Performance Tuning Overview

• Little’s Law: $L = A \times W$ (Queue Length = Average Arrival Rate x Wait Time)
  • The length of the waiting line for a resource depends on the average rate at which new requests arrive multiplied by the average amount of time a request spends in the system.

• Wait Time: $W = S + Q$ (Wait Time = Service Time + Queue Time)
  • The sum of the amount of time the request waits for a resource to become available combined with the amount of time it takes to service the request.

• Queue Stability & Forced Flow Law
  • For a queue to maintain the same length, the rate at which new requests arrive must be the same as the rate at which requests are completed
  • System Throughput = Resource Throughput x Resource Visit Count
Performance Tuning Overview (continued)

- Variables
  - Latency
  - Throughput
  - Power Usage
  - Jitter
- Standard Process
  - Baseline -> Make Change -> New Baseline
- Final Thoughts
  - Typically compared to the old system
  - Does not necessarily always mean better*
Standard Distributed Architecture

- Master Nodes: HDFS NameNode service, YARN resource manager and MapReduce Job History manager
- Data Nodes: HDFS DataNode service and the YARN distributed NodeManager
SOLID PERFORMANCE ACROSS WORKLOADS
RHEL 7 VS RHEL 6.5

PERFORMANCE GAINS ACROSS WIDE RANGE OF WORKLOADS AND MULTIPLE GENERATIONS OF HARDWARE

NETWORK   CPU   ERP   MEMORY   OLTP   ANALYTICS   OLTP   JAVA
RHEL 6.5   RHEL 7

PARITY +1% +2% +8% +10% +11% +13% +25%

NORMALIZED PERFORMANCE (%)
CHOICE OF FILE SYSTEMS

- Scale file systems to 500TB with new default filesystem XFS
- Scale to 50TB with ext4
- Btrfs also available\(^1\)
- Parallel NFS v4 provides improved performance and throughput

<table>
<thead>
<tr>
<th>Type</th>
<th>Supported Limit</th>
<th>Root</th>
<th>Boot</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-node</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>XFS</td>
<td>500TB</td>
<td>Yes</td>
<td>Yes</td>
<td>System default</td>
</tr>
<tr>
<td>ext4</td>
<td>50TB</td>
<td>Yes</td>
<td>Yes</td>
<td>Driver allow access to older versions (ext2, ext3).</td>
</tr>
<tr>
<td>btrfs(^2)</td>
<td>50TB</td>
<td>Yes</td>
<td>Yes</td>
<td></td>
</tr>
<tr>
<td>GFS2</td>
<td>2-16 nodes</td>
<td>Yes</td>
<td>No</td>
<td>Shared-storage file system</td>
</tr>
</tbody>
</table>

\(^1\) Available as a Technology Preview
Tuned Recommendations

• Physical
  • NameNode and DataNode:
    • tuned-adm profile enterprise-storage

• Virtualized
  • NameNode and DataNode:
    • tuned-adm profile virtual-guest
    • tuned-adm profile virtual-host (for underlying host)
Behind the Scenes

- Scheduler Changes
  - Deadline executes I/O Operations (IOPs) through the concept of "batches" which are sets of operation
  
  ```
  ELEVATOR="deadline"
  # These are the devices, that should be tuned with the
  ELEVATORELEVATOR_TUNE_DEVS="/sys/block/
  {sd,cciss,dm-,vd}*/queue/scheduler"
  ```

- Sysctl.conf Changes: Increases latency, but also throughput
  
  ```
  kernel.sched_min_granularity_ns = 10000000
  kernel.sched_wakeup_granularity_ns = 15000000
  # The generator of dirty data starts writeback at this
  # percentage (system default is 20%)
  vm.dirty_ratio = 40
  ```
Links

- Red Hat & HortonWorks Data Platform Reference Architecture

- Douglas Shakshober (Shak) and Larry Woodman Performance Tuning:
  - Part 1: https://www.youtube.com/watch?v=fATEiBJ3pKw
  - Part 2: https://www.youtube.com/watch?v=km-vLELmWLs

- Deadline Scheduler

- Adjusting CFS parameters
Thank You

Scott McCarty
Senior Solutions Architect, Red Hat
01/12/2015
PERFORMANCE ENHANCEMENTS WITH RED HAT ENTERPRISE LINUX 7

- BUILT-IN PERFORMANCE PROFILES SIMPLIFY CONFIGURATION
- MONITORING WITH PERFORMANCE CO-PILOT AND THERMOSTAT
- FINE-TUNE PERFORMANCE WITH ENHANCED TOOLING VIA TUNA AND TUNED
OPTIMAL PERFORMANCE VIA PROFILES

Optimal performance management via enhanced performance tuning at install, simplified instrumentation and tuning features, and performance monitoring tooling.

PERFORMANCE CO-PILOT (PCP)

THERMOSTAT (FOR JVMs)
PROFILING AND MONITORING WITH TUNA

- Tool for fine grained control
- Display applications / processes
- Displays CPU enumeration
- Socket (useful for NUMA tuning)
- Dynamic control of tuning
  - Process affinity
  - Parent & threads
  - Scheduling policy
  - Device IRQ priorities, etc