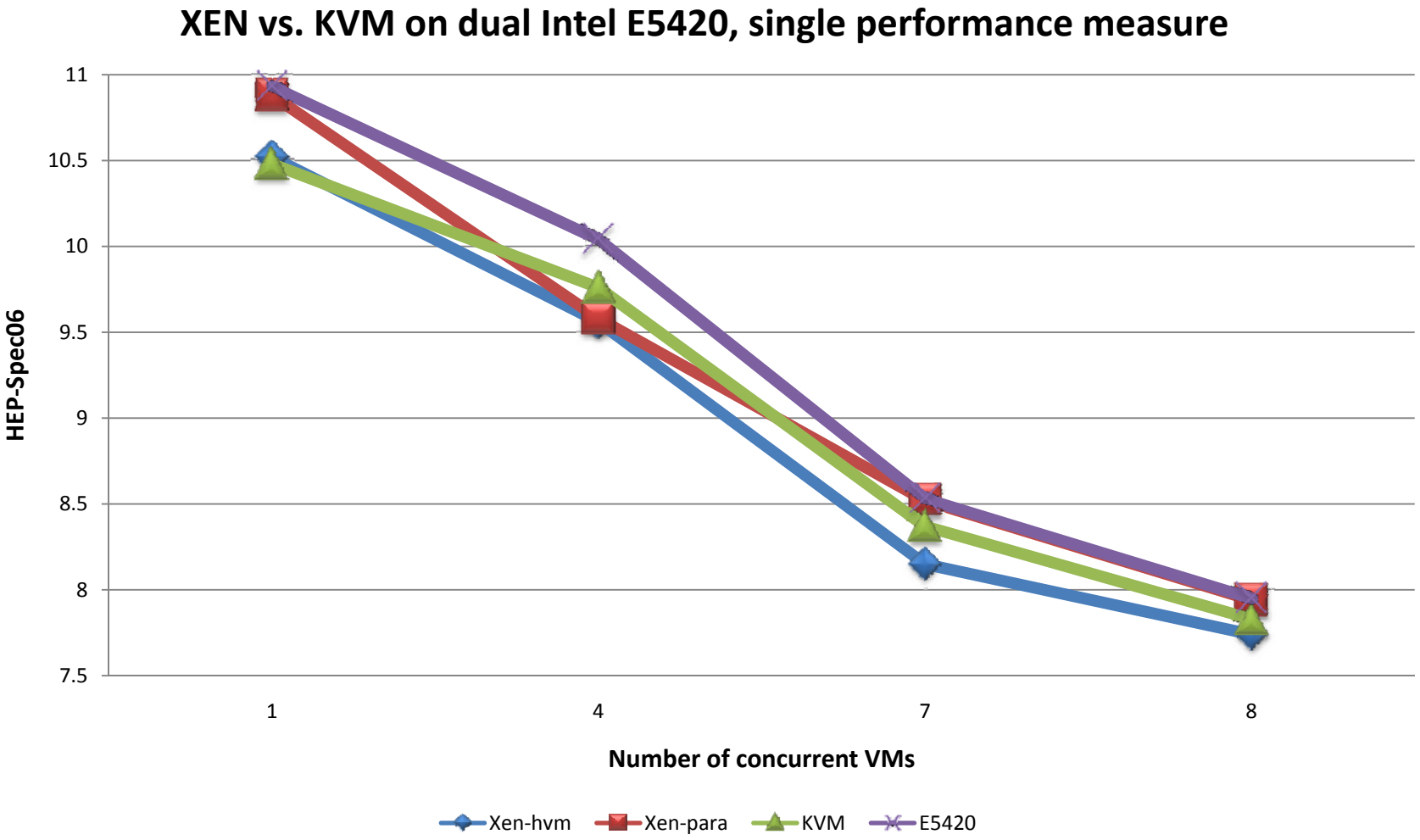
Quantitative test: description

- Need some measures to understand what is the best solution
- Tested 3 classic parameters
  - CPU → hep-spec06 (v1.1), PI test for cpu distribution
  - Network → iperf (v2.0.4)
  - Disk access → bonnie++ (v1.94)
- Compared Xen (para-virtualized and hvm) with KVM, using non virtualized machine as a baseline

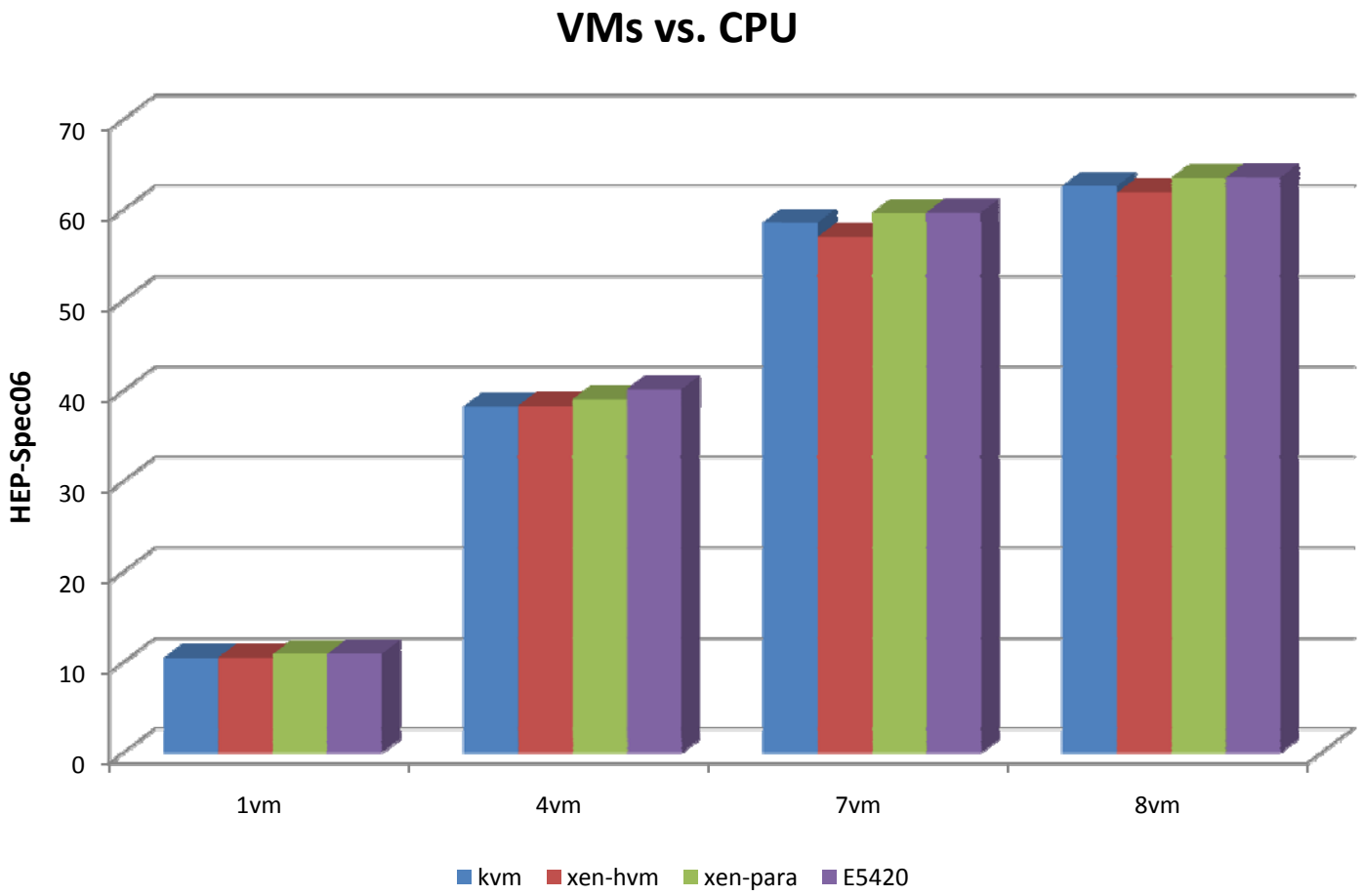
Quantitative test: description

- Hardware used: 1 blade, dual E5420, 16GB ram, 10k sas disk via LSI logic raid controller (raid0)
- Xen-para VM specs: 1 vcpu, 2 GB ram, disk on a file
- Xen-hvm VM specs: 1 vcpu, 2GB ram, disk on a file, “netfront” network driver
- KVM VM specs: 1 vcpu, 2GB ram, disk on a file, e1000 network driver emulation (non-virtio tests)
- Host OS: SL 5.2 x86_64, kernel 2.6.18-92.1.22.el5
- VM OS: SLC 4.5 i386, kernel 2.6.9-67.0.15.EL.cern
- VM OS: SLC 5.3 x86_64, kernel 2.6.18-128.1.1 for kvm virtio tests
- KVM version: 83
- Xen version: 3.2.1

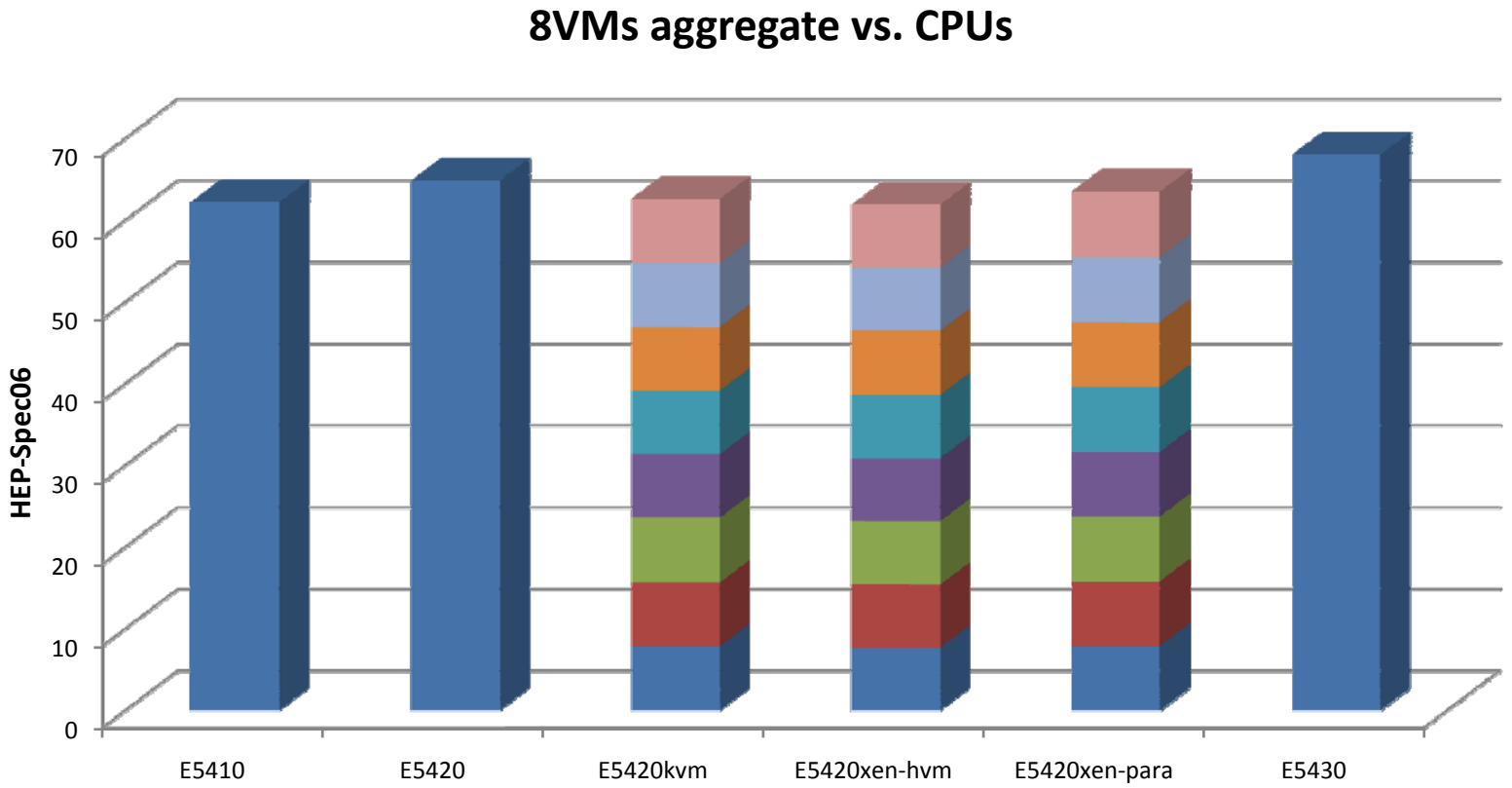
Benchmarks: HEP-Spec06



Benchmarks: HEP-Spec06



Benchmarks: HEP-Spec06



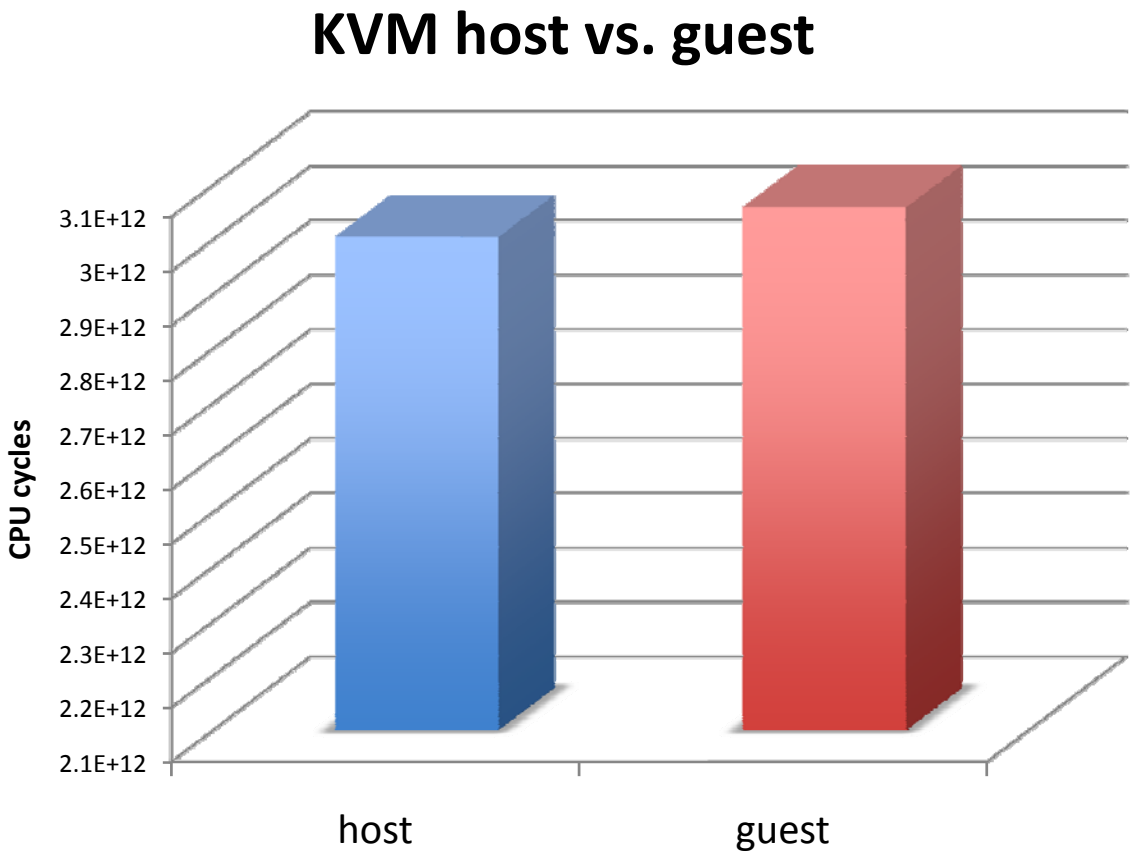
Benchmarks: HEP-Spec06

Virtualization Technology	% loss from non emulated CPU (E5420, 8vm)
E5420kvm	3,42
E5420xen-hvm	4,55
E5420xen-para	2,02
E5410 vs. E5420	4,07

Benchmarks: PI test

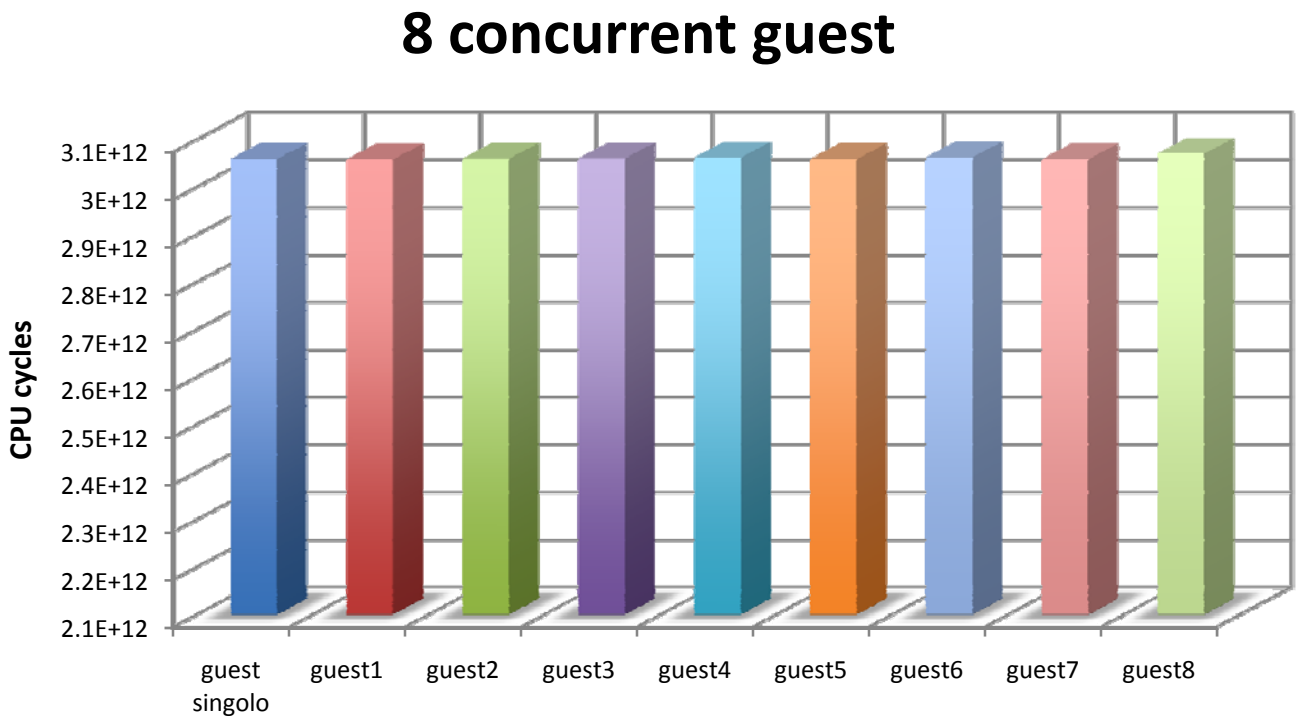
- System tools to measure CPU usage are not reliable enough
- Calculating PI with n arbitrary ciphers
  - Algorithm based on Numerical Recipes
  - Written in C
  - Compiled with options
    - gcc -O3 -ffast-math -finline-limit=1000
- Returns the number of CPU cycles
  - An external function included into the code calculates the number of CPU cycles, using rdtsc IA32 registry

Benchmarks: PI test



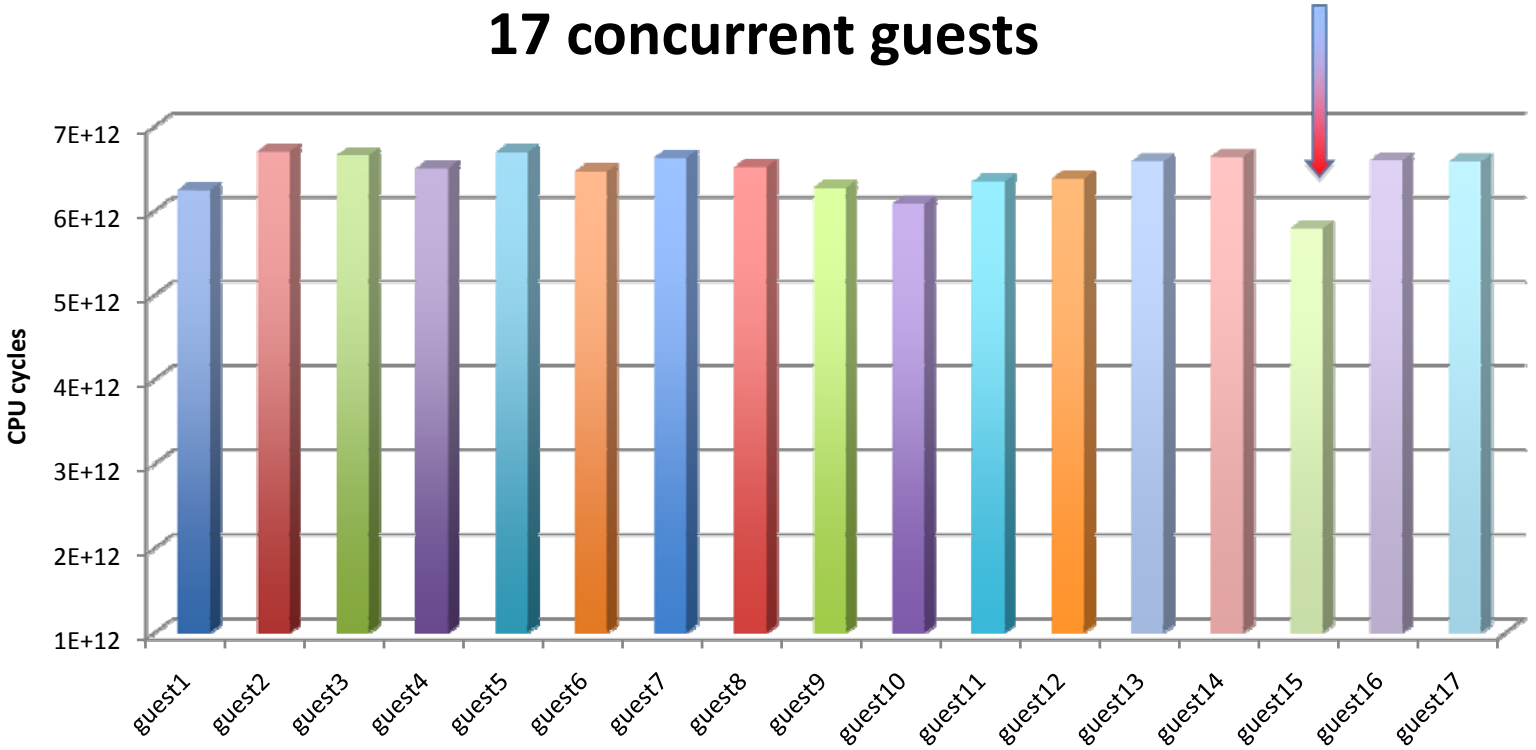
difference < 2%

Benchmarks: PI test



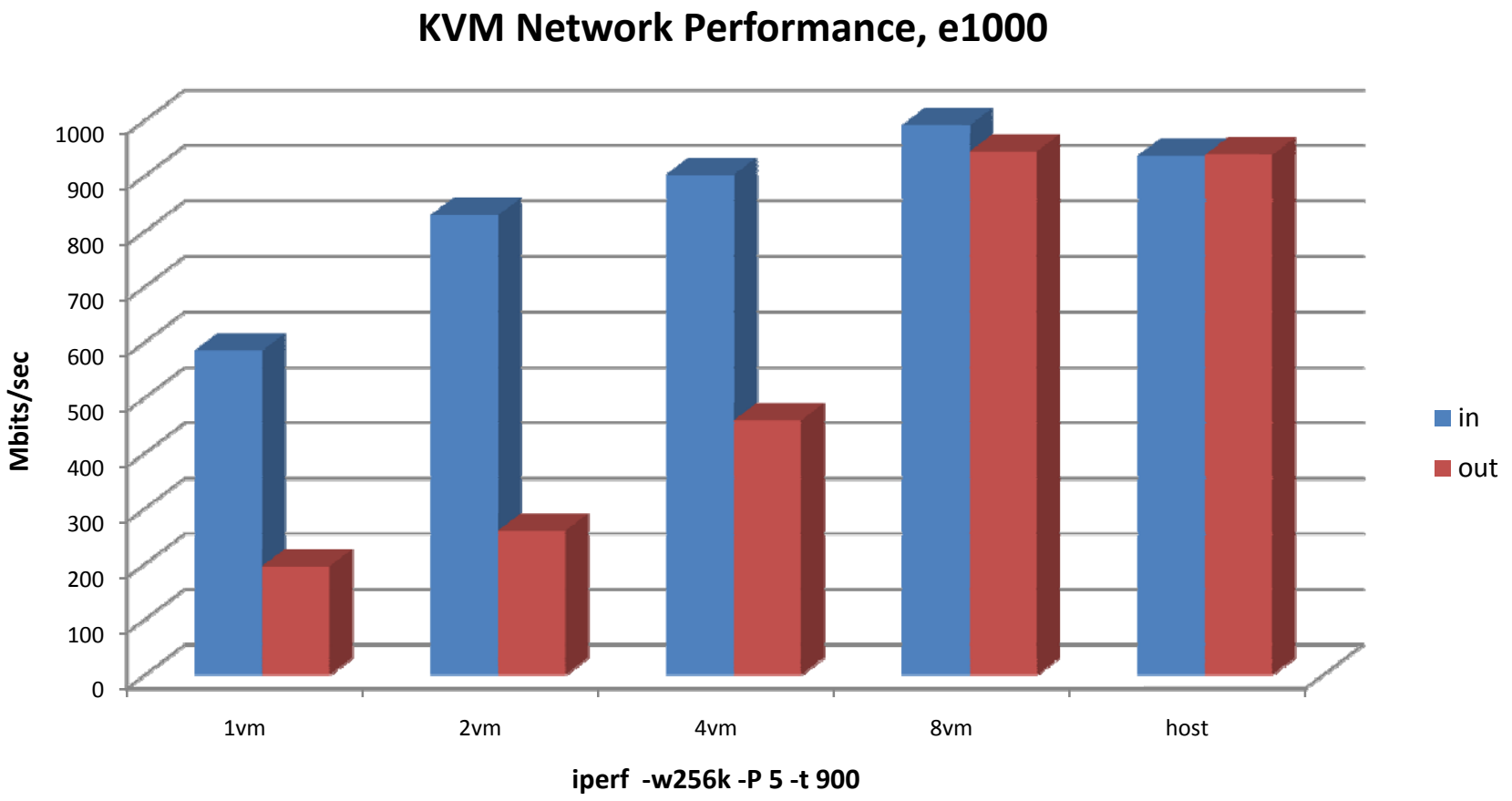
- 8 core
- Balanced Performance
- Every guest uses one core at 100%

Benchmarks: PI test

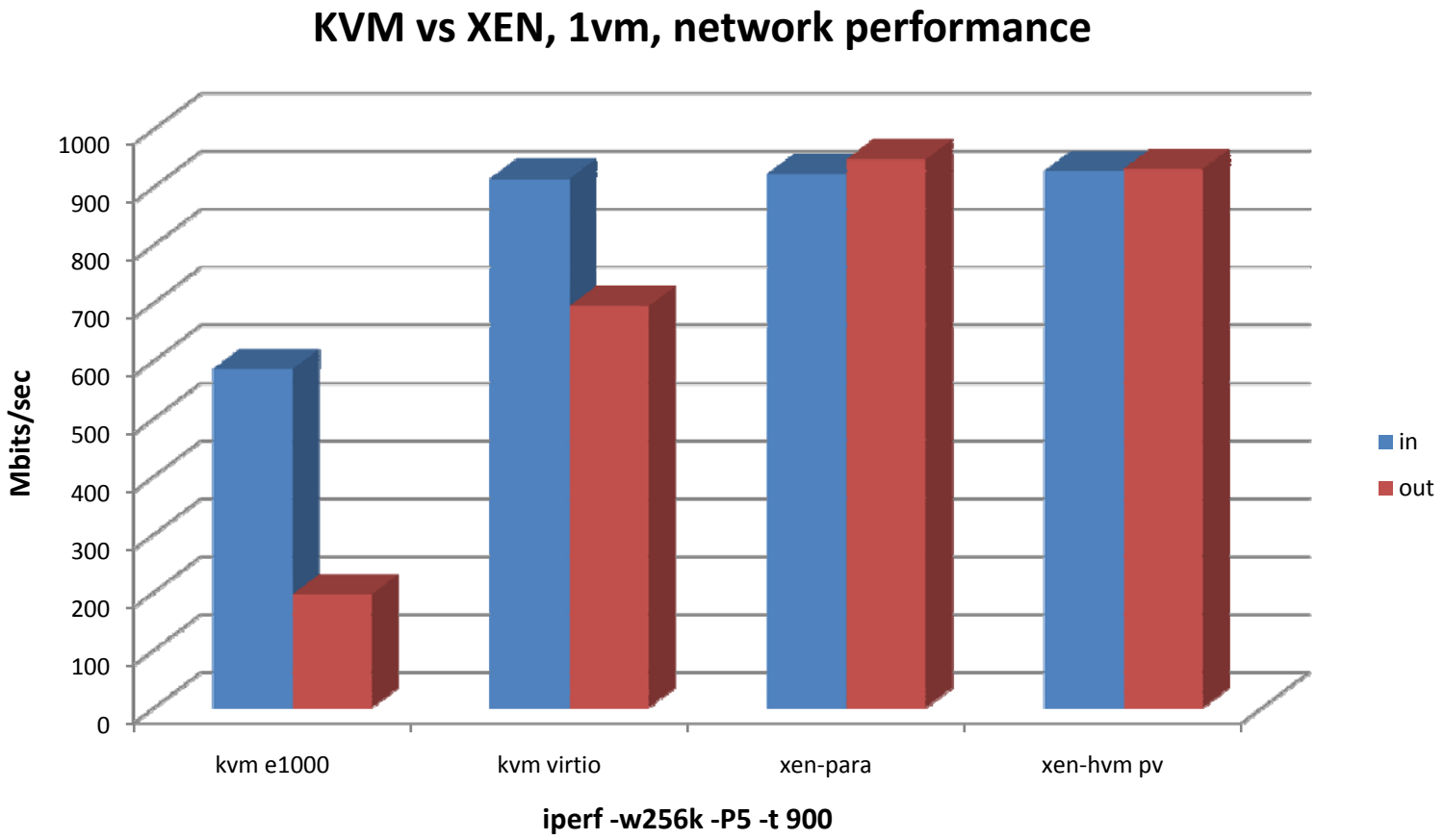


- Good balancing on guest
- Max difference 14%, average ~5%

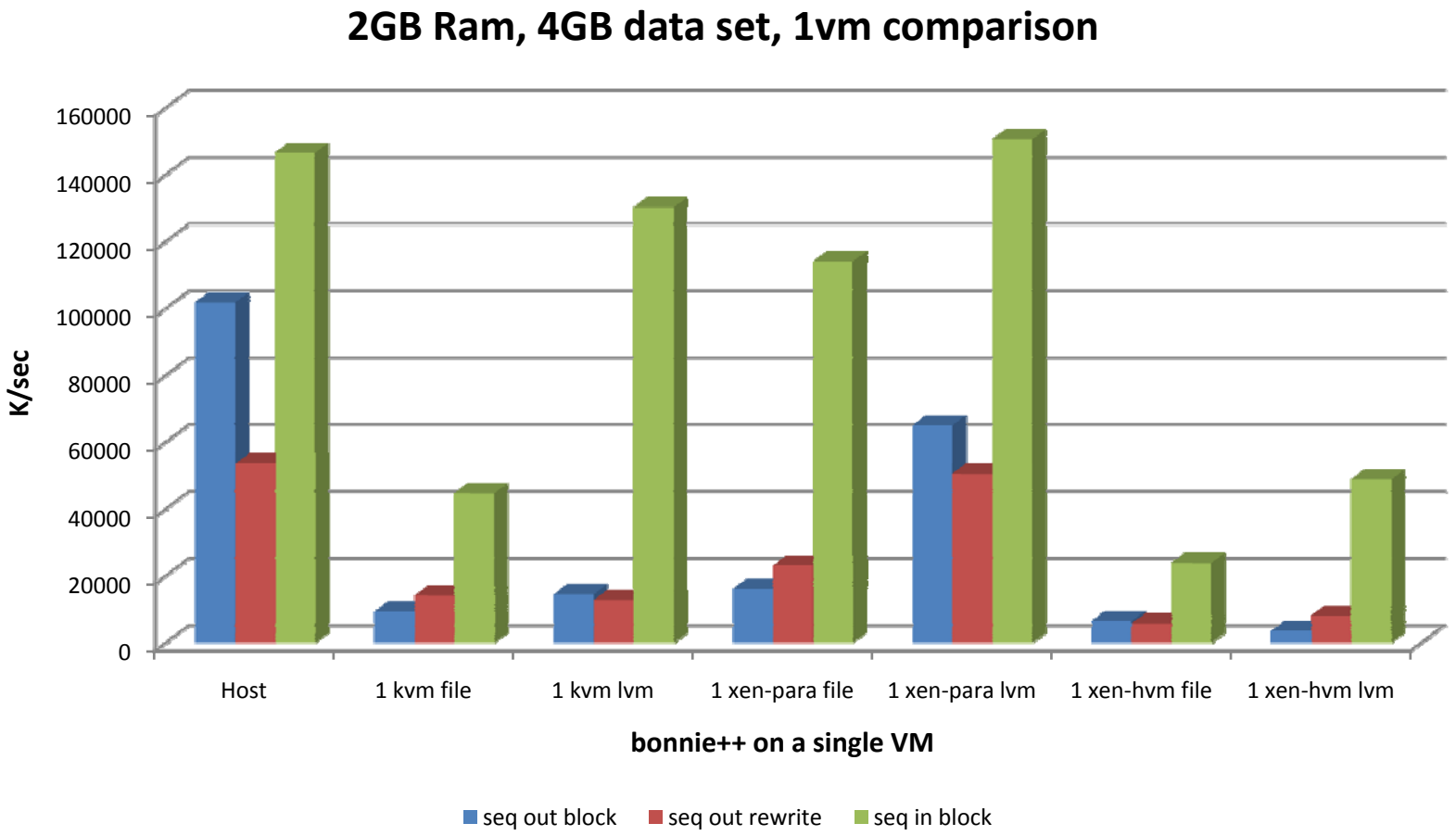
Benchmarks: Iperf



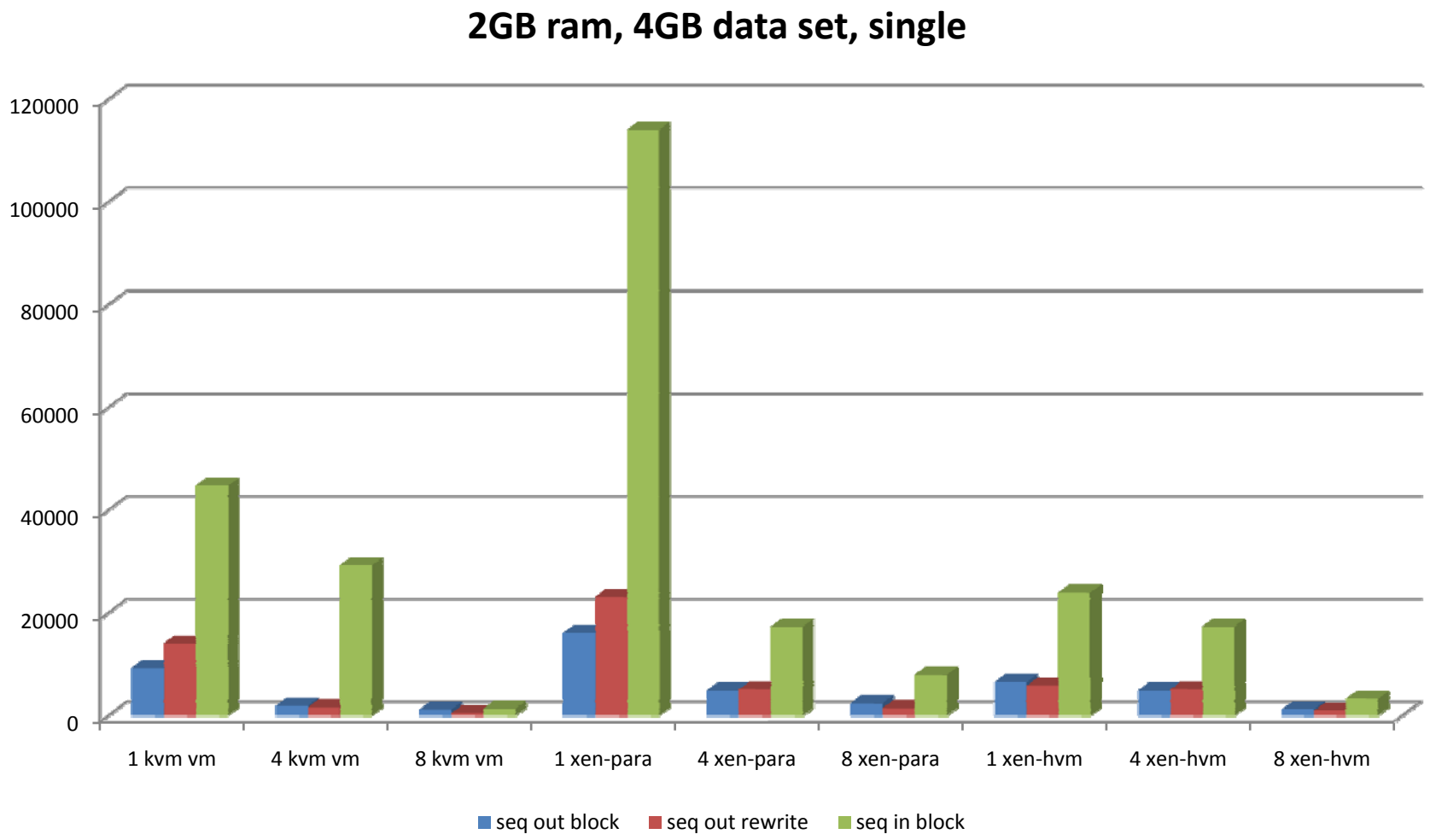
Benchmarks: Iperf



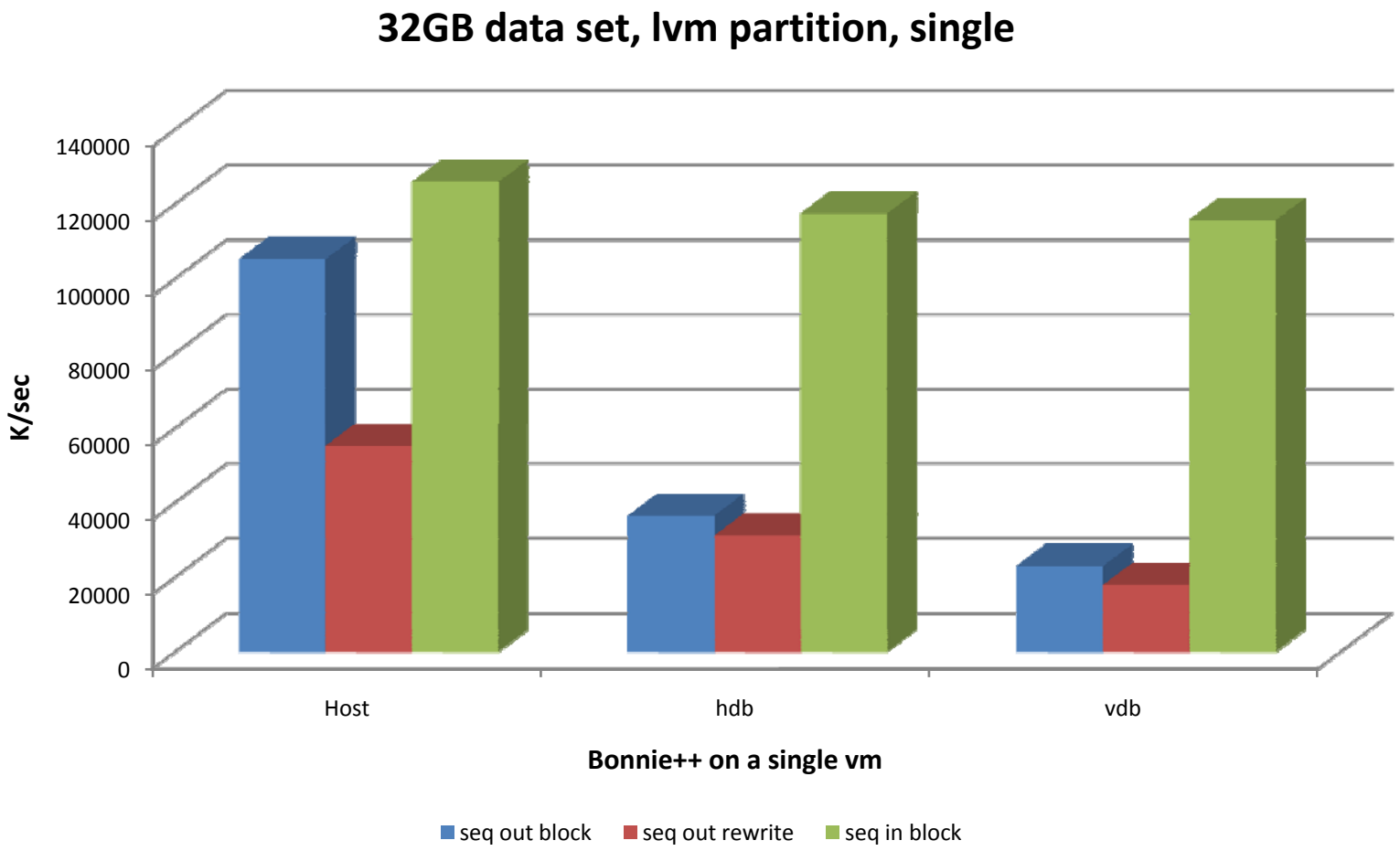
Benchmarks: bonnie++, file vs lvm



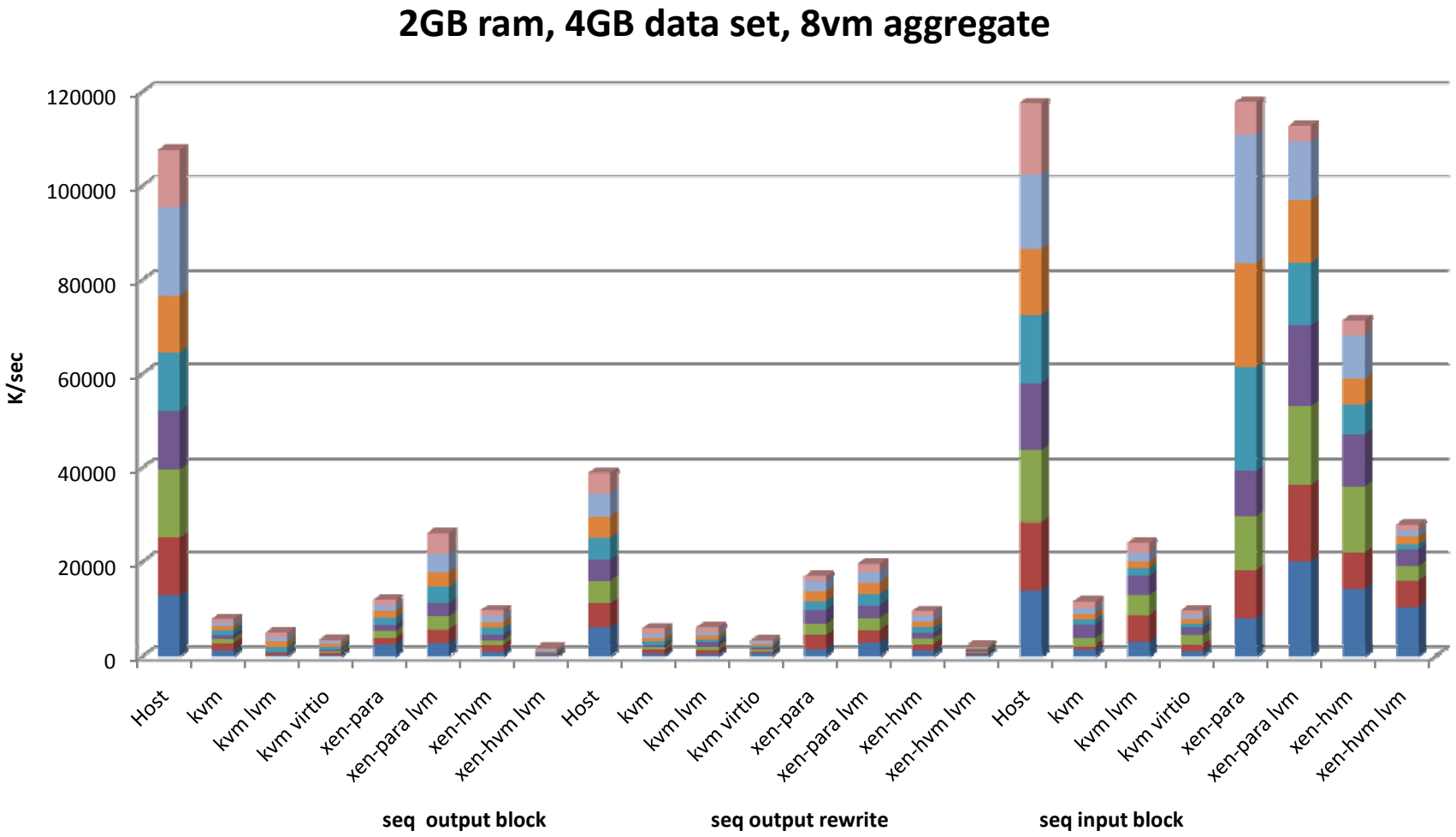
Benchmarks: bonnie++



Benchmarks: bonnie++, kvm virtio



Benchmarks: bonnie++



Conclusions

- KVM proved good stability and reliability
  - No problems on running production machines for more than 4 weeks
  - CPU performances are extremely good
    - Modern CPU virtualization technologies help!
  - Network performances are fair with e1000, good with virtio
  - Disk I/O seems the most problematic aspect
    - Other solutions have problems too
  - Requires sysadmins only a small effort
- Even if looking promising, right now xen is the most performing solution

Future work

- qemu snapshot features
  - Backup and cloning of nodes
- high-level VM managers
  - Ovirt, enomalism, ganeti

Bibliography and links

- Xen
  - Xen repository
- KVM
  - Kvm repository
- Quattor
- HEP-SPEC
- Numerical Recipes: second edition in C
- Computer Systems: A Programmer's Perspective