LEARN. NETWORK. EXPERIENCE OPEN SOURCE.

June 11-14, 2013
Boston, MA
Agenda: Performance Analysis Tuning Part II

Part I
- RHEL “tuned” profiles, top benchmark results
- Scalability – CFS Scheduler tunables / Cgroups
- Hugepages – Transparent Hugepages, 2MB/1GB
- NonUniform Memory Access (NUMA) and NUMAD

Part II
- Network Performance and Latency-performance
- Disk and Filesystem IO - Throughput-performance
- System Performance/Tools – perf, tuna, systemtap

Q & A
RHEL 6 Networking performance – System setup

- Disable unnecessary services, runlevel 3
- Follow vendor guidelines for BIOS Tuning
  - Logical cores? Power Management? Turbo?
- In the OS, consider
  - Disabling filesystem journal
  - Ensure mount using relatime
  - SSD/Memory Storage
  - Running swapless
  - Reducing writeback thresholds if your app does disk I/O
RHEL6 “tuned” package

# yum install tune*
# tuned-adm profile latency-performance
# tuned-adm list
Available profiles:
- latency-performance
- default
- enterprise-storage
- virtual-guest
- throughput-performance
- virtual-host

Current active profile: latency-performance
# tuned-adm profile default  (to disable)
## tuned Profile Comparison Matrix

<table>
<thead>
<tr>
<th>Tunable</th>
<th>default</th>
<th>enterprise-storage</th>
<th>virtual-host</th>
<th>virtual-guest</th>
<th>latency-performance</th>
<th>throughput-performance</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>kernel.sched_min_granularity_ns</code></td>
<td>4ms</td>
<td>10ms</td>
<td>10ms</td>
<td>10ms</td>
<td></td>
<td>10ms</td>
</tr>
<tr>
<td><code>kernel.sched_wakeup_granularity_ns</code></td>
<td>4ms</td>
<td>15ms</td>
<td>15ms</td>
<td>15ms</td>
<td></td>
<td>15ms</td>
</tr>
<tr>
<td><code>vm.dirty_ratio</code></td>
<td>20% RAM</td>
<td>40%</td>
<td>10%</td>
<td>40%</td>
<td></td>
<td>40%</td>
</tr>
<tr>
<td><code>vm.dirty_background_ratio</code></td>
<td>10% RAM</td>
<td></td>
<td>5%</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><code>vm.swappiness</code></td>
<td>60</td>
<td>10</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O Scheduler (Elevator)</td>
<td>CFQ</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
</tr>
<tr>
<td>Filesystem Barriers</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU Governor</td>
<td>ondemand</td>
<td>performance</td>
<td></td>
<td></td>
<td></td>
<td>performance</td>
</tr>
<tr>
<td>Disk Read-ahead</td>
<td></td>
<td>4x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disable THP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>CPU C-States</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Locked @ 1</td>
</tr>
</tbody>
</table>

[https://access.redhat.com/site/solutions/369093](https://access.redhat.com/site/solutions/369093)
Locality of Packets

Stream from Customer 1 → Socket1/Core1

Stream from Customer 2 → Socket1/Core2

Stream from Customer 3 → Socket1/Core3

Stream from Customer 4 → Socket1/Core4
Network Tuning: IRQ affinity

- Use irqbalance for the common case
- New irqbalance automates NUMA affinity for IRQs
- Flow-Steering Technologies

- Move 'p1p1*' IRQs to Socket 1:
  # tuna -q p1p1* -S1 -m -x
  # tuna -Q | grep p1p1

- Manual IRQ pinning for the last X percent/determinism
  - Guide on Red Hat Customer Portal
NUMA Affinity CLI Reference

```
# numactl -N1 -m1 ./command
```

- Sets CPU affinity for 'command' to CPU node 1
- Allocates memory out of Memory node 1
- Chose node 1 because of PCI-bus wiring
  - Upstream kernel community working on automatic NUMA balancing.
  - Test numad in RHEL6
Network Tuning: NIC Offloads favor Throughput

- Reduce the # of packets/IRQs the kernel processes
- Throughput vs Latency trade-off

<table>
<thead>
<tr>
<th>Offload</th>
<th>Summary</th>
<th>Protocol</th>
<th>Direction</th>
</tr>
</thead>
<tbody>
<tr>
<td>TSO tcp segment offload</td>
<td>MTU-chunking offloaded to NIC</td>
<td>TCP</td>
<td>TX</td>
</tr>
<tr>
<td>UFO udp fragment offload</td>
<td>MTU-chunking offloaded to NIC</td>
<td>UDP</td>
<td>TX</td>
</tr>
<tr>
<td>GSO generic segment offload</td>
<td>MTU-chunking done in-kernel</td>
<td>TCP, UDP</td>
<td>TX</td>
</tr>
<tr>
<td>GRO generic receive offload</td>
<td>NIC/driver batches certain RX packets</td>
<td>TCP, UDP</td>
<td>RX</td>
</tr>
<tr>
<td>LRO large receive offload</td>
<td>NIC/driver batches all RX packets</td>
<td>TCP</td>
<td>RX</td>
</tr>
</tbody>
</table>
Network Tuning: Buffer Bloat

- **Kernel buffers:**

```
# ss | grep -v ssh
```

<table>
<thead>
<tr>
<th>State</th>
<th>Recv-Q</th>
<th>Send-Q</th>
<th>Local Address:Port</th>
<th>Peer Address:Port</th>
</tr>
</thead>
<tbody>
<tr>
<td>ESTAB</td>
<td>0</td>
<td>0</td>
<td>172.17.1.36:38462</td>
<td>172.17.1.34:12865</td>
</tr>
<tr>
<td>ESTAB</td>
<td>0</td>
<td>3723128</td>
<td>172.17.1.36:58856</td>
<td>172.17.1.34:53491</td>
</tr>
</tbody>
</table>

- 10G line-rate
- ~4MB queue depth
- Matching servers
SR-IOV: RHEL 6.4

Round-trip Latencies Into Guest
(Lower is Better)

Latency (Microseconds)

<table>
<thead>
<tr>
<th>Bare metal</th>
<th>KVM + SR-IOV</th>
<th>KVM + Bridge</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHEL6.4 (tuned)</td>
<td>13</td>
<td>165</td>
</tr>
<tr>
<td>RHEL6.4 (untuned)</td>
<td>17</td>
<td>94</td>
</tr>
<tr>
<td>RHEL6.4 (SR-IOV tuned)</td>
<td>13</td>
<td>35</td>
</tr>
<tr>
<td>RHEL6.4 (Bridge tuned)</td>
<td>16</td>
<td>22</td>
</tr>
</tbody>
</table>

Min | Mean | 99.9% | StdDev

#redhat #rhsummit
CPU Tuning: P-states (frequency)

- Variable frequencies for each core

**P-state Impact on Latency**
(Lower is better)
CPU Tuning: C-states (idle states)

- Variable idle states for each core
- Cstate lock disables turbo, but increases determinism

### C-state Impact on Jitter

![Graph showing latency in micro-seconds over time (1-sec intervals) with C6, C3, C1, and C0 states highlighted. The graph indicates a decrease in latency with the transition from C6 to C0.]
# Turbostat shows P/C-states on Intel CPUs

*turbostat begins shipping in RHEL6.4, cpupowerutils package*

## Default

<table>
<thead>
<tr>
<th>pk</th>
<th>cor</th>
<th>CPU</th>
<th>%c0</th>
<th>GHz</th>
<th>TSC</th>
<th>%c1</th>
<th>%c3</th>
<th>%c6</th>
<th>%c7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.24</td>
<td>2.93</td>
<td>2.88</td>
<td>5.72</td>
<td>1.32</td>
<td>0.00</td>
<td>92.72</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>2.54</td>
<td>3.03</td>
<td>2.88</td>
<td>3.13</td>
<td>0.15</td>
<td>0.00</td>
<td>94.18</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>2.29</td>
<td>3.08</td>
<td>2.88</td>
<td>1.47</td>
<td>0.00</td>
<td>0.00</td>
<td>96.25</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>3</td>
<td>1.75</td>
<td>1.75</td>
<td>2.88</td>
<td>1.21</td>
<td>0.47</td>
<td>0.12</td>
<td>96.44</td>
</tr>
</tbody>
</table>

## latency-performance

<table>
<thead>
<tr>
<th>pk</th>
<th>cor</th>
<th>CPU</th>
<th>%c0</th>
<th>GHz</th>
<th>TSC</th>
<th>%c1</th>
<th>%c3</th>
<th>%c6</th>
<th>%c7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0.00</td>
<td>3.30</td>
<td>2.90</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0.00</td>
<td>3.30</td>
<td>2.90</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>2</td>
<td>2</td>
<td>0.00</td>
<td>3.30</td>
<td>2.90</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
<tr>
<td>0</td>
<td>3</td>
<td>3</td>
<td>0.00</td>
<td>3.30</td>
<td>2.90</td>
<td>100.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
Power Consumption RHEL6.4 vs RHEL6.4@C0

- C-state lock increases power draw over “out of the box”

<table>
<thead>
<tr>
<th>Test</th>
<th>Efficiency [Wh] % Diff</th>
</tr>
</thead>
<tbody>
<tr>
<td>Kernel Build</td>
<td>+12.5%</td>
</tr>
<tr>
<td>Disk Read</td>
<td>+32.2%</td>
</tr>
<tr>
<td>Disk Write</td>
<td>+25.6%</td>
</tr>
<tr>
<td>Unpack tar.gz</td>
<td>+23.3%</td>
</tr>
<tr>
<td>Active Idle</td>
<td>+41%</td>
</tr>
</tbody>
</table>

- Use cron to set latency-performance tuned profile when necessary.
- Set tuned profile in application init script
Memory Tuning: Transparent Hugepages

Introduced in RHEL 6

- Added counters in RHEL 6.2
- Enhanced again to reduce overhead in 6.4

```bash
# egrep 'trans|thp' /proc/vmstat
nr_anon_transparent_hugepages 2018
thp_fault_alloc 7302
thp_fault_fallback 0
thpCollapse_alloc 401
thpCollapse_alloc_failed 0
thp_split 21
```
Transparent Hugepages

Approx 20 spikes every 10 seconds: i.e., a spike per ~500ms
Transparent Hugepages Disabled

Nada
# Tuna (new in RHEL6.4)

## CPU Affinity and Events

<table>
<thead>
<tr>
<th>IRQ</th>
<th>Affinity</th>
<th>Events</th>
<th>Users</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0.23</td>
<td>12994</td>
<td>timer</td>
</tr>
<tr>
<td>1</td>
<td>0.2,4,6,8,10</td>
<td>2</td>
<td>i86042</td>
</tr>
<tr>
<td>3</td>
<td>0.2,4,6,8,10</td>
<td>268</td>
<td>serial</td>
</tr>
<tr>
<td>4</td>
<td>0.2,4,6,8,10</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>0.2,4,6,8,10</td>
<td>1</td>
<td>rtc0</td>
</tr>
<tr>
<td>9</td>
<td>0.2,4,6,8,10</td>
<td>0</td>
<td>acpi</td>
</tr>
<tr>
<td>12</td>
<td>0.2,4,6,8,10</td>
<td>4</td>
<td>i86042</td>
</tr>
<tr>
<td>14</td>
<td>0.2,4,6,8,10</td>
<td>0</td>
<td>pata_etikp</td>
</tr>
<tr>
<td>15</td>
<td>0.2,4,6,8,10</td>
<td>0</td>
<td>pata_etikp</td>
</tr>
<tr>
<td>16</td>
<td>0.2,4,6,8,10</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>0.2,4,6,8,10</td>
<td>20</td>
<td>radeon,ahci</td>
</tr>
<tr>
<td>23</td>
<td>0.2,4,6,8,10</td>
<td>4</td>
<td>ehci_hcd,usb2,ohci_hcd,usb3,ohci_hcd,usb4</td>
</tr>
<tr>
<td>44</td>
<td>0.2,4,6,8,10</td>
<td>12,14,16,18,20,22</td>
<td>25</td>
</tr>
</tbody>
</table>

### MD Table

<table>
<thead>
<tr>
<th>MD</th>
<th>Policy</th>
<th>Priority</th>
<th>Affinity</th>
<th>VolCtxtSwitch</th>
<th>NonVolCtxtSwitch</th>
<th>Command Line</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>1342</td>
<td>55</td>
<td>/sbin/init</td>
</tr>
<tr>
<td>363</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>1</td>
<td>0</td>
<td>/sbin/udevd -d</td>
</tr>
<tr>
<td>404</td>
<td>OTHER</td>
<td>0</td>
<td>0.2,4,6,8,10</td>
<td>59280707</td>
<td>77026</td>
<td>/usr/libexec/qemu-kvm -name-broker -S -M rhel6.4 -cpu Opteron_63 -nodeid_msr -wait -ssh</td>
</tr>
<tr>
<td>911</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>668</td>
<td>91</td>
<td>/sbin/udevd -d</td>
</tr>
<tr>
<td>2428</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>111956</td>
<td>0</td>
<td>auditd</td>
</tr>
<tr>
<td>2446</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>1</td>
<td>0</td>
<td>/sbin/parapreserve</td>
</tr>
<tr>
<td>2453</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>51</td>
<td>0</td>
<td>/sbin/ryskd -i /var/run/skqd.pid -c 5</td>
</tr>
<tr>
<td>2482</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>375632</td>
<td>1387</td>
<td>irqbalance</td>
</tr>
<tr>
<td>2503</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>126446</td>
<td>0</td>
<td>rpcbind</td>
</tr>
<tr>
<td>2510</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>10356</td>
<td>34</td>
<td>sshd: root@pts/2</td>
</tr>
<tr>
<td>2513</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>49</td>
<td>6</td>
<td>bash</td>
</tr>
<tr>
<td>2521</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>12</td>
<td>0</td>
<td>rc-pid</td>
</tr>
<tr>
<td>2542</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>5567</td>
<td>1302</td>
<td>/usr/bin/python/usr/bin/tuna</td>
</tr>
<tr>
<td>2577</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>1</td>
<td>0</td>
<td>rpcidmapd</td>
</tr>
<tr>
<td>2677</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>2485</td>
<td>3</td>
<td>dbus-daemon --system</td>
</tr>
<tr>
<td>2689</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>7745159</td>
<td>43353</td>
<td>avahi-daemon</td>
</tr>
<tr>
<td>2690</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>3</td>
<td>0</td>
<td>avahi-daemon</td>
</tr>
<tr>
<td>2718</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>2</td>
<td>0</td>
<td>/usr/sbin/acpid</td>
</tr>
<tr>
<td>2310</td>
<td>OTHER</td>
<td>0</td>
<td>0.23</td>
<td>323236</td>
<td>2</td>
<td>hold</td>
</tr>
</tbody>
</table>

[@redhat](https://www.redhat.com)  #rhsummit
Tuna IRQ/CPU affinity context menus

- CPU affinity for IRQs
- CPU affinity for PIDs
- Scheduler Policy
- Scheduler Priority

Set attributes for this IRQ:

<table>
<thead>
<tr>
<th>Policy</th>
<th>Scheduler priority</th>
<th>Affinity</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0</td>
<td>12</td>
</tr>
</tbody>
</table>

Set for these processes:

- Just the selected thread
- All threads of the selected process
- All command lines matching the regex below:

Command line regex: `/usr/libexec/qemu-kvm -name ose-broker -S -M rhel6.4.0 -cpu Opteron_G3,+nodeid_msr, +wtd, +skinit, +ib`
# tuna -t netserver -P

<table>
<thead>
<tr>
<th>thread</th>
<th>ctxt_switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>SCHED_ rtpri</td>
</tr>
<tr>
<td>13488</td>
<td>OTHER</td>
</tr>
</tbody>
</table>

# tuna -c2 -t netserver -m

# tuna -t netserver -P

<table>
<thead>
<tr>
<th>thread</th>
<th>ctxt_switches</th>
</tr>
</thead>
<tbody>
<tr>
<td>pid</td>
<td>SCHED_ rtpri</td>
</tr>
<tr>
<td>13488</td>
<td>OTHER</td>
</tr>
</tbody>
</table>
Tuna – for IRQs

- Move 'p1p1*' IRQs to Socket 1:
  ```
  # tuna -q p1p1* -S0 -m -x
  # tuna -Q | grep p1p1
  78  p1p1-0
  79  p1p1-1
  80  p1p1-2
  81  p1p1-3
  82  p1p1-4
  ...
  ```
Tuna – for core/socket isolation

# tuna -S1 -i

# grep Cpus_allowed_list /proc/`pgrep rsyslogd`/status
Cpus_allowed_list: 0-15

# tuna -S1 -i (tuna sets affinity of 'init' task as well)

# grep Cpus_allowed_list /proc/`pgrep rsyslogd`/status
Cpus_allowed_list: 0,1,2,3,4,5,6,7
NUMA Topology and PCI Bus

- Servers may have more than 1 PCI bus.
- Install adapters “close” to the CPU that will run the performance critical application.
- When BIOS reports locality, irqbalance handles NUMA/IRQ affinity automatically.

```bash
42:00.0 Network controller: Mellanox Technologies MT27500 Family [ConnectX-3]
# cat /sys/devices/pci0000:40/0000:40:03.0/0000:42:00.0/local_cpulist
   1,3,5,7,9,11,13,15

# dmesg | grep "NUMA node"
pci_bus 0000:00: on NUMA node 0 (pxm 1)
pci_bus 0000:40: on NUMA node 1 (pxm 2)
pci_bus 0000:3f: on NUMA node 0 (pxm 1)
pci_bus 0000:7f: on NUMA node 1 (pxm 2)
```
Know Your Hardware (hwloc)

PCI Bus “local” to this NUMA node
PCI Device Affinity

```bash
# lstopo-no-graphics | egrep 'NUMA|eth4'
NUMANode L#0 (P#0 144GB)
NUMANode L#1 (P#1 144GB)
Net L#10 "eth4"
```
Performance Monitoring Tool - perf

- Userspace tool to read CPU counters and kernel tracepoints

- RHEL 6.4 includes perf from upstream kernel 3.6
- https://perf.wiki.kernel.org
- perf top (dynamic)
- perf record / report (save and replay)
- perf stat <command> (analyze a particular workload)
Tracing with 'perf': perf top

- System-wide 'top' view of active processes
Tracing with 'perf': perf stat

- Attach to existing PID and report all kmem tracepoints:
  
  # perf stat -a ./my_cmd

<table>
<thead>
<tr>
<th>Count</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>118736.543926</td>
<td>task-clock</td>
</tr>
<tr>
<td>7,150</td>
<td>context-switches</td>
</tr>
<tr>
<td>123</td>
<td>CPU-migrations</td>
</tr>
<tr>
<td>66,130</td>
<td>page-faults</td>
</tr>
<tr>
<td>3,075,630,979</td>
<td>cycles</td>
</tr>
<tr>
<td>169,711,604</td>
<td>stalled-cycles-frontend</td>
</tr>
<tr>
<td>3,837,328,065</td>
<td>stalled-cycles-backend</td>
</tr>
<tr>
<td>531,775,550</td>
<td>instructions</td>
</tr>
<tr>
<td>147,957,418</td>
<td>branches</td>
</tr>
<tr>
<td>6,019,001</td>
<td>branch-misses</td>
</tr>
</tbody>
</table>
Tracing with 'perf': perf diff

- Show differences between 2 perf.data recordings
- Run perf record twice, each with different tuning

```bash
# perf diff -M perf.data.data.old perf.data
```

<table>
<thead>
<tr>
<th>Baseline</th>
<th>Delta</th>
<th>Displ</th>
<th>Shared Object</th>
</tr>
</thead>
<tbody>
<tr>
<td>66.66%</td>
<td>-15.24%</td>
<td></td>
<td>pig</td>
</tr>
<tr>
<td>14.62%</td>
<td>+9.61%</td>
<td></td>
<td>libc-2.12.so</td>
</tr>
<tr>
<td>6.73%</td>
<td>+1.44%</td>
<td></td>
<td>[kernel.kallsyms]</td>
</tr>
<tr>
<td>1.94%</td>
<td>+1.81%</td>
<td></td>
<td>[kernel.kallsyms]</td>
</tr>
<tr>
<td>0.81%</td>
<td>+0.24%</td>
<td></td>
<td>[kernel.kallsyms]</td>
</tr>
</tbody>
</table>
Tracing with 'perf': perf script + gprof2dot

# perf script -i perf.data | \
gprof2dot -f perf | \
odot -Tsvg -o output.svg
Interesting new Network/Perf things in RHEL6.4

- tuna included
- latency-performance “tuned” profile beefed up
  - Lock C-states
  - Disable Transparent Hugepages
- turbostat included in cpupowerutils package
- hwloc now reports PCI bus topology
- PTP Tech Preview
- Mellanox Infiniband SR-IOV Tech Preview
• In both OOB and Optimized cases Red Hat Enterprise Linux delivers better throughput and lower latency to critical network-heavy applications

I/O Tuning – Understanding I/O Elevators

- **Deadline**
  - Two queues per device, one for read and one for writes
  - I/Os 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
Iozone Performance Comparison EXT4/XFS/GFS

The diagrams show the performance comparison of different file systems (ext3, ext4, gfs2, xfs) across RHEL 6.3 and RHEL 6.4. The y-axis represents the performance metric, while the x-axis represents the file systems. The red bars represent RHEL 6.3, and the blue bars represent RHEL 6.4.
SAS Application on Standalone Systems
Picking a RHEL File System

**xfs**  most recommended
- Max file system size 100TB
- Max file size 100TB
- Best performing

**ext4** recommended
- Max file system size 16TB
- Max file size 16TB

**ext3** not recommended
- Max file system size 16TB
- Max file size 2TB

---

### SAS Mixed Analytics 9.3 running RHEL6.3

Comparing Total time and System CPU usage

<table>
<thead>
<tr>
<th>File system</th>
<th>TOTALtime</th>
<th>SystemTime</th>
</tr>
</thead>
<tbody>
<tr>
<td>xfs</td>
<td>9600</td>
<td>400</td>
</tr>
<tr>
<td>ext4</td>
<td>10200</td>
<td>200</td>
</tr>
<tr>
<td>ext3</td>
<td>10800</td>
<td>600</td>
</tr>
</tbody>
</table>

**File system size**
- ext3: 100TB
- ext4: 16TB
- xfs: 16TB

**Max file size**
- ext3: 2TB
- ext4: 16TB
- xfs: 100TB
RHEL6 “tuned” package

# yum install tune*
# tuned-adm profile enterprise-storage
# tuned-adm list

Available profiles:
- latency-performance
- default
- enterprise-storage
- virtual-guest
- throughput-performance
- virtual-host

Current active profile: enterprise-storage
# tuned-adm profile default   (to disable)
## tuned Profile Comparison Matrix

<table>
<thead>
<tr>
<th>Tunable</th>
<th>default</th>
<th>enterprise-storage</th>
<th>virtual-host</th>
<th>virtual-guest</th>
<th>latency-performance</th>
<th>throughput-performance</th>
</tr>
</thead>
<tbody>
<tr>
<td>kernel.sched.min粒度_min granularity_ns</td>
<td>4ms</td>
<td>10ms</td>
<td>10ms</td>
<td>10ms</td>
<td>10ms</td>
<td>10ms</td>
</tr>
<tr>
<td>kernel.sched.wakeup粒度_min granularity_ns</td>
<td>4ms</td>
<td>15ms</td>
<td>15ms</td>
<td>15ms</td>
<td>15ms</td>
<td>15ms</td>
</tr>
<tr>
<td>vm.dirty_ratio</td>
<td>20% RAM</td>
<td>40%</td>
<td>10%</td>
<td>40%</td>
<td>40%</td>
<td>40%</td>
</tr>
<tr>
<td>vm.dirty_background粒度_min ratio</td>
<td>10% RAM</td>
<td>5%</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>vm.swappiness</td>
<td>60</td>
<td>10</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>I/O Scheduler (Elevator)</td>
<td>CFQ</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
<td>deadline</td>
</tr>
<tr>
<td>Filesystem Barriers</td>
<td>On</td>
<td>Off</td>
<td>Off</td>
<td>Off</td>
<td></td>
<td></td>
</tr>
<tr>
<td>CPU Governor</td>
<td>ondemand</td>
<td>performance</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disk Read-ahead</td>
<td></td>
<td>4x</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Disable THP</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
<tr>
<td>Disable C-States</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Yes</td>
</tr>
</tbody>
</table>

https://access.redhat.com/site/solutions/369093
Tuning Memory – **Flushing Caches**

- Drop unused Cache – to control pagecache dynamically
  - ✓ Frees most pagecache memory
  - ✓ File cache
  - ✗ If the DB uses cache, may notice slowdown
- NOTE: Use for benchmark environments.
- **Free pagecache**
  - `# sync; echo 1 > /proc/sys/vm/drop_caches`
- **Free slabcache**
  - `# sync; echo 2 > /proc/sys/vm/drop_caches`
- **Free pagecache and slabcache**
  - `# sync; echo 3 > /proc/sys/vm/drop_caches`
Iozone Performance Effect of TUNED EXT4/XFS/GFS

RHEL6.4 File System In Cache Performance

Intel Large File I/O (iozone)

- not tuned
- tuned

Throughput in MB/Sec

ext3  ext4  xfs  gfs2

RHEL6.4 File System Out of Cache Performance

Intel Large File I/O (iozone)

- not tuned
- tuned

Throughput in MB/Sec

ext3  ext4  xfs  gfs2
RHEL BIOS and Tuned profiles

BIOS to OS controlled

# tuned-adm profile enterprise-storage

kernel.sched_min_granularity_ns = 10000000

kernel.sched_wakeup_granularity_ns = 15000000

vm.dirty_ratio = 40

ELEVATOR="deadline"

If

/sys/block/sdX/device/scsi_disk/X:X:X:X/cache_type:write back

then

BARRIERS=off (for mounts other than root/boot vols)

# set cpuspeed governors=performance
RHEL vs Windows Server 2012 Comparison
File system: in-cache file-access method

Comparison of file system performance - In cache

- Red Hat Enterprise Linux 6 ext4
- Microsoft Windows Server 2012 NTFS
- Red Hat Enterprise Linux 6 XFS
- Microsoft Windows Server 2012 ReFS

Average performance (KB/s)

Out-of-box

Optimized

Comparison of file system performance - Direct I/O

- Red Hat Enterprise Linux 6 ext4
- Microsoft Windows Server 2012 NTFS
- Red Hat Enterprise Linux 6 XFS
- Microsoft Windows Server 2012 ReFS

Out-of-box

- Average performance (KB/s)

Optimized

Per device/file/LUN page flush daemon

- Each file system or block device has its own flush daemon
- Allows different flushing thresholds and resources for each daemon/device/file system.
- Prevents some devices from not getting flushed because a shared daemon blocks used all resources.
- Replaces pdflushd where a pool of threads flushed all devices.
High End HP DL 980 AIM7 results w/ “ktune” (r5) “tuned-adm” (r6)

HP DL980 64-core/256GB/30 FC/480 lun  AIM7 results w/ “tuned”
Virtual Memory Manager (VM) Tunables

• Reclaim Ratios
  • /proc/sys/vm/swappiness
  • /proc/sys/vm/vfs_cache_pressure
  • /proc/sys/vm/min_free_kbytes

• Writeback Parameters
  • /proc/sys/vm/dirty_background_ratio
  • /proc/sys/vm/dirty_ratio

• Readahead parameters
  • /sys/block/<bdev>/queue/read_ahead_kb
dirty_ratio and dirty_background_ratio

pagecache

100% of pagecache RAM dirty

flushd and write()'ng processes write dirty buffers

dirty_ratio(20% of RAM dirty) – processes start synchronous writes

flushd writes dirty buffers in background

dirty_background_ratio(10% of RAM dirty) – wakeup flushd

do_nothing

0% of pagecache RAM dirty
KVM / RHS Tuning

- gluster volume set `<volume>` group virt
- XFS mkfs -n size=8192, mount inode64, noatime
- RHS server: `tuned-adm profile rhs-virtualization`
  - Increase in readahead, lower dirty ratio's
- KVM host: `tuned-adm profile virtual-host`
  - Better response time shrink guest block device queue
    - `/sys/block/vda/queue.nr_request` (16 or 8)
  - Best sequential read throughput, raise VM read-ahead
    - `/sys/block/vda/queue/read_ahead_kb` (4096/8192)
Iozone Performance Comparison RHS2.1/XFS w/ RHEV

![Graph showing performance comparison between Out-of-the-box and tuned rhs-virtualization for different I/O operations (rnd-write, rnd-read, seq-write, seq-read).]
Summary / Questions

- Red Hat Enterprise Linux 6 Performance Features
  - “TUNED” tool – adjusts system parameters to match environments - throughput/latency.
  - Transparent Huge Pages – auto select large pages for anonymous memory, static hugepages for shared mem
  - Non-uniform Memory Access (NUMA)
    - NUMAstat enhancements
    - NUMActl for manual control
    - NUMAD daemon for auto placement
  - TUNA – integration w/ RHEL6.4
Helpful Links

- Performance Tuning Guide
- Low Latency Performance Tuning Guide
- Optimizing RHEL Performance by Tuning IRQ Affinity
- KVM Performance Guide
- STAC Network I/O SIG

Blog: http://www.breakage.org/ or @jeremyeder
Comparison of TPC-C results using the Red Hat operating system. For more information about the TPC and the benchmark results referenced here see www.tpc.org.
RHEL6 Benchmark TPC-C- 2 socket improvements

<table>
<thead>
<tr>
<th>System</th>
<th>tpmC</th>
<th>$/tpmC</th>
<th>cpus</th>
</tr>
</thead>
<tbody>
<tr>
<td>RHEL 3 HP x280</td>
<td>51,506</td>
<td>$1.81</td>
<td></td>
</tr>
<tr>
<td>RHEL 4 IBM iServer p570</td>
<td>197,669</td>
<td>$3.93</td>
<td></td>
</tr>
<tr>
<td>RHEL 5 IBM Power 550 Express</td>
<td>276,383</td>
<td>$2.55</td>
<td></td>
</tr>
<tr>
<td>RHEL 6 IBM Flex System x260</td>
<td>1,503,544</td>
<td>$0.53</td>
<td></td>
</tr>
<tr>
<td>RHEL 6 IBM Flex System x260</td>
<td>1,320,082</td>
<td>$0.51</td>
<td></td>
</tr>
</tbody>
</table>
Comparison of TPC-C results using the Red Hat operating system. For more information about the TPC and the benchmark results referenced here see www.tpc.org.
Comparison of TPC-C results using the Red Hat operating system. For more information about the TPC and the benchmark results referenced here see www.tpc.org.