SUMMIT

PRESENTED BY RED HAT

LEARN. NETWORK. EXPERIENCE OPEN SOURCE.

www.theredhatsummit.com
Red Hat Enterprise MRG Messaging
Performance Seminar

Mark Wagner
Principle Engineer,
Red Hat

June 25, 2010

Carl Trieloff
Technical Director,
Red Hat
Overview

- Brief overview of MRG
- Guidelines for tuning Red Hat Enterprise MRG Messaging
- Understanding differences between technologies
  - 1GB / 10GB / IB / RDMA
  - SCSI / SATA / Fibre / FusionIO
- Best practices - Enterprise MRG in various configurations
  - Standalone/ Cluster / Grid
- The updated benchmark data across the newest platforms
- Open source performance tools for Red Hat Enterprise MRG
Red Hat Enterprise MRG

- Deploy apps at scale to any resource
- Run with Realtime performance
- Interoperate and send data with fast, reliable, AMQP-compliant messaging

_RED HAT ENTERPRISE MRG_

- Desktop PC Cycle-Stealing
- Local Grid
- Remote Grid
- Remote Cloud
- Remote Server
Messaging
MRG Messaging

Enterprise Messaging System that

**Implements AMQP** (Advanced Message Queuing Protocol), the first open messaging standard

Participation from Red Hat, JPMC, Goldman, Credit Suisse, Deutsche Borse, Barclays, Bank of America, Microsoft, Cisco, etc

**Spans many use cases in one implementation** to consolidate architectural silos (fast messaging, reliable messaging, large file transfer, publish/subscribe, eventing, etc)

**Uses Linux-specific optimizations** to achieve breakthrough performance on Red Hat Enterprise Linux and MRG Realtime

**Runs on non-Linux platforms** without the full performance and quality of service benefits that Red Hat Enterprise Linux provides

Provides open, high performance system for everything from financial exchanges to infrastructure management
MRG Messaging Feature Highlights

Core Messaging
- P2P, fanout, pub-sub, sync, async
- Reliable messaging
- Transactions local to dtx
- Multiple clients (C++, Java, JMS, .NET Python, Ruby, {WCF})

High Performance
- C++ broker, optimized for RHEL
- O-direct AIO for high-speed durability
- RDMA support for ultra low latency

Management tools
- Web-based GUI, cmd line tools
- AMQP-based framework & APIs (QMF)

Advanced Features
- Queue Semantics: Ring Queue, Last Value Queue, TTL, Initial Value Exchange, etc
- Routing patterns, including XML XQuery
- Federation with dynamic routes

High Availability
- Active-Standby/Active-Active Broker Clustering
- Federated disaster recovery

Security
- SASL authentication
- SSL/TLS/ Kerberos encryption
- Role-based Access Control (ACL)
MRG 1.3 / RHEL6, what is new with regard to 'M'

Updated clients

- New protocol-independent C++ and Python clients
- Additional python & ruby clients that wrap a native C implementation for improved performance
- Windows C++ client support (including .NET support) & Python
- Additional QMF APIs
- Map message support
- JBoss SOA-P and EAP certification

Broker

- Addition of iWARP, & RoCE in conjunction with RHEL 6
- RDMA for openAIS/corosync in conjunction with RHEL 6
- Offline Storage Management
- And much more... (includes ~300 updates/ improvements)

(Go to Bryan Che's presentation for full roadmap details)
MRG 1.2 / AMQP Scale up.

Single HP Nehalem BL460c 40G Infiniband AMQP Perftest

Number of Brokers on the Server

- 8 Broker
- 4 Broker
- 2 Broker
- 1 Broker

Messages/Sec

- 8 bytes
- 64 Bytes
- 256 Bytes
- 1024 Bytes

2 Intel(R) Xeon(R) CPU X5570 @ 2.93GHz per blade (Nehalem) (2.93 GHz, 8MB L3 cache, 95W, 2/2/3/3)

Memory 24GB(6x4GB), Memory Type DDR3-1333, HT, Turbo 2/2/3/3

Infiniband 4X QDR IB Dual-port Mezzanine HCAs (1 port connected)

Infiniband Switch BLc 4X QDR IB Switch

SUMMIT JBoss WORLD
PRESENTED BY RED HAT
PerfTest - Bare Metal and KVM

Message Rates with Different Technologies

- Bare Metal Msg/Sec
- 1 Bridged Guest Msg/Sec
- 2 Bridged Guests Msg/Sec
- 1 SR-IOV Guests Msg/Sec
- 2 SR-IOV GuestsMsg/Sec

Message Size (Bytes) vs. Messages / Sec
MRG 1.3 / AMQP Scale up, KVM ~5%

Perftest - Bare Metal and KVM

Lines = Messages / Sec
Columns = MBytes/sec

Throughput (MBytes/sec)
Message Size (Bytes)

1 Bridged Guest MB/sec
2 Bridged Guests MB/Sec
1 SR-IOV Guests MB/Sec
2 SR-IOV Guests MB/Sec
Bare Metal MB/sec

1 Bridged Guest Msg/Sec
2 Bridged Guests Msg/Sec
1 SR-IOV Guests Msg/Sec
2 SR-IOV GuestsMsg/Sec
Bare Metal Msg/Sec

SUMMIT
JBoss WORLD
PRESENTED BY RED HAT
MRG 1.3 RHEL6 A quick peak at RoCE (RoE)

10Gbit Mellanox w/wo RDMA - 8 Byte
Latency vs Message Rates - Lower is Better

Message Rate (Msg / Sec )
- 8 Bytes
- 8 Bytes with RDMA

Latency in microseconds
MRG 1.3 RHEL6 A quick peak at RoCE (cont)

10Gbit Mellanox w/wo RDMA

Latency vs Message Sizes

Latency in microseconds

Message Rate (Msg / Sec )

Latency in microseconds

8 Bytes with RDMA
16 Bytes with RDMA
32 Bytes with RDMA
256 Bytes with RDMA
1024 Bytes with RDMA
8 Bytes
16 Bytes
32 Bytes
256 Bytes
1024 Bytes
RHEL5.5 Infiniband vs. RHEL6 10Gbit – Messages / Sec

Comparing RHEL5.5 Mellanox Infiniband and RHEL6 Mellanox 10Gb with RoCE

Message Size (Bytes)

<table>
<thead>
<tr>
<th>Message Size (Bytes)</th>
<th>RHEL5.5 Infiniband</th>
<th>RHEL6 RoCE</th>
</tr>
</thead>
<tbody>
<tr>
<td>8</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>16</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>32</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>64</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>128</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>256</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>512</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>1024</td>
<td>800000</td>
<td>1200000</td>
</tr>
<tr>
<td>2048</td>
<td>800000</td>
<td>1200000</td>
</tr>
</tbody>
</table>
RHEL5.5 to RHEL6

10 Gbit Ethernet (Mellanox)

RHEL5 vs RHEL6 (preliminary)

Message Rates

Message Size (Bytes)

Messages / Second

10 Gbit Ethernet (Mellanox)
Tuning

Tuning can provide excellent improvements

Steps are different for throughput vs latency, goal is the same.

Try to maximize CPU cache hits and localize memory

Use NUMA if possible

```
numactl -c1 -m1 /root/qpid/cpp/src/qpidd --auth no -m no --pid-dir /var/run/qpidd --data-dir /var/lib/qpidd --load-module /root/qpid/cpp/src/.libs/rdma.so -P rdma
```

Move IRQ handlers as needed

Understand the NIC parameters, tune as necessary
The impact of tuning
Clustering and Federation

Active/Active **Clustering** provides
Cluster acts as virtual single broker
scalability and enhanced load-balancing
Producers and consumers can be connected to any broker in the cluster based on RHEL5 OpenAIS technology

**Federation** and **Disaster Recovery** provides
Geographical distribution of brokers configured via *links* and *routes*

*DR is setup via queue state replication*

*link*: connection between two brokers that allows messages to be passed between them - can be RDMA, SSL, TCP etc.

*route*: path that messages take from one broker to another; can run along one or more links to the final destination. Routes can be dynamic or static
MRG 1.2 Messaging Clustered Throughput with RDMA

On prebuild of corosync for RHEL 6, Final RHEL6 / MRG 1.3 data not yet avail

3-Node cluster
IBM x3550
2 x 4-core Xeon E5420 2.5MHz (Harpertown)
16GB ram 266 MHz Mellanox MT25204 [InfiniHost III Lx HCA]
MRG Messaging Infiniband RDMA Latency: Under 40 Microseconds Reliably Acknowledged

MRG Messaging Latency Test on HP BL460c G6 Infiniband
100K Message Rate

Average Latency (ms)

- 32 Bytes RDMA Nehalem
- 256 Bytes RDMA Nehalem
- 1024 Bytes RDMA Nehalem
MRG 1.2 Messaging on KVM Virtualized Performance: <200 Microsecond Latency, Reliably Acknowledged

RHEL5.4 KVM AMQP Messaging Perf
Dell Poweredge R710 Intel Nehalem, 2 10Gbit VT-d

(Samples of 10k messages)
MRG 1.2 Messaging Durable Messaging Throughput

MRG Durable Messaging Throughput Across Different Storage Types

Intel 16 CPU Hapertown
12GB memory 667 Memory speed
Intel 82571EB Gigabit Ethernet
HP IO Fusion
32-byte messages

- 1 NIC
- 1 NIC Durable IO Fusion Card
- 1 NIC Durable Fiber Disk
- 1 NIC Durable Internal SCSI drive
MRG 1.2 Messaging Durable Messaging Latency

Latency test with Durable Store Different Storage Types

Intel 16 CPU Harpertown
12GB memory 667 Memory speed
Intel 82571EB Gigabit Ethernet
HP IO Fusion
32-byte messages

Average Latency (ms)
All Layers of the stack matter

If you are concerned about performance, source of latency, or issues can come from all layers...

- Tuning & working in labs
- Red Hat MRG – Tuning tools
- Red Hat MRG – Messaging / Grid
- Red Hat / HP Systems
- HP – Voltaire / Red Hat RDMA
- Red Hat MRG - Realtime
- HP reduced SMI BIOS's
- HP compute & storage

FSI-HPC Solution Stack

Determinism, and performance needs to work at each layer, HP & Red Hat for example are partnered across the stack
Dealing with SMIs

HP BIOS Option for Low Latency Apps (G5,G6,G7)
Disable frequent SMIs used for Dynamic Power Savings Mode, CPU Utilization monitoring, P-state monitoring and ECC monitoring
Benefits both RHEL & MRG operating environments.

Latency spikes with standard BIOS settings
Latency when SMIs disabled in BIOS
Real Time
MRG – Realtime RHEL on HP systems

Enables applications and transactions to run predictably, with guaranteed response times.

Upgrades RHEL 5 to realtime OS

Provides replacement kernel for RHEL5; x86/x86_64

Preserves RHEL Application Compatibility

For certified hardware, see www.redhat.com/mrg/hardware
**MRG 1.2 Realtime Scheduling Latency**

Vanilla 2.6.24.7 versus MRG RT (500K loops)

- **Vanilla**
  - Min: 1
  - Max: 2857
  - Mean: 11.47
  - Mode: 9.00
  - Median: 9.00
  - Std. Deviation: 54.94

- **MRG RT**
  - Min: 4
  - Max: 43
  - Mean: 8.34
  - Mode: 8.00
  - Median: 8.00
  - Std. Deviation: 1.49
MRG 1.3, what is new with regard to 'R'

2.6.33 kernel
- Realtime will move to a 2.6.33-based kernel

Perf tool
- Realtime will include the new Performance Counter subsystem in the kernel and also the new associated perf performance tool

Improved performance
- Realtime performance will continue to improve

Certification
- Additional hardware enablement/hardware
- New Realtime self-service hardware certification program
Grid
MRG Grid

Provides leading High Performance & High Throughput Computing

- Brings advantages of scale-out and flexible deployment to any application or workload
- Delivers better asset utilization, allowing applications to take advantage of all available computing resources

Enables building cloud infrastructure and aggregating multiple clouds

- Integrated support for virtualization as well as public clouds
- Seamlessly aggregates multiple cloud resources into one compute pool

Provides seamless and flexible computing across:

- Local grids
- Remote grids
- Private and hybrid clouds
- Public clouds (Amazon EC2)
- Cycle-harvesting from desktop PCs
Based on Condor and Includes:

**Enterprise Supportability**
From Red Hat

**Web-Based Management Console**
Unified management across all of MRG for job, system, license management, and workload management/monitoring

**Low Latency Scheduling**
Enable job submission to Condor via AMQP Messaging clients
Enable sub-second, low-latency scheduling for sub-second jobs

**Virtualization Support via libvirt Integration**
Support scheduling of virtual machines on Linux using libvirt API's

**Cloud Integration with Amazon Ec2**
Enable automatic cloud provisioning, job submission, results storage, teardown via Condor scheduler
Extensible, it can be a dependency for other jobs or executed based on rules (e.g. add capacity in the cloud if local grid out of capacity)

**Concurrency Limits**
Set limits on how much of a certain resource (e.g. software licenses, db connections) can be used at once

**Dynamic Slots**
Mark slots as partitionable and sub-divide them dynamically so that more than one job can occupy a slot at once
MRG 1.3, what is new with regard to 'G'

- **New User Tools**
  - Job submission interface
  - User centric UI with submissions

- **New Admin Tools**
  - Configuration management
  - Trigger/event service (diagnostic)
  - Multiple pool/grid representation (e.g. EC2 nodes)
  - Facility for administrators to adjust User/Group priorities and do so in a hierarchical manner for hierarchical fair share

- **Engine**
  - Windows Execute Node Support
  - Enhanced workflow (DAGMan) management
  - **Enhanced scalability**
Grid scalability data

- Enterprise workload focus
  - Previous optimized for long duration jobs at scale
  - Enhancements for short running jobs at scale
- Performance testing showed room for improvement
  - Our testing showed some cyclic behavior
    - Mostly visible on short duration jobs
  - Multiple improvements lead to good utilization
    - Can now achieve ~98% utilization with 4 minute jobs
    - Can now achieve ~88% utilization with 15 second jobs
Grid scalability - Before

Active Jobs - fixed 15 sec duration
Grid scalability - After

Active Jobs - fixed 15 sec duration
WORKLIFE=7200  MAX_ACCEPT=5
MRG Realtime Tools

**TUNA: System Tuning Tool**

Dynamically control tuning parameters like process affinity, parent & threads, scheduling policy, device IRQ priorities, etc.

**FTrace: Latency Tracer**

Runtime trace capture of longest latency codepaths – both kernel and application. Peak detector

Selectable triggers for threshold tracing

Detailed kernel profiles based on latency triggers

**RTEval: Hardware Latency Detector**

Tool that finds hardware latencies in your system so that you can achieve low latency across your entire platform

Complements MRG Realtime hardware certification program

Existing standard RHEL5 based performance monitoring tools remain relevant
Take Home Exam
An example to try

Broker

```bash
numactl -c1 -m1 /root/qpid/cpp/src/qpidd --auth no -m no --pid-dir /var/run/qpidd --data-dir /var/lib/qpidd -P rdma
```

Driver

```bash
cat run_latencytest_rdma.sh

#!/bin/bash

for rate in `seq 1 10`; do  echo "rate at ${rate}0000"; /usr/bin/latencytest -b $1 --size $2 --rate ${rate}0000 -P rdma & sleep 60; kill %1; echo ;  sleep 1; done 2>&1 > $3

./run_latencytest_rdma.sh 172.168.10.18 32 perf18_rhel6_mellanox_10gb_rdma.log &
```
Questions ?
FOLLOW US ON TWITTER
www.twitter.com/redhatsummit

TWEET ABOUT IT
#summitjbw

READ THE BLOG
http://summitblog.redhat.com/