Managing Resources on Overcommitted Virtualization Hosts

Adam Litke <agl@us.ibm.com>
IBM Corporation
Memory Overcommitment Today

- Linux is designed to over-commit process memory
  - Virtual memory and demand paging
  - Page caching and sharing
  - Swap
- KVM guests are still processes but they are different
  - Long running with variable resource requirements
  - Static resource allocations are often over-provisioned
  - Host and guest are managing the same memory
- Virtualization tools: KSM, memory ballooning, etc
- Modest overcommitment possible
Improving Memory Overcommitment

• **Real-time tuning**
  - KSM and Memory ballooning require external control
  - Optimal settings require host and guest statistics
  - ksmtuned is the perfect example of this

• **Manage interactions**
  - Interference: Ballooning decreases KSM effectiveness
  - Side-effects: Ballooning can increase I/O load

• **Flexibility**
  - Diverse configuration scenarios
  - Evolving overcommitment management techniques
  - Density Vs. Performance trade-off
Memory Overcommitment Manager

- Guest tracking
- Host and guest statistics collection
- Policy engine
- Control KSM and memory ballooning
- Policies can be customized
MOM Policy Format

- Lightweight policy language
- Access to stats and controls through simple variables
- Functions, conditionals, variables, constants, math
- No looping (except built-in guest iteration)
- Currently Python-based but this may change

```python
host_free_percent = Host.StatAvg('mem_free') / Host.mem_available
if host_free_percent < pressure_threshold:
    # We are under memory pressure
    for_each_guest(shrink_guest)
else:
    # We are not under memory pressure
    for_each_guest(grow_guest)
```
MOM Policy: Memory Ballooning

- Under pressure, guests should swap, not host
- Incremental balloon adjustments

<table>
<thead>
<tr>
<th>Host memory pressure</th>
<th>Take this action ...</th>
</tr>
</thead>
<tbody>
<tr>
<td>Moderate</td>
<td>Inflate balloons. Guests retain some free memory.</td>
</tr>
<tr>
<td>Severe</td>
<td>Inflate balloons more. This will cause cache pressure and guest swapping.</td>
</tr>
<tr>
<td>Low</td>
<td>Deflate balloons. Gradually return guests to full size.</td>
</tr>
</tbody>
</table>
MOM Policy: KSM

- Run ksmd only when necessary to reduce overhead:
  - When free memory is low
  - When memory committed to virtualization is high
- Dynamic adjustment of scanning behavior
  - Frequency is proportional to total memory size
  - Duration is proportional to level of memory pressure
Workload #1: Memknobs

- A simple C program is run in each guest.
- Allocates a large buffer of anonymous memory and touches pages in a loop to create memory pressure.
- Memknobs parameters are varied across 32 guests to create a variable, memory intensive workload.
Workload #2: Cloudy

- New open LAMP virtualization benchmark
- Each guest is a standalone MediaWiki instance
  - Actual Wikipedia content
  - Random image data
- A JMeter test plan exercises all instances and provides quality of service metrics
  - Total request throughput
  - 95th percentile request duration
- Cloudy is I/O intensive
Results: Memknobs

- Ballooning redirected swapping to the guests which increased throughput by 20%
- KSM was not a factor
Results: Cloudy

- Policy had no effect on throughput or QOS
- Negligible swap activity
- Ballooning caused cache pressure and an increase in I/O

<table>
<thead>
<tr>
<th># of VMs</th>
<th>MOM Policy</th>
<th>QOS</th>
<th>Throughput</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>No</td>
<td>1669</td>
<td>710007</td>
</tr>
<tr>
<td>32</td>
<td>No</td>
<td>3240</td>
<td>774555</td>
</tr>
<tr>
<td>32</td>
<td>Yes</td>
<td>3231</td>
<td>764762</td>
</tr>
</tbody>
</table>

![Graph showing memory size and disk utilization over time]
The Future

• Policy research and improvements
  - There is no “One size fits all” policy
  - Increase applicability of the default policy
  - Safeguards to avoid performance degradation

• Support additional overcommitment technologies
  - Cgroups for hard guest RSS limits
  - Host / guest page cache control
  - Swap tuning / Compcache
  - Follow other developments in this community
The Future

- Standardized host ↔ guest communication
  - Notably missing from KVM virtualization
  - Needed for guest statistics collection
  - Useful for many other things
    - Copy and paste
    - Installation and administration tasks
  - Host side integrated into QEMU
  - Guest side “qemu-guest-tools” package
  - Data transport via virtio-serial with fallback to older methods such as emulated serial and networking
Links

- Memory Overcommitment Manager
  http://wiki.github.com/aglitke/mom/
  mom-devel@googlegroups.com

- Cloudy Benchmark
  http://github.com/aglitke/cloudy

- Apache JMeter
  http://jakarta.apache.org/jmeter/

- Memknobs Program
  http://git.sr71.net/?p=memknobs.git;a=summary
Q & A