Accelerating Virtual Machine Storage I/O for Multicore Systems
Dongwoo Lee, Junghan Kim, Junghoon Kim, Changwoo Min, Young Ik Eom

Motivation and Goal

I/O workloads in VM suffer performance degradation due to virtualization overhead.

<table>
<thead>
<tr>
<th>I/O Request Cost</th>
<th>Device Emulation Cost</th>
<th>Completion Notification Cost</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct Cost</td>
<td>VM-to-VMM world switching (Issue I/O requests to VMM)</td>
<td>Domain or user-kernel mode switching (Emulate requests in a separated domain or process)</td>
</tr>
<tr>
<td>Indirect Cost</td>
<td>Cache pollution TLB flush</td>
<td>Cache pollution TLB flush</td>
</tr>
<tr>
<td>Synchronous Cost</td>
<td>ALL processes on a VM are stopped.</td>
<td>A process on a CPU is stopped.</td>
</tr>
</tbody>
</table>

Goal: Accelerating I/O performance in VM by reducing the cost induced by exits

Key Techniques

Reducing the I/O virtualization overhead by exploiting multicore architecture

Exitless I/O Request
- Using para-virtualized device driver that communicates with VMM through shared request queue.
- Issuing an I/O request is just enqueueing a request in the shared queue.
- IOCore thread checks if there are new requests and dequeues them for further processing.

In-VMM Device Emulation
- IOCore thread passes the dequeued request to the device emulation thread.
- Each emulated device has its own emulation thread.
- Each device thread handles the I/O requests for the corresponding device.

Exitless Completion Notification
- IDT in guest is shadowed and configured to directly deliver IPI to the guest OS.
- Configuring x2APIC to directly expose EOI register.
- Guest OS writes the EOI registers without exits when guest completes interrupt handling.

Experimental Results

Sequential reads and random reads on high-end SSD

<table>
<thead>
<tr>
<th></th>
<th>KVM-QEMU</th>
<th>KVM-virtio</th>
<th>In-VMM</th>
<th>In-VMM + IOCore</th>
</tr>
</thead>
<tbody>
<tr>
<td>Sequential Read</td>
<td>58%</td>
<td>92%</td>
<td>94%</td>
<td>98%</td>
</tr>
<tr>
<td>Random Read</td>
<td>46%</td>
<td>71%</td>
<td>82%</td>
<td>95%</td>
</tr>
</tbody>
</table>

Our approach highly close to the bare-metal performance
Seq. Reads by 98% and Rnd. Reads by 95%
In-VMM device emulation benefit: 2~11%
Exitless I/O Request benefit: 4~13%

Considering our prototype is not optimized and exitless completion notification is not implemented yet, the performance result is quite encouraging.

For Further Information, please contact Dongwoo Lee (dwlee@ece.skku.ac.kr)