



Red Hat Performance Briefs

KVM Virtualized I/O Performance

Achieving Leadership I/O Performance Using Virtio-Blk-Data-Plane Technology Preview in Red Hat Enterprise Linux 6.4

Khoa Huynh, Ph.D. - Linux Technology Center, IBM

Andrew Theurer - Linux Technology Center, IBM

Stefan Hajnoczi - Red Hat

February 2013

Version 1.7





Linux is a registered trademark of Linus Torvalds.

Red Hat, Red Hat Enterprise Linux and the Red Hat "Shadowman" logo are registered trademarks of Red Hat, Inc. in the United States and other countries.

IBM, the IBM logo, ibm.com, ServeRAID, System x, X-Architecture are trademarks of International Business Machines Corporation in the United States, other countries, or both

UNIX is a registered trademark of The Open Group.

Intel, the Intel logo and Xeon are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

All other trademarks referenced herein are the property of their respective owners.

© 2013 by Red Hat, Inc and IBM Corp. This material may be distributed only subject to the terms and conditions set forth in the Open Publication License, V1.0 or later (the latest version is presently available at <http://www.opencontent.org/openpub/>).

The information contained herein is subject to change without notice. IBM Corp. or Red Hat, Inc. shall not be liable for technical or editorial errors or omissions contained herein.

Distribution of modified versions of this document is prohibited without the explicit permission of IBM Corp. and Red Hat Inc.

Distribution of this work or derivative of this work in any standard (paper) book form for commercial purposes is prohibited unless prior permission is obtained from IBM Corp. or Red Hat Inc.

Table of Contents

Executive Summary.....	4
1. Introduction.....	5
1.1. The KVM hypervisor.....	5
1.2. Red Hat® Enterprise Linux® 6 Virtualization.....	6
1.3. Technology Preview – Virtio-Blk-Data-Plane.....	6
1.4. IBM® System x® Servers.....	7
1.5. Motivation.....	8
2. Test setup.....	9
2.1. Test Hardware.....	9
2.2. Workload.....	9
2.3. KVM Configuration.....	10
3. Results.....	12
Summary.....	15
References.....	16



Executive Summary

The Kernel-based Virtual Machine (KVM) hypervisor has earned a reputation as the highest performing hypervisor in virtualization benchmarks, holding the top seven results in SPECvirt_sc2010 [1] and recently achieving a leadership spot among x86-virtualized results on SAP's 2-tier SD benchmark [2]. One of the key ingredients to this success is KVM hypervisor's ability to handle the high I/O rates required by enterprise workloads, such as databases, ERP systems, and low-latency financial trading applications that are running in virtual machines.

This paper describes a test environment that included an IBM® System x3850 X5 host server with QLogic® QLE 256x Host Bus Adapters (HBAs), Red Hat® Enterprise Linux® 6.4 hypervisor with a new I/O virtualization technology called ***virtio-blk-data-plane***, and Red Hat Enterprise Linux 6.4 guests. It also details test results that demonstrate that a single KVM guest can handle more than 1.2 million I/O operations per second (IOPS) at 8KB I/O request size and more than 1.5 million IOPS at 4KB and smaller request sizes – the highest storage I/O performance ever reported in a virtualized environment. This achievement is the result of a collaborative effort between IBM's Linux Technology Center's Performance organization and Red Hat's KVM Development team. These results show that KVM is ready for the most demanding enterprise workloads.

1. Introduction

A hypervisor is an object that manages virtual machines (guests) running on a physical machine (host). It can be a computer program, firmware, or hardware that provides the end user the ability to create, configure, and manage virtual machines that operate exactly as if they were physical machines.

1.1. The KVM hypervisor

The Kernel-based Virtual Machine (KVM) project represents the next generation in open-source hypervisors. KVM is fully integrated into the Linux operating system both as a host and a guest. Unlike other hypervisors, KVM makes no distinction between running in either host or hypervisor mode. This duality in design has helped KVM to rapidly mature into a stable, high-performing hypervisor, positioned to outperform other hypervisors available on the market today.

The first design principle includes the following:

- Leverage all hardware-assisted virtualization capabilities provided by Intel® Virtualization Technology (VT) and AMD® Secure Virtual Machine (SVM)
- Feature the latest hardware virtualization extensions, including:
 - Hardware nested paging (EPT/NPT)
 - Pause loop filtering
 - I/O off-load features, such as secure PCI pass-through using Intel VT-D or AMD I/O Memory Management Unit (IOMMU)
- Exploit hardware capabilities while keeping the KVM virtualization overhead to the absolute minimum

The second design principle includes the following:

- Leverage the Linux operating system
- Fulfill many components required by a hypervisor, such as memory management, scheduler, I/O stack, and device drivers by reusing optimized, off-the-shelf Linux implementations

The Linux kernel, with its 20 years of development, is the industry leader in terms of performance and availability. The Linux process scheduler, for example, provides completely fair scheduling (CFS) that is optimized to manage complex workloads and NUMA systems, while offering low latency, high performance determinism, and fine-grained Service Level Agreement (SLA) for applications. By placing the KVM hypervisor directly into the Linux kernel, all of these services and advantages have a direct impact on the hypervisor performance.



1.2. Red Hat® Enterprise Linux® 6 Virtualization

Red Hat's unique approach to virtualization is easy to adopt as it is delivered as an integral part of the Red Hat Enterprise Linux platform. Using KVM technology, Red Hat's virtualization capabilities are integrated into Red Hat Enterprise Linux and leverage the latest in hardware virtualization that Intel and AMD processor platforms can provide. The modular design of Red Hat Enterprise Linux allows customers to choose when and where to use virtualization. For additional flexibility, customers can deploy both Red Hat Enterprise Linux and Microsoft® Windows® as fully supported guests (virtual machines) within a Red Hat Enterprise Linux virtualized environment. Red Hat Enterprise Linux also supports multiple virtualization use cases, from hardware abstraction for existing software stacks and data center consolidation to virtualized clusters and private clouds.

Beyond core virtualization, Red Hat Enterprise Linux offers leading support for advanced virtualized I/O capabilities through Single Root I/O Virtualization (SR-IOV) and N-Port ID Virtualization (NPV) standards. The *libvirt* toolkit – a standard virtualization management infrastructure – was developed by Red Hat and adopted by other operating systems. The *libvirt* toolkit provides a flexible interface for defining, managing, and monitoring virtual machines.

The Red Hat Enterprise Linux 6.4 supports up to 160 virtual CPUs (vCPUs) per virtual machine, allowing even the largest workloads to be virtualized.

Although Red Hat Enterprise Virtualization (RHEV) is an enterprise-class management solution that uses the KVM technology, it is outside the scope of this paper.

1.3. Technology Preview – Virtio-Blk-Data-Plane

Red Hat Enterprise Linux 6.4 introduces, as a Technology Preview, a new I/O virtualization technology called *virtio-blk-data-plane*. This new technology achievement is the result of a collaborative effort between IBM's Linux Technology Center's Performance organization and Red Hat's KVM Development team. It accelerates I/O operations going through the para-virtualized I/O block driver (*virtio-blk*) with dedicated per-device threads. This approach, shown in **Figure 1**, allows the block I/O processing to run concurrently with other device emulation, and therefore, achieves some of the best I/O performance results to date. Device emulation in KVM is provided by the Quick EMULATOR (QEMU) running in user space.

The key concept of the *virtio-blk-data-plane* approach is that by isolating the *virtio-blk* processing into dedicated threads, synchronization with other components can be avoided. Consequently, these dedicated threads, called *virtio-blk-data-plane threads*, do not need to acquire the global mutex that protects the hardware emulation code in QEMU when they submit guest I/O operations to the host, resulting in higher performance. There is a dedicated *virtio-blk-data-plane* thread for each block device in the guest. For even better performance, *virtio-blk-data-plane* also exploits the *ioeventfd / irqfd* mechanism, which decouples the I/O processing from the guest's execution. The Asynchronous I/O (AIO) support in the host Linux kernel is used by *virtio-blk-data-plane* to process the actual I/O requests on behalf of the guest. Due to the architectural changes required to move *virtio-blk* processing to dedicated

threads, this technology preview in Red Hat Enterprise Linux 6.4 currently limits some storage features:

- Only raw image files are currently supported.
- Storage migration, hot unplug, I/O throttling, image streaming, and drive mirroring are currently not supported.

These limitations apply only to virtio-blk-data-plane and not to the existing virtio-blk device emulation.

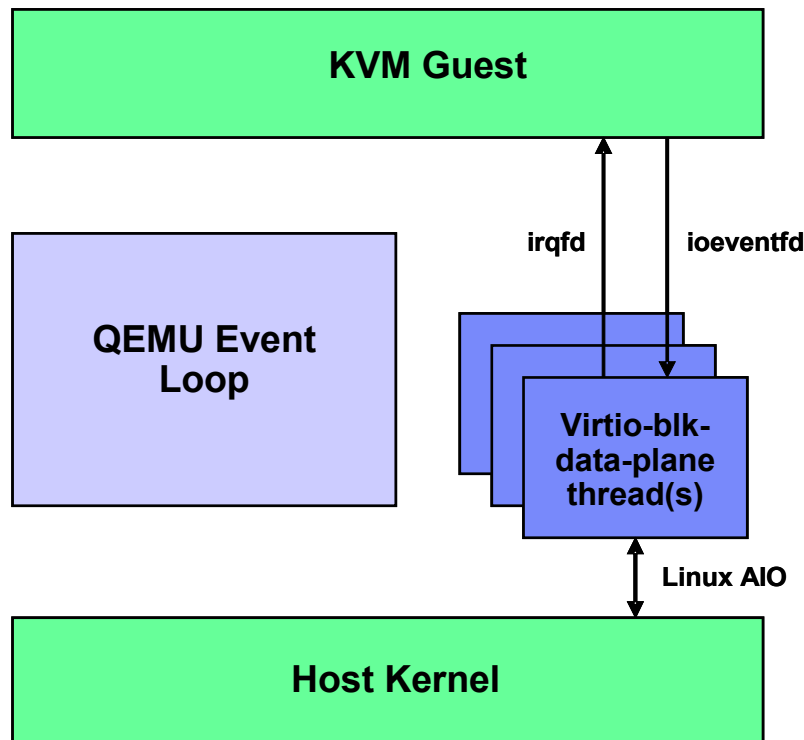


Figure 1. Virtio-blk-data-plane

1.4. IBM® System x® Servers

IBM System x servers support Microsoft Windows, Linux, and hypervisors. System x servers are intelligent systems, designed to reduce costs and complexity for enterprise workloads. With the introduction of eX5 – IBM's fifth-generation industry-leading enterprise X-Architecture® servers – IBM engineers have redefined x86 servers by expanding their capabilities. A member of the eX5 server family, the x3850 X5 is a scalable 4-socket, 4U, rack-optimized enterprise server that delivers the following benefits to enterprise customers:

- High memory capacity (up to 3TB, 3 times the memory capacity of other 4-socket x86 servers, using the industry-unique IBM MAX5 memory expansion unit)
- Processor scalability up to 8 sockets (up to 80 processor cores) by connecting two 4-socket x3850 X5 systems together and doubling all system resources (including up to

6TB of memory, using two MAX5 memory expansion units)

- The broadest range of network and storage support in the industry for ultimate flexibility and choice
- Support for IBM eXFlash solid-state storage technology for extreme storage I/O performance
- Integrated Emulex 10 GbE Virtual Fabric Adapter with capability for upgrades to Fiber Channel over Ethernet (FCoE)
- Fifth-generation eX5 chipset design enhancements, built on the latest X-Architecture blueprint
- Balanced systems for virtualization, database and enterprise workloads
- Workload-optimized systems with customizable configurations for target workloads
- Greater performance and utilization at a lower total cost
- Mainframe-inspired reliability
- Simplified power and systems management with an energy-smart design and remote access

Figure 2 shows the front exterior of the IBM System x3850 X5 server.



Figure 2. IBM System x3850 X5 Server

1.5. Motivation

IBM and Red Hat customers drive enterprise workloads, such as databases, ERP systems, and low-latency financial trading applications. In the past, these workloads were seldom virtualized in production due to scaling and time-sensitive barriers, so they were unable to exploit the many benefits of virtualization, such as hardware abstraction, live migration, dynamic resource allocation, and more. Proving that KVM is able to sustain high I/O rates is very critical in enabling the migration of these workloads into the virtualized environment.

2. Test setup

2.1. Test Hardware

To demonstrate how KVM can handle extremely high I/O rates, it was necessary to set up a storage back-end that was capable of delivering at least one million I/O operations per second (IOPS). For a diagram of the test environment, refer to **Figure 3**.

The KVM host server was an IBM System x3850 X5 with four Intel Xeon® E7-4870 processors and 256 GB of memory. Each E7-4870 processor had 10 cores running at 2.40 GHz. The x3850 X5 server had seven available PCI slots, each of which were fitted with QLogic® QLE 256x Host Bus Adapters (HBAs). Each of these adapters had two ports, each supporting full-duplex, 8-Gigabit-per-second data links.

Each QLogic HBA in the KVM host was directly connected to a unique Fiber Channel SCSI target server. For more information about the Fiber Channel SCSI target server, see <http://linux-iscsi.org>. Each SCSI target server had four 15-GB RAM disks configured so that these RAM disks would appear as Logical Unit Numbers (LUNs) at the SCSI host (KVM host server). As a result, from the KVM host's perspective, the storage back-end had a total of 14 (7 PCI slots x 2 ports) PCI devices and 56 (14 PCI devices x 4 RAM disks) storage LUNs.

2.2. Workload

The Flexible I/O (FIO) benchmark (<http://linux.die.net/man/1/fio>) was used to generate disk I/O workloads and measure the resulting I/O rates, throughput, and latency. This workload had the following FIO parameter settings:

- Direct I/O operations
- Asynchronous I/O operations (engine = libaio)
- Random read and write operations (50% reads, 50% writes)
- I/O request sizes = 512 bytes, 1KB, 2KB, 4KB, 8KB
- One job per storage LUN
- Queue depth = 32

Both random read and write operations were included in the workload for two reasons. First, a workload with both random reads and writes would be more realistic and similar to actual enterprise workloads, such as database applications. Secondly, having both reads and writes in the workload allowed the ability to fully exploit the full-duplex data links supported by the QLogic HBAs. A range of I/O request sizes, from 512 bytes to 8KB, was also considered as these are the typical I/O sizes used in many real-world applications.

The FIO workload was first run directly on the KVM host server to determine the maximum I/O rates that this test setup could support. These “bare-metal” results indicated that our storage setup could deliver up to 200,000 IOPS per SCSI target server, or a total of 1.4 million IOPS

with all 7 SCSI target servers, using 8KB requests. This was determined to be sufficient for the KVM storage performance testing.

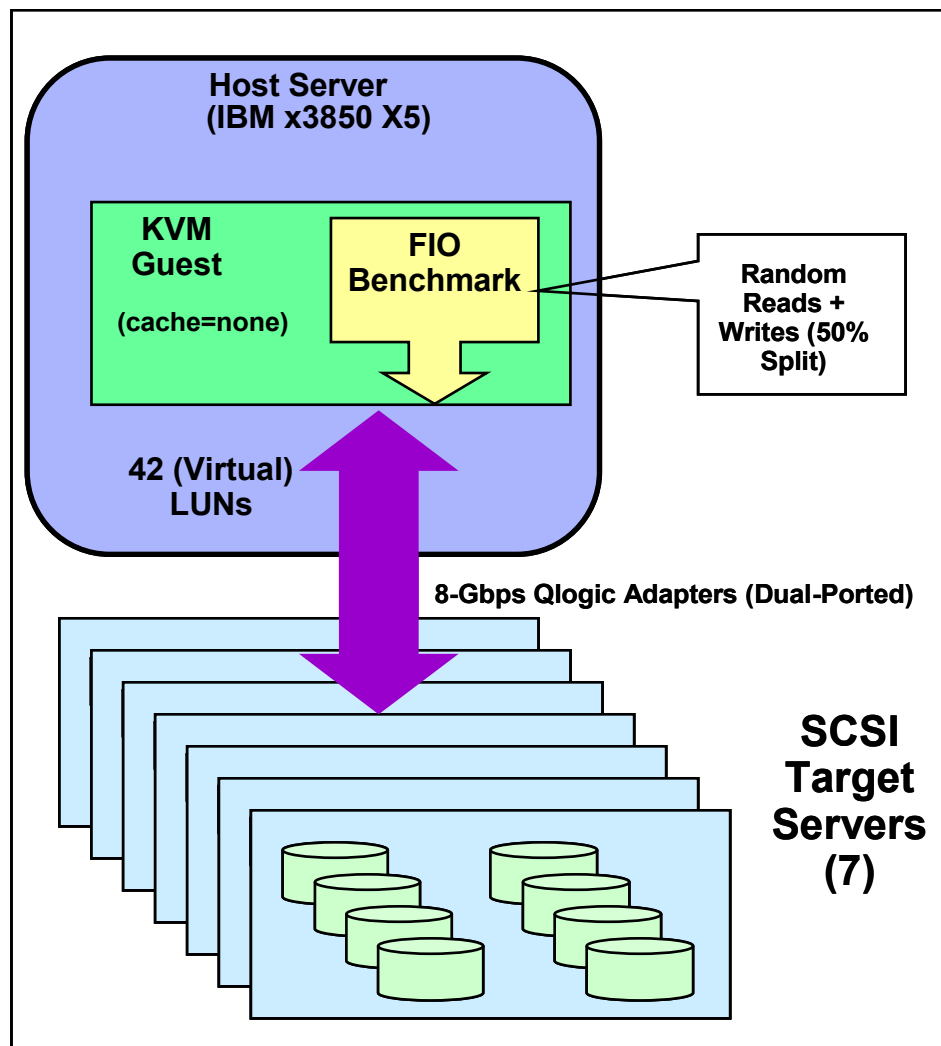


Figure 3. The test setup

2.3. KVM Configuration

To achieve the best possible I/O rates for the KVM guest, the `virtio-blk-data-plane` feature was enabled for each LUN (a disk or partition) that was passed from the host to the guest. To enable `virtio-blk-data-plane` for a LUN being passed to the guest, the `x-data-plane=on` option was added for that LUN in the `qemu-kvm` command line used to set up the guest. For example:

```
/usr/libexec/qemu-kvm -drive if=none,id=drive0,cache=none,aio=native,format=raw,file=<disk or partition> -device virtio-blk-pci,drive=drive0,scsi=off,x-data-plane=on
```



After the guest finished booting up, all I/O operations going through the para-virtualized block driver (virtio-blk) would use the fast I/O path enabled by the virtio-blk-data-plane feature for this specific LUN.

Red Hat Enterprise Linux 6.4 was used on both host and guest.

3. Results

In this test, the goal was to determine the maximum I/O rates that could be achieved with a single KVM guest. A very large guest was configured with 40 virtual CPUs to test this goal. Because our FIO workload was not very memory-intensive, 8GB of memory was configured for the guest. More specifically, the single-guest configuration included the following:

- Host Server: IBM System x3850 X5
 - 4 Intel Xeon E7-4870 processors (40 cores at 2.40 GHz), 256 GB memory (total)
 - Red Hat Enterprise Linux 6.4
- Storage:
 - 7 QLogic QLE 256x (8 Gbps, dual-ported) connected to 7 SCSI target servers
 - 56 LUNs
- KVM Guest (Virtual Machine):
 - 40 virtual CPUs, 8 GB memory
 - 42 virtual LUNs
 - Red Hat Enterprise Linux 6.4
- FIO Workload
 - Random reads and writes (50% reads, 50% writes)
 - 1 job per LUN
 - Direct I/O operations
 - Engine = libaio
 - Queue depth = 32

To optimize the virtualized I/O performance, the following performance tuning steps were done prior to the test runs:

- Leveraged caching efficiency in the host processors by binding dedicated virtio-blk-data-plane threads to specific CPUs in the host
- Used deadline I/O scheduler in the host
- Exploited interrupt-coalescing capability of the QLogic HBAs
- Disabled entropy random contribution from block devices
- Disabled all cgroup and CPU delay accounting in both host and guest
- Switched the clock source in the guest to TSC

Figure 4 shows the number of I/O operations per second (IOPS) and the average latencies for random read and write operations across several I/O request sizes. Using a single guest, KVM with virtio-blk-data-plane was able to achieve up to **1.2 million IOPS** for 8KB random I/O requests and **more than 1.5 million IOPS** for random I/O requests that were 4KB or less.

The average latencies for random reads and writes were very consistent up to 4KB I/O size – at about 0.8 milliseconds (ms) for reads and 1.0 ms for writes. The average latencies for 8 KB requests were a little higher – 0.9 ms for reads and 1.6 ms for writes – due to larger data transfer times. This data shows that KVM can sustain very high storage performance across all typical I/O sizes.

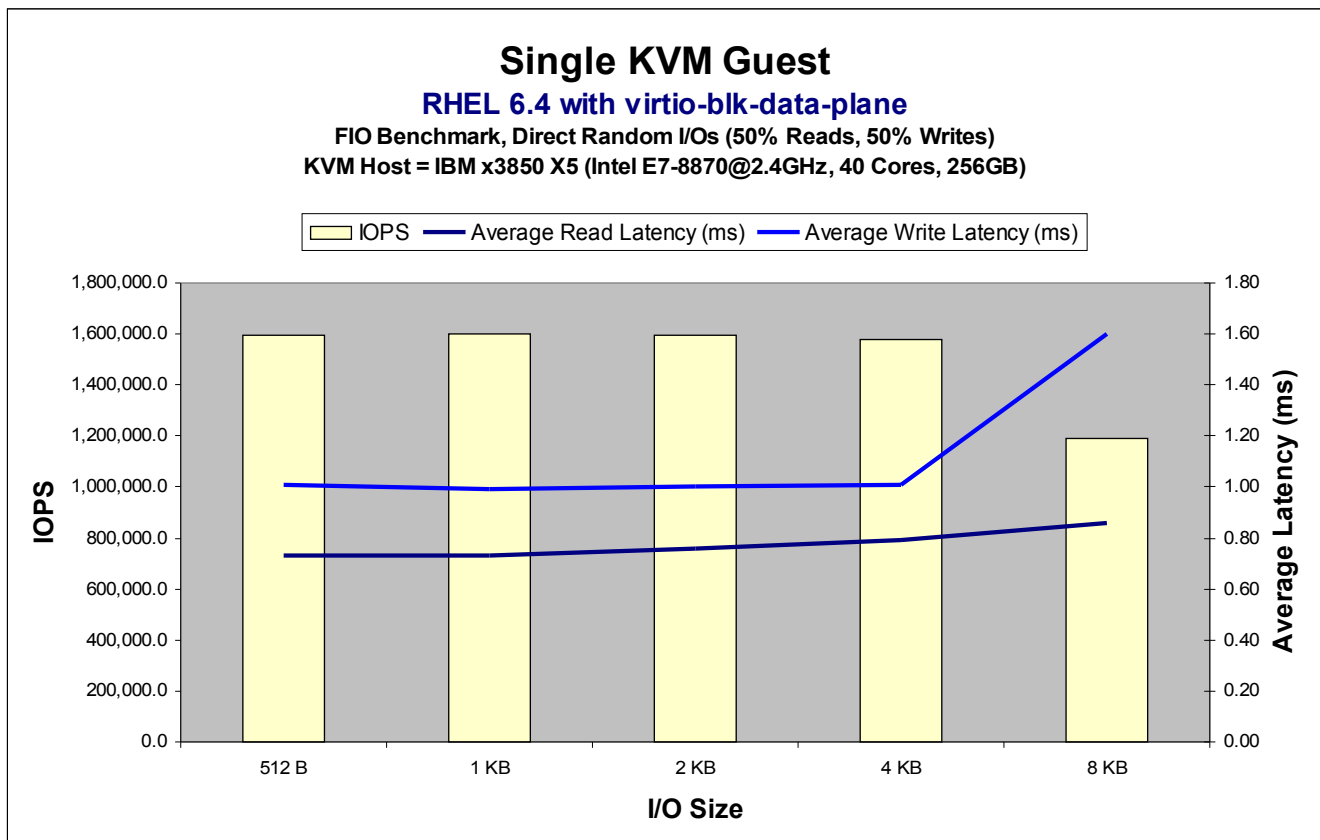


Figure 4. I/O rates and average latencies across multiple block sizes

At the I/O request size of 4KB or smaller, KVM with virtio-blk-data-plane could achieve close to 1.6 million IOPS for a single guest. **Figure 5** shows that, *without* virtio-blk-data-plane technology, KVM could achieve only about 150,000 I/O operations per second for a single guest, mostly due to the impact of the global mutex in QEMU. With the PCI pass-through (device assignment) approach, where the QLogic PCI physical functions were passed directly from the host to the guest using KVM's pass-through capability, KVM could achieve more than 900,000 IOPS per guest. However, using this approach, KVM could not reach 1 million IOPS per guest on our test setup, due to the limit of 8 PCI devices per guest imposed by Red Hat Enterprise Linux. With the virtio-blk-data-plane technology, KVM could achieve close to 1.6

million IOPS for a single guest. VMware® recently indicated that it could achieve almost 1.1 million IOPS for a single virtual machine (VM) running on a vSphere™ 5.1 host for 4KB I/O requests [3], although it used a different operating system in the VM and a different storage back-end. This means that KVM with virtio-blk-data-plane could achieve I/O rates that are **49% higher** than VMware vSphere 5.1.

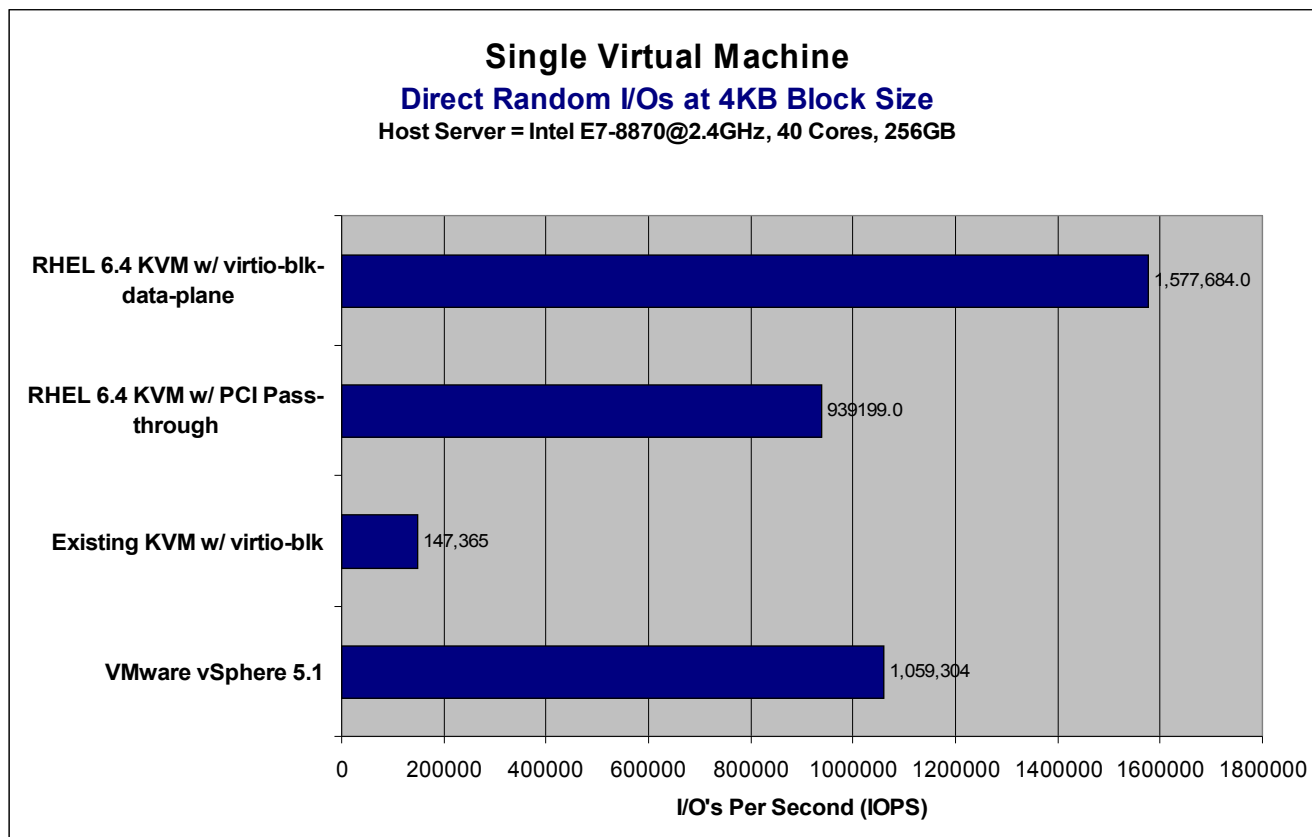


Figure 5. I/O rates at I/O request size of 4KB

In addition, KVM with virtio-blk-data-plane technology can achieve much higher I/O rates than what Microsoft has claimed for its Hyper-V hypervisor across multiple block sizes (albeit with different operating systems and hardware setups) [4]. From these data, it is quite clear that the virtio-blk-data-plane technology allows KVM to deliver much higher I/O rates than both of its major competing hypervisors.

Summary

Leveraging new silicon capabilities, with tight integration into the Linux kernel, KVM outperforms other competing hypervisor solutions in many aspects. With the newly available virtio-blk-data-plane technology for KVM in Red Hat Enterprise Linux 6.4, a single KVM guest can handle up to **1.2 million I/Os per second (IOPS)** for 8KB random I/O requests and more than **1.5 million IOPS** for smaller requests, approaching the “bare-metal” limit of the test storage setup. These I/O rates are almost **50% higher** than the highest claims by other competing hypervisors. The results in this paper have proven that KVM is ready for enterprise workloads that demand extremely high I/O rates, very low latencies, and very consistent storage I/O performance.

References

- [1] SPECvirt_sc2010 Results Published by SPEC,
http://www.spec.org/virt_sc2010/results/specvirt_sc2010_perf.html
- [2] SAP SD Standard Benchmark Application Results, Two-Tier Internet Configuration,
<http://www.sap.com/solutions/benchmark/sd2tier.epx>
- [3] VMware, Inc. “1 million IOPS on 1 VM”, August 27th, 2012.
<http://blogs.vmware.com/performance/2012/08/1millioniops-on-1vm.html>
- [4] Microsoft TechNet webcast. “Maximizing Hyper-V iSCSI Performance With Microsoft and Intel.” <https://msevents.microsoft.com/CUI/EventDetail.aspx?culture=en-US&EventID=1032432957&CountryCode=US>