Introduction to Realtime Linux

Bryan Che

Southern California Linux Expo

What is Realtime Performance?

- Realtime is not about faster performance
- Realtime is not about higher throughput
- Realtime may even be slower than non-realtime!

So, then what is realtime performance, and why would anyone want it?

Realtime is About Deterministic Performance

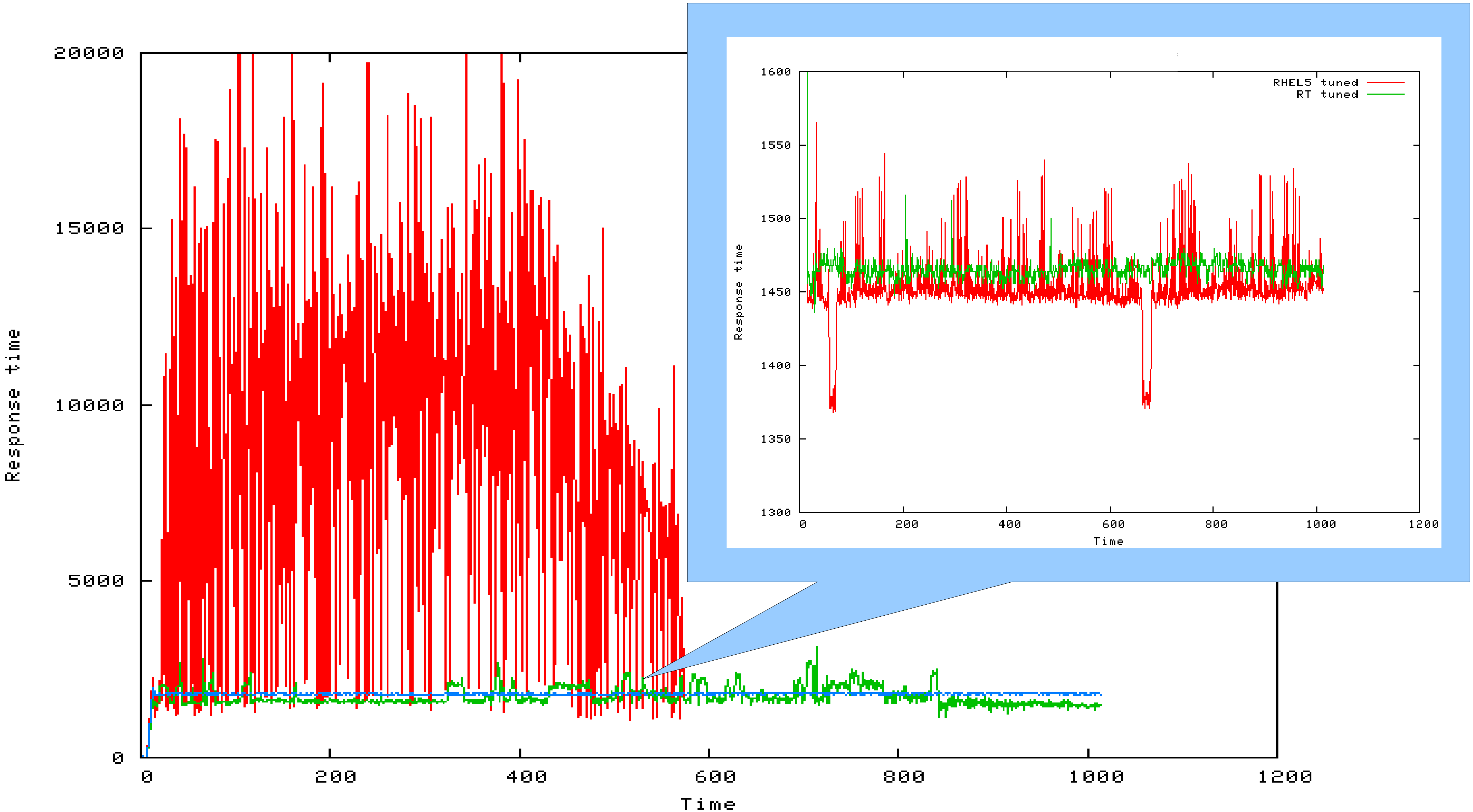
- Non-Realtime
  - Average times highly variable
  - No prioritization of traffic
  - Favors throughput

- Realtime
  - Highly deterministic time
  - Prioritization of traffic
  - Sacrifices throughput for low, deterministic latency

Why Realtime Linux?

- **Enables applications and transactions to run predictably, with guaranteed response times**
  - Provides microsecond accuracy
- **Provides competitive advantage & meets Service Level Agreements**
  - Multimedia: precise timing and synchronization
  - Travel web site: missed booking
  - Program trading: missed trades
  - Command & Control: life & death
- **Industries particularly interested in Realtime Linux include Government, Defense, Financial Services, Telecommunications, Manufacturing, etc.**

What Does Realtime Linux Provide?



What is Realtime Linux?

- **Patchset developed at kernel.org community which adds to the standard Linux kernel:**
  - Full preemption
  - Threaded IRQs
- **Breaks down long-running, un-preemptable code paths to provide responsive behavior**
- **Large patchset:**
  - Diffstat of patch set 2.6.26.6-rt11 shows:
    - 664 files changed, 37806 insertions(+), 4217 deletions(-)
- **Key Developers**
  - **Ingo Molnar** (Maintainer, Red Hat), **Thomas Gleixner** (Red Hat contractor), **Steven Rostedt** (Red Hat), **Paul McKenney** (IBM), **John Stultz** (IBM), **Gregory Haskins** (Novell), **Peter Zijlstra** (Red Hat), etc

Key Changes in the Realtime Kernel

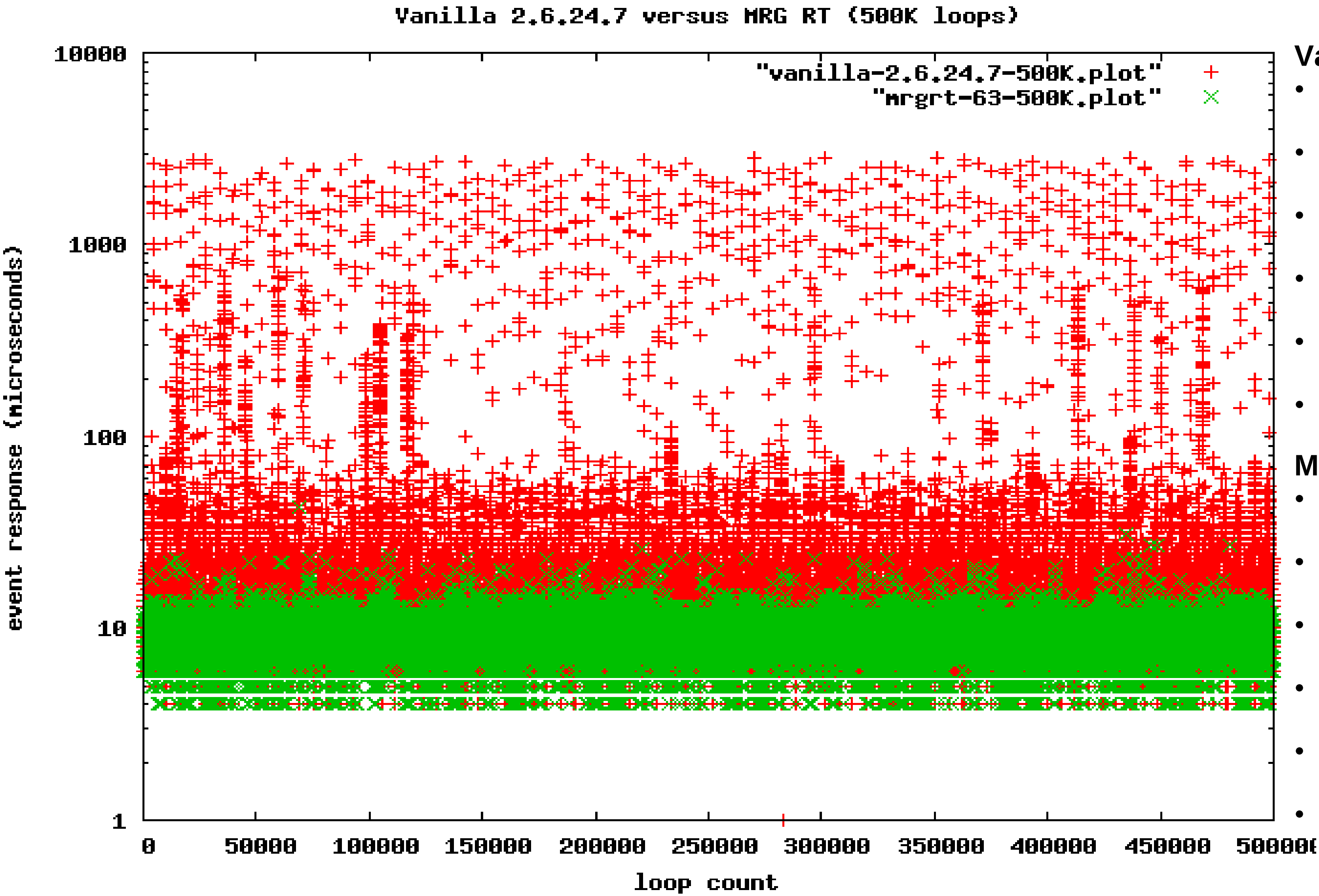
- **Preemption**
  - Most locks converted to rt_mutex
  - priority inheritance for mutexes
  - threaded interrupt handlers (both hard and soft)
  - Spinlocks can sleep
  - Interrupts not turned off for almost all operations
- **high-resolution timers**
- **Completely Fair Scheduler (CFS) ***
- **Read-Copy-Update (RCU) ***
- **Ftrace tracing logic**

Now in Upstream Kernel

Key Changes in the C Library for Realtime

- **pthread_mutex_t has kernel support for PRIO_INHERIT**
  - *Priority Inheritance* is a mechanism used to avoid the deadlock condition known as Priority Inversion
  - The realtime kernels implement priority inheritance (PI) in *futexes* (fast user-space mutexes) used by pthreads
- **Fast user-space mutexes (futexes) used for pthread mutexes**
- **POSIX interfaces to scheduler APIs**
  - sched_*
- **Timer interfaces**
- ***Note that you don't have to have a realtime kernel for most of these APIs to work***

Reducing Scheduling Latency



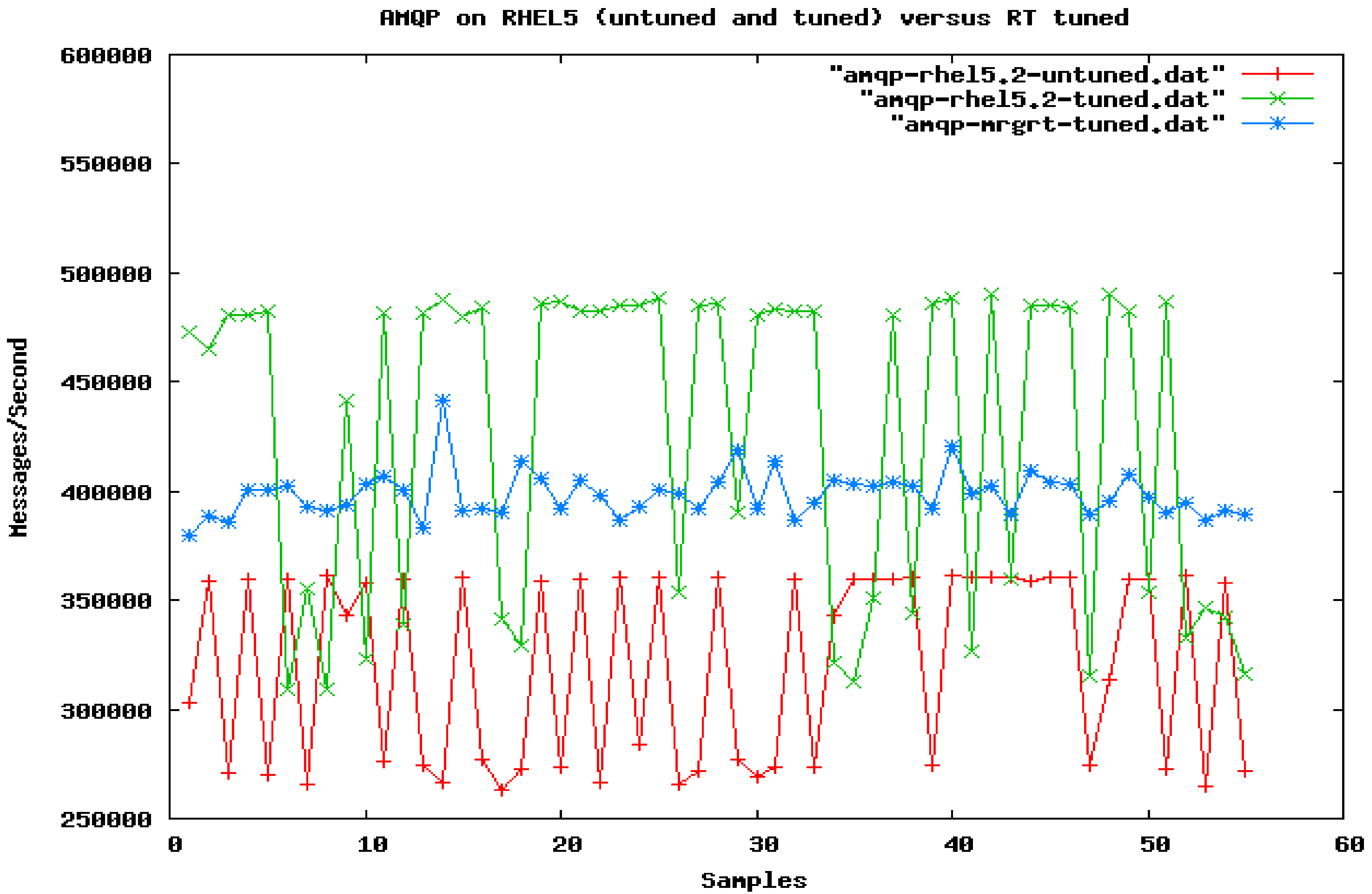
Vanilla

- 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

Realtime Throughput

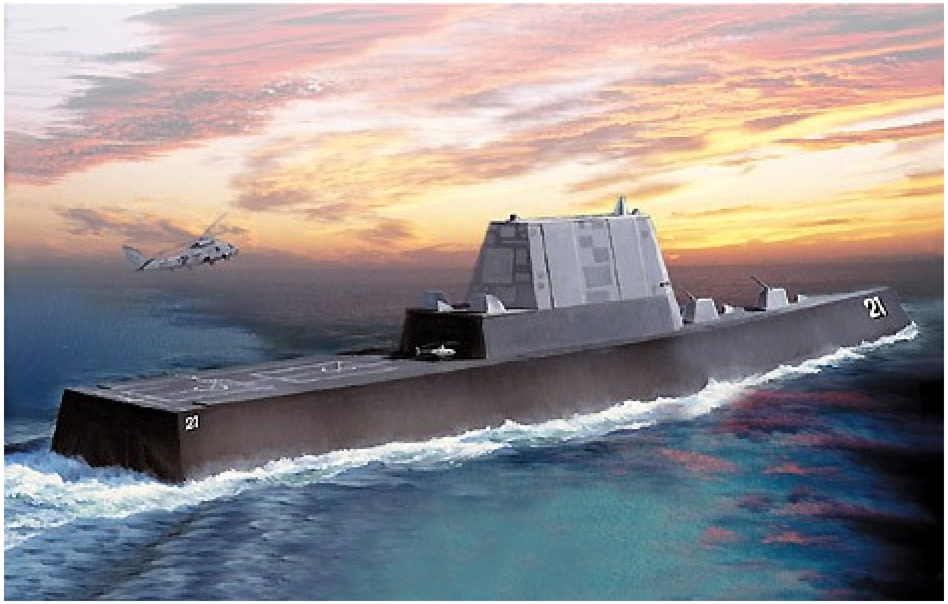


Realtime Performance Tools

- **FTrace**
  - 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
- **TUNA**
  - Dynamically control tuning parameters like process affinity, parent & threads, scheduling policy, device IRQ priorities, etc.
- **Standard Linux performance tools**
  - Gdb, OProfile Frysk – source level debuggers & profiler
  - SystemTap, kprobe – kernel event tracing and dynamic data collection
  - kexec/kdump standard kernel dump/save core capabilities

Realtime Java With Realtime Linux

- **Standard Java deployments typically have highly undeterministic performance—especially because of garbage collection**
- **JSR 1 provides a realtime specification for Java and realtime JVMs**
  - Requires an underlying realtime operating system to provide *priority inheritance* and *preemption*—like realtime Linux!
  - Provides deterministic garbage collection, realtime threads, and deterministic performance



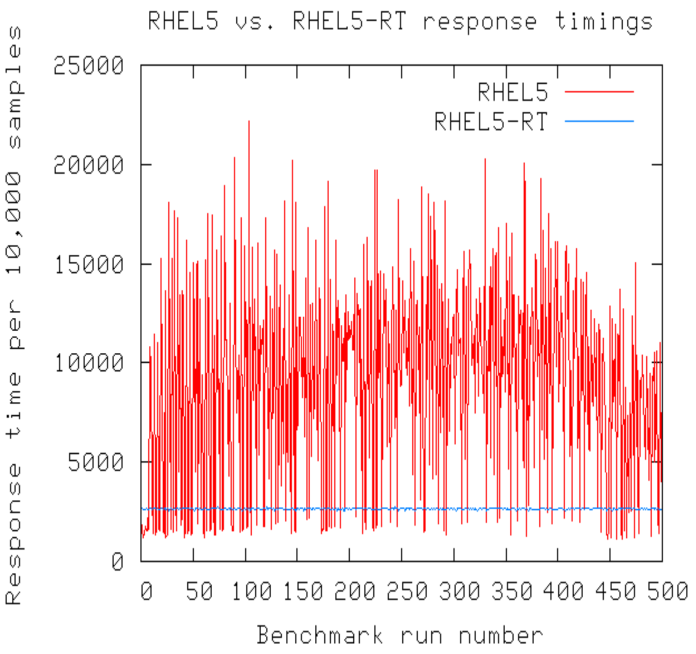
Red Hat and IBM have partnered to deliver Realtime Java on Realtime Linux for the US Navy DDG 1000 Zumwalt Class Destroyer Program

How to Develop for Realtime Linux

- **Use POSIX threads**
  - finer grained applications mean more parallelism, so can take advantage of multiple cores
- **Use POSIX threads synchronization mechanisms**
  - Mutexes
  - Barriers
  - Condition variables
- **Set appropriate priorities for your threads**
  - Any SCHED_FIFO thread is higher priority than any SCHED_OTHER thread
  - ensure that your high priority threads don't hog the processor

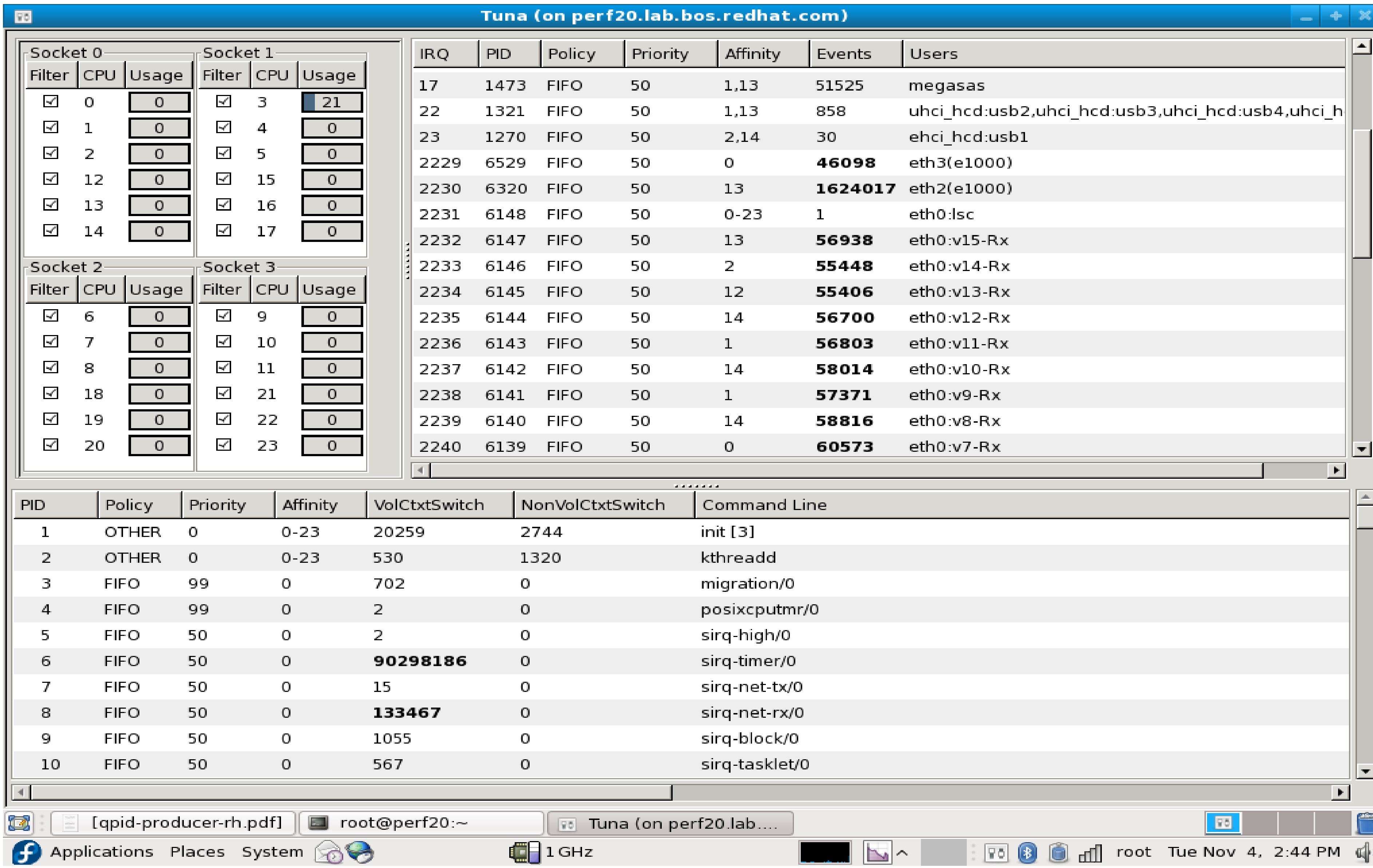
How to Deploy Realtime Linux

- **Tune your system!**
  - No two applications behave the same
  - use *tuna* to tweak priorities and affinities
  - use *oprofile* to find application hotspots
  - use *ftrace* to find long latency areas
- **Dedicate processors to your application threads**
  - Use *tuna* or *taskset* to bind threads to specific processors and move other threads off
  - 4-way and 8-way processors getting cheaper
- **Use cpu affinity field in /proc/irqs/<n>/smp_affinity to bind interrupts to specific processors**
  - *tuna* can do this easily



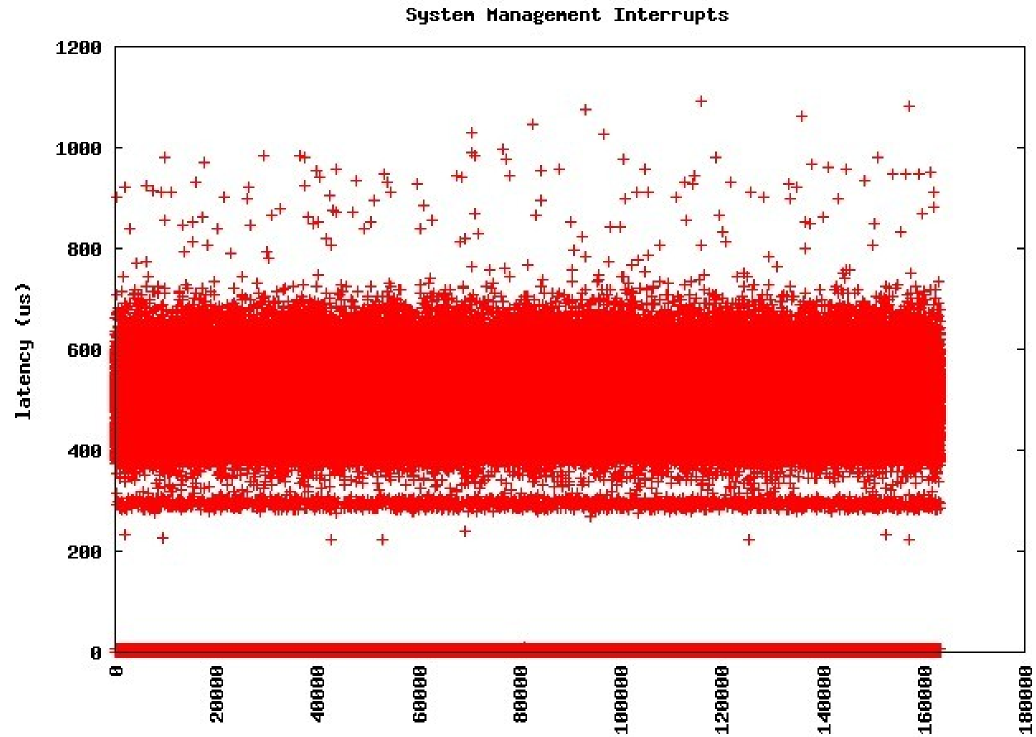
redhat

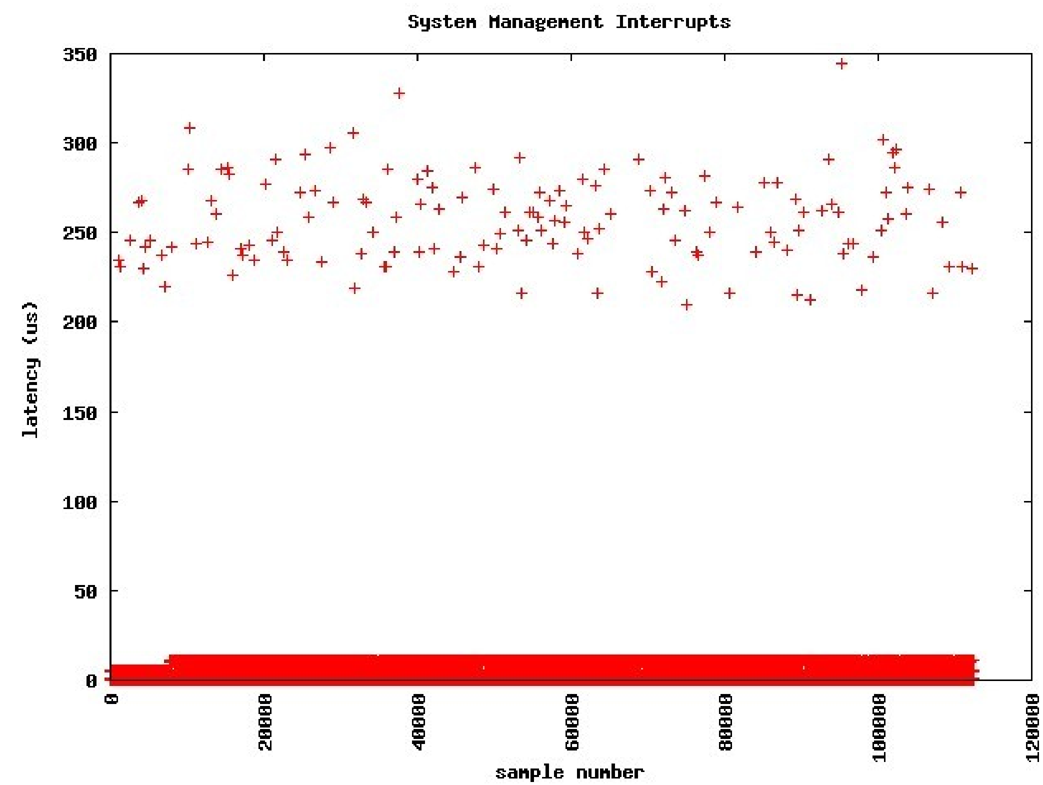
Use TUNA for Tuning



Hardware Matters

- **Hardware can have a big effect on realtime performance**
- **Hardware drivers may need to be updated to handle threaded interrupts**
- **Many system BIOS's include Service Management Interrupts (SMIs)**
  - Cause non-deterministic latency *beneath* the operation system by taking CPU cycles for things like power management, administration
  - SMI latencies *cannot* be resolved by realtime linux—they require the hardware OEM to remove SMIs or make them configurable





History of Realtime Linux

- **First steps (2000-2004)**
  - Ingo Molnar / Andrew Morton – low latency patch
  - Robert Love – preemption patch
- **Current State (2004 - today)**
  - First started on 2.6.9 kernel – Ingo Molnar's realtime patch
  - Originally called realtime-preempt patch
- **Moving From -rt to mainline:**
  - BKL preemptable (2.6.8)
  - Mutex patch (2.6.16)
  - Semaphore-to-Mutex conversion (ongoing ~85% done)
  - Hrtimers subsystem (2.6.16)
  - Robust futexes (2.6.17)
-
  - Priority inheritance futexes (PI-futex) (2.6.18)
  - Generic IRQ layer (2.6.18)
  - Core time re-write (2.6.18)
  - Sleepable RCU (2.6.19)
  - Latency Tracer (circa 2.6.18)
  - High-res+dynticks (2.6.21)
  - CFS – completely fair scheduler (2.6.23)
  - Conversion of spin-locks to mutex (2.6.23+)
  - All Interrupt handling in threads (~2.6.23+)
  - Full rt-preempt (~2.6.24+)

Realtime Linux Roadmap

- **Incorporate the realtime patchset into the mainline kernel**
  - Current realtime kernel is 2.6.26-based
  - Convert from patchset to GIT tree for 2.6.28
  - Merge threaded IRQs
    - Threaded device handler, allows driver to register as threaded interrupt handler
    - Target for 2.6.30
  - Merge preemption
    - Re-work 'macro magic' that implements preemption into lock_t abstraction
    - Depends heavily on acceptance of threaded interrupts
    - Target perhaps for 2.6.31/2.6.32
- **Improve performance**
  - Reduce determistic latency vs throughput tradeoff
  - Improve performance of surrounding IO systems

Red Hat Enterprise MRG Provides Realtime Linux

- **Red Hat Enterprise MRG (Messaging, Realtime, Grid) includes a Realtime kernel and performance tools**
- **Installs onto standard Red Hat Enterprise Linux 5 and preserves Red Hat Enterprise Linux application certifications**
  - No application or code changes necessary
  - Take advantage of Red Hat ecosystem
- **Aggressively tracks upstream kernel development for performance**
- **Red Hat has worked with OEMs to certify realtime hardware**
  - Including addressing SMIs
- **Red Hat has partnered with IBM and Sun to certify their realtime JVMs for MRG Realtime**

RED HAT ENTERPRISE MRG

Learn more.

Red Hat Enterprise MRG Demo

Additional Information

- **RT Wiki:** http://rt.wiki.kernel.org/index.php/Main_Page
- **Red Hat Enterprise MRG:** http://redhat.com/mrg
- **Realtime Tuning Guide:** http://www.redhat.com/docs/en-US/Red_Hat_Enterprise_MRG/1.1/html/Realtime_Tuning_Guide/

Southern California Linux Expo