KVM PERFORMANCE IMPROVEMENTS AND OPTIMIZATIONS

Mark Wagner
Principal SW Engineer, Red Hat
May 4, 2011
Overview

- Discuss a range of topics about KVM performance
  - Will show how some features impact SPECvirt results
  - Also show against real world applications
- RHEL5 and RHEL6
- Use libvirt where possible
  - Note that not all features in all releases
- We will not cover the RHEV products explicitly
  - Some stuff will apply but your mileage will vary...
Before we dive in...

Guest NFS Write Performance - are we sure?

Is this really a 10Gbit line?

By default the rtl8139 device is chosen
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
SPECvirt - Basics

- Virtual Machines are added in sets of six
  - Called a *Tile*
- Each guest has a specific function
  - Web (http)
  - Infrastructure (NFS for Web)
  - Application (Java Enterprise)
  - DB (for App)
  - Mail (imap)
  - Idle
SPECvirt - Basics

- Three SPEC workloads drive one Tile
  - SPECweb
  - SPECjApp
  - SPECmail
- Run as many Tiles as possible until *any* of the workloads fail any of the Quality of Service requirements
  - Tune, lather, rinse, repeat
SPECvirt - Basics

- Each workload is throttled
  - there are think times between requests
- SPECjApp workload has peaks/valleys to greatly vary resource usage in App & DB guests
- SPECvirt Home Page
  - http://www.spec.org/virt_sc2010/
A single QoS error invalidates the entire test

Each client runs a modified version of SPECweb, SPECjApp, and SPECmail

<table>
<thead>
<tr>
<th>Web</th>
<th>Infra</th>
<th>App</th>
<th>DB</th>
<th>Mail</th>
<th>Idle</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tile 2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tile 3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tile 4</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Tile n</td>
</tr>
</tbody>
</table>

Client
Client
Client
Client
Controller
SPECvirt_sc2010 Published Results – March 2011

SPECvirt_sc2010 2-Socket Results (x86_64 Servers) 3/2011

System

- RHEL 5.5 (KVM) / IBMx3650 M3 / 12 cores
- Vmware ESX 4.1 / HP D380 G7 / 12 cores
- RHEL 6.0 (KVM) / IBM HS22V / 12 cores
- RHEL 5.5 (KVM) / IBMx3690 X5 / 16 cores
- RHEL 6 (KVM) / IBMx3690 X5 / 16 cores

SPECvirt_sc2010 Score

SPECvirt Tiles/Core
SPECvirt_sc2010 Published Results – March 2011

SPECvirt_sc2010 2-4 Socket Results

(x86_64 Servers) 3/2011

SPECvirt_sc2010 Score

SPECvirt Tiles/Core

System

VMware ESX 4.1 / Bull SAS / 32 core
VMware ESX 4.1 / IBMx3850X5 / 32 core
VMware ESX 4.1 / HP DL580 G7 / 40 core
RHEL 6 (KVM) / IBMx3850 X5 / 64 core
SPECvirt_sc2010 Published Results – May 2011

SPECvirt_sc2010 2-Socket Results (x86_64 Servers) 5/2011

System

SPECvirt_sc2010 Score

Tiles / Core

SPECvirt_Score2010 Score

SPECvirt Tiles/Core

RHEL 5.5 (KVM) / IBMx3650 M3 / 12 cores
Vmware 4.1 / HP D380 G7 / 12 cores
ESX RHEL 6.0 (KVM) / IBM HS22V / 12 cores
RHEL 5.5 (KVM) / IBMx3690 X5 / 16 cores
RHEL 6 (KVM) / IBMx3690 X5 / 16 cores
Vmware ESX 4.1 / HP BL620c G7 / 20 cores
RHEL 6.1 (KVM) / HP BL620c G7 / 20 cores
Unofficial SPECvirt Tiles vs Tuning - RHEL5.5

Impact of Tuning KVM for SPECvirt

Not official SPECvirt Results

Based on presentation by Andrew Theurer at KVM Forum – August 2010
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Some block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Quick Overview – KVM Architecture

- Guests run as a process in userspace on the host
- A virtual CPU is implemented using a Linux thread
  - The Linux scheduler is responsible for scheduling a virtual CPU, as it is a normal thread
- Guests inherit features from the kernel
  - NUMA
  - Huge Pages
  - Support for new hardware
Quick Overview – KVM Architecture

- Disk and Network I/O through host (most of the time)
  - I/O settings in host can make a big difference in guest I/O performance
- Need to understand host buffer caching
  - Proper settings to achieve true direct I/O from the guest
  - Deadline scheduler (on host) typically gives best performance
- Network typically goes through a software bridge
Simplified view of KVM
Performance Improvements in RHEL6

- Performance enhancements in every component

<table>
<thead>
<tr>
<th>Component</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU/Kernel</td>
<td>NUMA – Ticketed spinlocks; Completely fair scheduler; Extensive use of Read Copy Update (RCU) Scales up to 64 vcpus per guest</td>
</tr>
<tr>
<td>Memory</td>
<td>Large memory optimizations: Transparent Huge Pages is ideal for hardware based virtualization</td>
</tr>
<tr>
<td>Networking</td>
<td>Vhost-net – a kernel based virtio w/ better throughput and latency. SRIOV for ~native performance</td>
</tr>
<tr>
<td>Block</td>
<td>AIO, MSI, scatter gather.</td>
</tr>
</tbody>
</table>
Performance Enhancements

• Vhost-net -
  • new host kernel networking backend providing superior throughput and latency over the prior userspace implementation

• FPU performance optimization
  • Avoid the need for host to trap guest FPU cr0 access
Performance Enhancements

• Disk I/O latency & throughput improvements,
  • using ioeventfd for faster notifications
• Qcow2 virt image format caching of metadata improves performance
  • batches writes upon I/O barrier, rather than fsync every time additional block storage needed (thin provisioning growth)
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Some block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Remember this?

Guest NFS Write Performance

Impact of not specifying OS at guest creation
Be Specific!

- virt-manager will:
  - Make sure the guest will function
  - Optimize as it can
  - The more info you provide the more tailoring will happen

Specify the OS details
Specify OS + flavor

- Specifying Linux / Red Hat will get you:
  - The virtio driver
  - Unless you are at RHEL6.1
    - vhost_net drivers
I Like This Much Better

Guest NFS Write Performance

Impact of specifying OS Type at Creation

Throughput (MB/second)

RHEL6-default

RHEL6-vhost

RHEL5-virtio

12.5 x
Low Hanging Fruit

- Remove unused devices
- Do you need sound in your web server?
- Remove / disable unnecessary services?
- Both host and guest
- Bluetooth in a guest?
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Some block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Memory Enhancements for Virtualization

- Extended Page Table (EPT) age bits
- Kernel Same-page Merging (KSM)
- Transparent Huge Pages
Memory Enhancements for Virtualization

- Extended Page Table (EPT) age bits
  - Allow host to make smarter swap choice when under pressure.
- Kernel Same-page Merging (KSM)
  - Consolidate duplicate pages.
  - Particularly efficient for Windows guests.
- Transparent Huge Pages
  - Efficiently manage large memory allocations as one unit
Memory sharing

- KSM
  - Scan into (transparent) huge pages
    - Help delay/avoid swapping
    - At cost of slower memory access
    - That's the major issue – to make sure THP won't reduce the memory over commit capacity of the host
  - Classical speed vs. space trade off
    - Throughput vs. density
  - Add KSM on/off per VM
Memory Tuning – Huge Pages

- 2M pages vs 4K standard Linux page
  - Virtual to physical page map is 512 times smaller
  - TLB can map more physical page resulting fewer misses
- Traditional Huge Pages always pinned
- Transparent Huge Pages in RHEL6
- Most databases support Huge Pages
- How to configure Huge Pages (16G)
  - echo 8192 > /proc/sys/vm/nr_hugepages
  - vi /etc/sysctl.conf (vm.nr_hugepages=8192)
Memory Tuning – Huge Pages

- Benefits not only Host but guests
  - Try them in a guest too!
Try Huge Pages on Host and Guest

Impact of Huge Pages on SPECjbb

RHEL5.5 - 8 threads

- No huge pages: 24%
- Host using huge pages: 46%
- Guest & host using huge pages: 46%
Impact of Huge Pages

KVM guest runs with and without Huge Pages

RHEL5.5 – KVM - OLTP Workload

Number of users (x 100)

Aggregate Transactions / Minute

10U 20U 40U

1 Guest 1 Guest – Huge pages 4 Guests 4 Guests – Huge pages
Transparent Huge Pages

RHEL6/6.1 SPECjbb

24-cpu, 24 vcpu Westmere EP, 24GB

Transactions Per Minute

No-THP

THP

r6-guest

r6-metal
Impact of Tuning KVM for SPECvirt

Not official SPECvirt Results

10%

Based on presentation by Andrew Theurer at KVM Forum – August 2010
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- **Networking**
- Some block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Virtualization Tuning - Network

- General Tips
- Virtio
- vhost_net
- PCI passthrough
- SR-IOV (Single root I/O Virtualization)
Network Tuning Tips

- Separate networks for different functions
  - Use arp_filter to prevent ARP Flux
    - echo 1 > /proc/sys/net/ipv4/conf/all/arp_filter
    - Use /etc/sysctl.conf for permanent
  - Don't need HW to bridge intra-box communications
    - VM traffic never hits the HW on same box
    - Can really kick up MTU as needed
- Packet size
  - Need to make sure it is set across all components
Virtualization Tuning - Network

- **Virtio**
  - VirtIO drivers for network

- **vhost_net (low latency – close to line speed)**
  - Bypass the qemu layer

- **PCI passthrough**
  - Bypass the host and pass the PCI device to the guest
  - Can be passed only to one guest

- **SR-IOV (Single root I/O Virtualization)**
  - Can be shared among multiple guests
  - Limited hardware support
  - Pass through to the guest
KVM Network Architecture - VirtIO

- Virtual Machine sees paravirtualized network device – VirtIO
  - VirtIO drivers included in Linux Kernel
  - VirtIO drivers available for Windows
- Network stack implemented in userspace
KVM Network Architecture

Virtio

Virtual Machine

QEMU

TAP

Kernel

Bridge

TX  RX
Latency comparison – RHEL 6

Network Latency virtio

Guest Receive (Lower is better)

4X gap in latency
KVM Network Architecture – vhost_net

- New in RHEL6.1
- Moves QEMU network stack from userspace to kernel
- Improved performance
- Lower Latency
- Reduced context switching
- One less copy
KVM Network Architecture – vhost_net
Latency comparison – RHEL 6

Network Latency - vhost_net
Guest Receive (Lower is better)

Latency (usecs)

Message Size (Bytes)

Latency much closer to bare metal
KVM Network Architecture – VirtIO vs vhost_net

Virtio

VirtIO

vhost_net

vhost_net
Host CPU Consumption virtio vs vhost_net

Host CPU Consumption, virtio vs Vhost

Major difference is usr time

Message Size (Bytes)
vhost_net Efficiency

8 Guest Scale Out RX Vhost vs Virtio - % Host CPU

Mbit per % CPU netperf TCP_STREAM

Message Size (Bytes)
KVM Network Architecture – PCI Device Assignment

- Physical NIC is passed directly to guest
- Guest sees real physical device
  - Needs physical device driver
- Requires hardware support
  - Intel VT-D or AMD IOMMU
- Lose hardware independence
- 1:1 mapping of NIC to Guest
- BTW - This also works on some I/O controllers
KVM Network Architecture – Device Assignment

Device Assignment

Virtual Machine

QEMU

Kernel

Physical NIC

TX

RX
KVM Network Architecture – SR-IOV

- Single Root I/O Virtualization
  New class of PCI devices that present multiple virtual devices that appear as regular PCI devices
- Guest sees real physical device
  - Needs physical device driver
- Requires hardware support
- Low overhead, high throughput
- No live migration
- Lose hardware independence
KVM Architecture – SR-IOV

SR-IOV

Virtual Machine

QEMU

Kernel

Virtual NICs

Physical NIC

Virtual NICs
Latency comparison – RHEL 6 based methods

Network Latency by guest interface method

Guest Receive (Lower is better)

Message Size (Bytes)

SR-IOV latency close to bare metal
RHEL 6 KVM w/ SR-IOV Intel Niantic 10Gb Postgres DB

**DVD Store Version 2 results**

<table>
<thead>
<tr>
<th>Throughput in Order/min (OPM)</th>
<th>1 Red Hat KVM bridged guest</th>
<th>1 Red Hat KVM SR-IOV guest</th>
<th>1 database instance (bare metal)</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>69,984</td>
<td>86,469</td>
<td>92,680</td>
</tr>
</tbody>
</table>

93% Bare Metal
Unofficial SPECvirt Tiles vs Tuning - RHEL5.5

Impact of Tuning KVM for SPECvirt

Not official SPECvirt Results

Based on presentation by Andrew Theurer at KVM Forum – August 2010
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Block

• Recent improvements
• Tuning hardware
• Choosing an elevator
• Choosing the caching model
• Tuned / ktune
• Device assignment
Block Improvements

- loeventd performance enhancement
- 4K sector size (RHEL6)
- Windows drivers
  - MSI support
- AIO implementation
I/O Tuning - Hardware

• Know your Storage
  • SAS or SATA?
  • Fibre Channel, Ethernet or SSD?
  • Bandwidth limits
• Multiple HBAs
  • Device-mapper-multipath
  • Provides multipathing capabilities and LUN persistence
• How to test
  • Low level I/O tools – dd, iozone, dt, etc
I/O Tuning – Understanding I/O Elevators

- I/O Elevators of Interest
  - Deadline
  - CFQ
  - Noop

- Use deadline on the host
  - Your mileage may vary
  - Experiment!
I/O Tuning – Understanding I/O Elevators

- **Deadline**
  - Two queues per device, one for read and one for writes
  - IOs dispatched based on time spent in queue

- **CFQ**
  - Per process queue
  - Each process queue gets fixed time slice (based on process priority)

- **Noop**
  - FIFO
  - Simple I/O Merging
  - Lowest CPU Cost
I/O Tuning – How to configure I/O Elevators

- Boot-time
  - Grub command line – elevator=deadline/cfq/noop
- Dynamically – per device
  - echo “deadline” > /sys/class/block/sda/queue/scheduler
Virtualization Tuning – I/O elevators - OLTP

Performance Impact of I/O Elevators on OLTP Workload

Host running Deadline Scheduler

- Noop
- CFQ
- Deadline

Transactions per Minute

1 Guest | 2 Guests | 4 Guests

K | 50K | 100K | 150K | 200K | 250K | 300K

SUMMIT JBoss WORLD
PRESENTED BY RED HAT
Virtualization Tuning – I/O Cache

• Three Types
  • Cache=none
  • Cache=writethrough
  • Cache=writeback - Not supported
Virtualization Tuning - Caching

- **Cache=none**
  - I/O from the guest in not cached

- **Cache=writethrough**
  - I/O from the guest is cached and written through on the host
  - Potential scaling problems with this option with multiple guests (host cpu used to maintain cache)

- **Cache=writeback - Not supported**
Virtualization Tuning - Caching

- Configure I/O-Cache per disk in qemu command line or libvirt
  - Virt-manager – drop-down option under “Advanced Options”
  - Libvirt XML file - driver name='qemu' type='raw' cache='none' io='native'
Effect of I/O Cache settings on Guest performance

OLTP like workload

FusionIO storage

Transaction Per Minute

1Guest

4Guests

Cache=WT

Cache=none
I/O Tuning - Filesystems

- RHEL6 introduced barriers
  - Needed for data integrity
  - On by default
  - Can disable on Enterprise class storage
- Configure read ahead
  - Database (parameters to configure read ahead)
  - Block devices (getra, setra)
- Asynchronous I/O
  - Eliminate Synchronous I/O stall
  - Critical for I/O intensive applications
AIO – Native vs Threaded (default)

Impact of AIO selection on OLTP Workload
"cache=none" setting used - Threaded is default

Configurable per device (only by xml configuration file)
Libvirt xml file - driver name='qemu' type='raw' cache='none' io='native'
RHEL6 “tuned-adm” profiles

# tuned-adm list

Available profiles:
- default
- latency-performance
- throughput-performance
- enterprise-storage

Example
  # tuned-adm profile enterprise-storage

Recommend “enterprise-storage” w/ KVM
RHEL6 “tuned-adm” profiles -default

- default
  - CFQ elevator (cgroup)
  - I/O barriers on
  - ondemand power savings
  - upstream VM
  - 4 msec quantum
- Example
  # tuned-adm profile default
RHEL6 “tuned-adm” profiles - latency-performance

- latency-performance
  - elevator=deadline
  - power=performance

- Example
  # tuned-adm profile latency-performance
RHEL6 “tuned-adm” profiles - throughput-performance

- throughput-performance
  - latency + 10 msec quantum
  - readahead 4x
  - VM dirty_ratio=40
- Example
  
  # tuned-adm profile throughput-performance
Remember Network Device Assignment?

- Device Assignment
  - It works for Block too!
  - Device Specific
  - Similar Benefits
  - And drawbacks...
• Block Device Passthrough - SAS Workload

RHEL6.1 SAS Mixed Analytics Workload - Metal/KVM

Intel Westmere EP 12-core, 24 GB Mem, LSI 16 SAS

- KVM VirtIO
- KVM/PCI-PassThrough
- Bare-Metal

- 79%
- 94%

Time to complete (secs)
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
NUMA (Non Uniform Memory Access)

- Multi Socket – Multi core architecture
  - NUMA is needed for scaling
  - RHEL 5 / 6 completely NUMA aware
  - Additional performance gains by enforcing NUMA placement

- How to enforce NUMA placement
  - numactl – CPU and memory pinning
Memory Tuning - NUMA

# numactl --hardware
available: 8 nodes (0-7)
ode 0 cpus: 0 1 2 3 4 5
node 0 size: 8189 MB
node 0 free: 7220 MB
node 1 cpus: 6 7 8 9 10 11
node 1 size: 8192 MB
...
node 7 cpus: 42 43 44 45 46 47
node 7 size: 8192 MB
node 7 free: 7816 MB

node distances:
node distances:

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
<th>7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>10</td>
<td>16</td>
<td>16</td>
<td>22</td>
<td>16</td>
<td>22</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>1</td>
<td>16</td>
<td>10</td>
<td>22</td>
<td>16</td>
<td>16</td>
<td>22</td>
<td>22</td>
<td>16</td>
</tr>
<tr>
<td>2</td>
<td>16</td>
<td>22</td>
<td>10</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>3</td>
<td>22</td>
<td>16</td>
<td>16</td>
<td>10</td>
<td>16</td>
<td>16</td>
<td>22</td>
<td>22</td>
</tr>
<tr>
<td>4</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>10</td>
<td>16</td>
<td>16</td>
<td>22</td>
</tr>
<tr>
<td>5</td>
<td>22</td>
<td>22</td>
<td>16</td>
<td>16</td>
<td>16</td>
<td>10</td>
<td>22</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>16</td>
<td>22</td>
<td>16</td>
<td>22</td>
<td>16</td>
<td>22</td>
<td>16</td>
<td>16</td>
</tr>
<tr>
<td>7</td>
<td>22</td>
<td>16</td>
<td>16</td>
<td>22</td>
<td>22</td>
<td>16</td>
<td>16</td>
<td>10</td>
</tr>
</tbody>
</table>

Internode Memory distance From SLIT table

Note variation in internode distances
NUMA and Huge Pages -Not an issue with THP

- Static Huge Page allocation takes place uniformly across NUMA nodes
  - Make sure that guests are sized to fit
- Workaround 1 – Use Transparent Huge Pages
- Workaround 2 – Allocate Huge pages / Start Guest / De-allocate Huge pages
OLTP Workload – Effect of NUMA and Huge Pages

OLTP workload - Multi Instance

Effect of NUMA: 8%
OLTP Workload – Effect of NUMA and Huge Pages

OLTP workload - Multi Instance

Effect of NUMA and Huge Pages

- Non NUMA: 8%
- NUMA: 12%
- non NUMA + Huge Pages: 18%
- NUMA + Huge Pages: 18%
Comparison between HugePages vs HugePages + affinity
Four RHEL5.5 Guests using libvirt

Huge Pages Leveled off
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
CPU Performance

- Improvements
- CPU Type
- CPU Topology
- CPU Pin - Affinity
CPU Improvements

- RCU for the KVM module locks
  - we scale to 64 vcpus!
- Dynamic cancel ticket spinlocks
- Add user return notifiers in the kernel
- X2apic
  - Use MSR access to limit mmio accesses to the irq chip
Specifying Processor Details

- Mixed results with CPU type and topology
- Experiment and see what works best in your case
CPU Pinning - Affinity

- Virt-manager allows selection based on NUMA topology
  - True NUMA support in the works
- Virsh pinning allows finer grain control
  - 1:1 pinning
- Good gains with pinning
Impact of Tuning KVM for SPECvirt

Not official SPECvirt Results

Based on presentation by Andrew Theurer at KVM Forum – August 2010
Performance monitoring tools

- Monitoring tools
  - `top, vmstat, ps, iostat, netstat, sar, perf`
- Kernel tools
  - `/proc, sysctl, AltSysrq`
- Networking
  - `ethtool, ifconfig`
- Profiling
  - `oprofile, strace, ltrace, systemtap, perf`
Agenda

- What is SPECvirt
- A quick, high level overview of KVM
- Low Hanging Fruit
- Memory
- Networking
- Block I/O basics
- NUMA and affinity settings
- CPU Settings
- Wrap up
Wrap up

- KVM can be tuned effectively
  - Understand what is going on under the covers
  - Turn off stuff you don't need
  - Be specific when you create your guest
  - Look at using NUMA or affinity
  - Choose appropriate elevators (Deadline vs CFQ)
  - Choose your cache wisely
For More Information – Other talks

- Campground 2 – Thurs 10:20
- Performance Analysis & Tuning of Red Hat Enterprise Linux – Shak and Larry
  - Part 1 - Thurs 10:20
  - Part 2 - Thurs 11:30
- Tuning Your Red Hat System for Databases
  - Sanjay Rao - Fri 9:45
- KVM Performance Optimizations
  - Rik van Riel - Thurs 4:20
For More Information

- KVM Wiki
  - http://www.linux-kvm.org/page/Main_Page
- irc, email lists, etc
  - http://www.linux-kvm.org/page/Lists%2C_IRC
- libvirt Wiki
  - http://libvirt.org/
- New, revamped edition of the “Virtualization Guide”
  - Should be available soon !
For More Information

- Reference Architecture Website
  - https://access.redhat.com/knowledge/refarch/TBD
- Andrew Theurers' original presentation
- SPECvirt Home Page
  - http://www.spec.org/virt_sc2010/
- Principled Technologies
SPECvirt_sc2010 2-Socket Results (x86_64 Servers) 5/2011

- RHEL 5.5 (KVM) / IBMx3650 M3 / 12 cores - SPECvirt_sc2010 Score: 1169
- Vmware 4.1 / HP D380 G7 / 12 cores - SPECvirt Tiles/Core: 1.2
- ESX 6.0 (KVM) / IBM HS22V / 12 cores - SPECvirt_sc2010 Score: 369
- RHEL 6.0 (KVM) / IBMx3690 X5 / 16 cores - SPECvirt Tiles/Core: 1.0
- RHEL 5.5 (KVM) / IBMx3690 X5 / 16 cores - SPECvirt_sc2010 Score: 1763
- RHEL 6 (KVM) / IBMx3690 X5 / 16 cores - SPECvirt Tiles/Core: 0.8
- Vmware ESX 4.1 / HP BL620c G7 / 20 cores - SPECvirt_sc2010 Score: 1221
- RHEL6.1 (KVM) / HP BL620c G7 / 20 cores - SPECvirt Tiles/Core: 1.0
SPECvirt_sc2010 Published Results – May 2011

SPECvirt_sc2010 2-4 Socket Results

(x86_64 Servers) 4/2011

SPECvirt_sc2010 Score

SPECvirt Tiles/Core

System

SPECvirt_sc2010 Score

Tiles / core