

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**

**LEARN. NETWORK.  
EXPERIENCE OPEN SOURCE.**

[www.theredhatsummit.com](http://www.theredhatsummit.com)

# Red Hat Enterprise MRG Messaging Performance Seminar

Mark Wagner  
Principle Engineer,  
Red Hat

Carl Trieloff  
Technical Director,  
Red Hat

June 25, 2010

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY 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

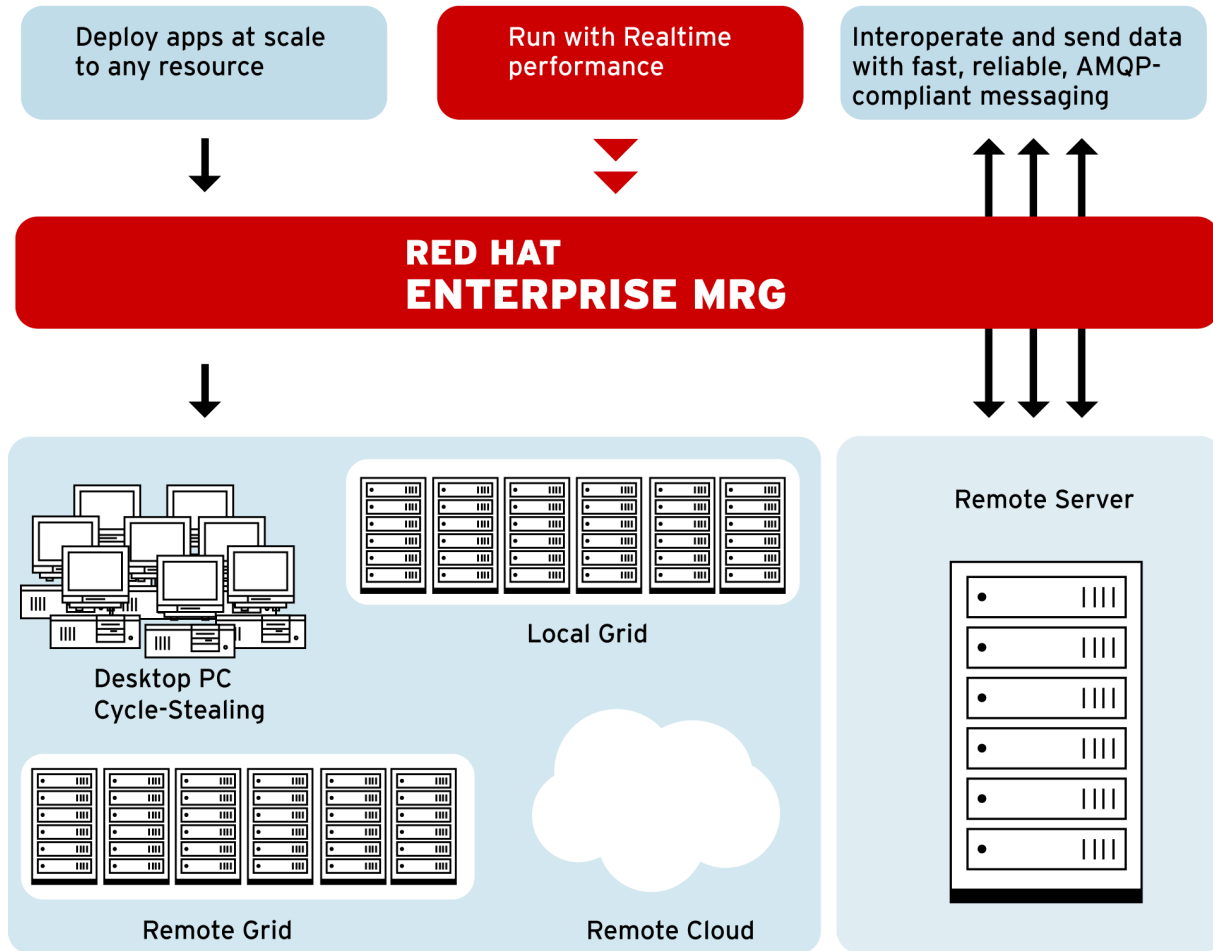
**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# Red Hat Enterprise MRG



SUMMIT

JBoss  
WORLD

PRESENTED BY RED HAT



# Messaging

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# 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

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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)

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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)

**SUMMIT**

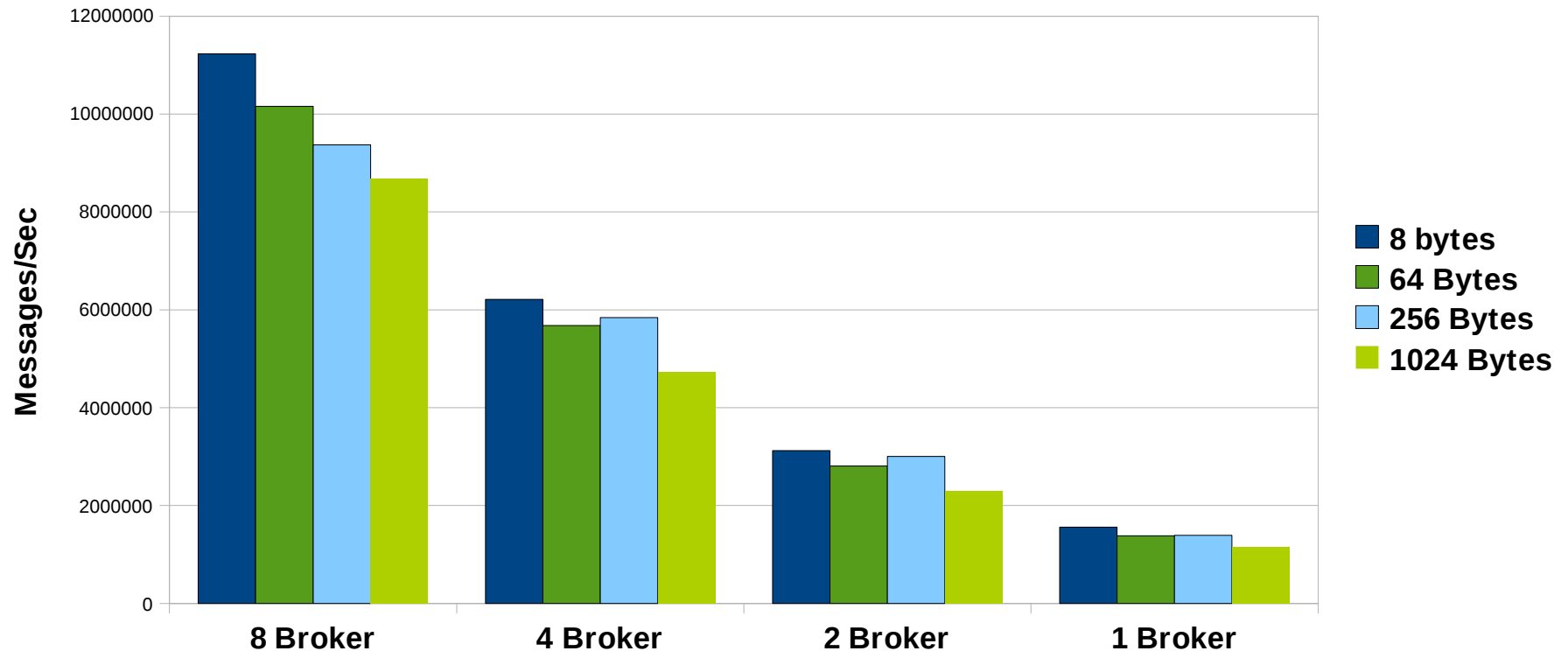
**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# MRG 1.2 / AMQP Scale up.

Single HP Nehalem BL460c 40G Infiniband AMQP Perf test



## Number of Brokers on the Server

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

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**

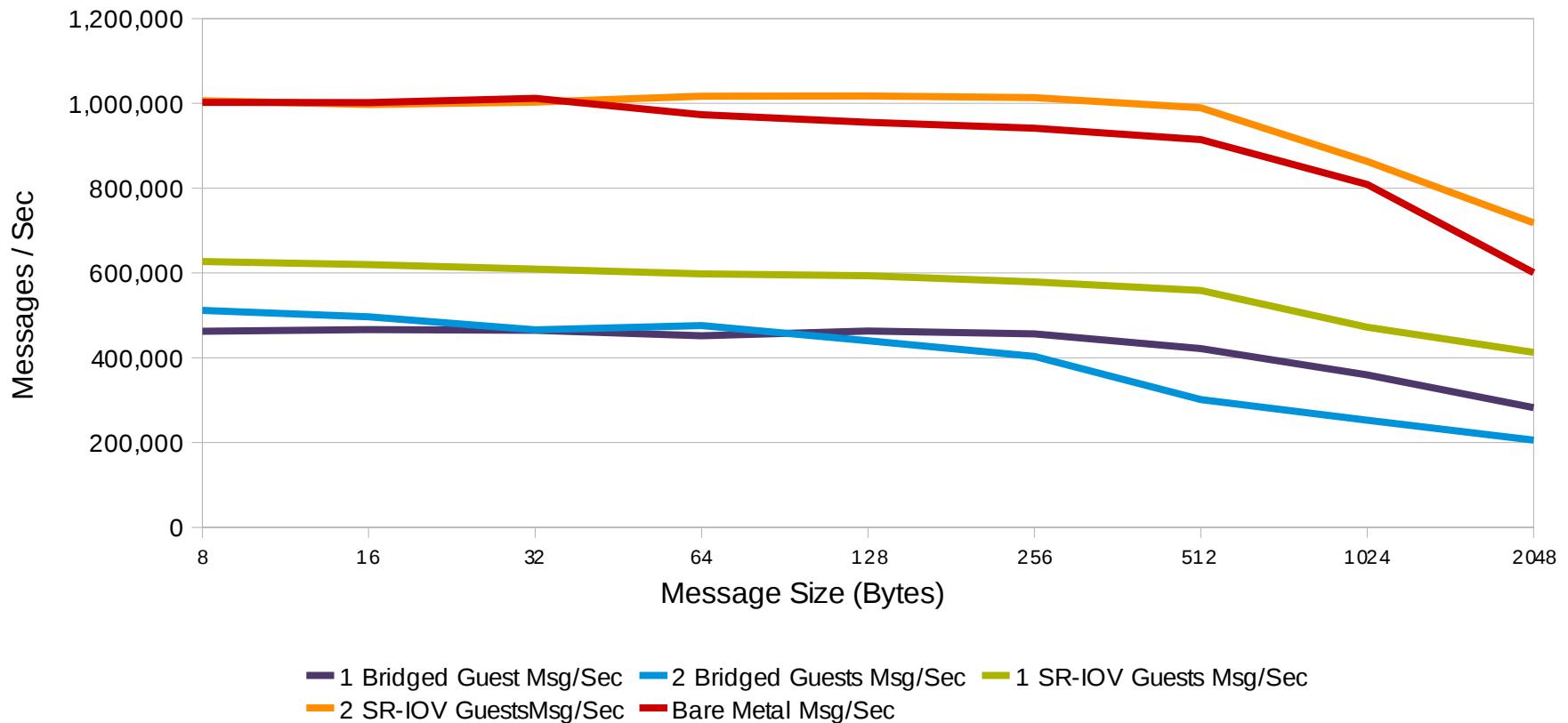
PRESENTED BY RED HAT



# MRG 1.3 / AMQP Scale up, KVM ~5%

Perftest - Bare Metal and KVM

Message Rates with Different Technologies



SUMMIT

JBoss  
WORLD

PRESENTED BY RED HAT

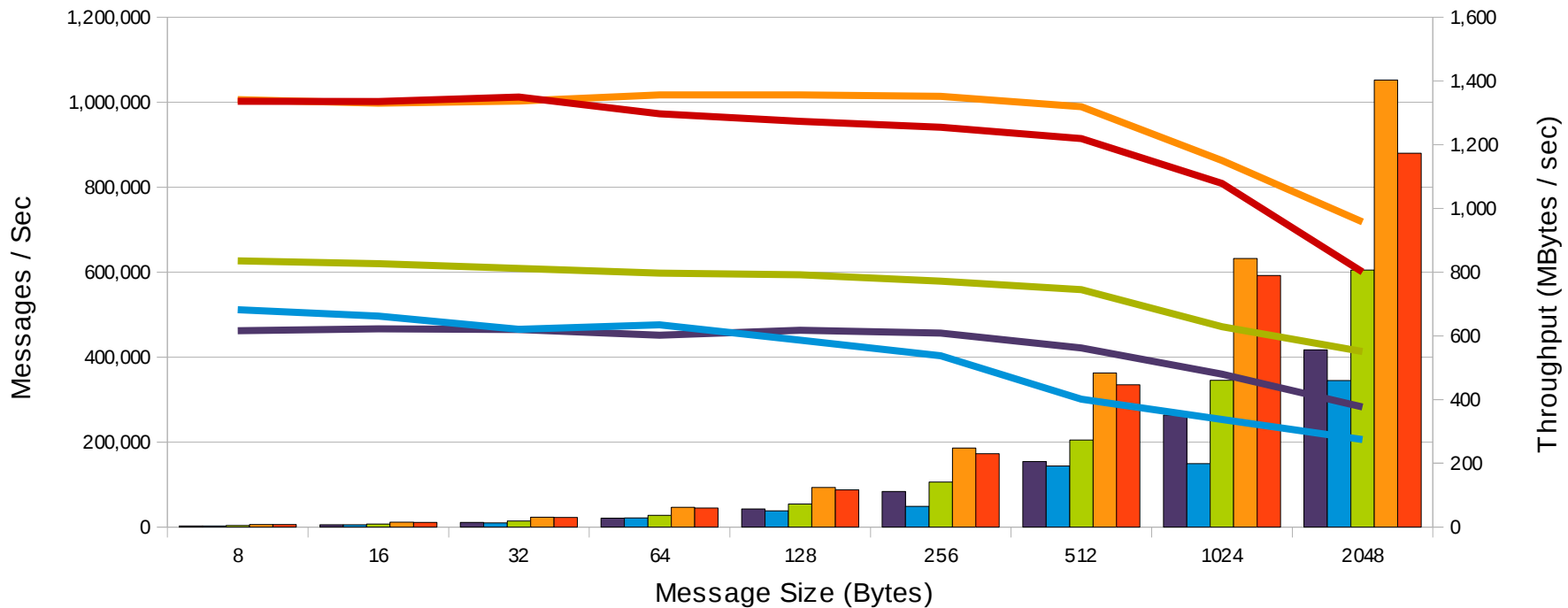


# MRG 1.3 / AMQP Scale up, KVM ~5%

Perftest - Bare Metal and KVM

Lines = Messages / Sec

Columns = MBytes/sec



■ 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**

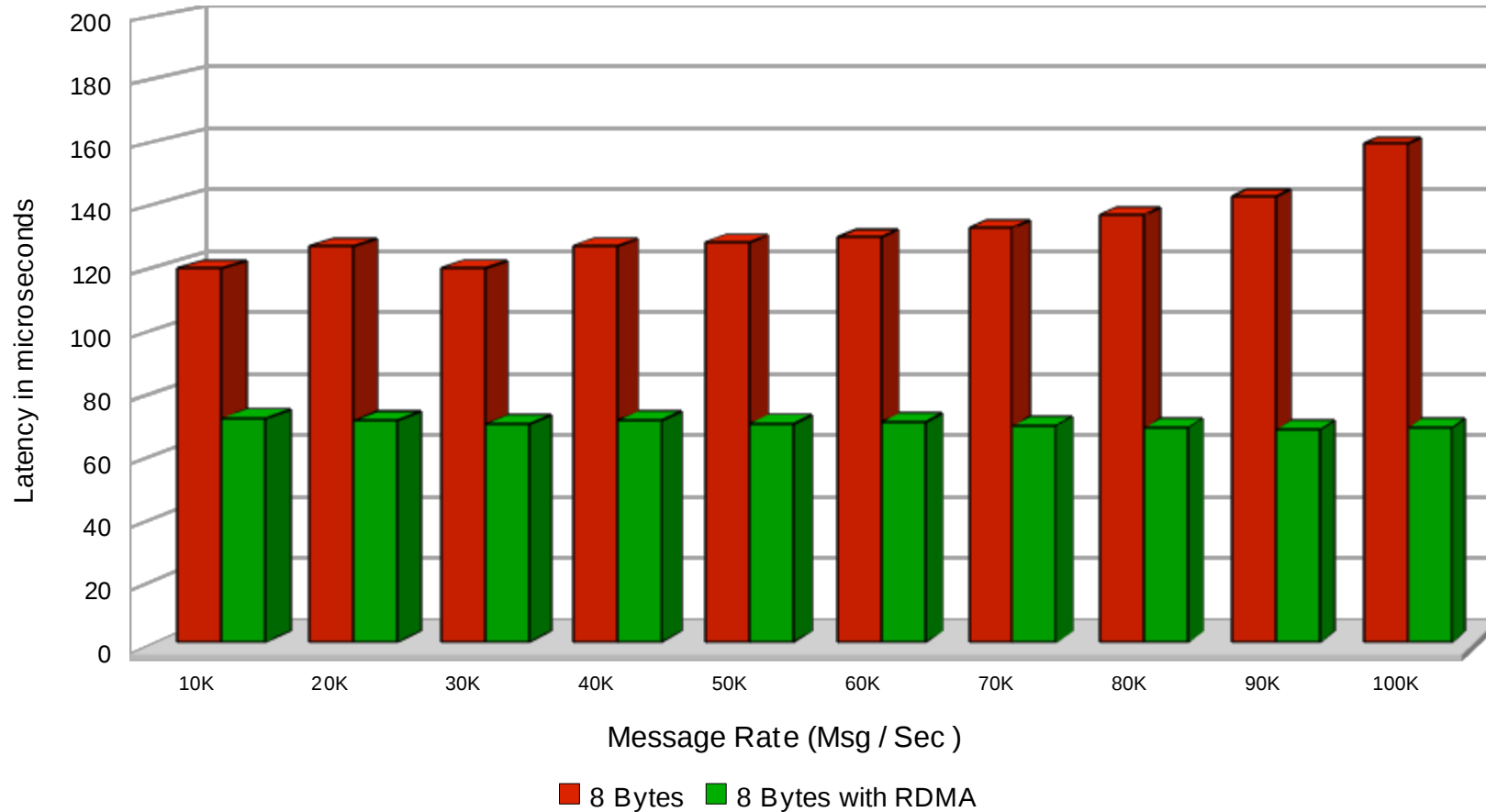
**PRESENTED BY RED HAT**



# MRG 1.3 RHEL6 A quick peak at RoCE (RoE)

10Gbit Mellanox w/wo RDMA - 8 Byte

Latency vs Message Rates - Lower is Better



**SUMMIT**

**JBoss  
WORLD**

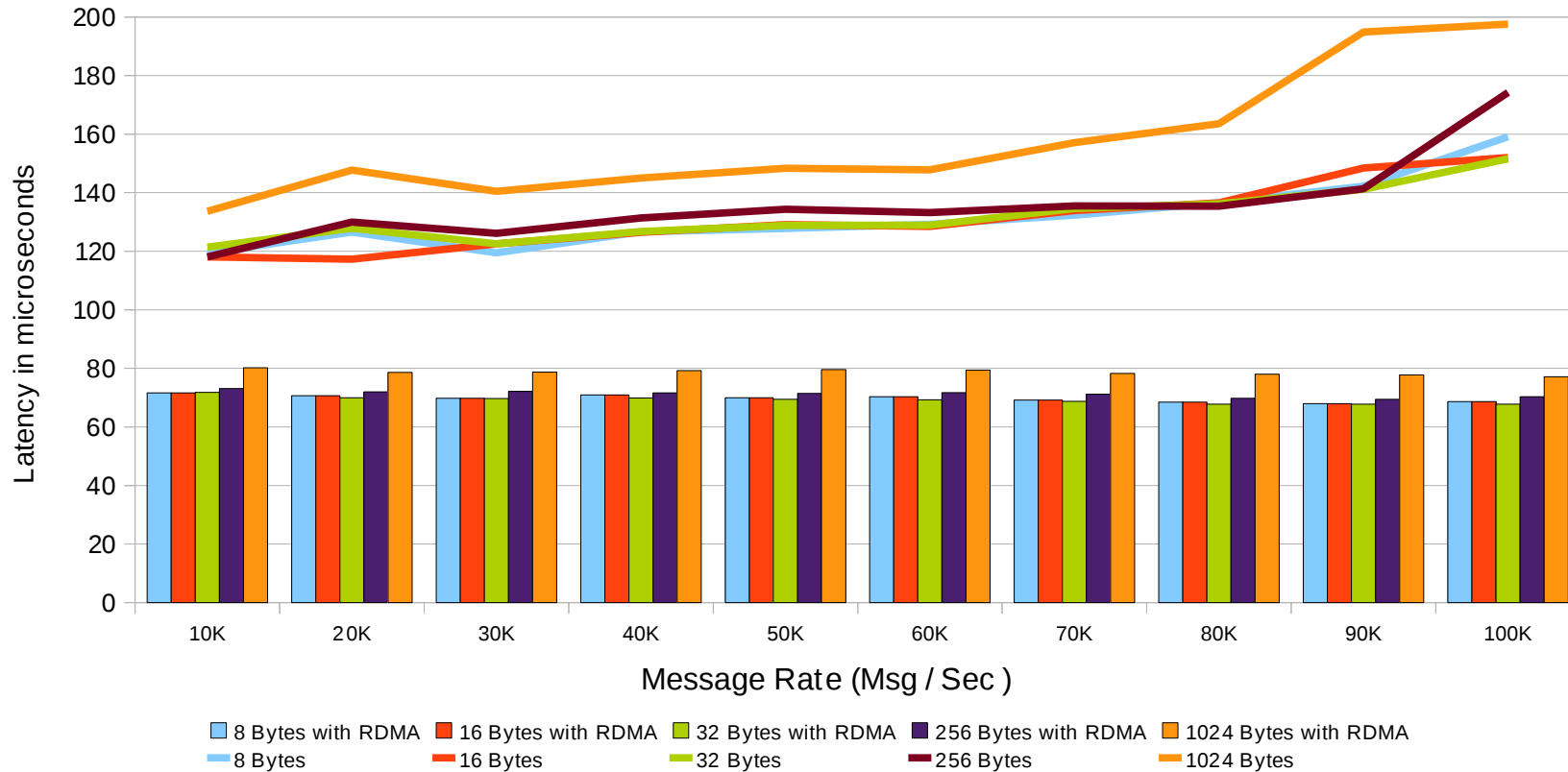
PRESENTED BY RED HAT



# MRG 1.3 RHEL6 A quick peak at RoCE (cont)

10Gbit Mellanox w/wo RDMA

Latency vs Message Sizes



SUMMIT

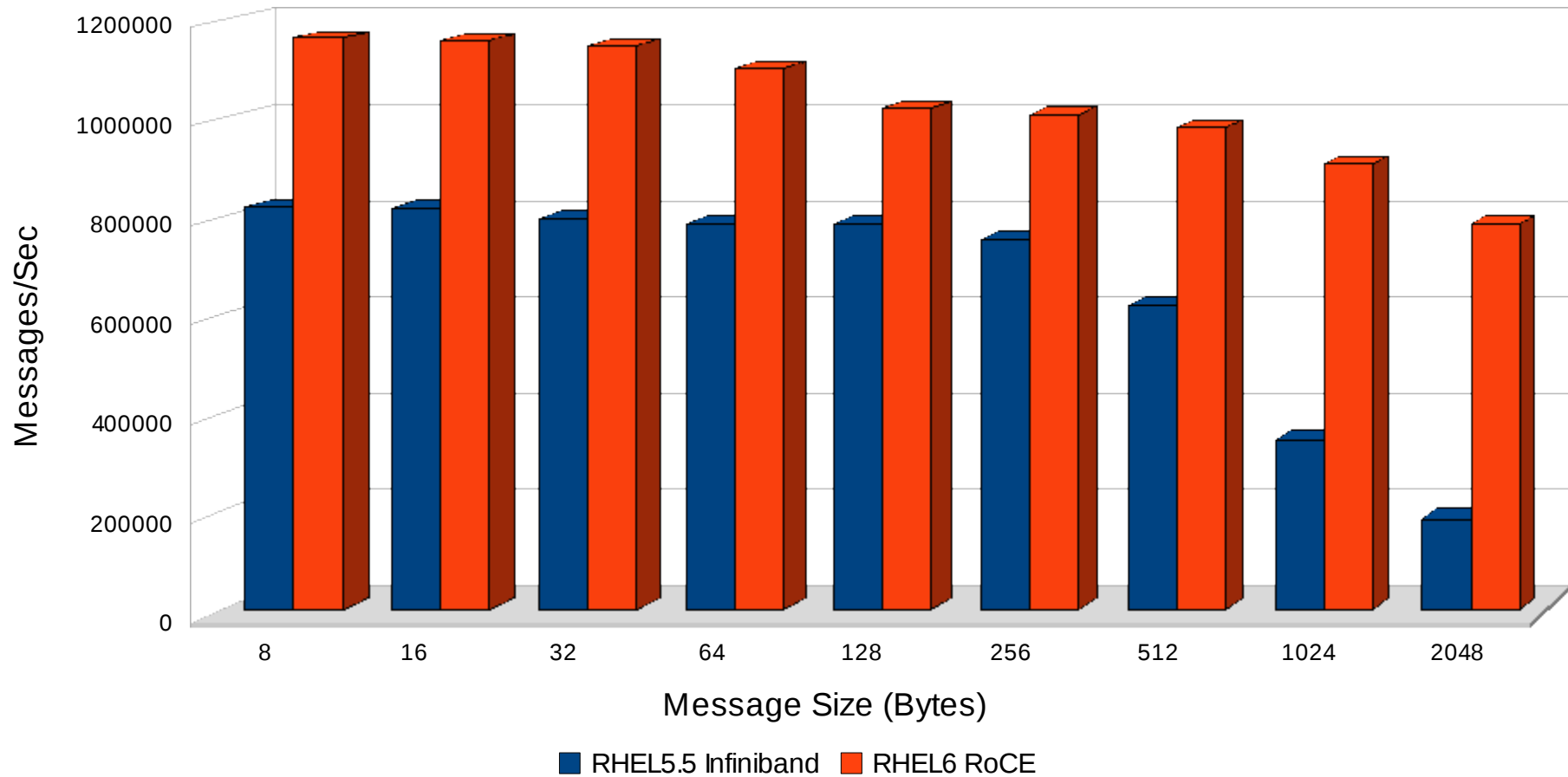
JBoss  
WORLD

PRESENTED BY RED HAT



# RHEL5.5 Infiniband vs. RHEL6 10Gbit – Messages / Sec

Comparing RHEL55 Mellanox Infiniband and RHEL6 Mellanox 10Gb with RoCE



**SUMMIT**

**JBoss  
WORLD**

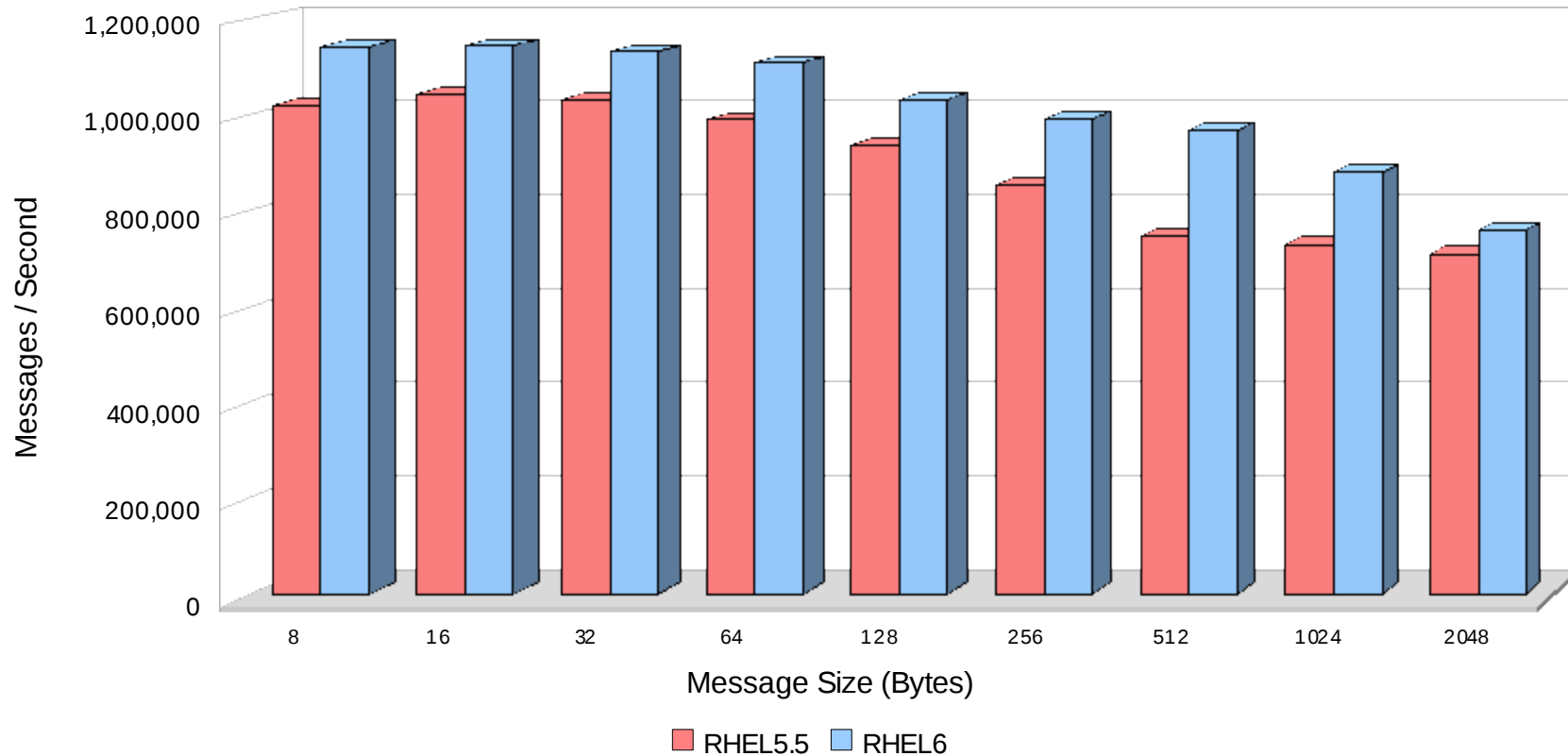
PRESENTED BY RED HAT



# RHEL5.5 to RHEL6

RHEL5 vs RHEL6 (preliminary)

Message Rates



10 Gbit Ethernet (Mellanox)

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# 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

**SUMMIT**

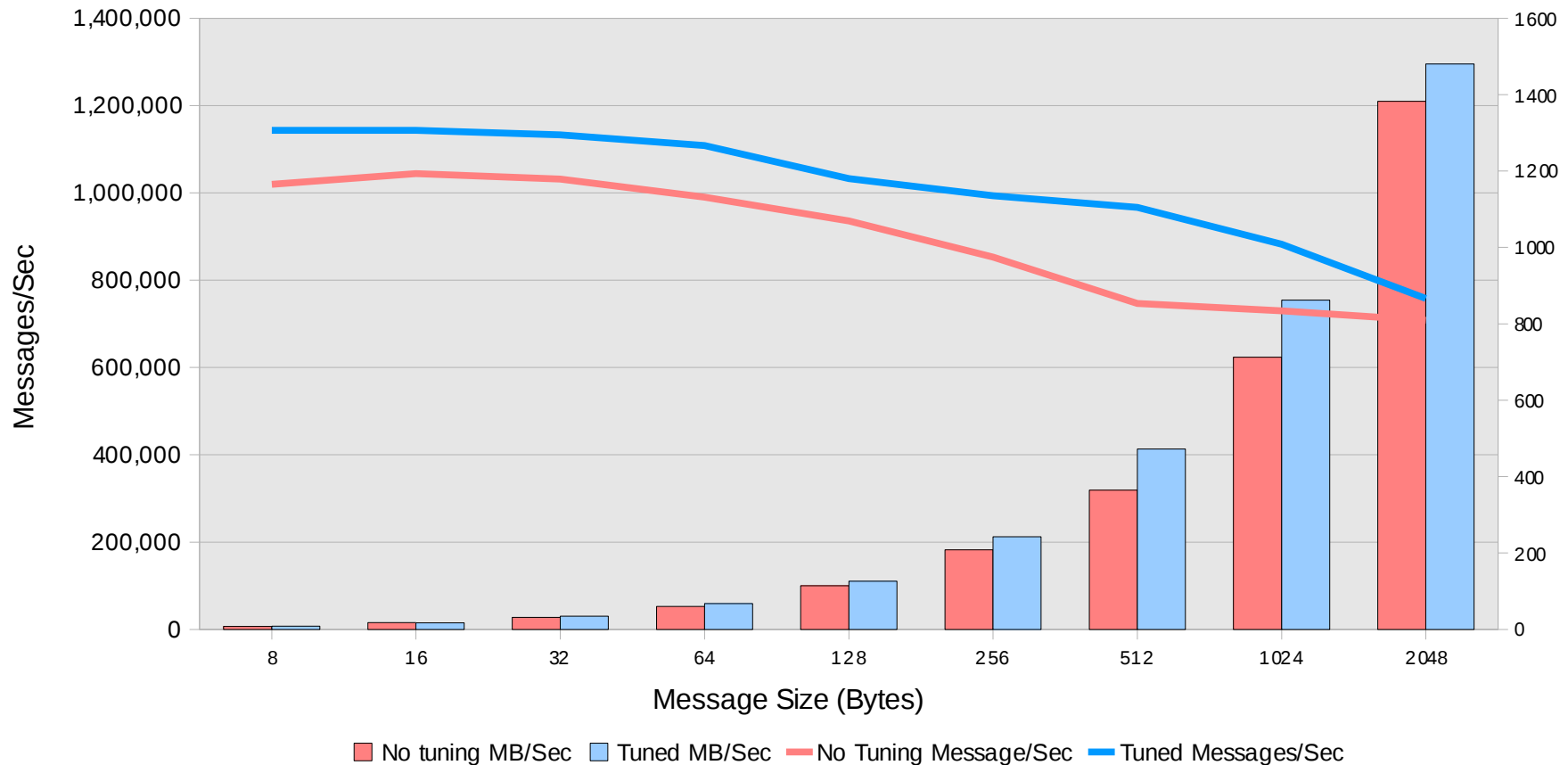
JBoss  
WORLD

PRESENTED BY RED HAT



# The impact of tuning

Intel Westmere Mellanox ConnectX RHEL6 AMQP Perfest



SUMMIT

JBoss  
WORLD

PRESENTED BY RED HAT



# 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

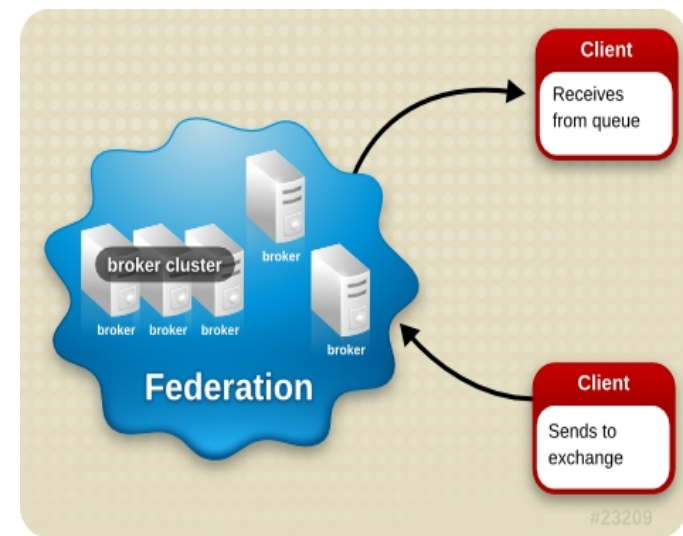
**Federation** and **Disaster Recovery** provides

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*



**SUMMIT**

JBoss  
WORLD

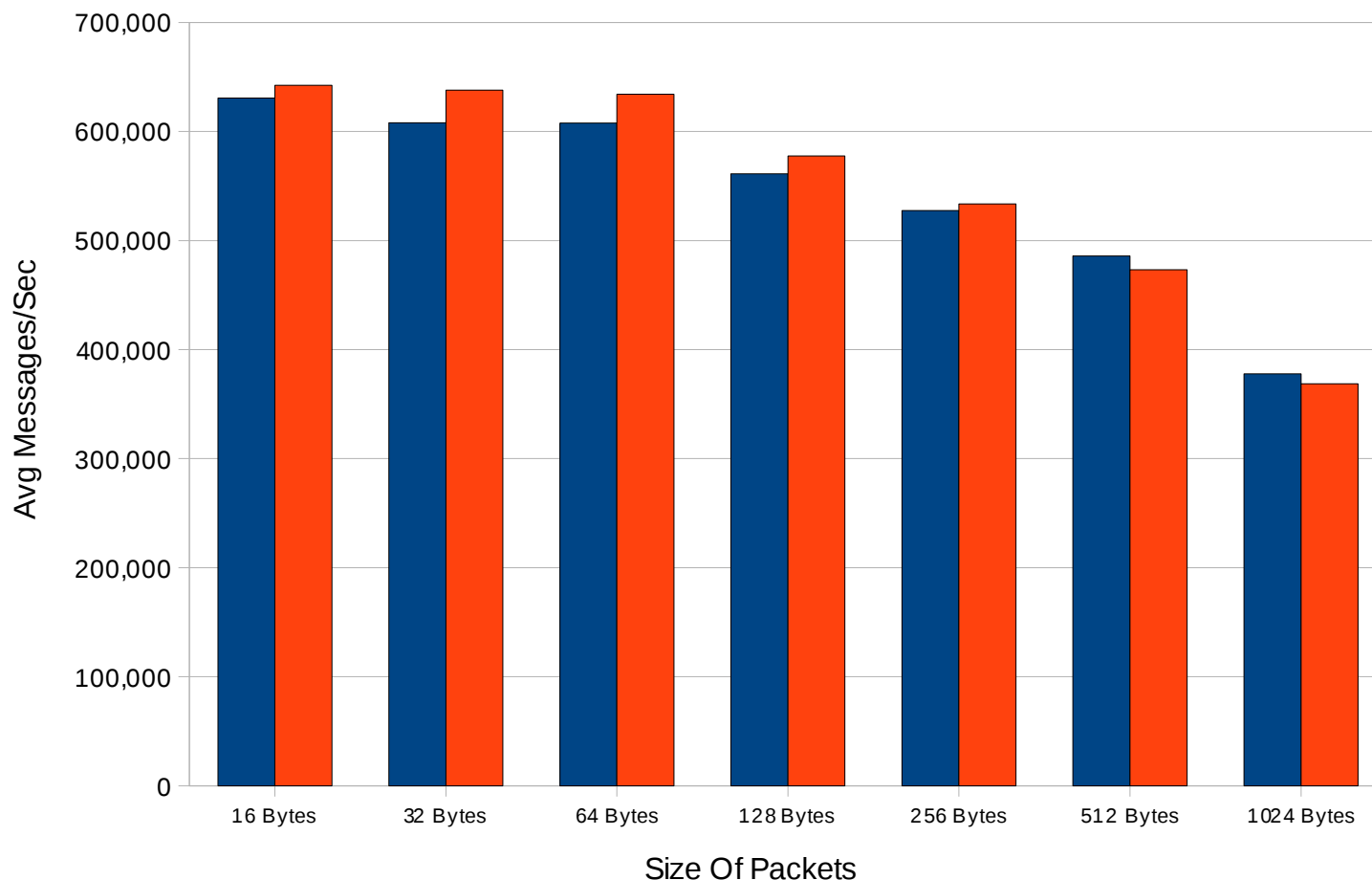
PRESENTED BY RED HAT



# MRG 1.2 Messaging Clustered Throughput with RDMA

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

AMQP Perfetest tesing Corosync



3-Node cluster  
IBM x3550  
2 x 4-core Xeon  
E5420 2.5MHz  
(Harpertown)  
16GB ram 266 MHz  
Mellanox MT25204  
[InfiniHost III Lx HCA]

■ No CoroSnc  
■ CoroSnc

SUMMIT

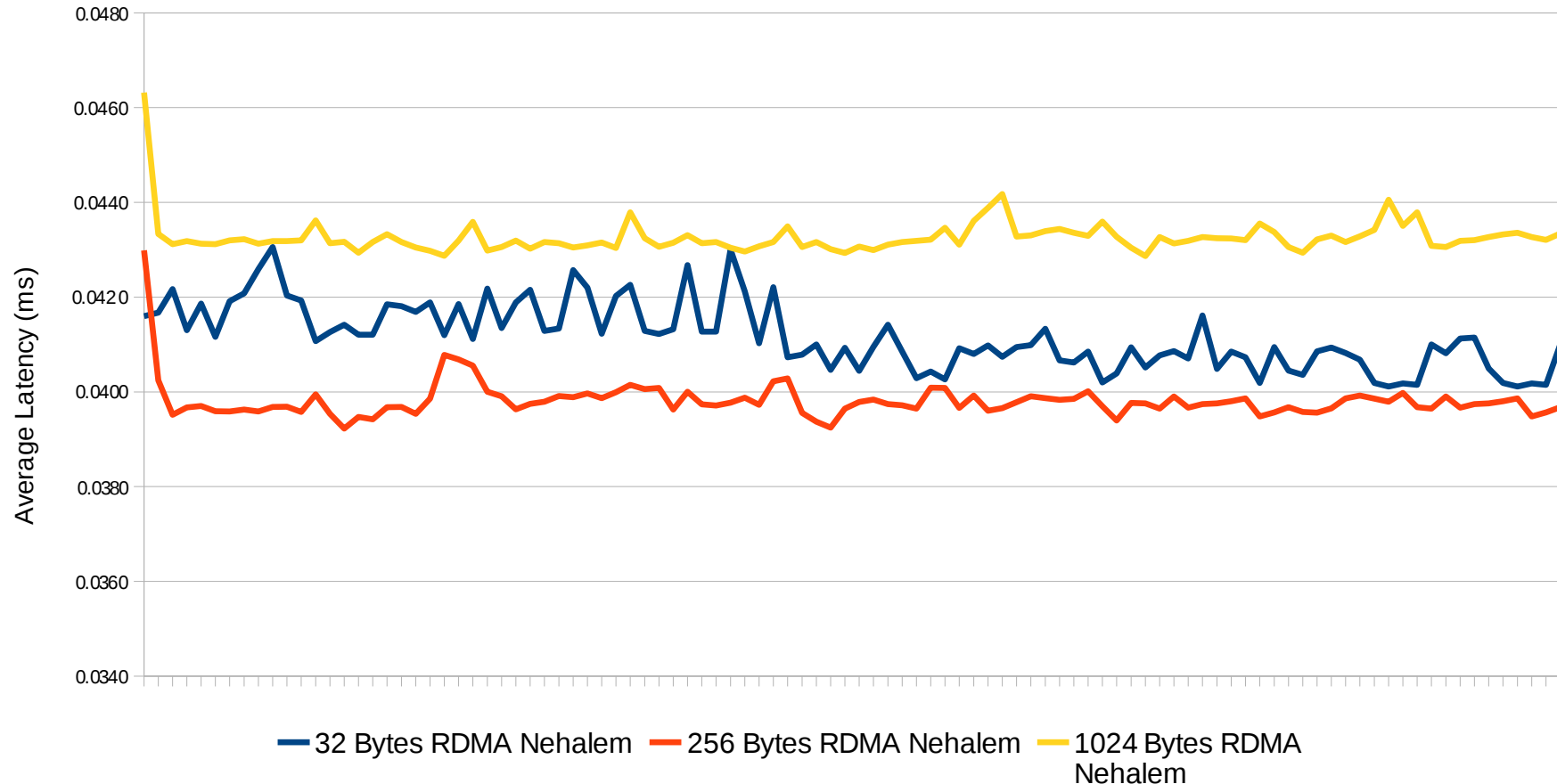
JBoss  
WORLD

PRESENTED BY RED HAT



# MRG 1.2 Messaging Infiniband RDMA Latency: Under 40 Microseconds Reliably Acknowledged

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



**SUMMIT**

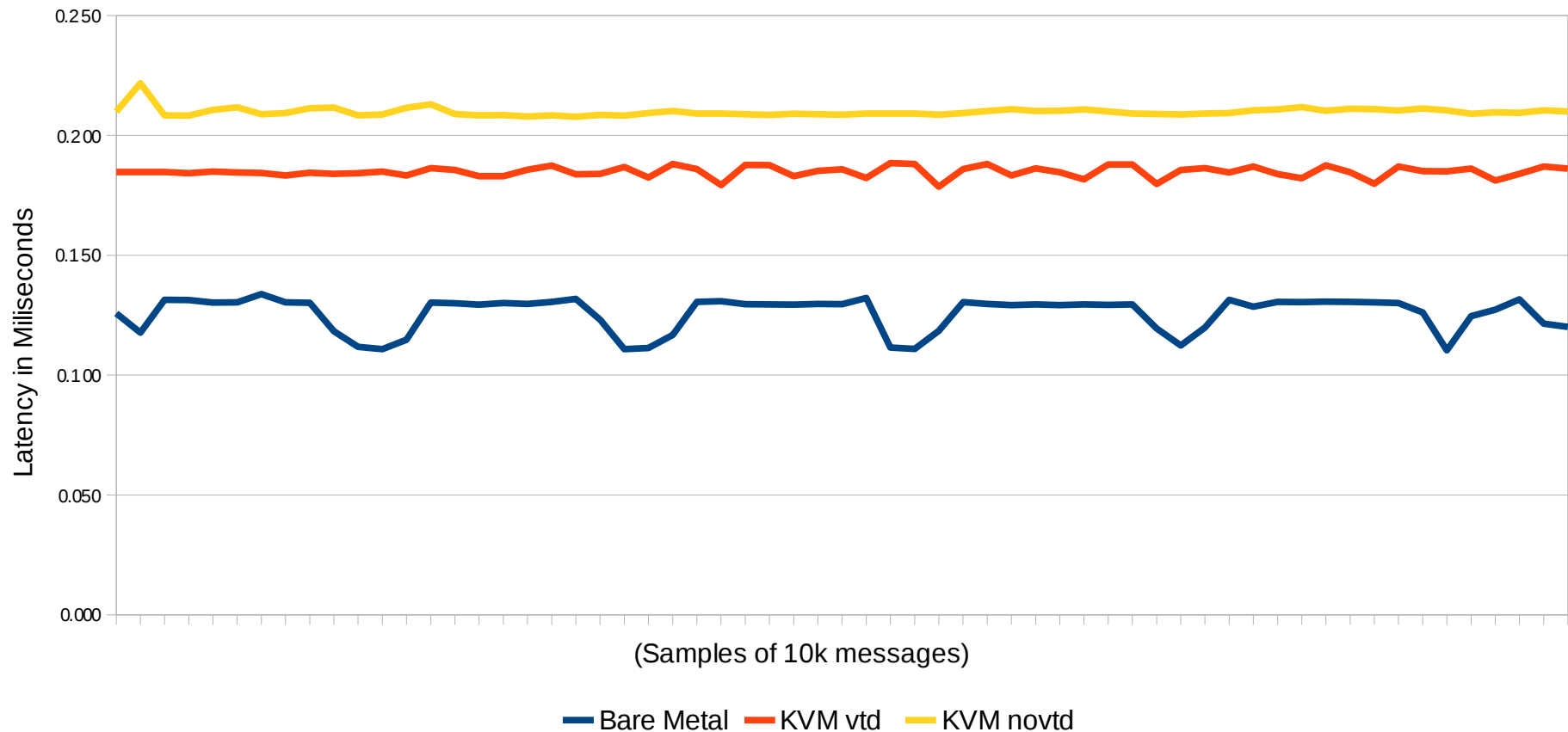
**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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



SUMMIT

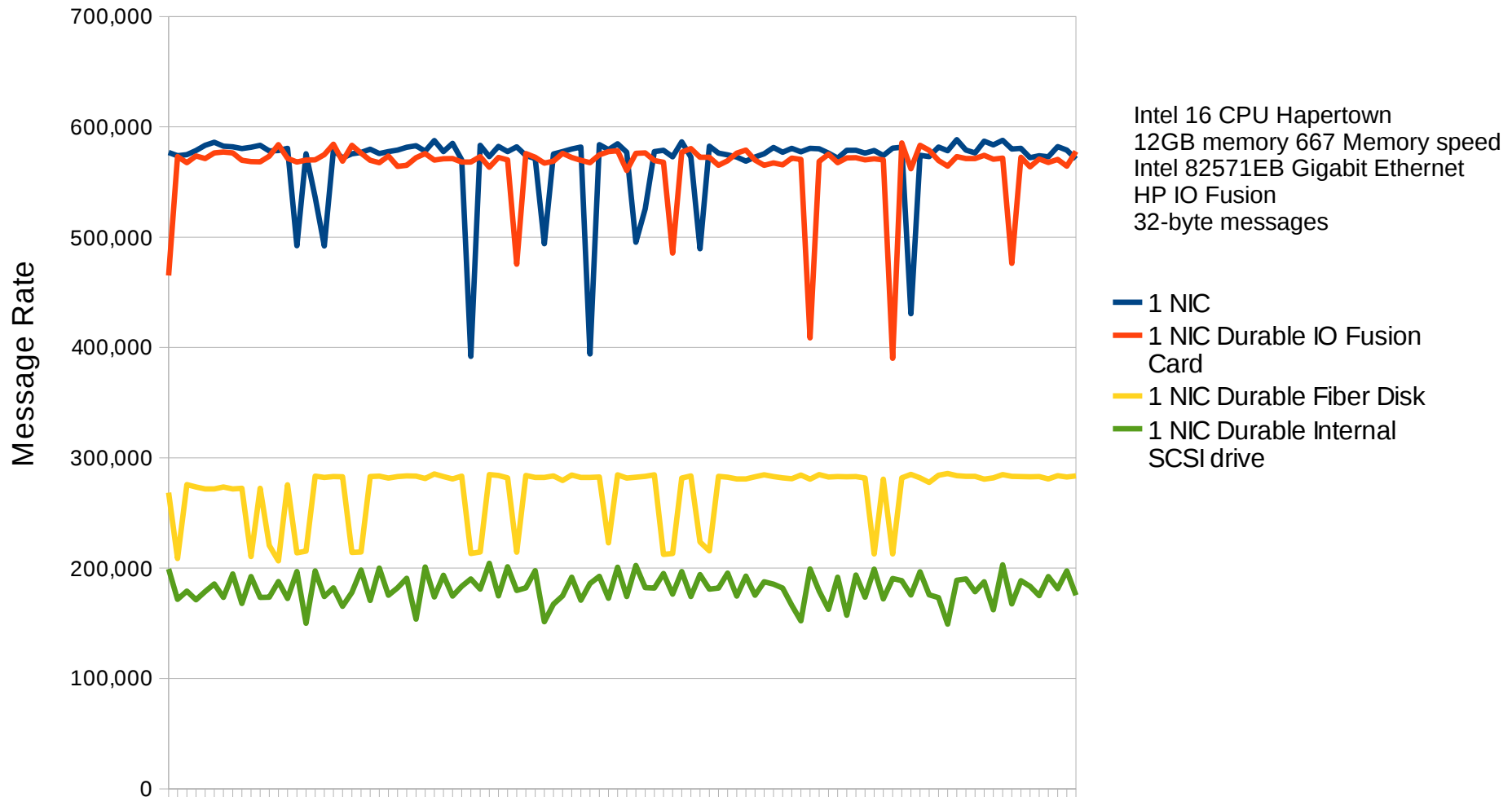
JBoss  
WORLD

PRESENTED BY RED HAT



# MRG 1.2 Messaging Durable Messaging Throughput

MRG Durable Messaging Throughput Across Different Storage Types



SUMMIT

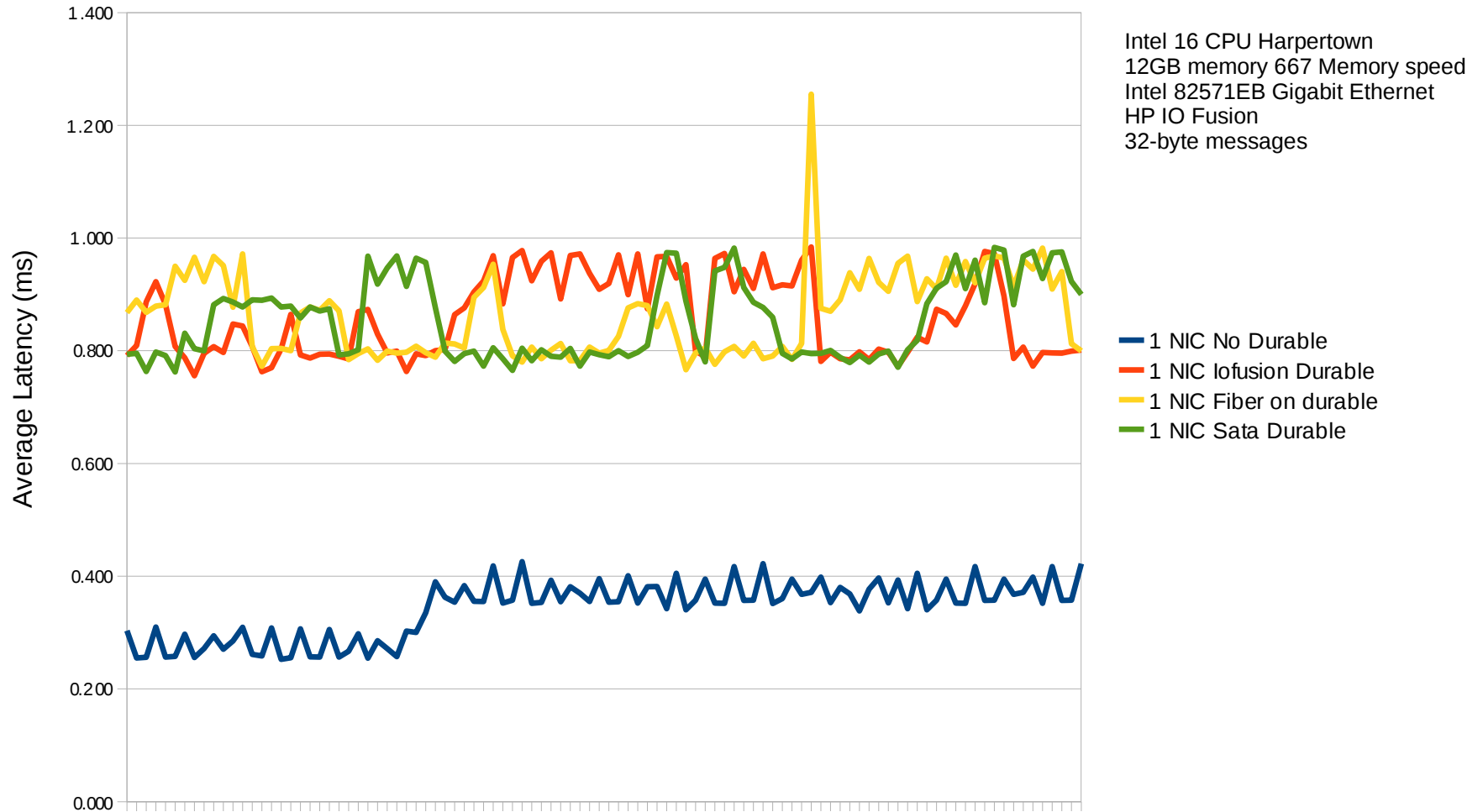
JBoss  
WORLD

PRESENTED BY RED HAT



# MRG 1.2 Messaging Durable Messaging Latency

Latencytest with Durable Store Different Storage Types



SUMMIT

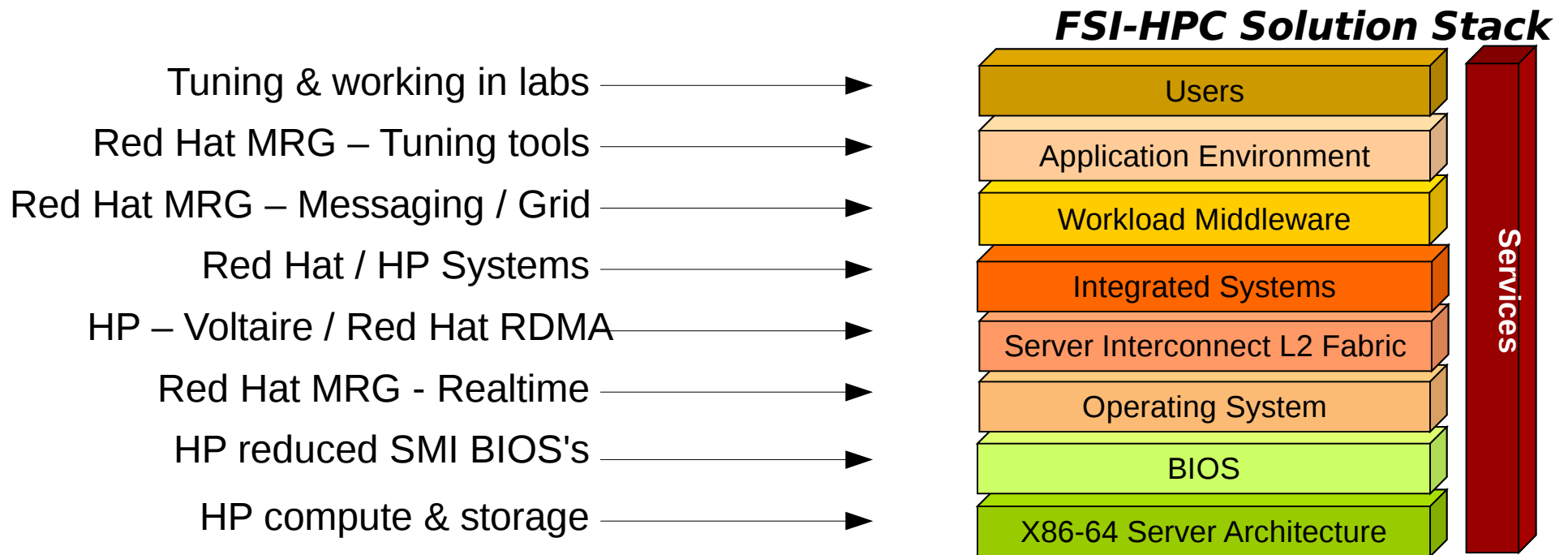
JBoss  
WORLD

PRESENTED BY RED HAT



# All Layers of the stack matter

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



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

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



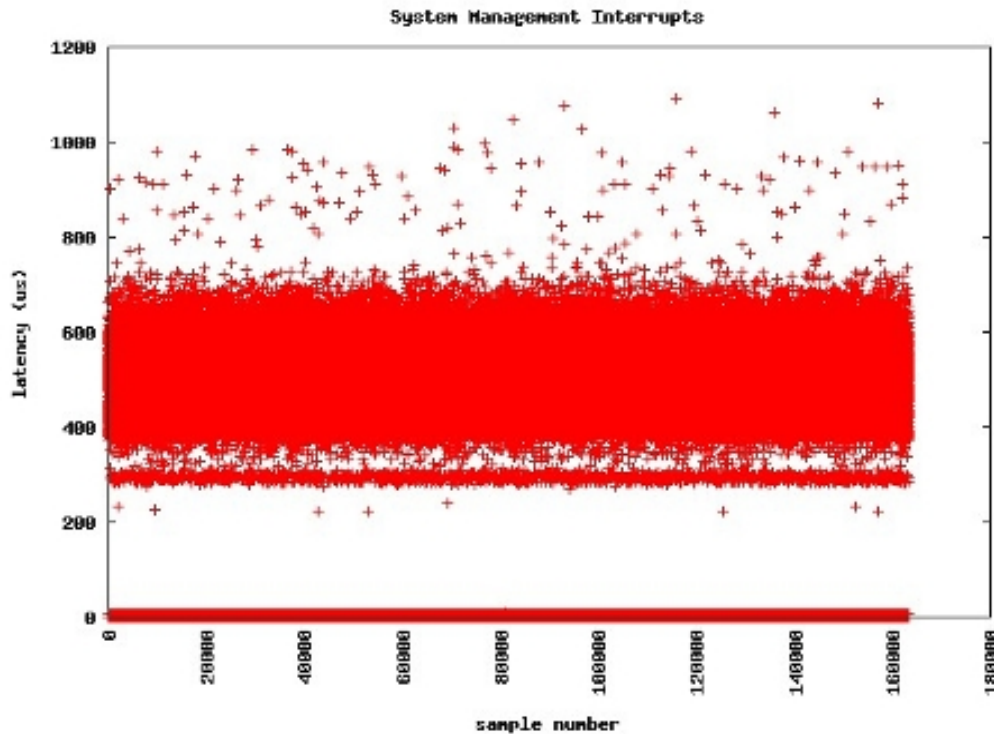
# Dealing with SMIs

## HP BIOS Option for Low Latency Apps (G5,G6,G7)

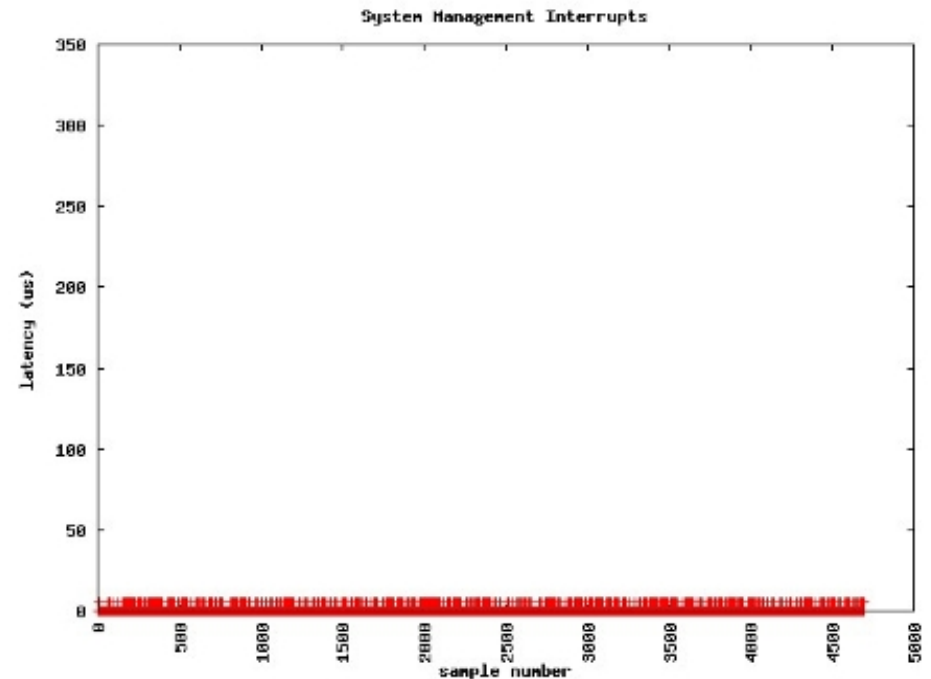
Disable frequent SMIs used for Dynamic Power Savings Mode, CPU Utilization monitoring, P-state monitoring and ECC monitoring

Benefits both RHEL & MRG operating environments.

Latency spikes with standard BIOS settings



Latency when SMIs disabled in BIOS



**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# Real Time

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# MRG – Realtime RHEL on HP systems

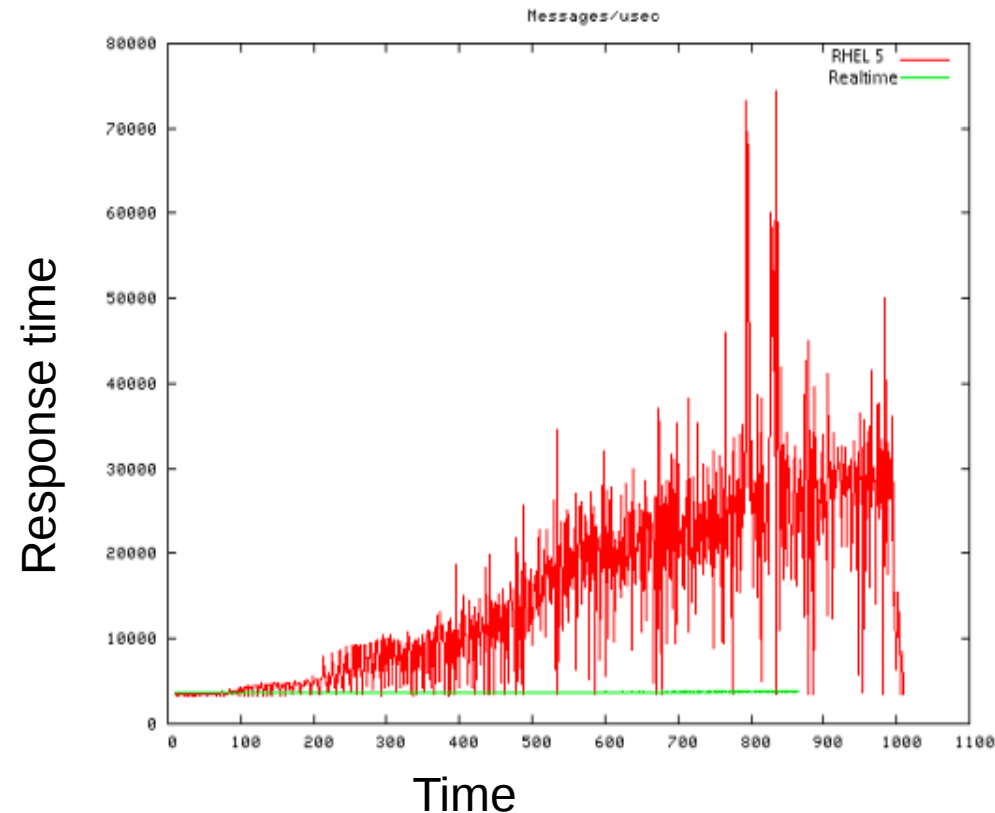
Enables applications and transactions to run predictably, with guaranteed response times

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](http://www.redhat.com/mrg/hardware)



**SUMMIT**

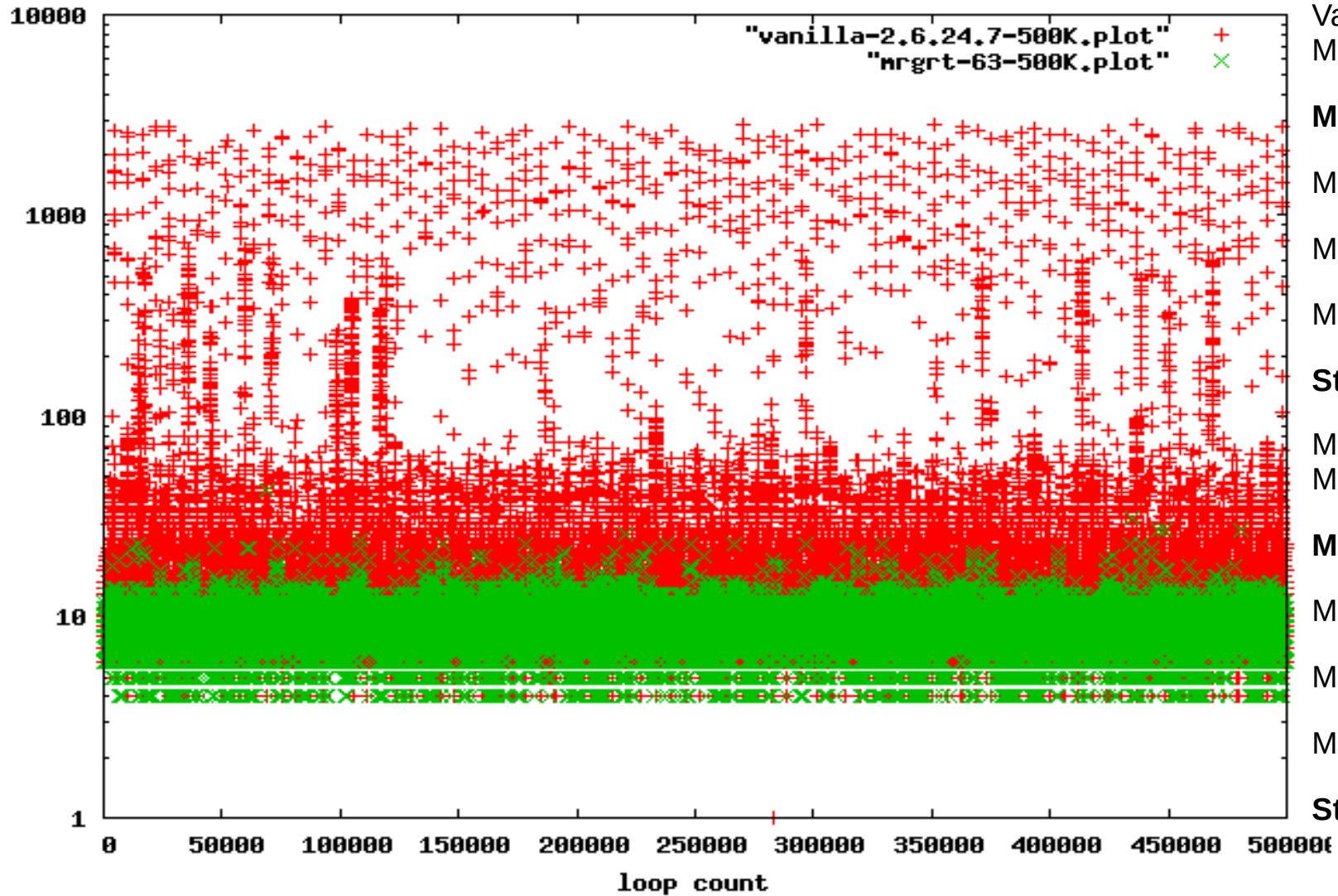
**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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**

SUMMIT

JBoss  
WORLD

PRESENTED BY RED HAT



# 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

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# Grid

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# 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

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# 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

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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**

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# 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

**SUMMIT**

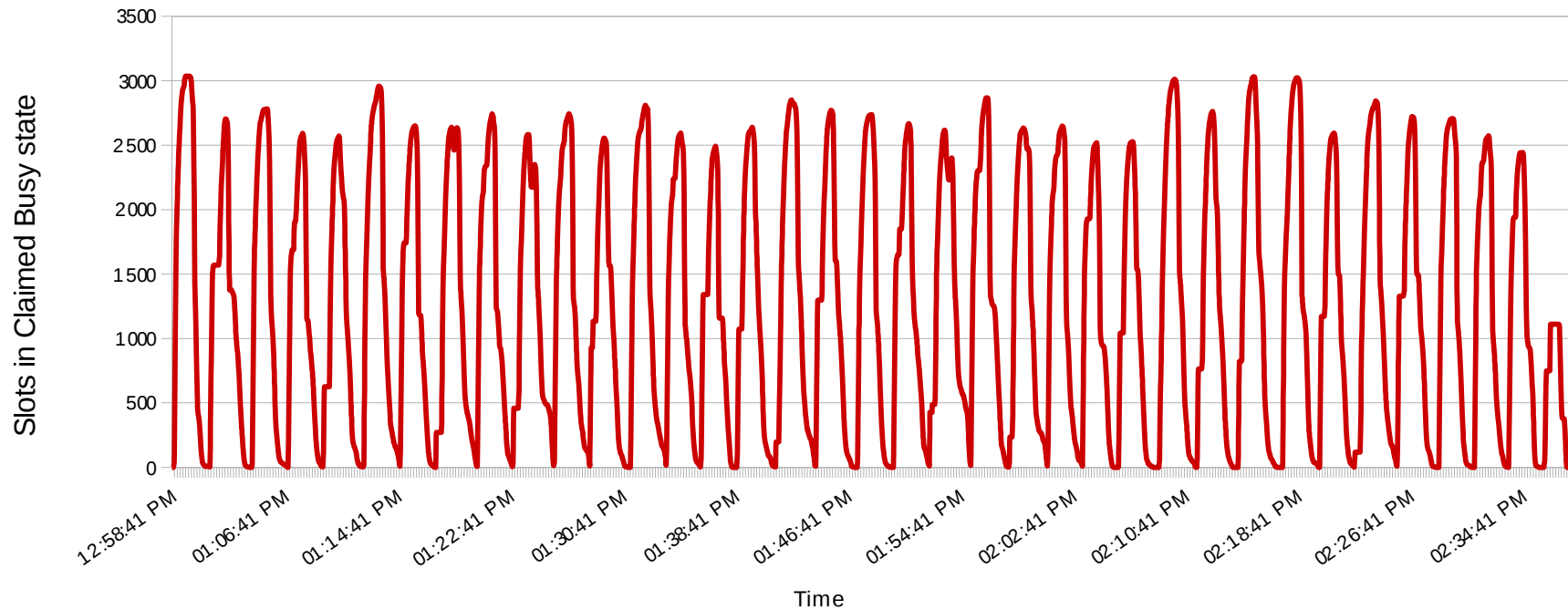
**JBoss  
WORLD**

PRESENTED BY RED HAT



# Grid scalability - Before

Active Jobs - fixed 15 sec duration



**SUMMIT**

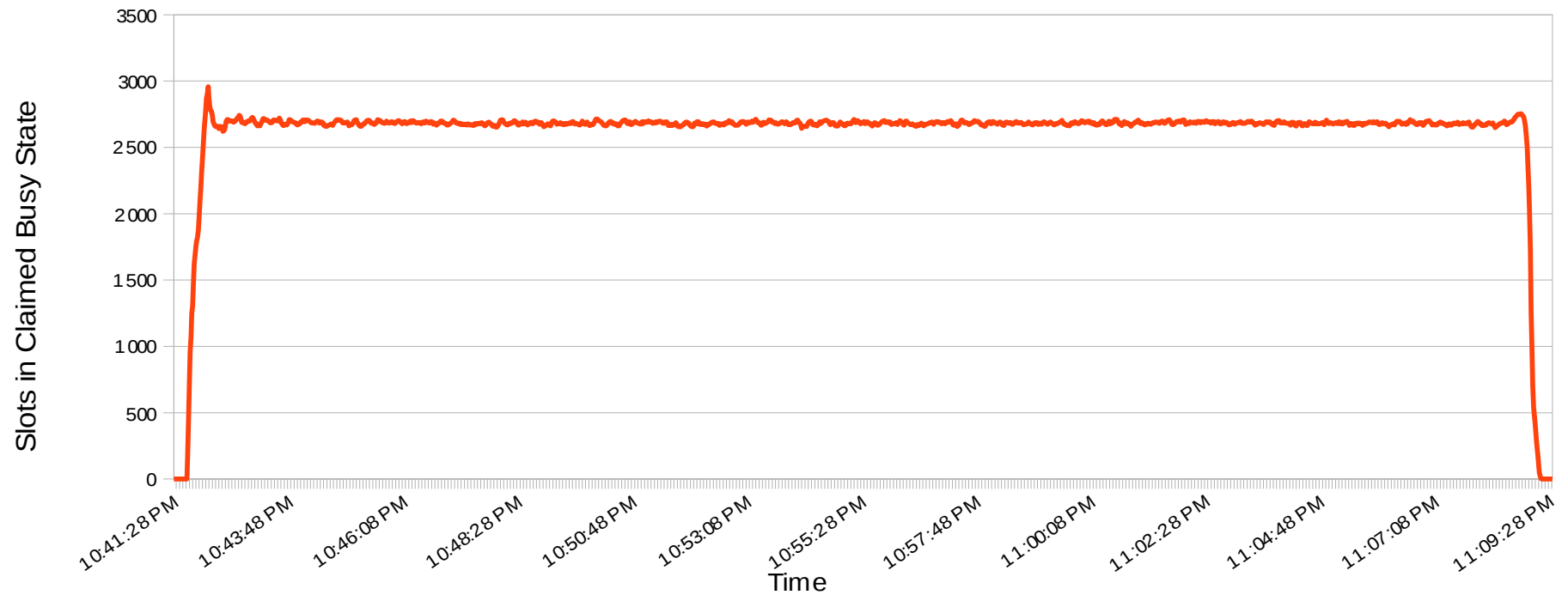
**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# Grid scalability - After

Active Jobs - fixed 15 sec duration  
WORKLIFE=7200 MAX\_ACCEPT=5



**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT



# Tools

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# 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

Socket	Filter	CPU	Usage
Socket 0	<input checked="" type="checkbox"/>	0	0
Socket 0	<input checked="" type="checkbox"/>	1	0
Socket 0	<input checked="" type="checkbox"/>	2	0
Socket 0	<input checked="" type="checkbox"/>	12	0
Socket 0	<input checked="" type="checkbox"/>	13	0
Socket 0	<input checked="" type="checkbox"/>	14	0
Socket 1	<input checked="" type="checkbox"/>	3	21
Socket 1	<input checked="" type="checkbox"/>	4	0
Socket 1	<input checked="" type="checkbox"/>	5	0
Socket 1	<input checked="" type="checkbox"/>	15	0
Socket 1	<input checked="" type="checkbox"/>	16	0
Socket 1	<input checked="" type="checkbox"/>	17	0
Socket 2	<input checked="" type="checkbox"/>	6	0
Socket 2	<input checked="" type="checkbox"/>	7	0
Socket 2	<input checked="" type="checkbox"/>	8	0
Socket 2	<input checked="" type="checkbox"/>	18	0
Socket 2	<input checked="" type="checkbox"/>	19	0
Socket 2	<input checked="" type="checkbox"/>	20	0
Socket 3	<input checked="" type="checkbox"/>	9	0
Socket 3	<input checked="" type="checkbox"/>	10	0
Socket 3	<input checked="" type="checkbox"/>	11	0
Socket 3	<input checked="" type="checkbox"/>	21	0
Socket 3	<input checked="" type="checkbox"/>	22	0
Socket 3	<input checked="" type="checkbox"/>	23	0

IRQ	PID	Policy	Priority	Affinity	Events	Users
17	1473	FIFO	50	1,13	51525	megasas
22	1321	FIFO	50	1,13	858	uhci_hcd:usb2,uhci_hcd:usb3,uhci_hcd:usb4,uhci_h
23	1270	FIFO	50	2,14	30	ehci_hcd:usb1
2229	6529	FIFO	50	0	46098	eth3(e1000)
2230	6320	FIFO	50	13	1624017	eth2(e1000)
2231	6148	FIFO	50	0-23	1	eth0:lsc
2232	6147	FIFO	50	13	56938	eth0:v15-Rx
2233	6146	FIFO	50	2	55448	eth0:v14-Rx
2234	6145	FIFO	50	12	55406	eth0:v13-Rx
2235	6144	FIFO	50	14	56700	eth0:v12-Rx
2236	6143	FIFO	50	1	56803	eth0:v11-Rx
2237	6142	FIFO	50	14	58014	eth0:v10-Rx
2238	6141	FIFO	50	1	57371	eth0:v9-Rx
2239	6140	FIFO	50	14	58816	eth0:v8-Rx
2240	6139	FIFO	50	0	60573	eth0:v7-Rx

PID	Policy	Priority	Affinity	VolCtxtSwitch	NonVolCtxtSwitch	Command Line
1	OTHER	0	0-23	20259	2744	init [3]
2	OTHER	0	0-23	530	1320	kthreadd
3	FIFO	99	0	702	0	migration/0
4	FIFO	99	0	2	0	posixcpumr/0
5	FIFO	50	0	2	0	sirq-high/0
6	FIFO	50	0	90298186	0	sirq-timer/0
7	FIFO	50	0	15	0	sirq-net-tx/0
8	FIFO	50	0	133467	0	sirq-net-rx/0
9	FIFO	50	0	1055	0	sirq-block/0
10	FIFO	50	0	567	0	sirq-tasklet/0

SUMMIT

JBoss  
WORLD

PRESENTED BY RED HAT



# Take Home Exam

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# An example to try

## Broker

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

## Driver

```
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  
&
```

**SUMMIT**

JBoss  
WORLD

PRESENTED BY RED HAT



Questions ?

**SUMMIT**

**JBoss  
WORLD**

**PRESENTED BY RED HAT**



# FOLLOW US ON TWITTER

[www.twitter.com/redhatsummit](http://www.twitter.com/redhatsummit)

## TWEET ABOUT IT

[#summitjbw](https://twitter.com/summitjbw)

## READ THE BLOG

<http://summitblog.redhat.com/>

**SUMMIT**

**JBoss  
WORLD**

PRESENTED BY RED HAT

