Megasas and VFIO
PCI device-assignment with qemu

Dr. Hannes Reinecke
SUSE Labs
hare@suse.de
PCI device-assignment

• Various I/O methods:
  - Emulated devices (qemu)
  - Virtual devices (virtio)
  - Accelerated virtio (vhost)
  - Direct access to hardware
Emulated devices

Guest userland
- Block layer
- SCSI / megaraid_sas
- Qemu megasas
- Qemu block

Guest kernel

Host qemu process

Host kernel
- Block layer
- SCSI / megaraid_sas

Hardware
- LSI HBA
Virtio: Efficient device interface

Guest userland

Block layer
virtio
Qemu virtio
Qemu block

Guest kernel

Host qemu process

Block layer
SCSI / megaraid_sas

Host kernel

Hardware

LSI HBA
Vhost: in-kernel I/O pass-through
SR-IOV: PCI device assignment

- Guest userland
- Block layer
- SCSI / megaraid_sas
- Guest kernel
- Host qemu process
- pci-vfio
- Host kernel
- IOMMU
- LSI HBA
- Hardware

Host kernel
Logical partitioning with Qemu
PCI device assignment

• Direct access to hardware:
  - Individual PCI devices are assigned to a guest
  - Guest can use unmodified drivers

• Prevent host access to assigned devices
  - pci-stub
PCI device assignment

- Guest and host have a different memory mapping
- DMA addresses need to be translated
- Hardware support needed
- Interrupts might need to be remapped
- VFIO
LPAR guest

- Create a guest with just VFIO devices
- No emulation
- Simple commandline:

```bash
# qemu-system-x86_64 -enable-kvm -net none \ 
  -device vfio-pci,host=01:10.0,id=igbvf0 \ 
  -device vfio-pci,host=07:00.0,id=megasas0 -m 1024
```
LPAR guest

RAID Controller BIOS Version 3.06.00 (Build March 25, 2009)

HA -0 (Bus 0 Dev 4) Intel (R) RAID Controller RSZBL080
Battery Status: Not present

<table>
<thead>
<tr>
<th>PCI SLOT ID</th>
<th>LUN VENDOR</th>
<th>PRODUCT</th>
<th>REVISION</th>
<th>CAPACITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>INTEL</td>
<td>Intel (R) RAID Controller</td>
<td>0003</td>
<td>512MB</td>
</tr>
<tr>
<td>3</td>
<td>10 0</td>
<td>SEAGATE ST9146802SS</td>
<td>0003</td>
<td>140014MB</td>
</tr>
<tr>
<td>3</td>
<td>11 0</td>
<td>SEAGATE ST9146802SS</td>
<td>0003</td>
<td>140014MB</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>INTEL Virtual Drive</td>
<td>RAID0</td>
<td>270472MB</td>
</tr>
</tbody>
</table>

1 Virtual Drive(s) found on the host adapter.

1 Virtual Drive(s) handled by BIOS
Press <Ctrl><Y> for Preboot CLI _

<Ctrl><G> to enter the RAID BIOS Console <<<<<<<< 2009 LSI Corporation. All rights reserved !
Performance measurement
Measurement goals

- Compare different emulation methods
- Measure emulation overhead
- Identify possible bottlenecks
- Identify areas of improvement
Testcases

• Test 5 different cases:
  - Megasas (IOMMU enabled)
  - Megasas (IOMMU disabled)
  - Virtio-scsi (IOMMU enabled)
  - Virtio-scsi (IOMMU disabled)
  - VFIO

• Using same hardware for each test
• No modifications to host or guest installation
Test platform

• Intel development platform
• 4-socket 10-core Xeon
• Integrated LSI megaraid HBA
• 128 GB RAM
• Using mmtest / tiobench to generate test results
I/O Throughput results
I/O throughput (seq read)

Abs. Throughput (seq read)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O throughput (seq read)

Rel. Throughput (seq read)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O throughput (seq write)

Abs. Throughput (seq write)

virtio (no IOMMU)
virtio (IOMMU)
megasas (no IOMMU)
megasas (IOMMU)
vfio (IOMMU)
I/O throughput (seq write)

Rel. Throughput (seq write)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O throughput (rand read)

Abs. Throughput (rand read)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)

MB/s vs. # of procs
I/O throughput (rand read)

Rel. Throughput (rand read)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O throughput (rand write)

Abs. Throughput (rand write)

virtio (no IOMMU)
virtio (IOMMU)
megasas (no IOMMU)
megasas (IOMMU)
vfio (IOMMU)

MB/s

# of procs
I/O throughput (rand write)

Rel. Throughput (rand write)

virtio (no IOMMU)
virtio (IOMMU)
megasas (no IOMMU)
megasas (IOMMU)
vfio (IOMMU)
I/O Latency results
I/O latency (seq read)

Avg. Latency (seq read)

virtio (no IOMMU)
virtio (IOMMU)
megasas (no IOMMU)
megasas (IOMMU)
vfio (IOMMU)

ms

# of procs
I/O latency (seq read)
I/O latency (seq write)

Avg. Latency (seq write)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O latency (seq write)

Rel. Latency (seq write)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)
I/O latency (rand read)

Avg. Latency (rand read)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)

# of procs
I/O latency (rand read)
I/O latency (rand write)

Avg. Latency (rand write)

- virtio (no IOMMU)
- virtio (IOMMU)
- megasas (no IOMMU)
- megasas (IOMMU)
- vfio (IOMMU)

ms vs. # of procs
Results

• All methods yield nearly identical results (+/- 5%)
• VFIO is not the fastest I/O path
• Random I/O significantly slower than sequential
• 'sweet spot' at 8 concurrent processes; most likely hardware related (8-core processors)
• Random write latency significantly lower than random read; caching?
This document could include technical inaccuracies or typographical errors. Changes are periodically made to the information herein. These changes may be incorporated in new editions of this document. SUSE may make improvements in or changes to the software described in this document at any time.

Copyright © 2011 Novell, Inc. All rights reserved.

All SUSE marks referenced in this presentation are trademarks or registered trademarks of Novell, Inc. in the United States. All third-party trademarks are the property of their respective owners.