KVM PERFORMANCE IMPROVEMENTS AND OPTIMIZATIONS

Mark Wagner
Principal SW Engineer, Red Hat
August 14, 2011
Overview

• Discuss a range of topics about KVM performance
  – How to improve out of the box experience
  – But crammed into 30 minutes
• Use libvirt where possible
  – Note that not all features in all releases
Before we dive in...

**Guest NFS Write Performance - are we sure?**

Is this really a 10Gbit line?

By default the rtl8139 device is chosen

Arrow shows improvement
Agenda

• Low hanging fruit
• Memory
• Networking
• Block I/O basics
• NUMA and affinity settings
• CPU Settings
• Wrap up
Recent Performance Improvements

- **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>
Remember this?

Guest NFS Write Performance

Impact of not specifying OS at guest creation

Throughput (MBytes / second)
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 will get you:
  – The virtio driver
  – If the kernel is recent enough the vhost_net drivers
I Like This Much Better

Guest NFS Write Performance
Impact of specifying OS Type at Creation

Throughput (MB/second)

Default
vhost
virtio

12.5 x
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
- We now have Transparent Huge Pages
- Most databases support Huge Pages
- Benefits not only Host but guests
  - Try them in a guest too!
Transparent Huge Pages

SPECjbb workload
24-cpu, 24 vcpu Westmere EP, 24GB

Transactions Per Minute

No-THP

THP

30%

25%

guest

bare metal
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

• Packet size - MTU
  – Need to make sure it is set across all components

• 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
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
Virtio

Context switch host kernel <-> userspace
Latency comparison

Network Latency virtio
Guest Receive (Lower is better)

4X gap in latency

Message Size (Bytes)

Latency (usecs)
KVM Network Architecture – vhost_net

- Moves QEMU network stack from userspace to kernel
- Improved performance
- Lower Latency
- Reduced context switching
- One less copy
Latency comparison

Network Latency - vhost_net
Guest Receive (Lower is better)

Latency much closer to bare metal

Message Size (Bytes)
Host CPU Consumption, virtio vs Vhost

8 Guests TCP Receive

Message Size (Bytes)

% Total Host CPU (Lower is Better)

Two columns is a data set

Major difference is usur time

%usr
%soft
%guest
%sys
vhost_net Efficiency

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

Mbit per % CPU netperf TCP_STREAM

Message Size (Bytes)

Mbit / % CPU (bigger is better)

Vhost

Virtio
KVM Architecture – Device Assignment vs SR/IOV

Device Assignment

SR-IOV
KVM Network Architecture – PCI Device Assignment

• Physical NIC is passed directly to guest
  – Device is not available to anything else on the host
• 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 – 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 (virtual) device driver
- Requires hardware support
- Actual device can still be shared
- Low overhead, high throughput
- No live migration – well its difficult
- Lose hardware independence
Latency comparison

Network Latency by guest interface method

Guest Receive (Lower is better)

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

![DVDStore Version 2 results](chart.png)

Throughput in Order/min (OPM)

- **1 Red Hat KVM bridged guest**: 69,984
- **1 Red Hat KVM SR-IOV guest**: 86,469
- **1 database instance (bare metal)**: 92,680

- **76% Bare Metal**
- **93% Bare Metal**
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

• 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

• Can set at Boot-time
  − Grub command line – elevator=deadline/cfq/noop

• Or 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
Virtualization Tuning - Caching

• Cache=none
  – I/O from the guest is 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
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

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

Transactions Per Minute

<table>
<thead>
<tr>
<th>Number of Users (x 100)</th>
<th>AIO Threaded</th>
<th>AIO Native</th>
</tr>
</thead>
<tbody>
<tr>
<td>10U</td>
<td>700K</td>
<td>600K</td>
</tr>
<tr>
<td>20U</td>
<td>900K</td>
<td>800K</td>
</tr>
</tbody>
</table>

Configurable per device (only by xml configuration file)
Libvirt xml file - driver name='qemu' type='raw' cache='none' io='native'
Remember Network Device Assignment?

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

**SAS Mixed Analytics Workload - Metal/KVM**

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

![Graph showing performance comparison]

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

- **Time to complete (secs)**
- **SAS system**
- **SAS Total**

- 25% longer
- 6% longer
NUMA (Non Uniform Memory Access)

- Multi Socket – Multi core architecture
  - NUMA is needed for scaling
    - Keep memory latencies low
  - Linux completely NUMA aware
  - Additional performance gains by enforcing NUMA placement
  - Still some “out of the box” work is needed

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

- One way to test if you get a gain is to **mistune** it.
- Libvirt now supports some NUMA placement
Memory Tuning - NUMA

```bash
# numactl --hardware
available: 8 nodes (0-7)
node 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   0   1   2   3   4   5   6   7
0:  10  16  16  22  16  22  16  22
1:  16  10  22  16  16  22  22  16
2:  16  22  10  16  16  16  16  16
3:  22  16  16  10  16  16  22  22
4:  16  16  16  16  10  16  16  22
5:  22  22  16  16  16  10  22  16
6:  16  22  16  22  16  22  10  16
7:  22  16  16  22  22  16  16  10
```

Internode Memory distance
From SLIT table

Note variation in internode distances
Virtualization Tuning – Using NUMA

Impact of NUMA in multiguest OLTP

location, location, location

<table>
<thead>
<tr>
<th>Transactions Per Second</th>
</tr>
</thead>
<tbody>
<tr>
<td>400K</td>
</tr>
<tr>
<td>350K</td>
</tr>
<tr>
<td>300K</td>
</tr>
<tr>
<td>250K</td>
</tr>
<tr>
<td>200K</td>
</tr>
<tr>
<td>150K</td>
</tr>
<tr>
<td>100K</td>
</tr>
<tr>
<td>50K</td>
</tr>
<tr>
<td>0</td>
</tr>
</tbody>
</table>

4Guest-24vcpu-56G       4Guest-24vcpu-56G-NUMA

- Guest 4
- Guest 3
- Guest 2
- Guest 1
Specifying Processor Details

- Mixed results with CPU type and topology
- The Red Hat team is still exploring some topology performance quirks
  - Both model and topology
- Experiment and see what works best in your case
CPU Pinning - Affinity

- Virt-manager allows CPU selection based on NUMA topology
  - True NUMA support in libvirt
- Virsh pinning allows finer grain control
  - 1:1 pinning
- Good gains with pinning
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
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

• 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!