SUMMIT JBoss WORLD

PRESENTED BY RED HAT

LEARN. NETWORK. EXPERIENCE OPEN SOURCE.

redhat.

Achieving Top Network Performance

Mark Wagner
Principal Engineer, Red Hat Inc.
06.27.12

Not Covered

- Bonding
- RHEL5
- Wireless
- Coding examples
  - Some mentions of tips

Take Aways

- Awareness of the issues
- Awareness of tools
- Guidelines

Take Aways

- You will leave this discussion with:
  - An understanding of some issues affecting server network performance
  - Tools to help you evaluate your network performance
  - Some guidelines to try in your environment

Some Quick Disclaimers

- We do not recommend one vendor over another
- Test data used is based on “performance mode”
  - Maximize a particular thing at the expense of other things
  - Not recommended for production
- Don't assume settings shown will work for you without some tweaks
  - Always experiment to find what works best in your environment

Agenda

- Why Bother ?
- Basic Concepts
- RHEL5 -> RHEL6
- Tuning Knobs and Auto Tuning
- Real World Debug
- Wrap Up

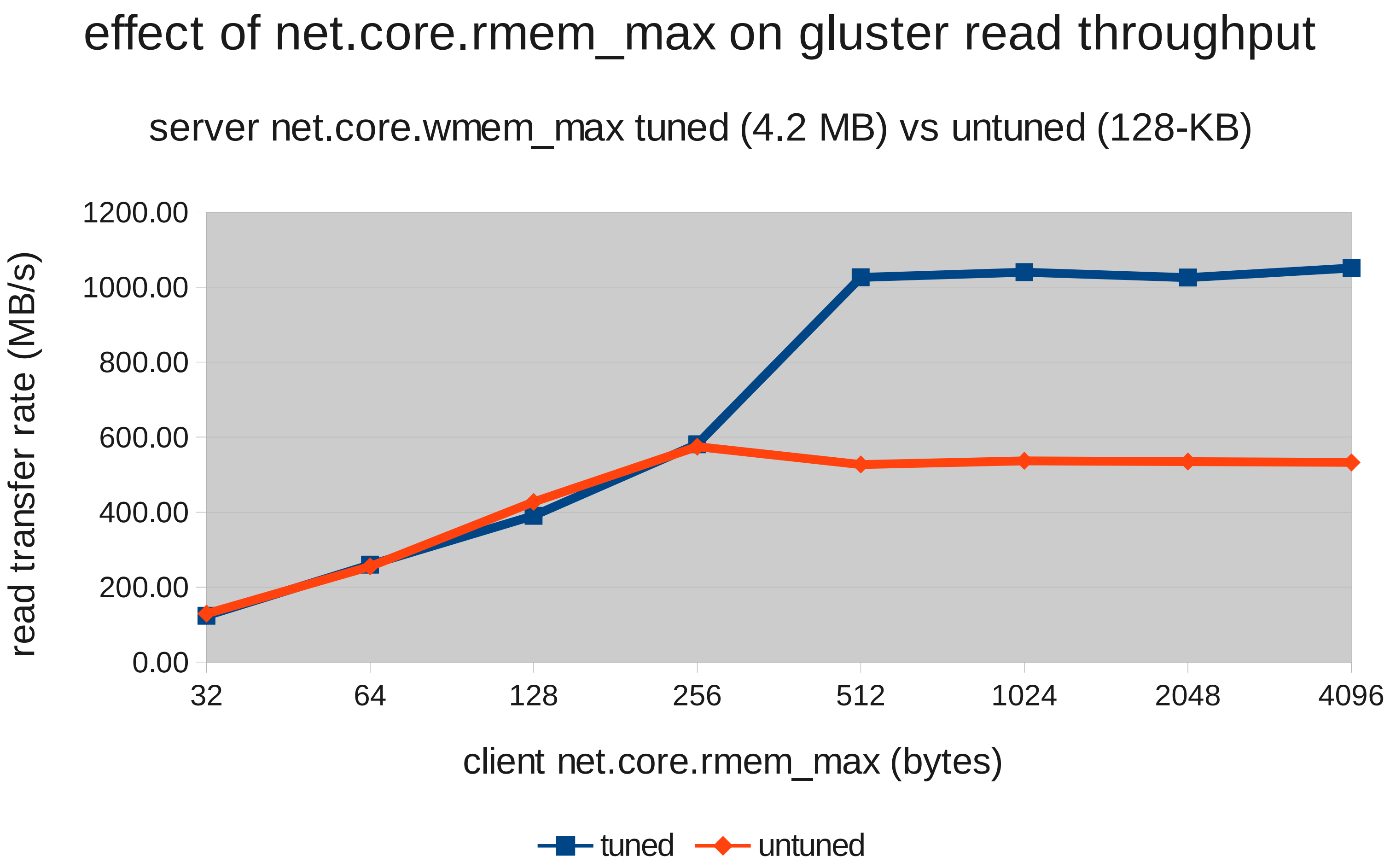
Agenda

- Why Bother ?
  - 40 gbit, gluster,latency
- Basic Concepts
  - Pci, cpu, numa
  - Offloads, rdma, solarflare, etc
  - The Virtual World
- RHEL5 -> RHEL6
  - Multiqueue, cgroups,steering, sctp, congestion control

Agenda

- Tunings
  - Drivers
    - Modinfo, modprobe, ethtool
  - Sysctl
  - Application, System
- Debug examples
  - Netperf throughput,Latency,Gluster
- Wrap up

Teaser 1 – Gluster



Teaser 2 – 40 Gbit / sec netperf

- Two 40Gbit cards back to back (no switch).

```
# ./netperf -l 30 -H 172.17.200.82
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
172.17.200.82 (172.17.200.82) port 0 AF_INET : spin interval : demo
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00      8868.76
```

Teaser 3 – latency

- Font size 28
    - Font size 26
        - Font size 22
- Font size 28
    - Font size 26
        - Font size 22
- Font size 28
    - Font size 26
        - Font size 22

Basics Concepts

- NUMA
- PCI bus
- CPU Characteristics
- Power Management
- The Virtual World

Memory Characteristics

- Memory Speed is crucial
  - Faster is better
- Understand layout and its impact
  - On middle age systems, fully populating the memory will slow it down
- Triple check your BIOS settings
  - Make sure that you pick settings for optimal performance

What is NUMA ?

- NUMA – Non Uniform Memory Architecture
  - Make bigger systems scalable by distributing system memory near individual CPUs
- NUMA has been around for a long time
  - In the past was in specialized high end systems
  - now the norm across the board for servers
- Most current multi-socket systems...
  - Recent AMD systems have 2 nodes / socket

“Issues” that NUMA makes visible

- System scheduler
- Non-local memory accesses

“Issues” that NUMA makes visible

- RHEL6 system scheduler appears biased towards responsiveness and optimizing for CPU utilization
  - It will often align network app on same core as interrupt
- Tries to use idle CPUs, regardless of where process memory is located!
- Non-local memory accesses have higher access latency, which degrades performance

NUMA - Latency

```
[root@perf ~]# numactl --hardware
available: 4 nodes (0-3)
node 0 cpus: 0 4 8 12 16 20 24 28 32 36 40 44 48 52 56 60
node 0 size: 32649 MB
node 0 free: 30868 MB
node 1 cpus: 1 5 9 13 17 21 25 29 33 37 41 45 49 53 57 61
node 1 size: 32768 MB
node 1 free: 29483 MB
node 2 cpus: 2 6 10 14 18 22 26 30 34 38 42 46 50 54 58 62
node 2 size: 32768 MB
node 2 free: 31082 MB
node 3 cpus: 3 7 11 15 19 23 27 31 35 39 43 47 51 55 59 63
node 3 size: 32768 MB
node 3 free: 31255 MB
node distances:
node   0   1   2   3
  0:  10  21  21  21
  1:  21  10  21  21
  2:  21  21  10  21
  3:  21  21  21  10
```

Numa – Latency

- Sample inter-NUMA-node relative latency:
  - Intel 4 socket / 4 node: 1.5x
  - AMD 4 socket / 8 node: 2.7x
  - 8 socket / 8 node: 2.8x
  - 32 node blade system: 5.5x

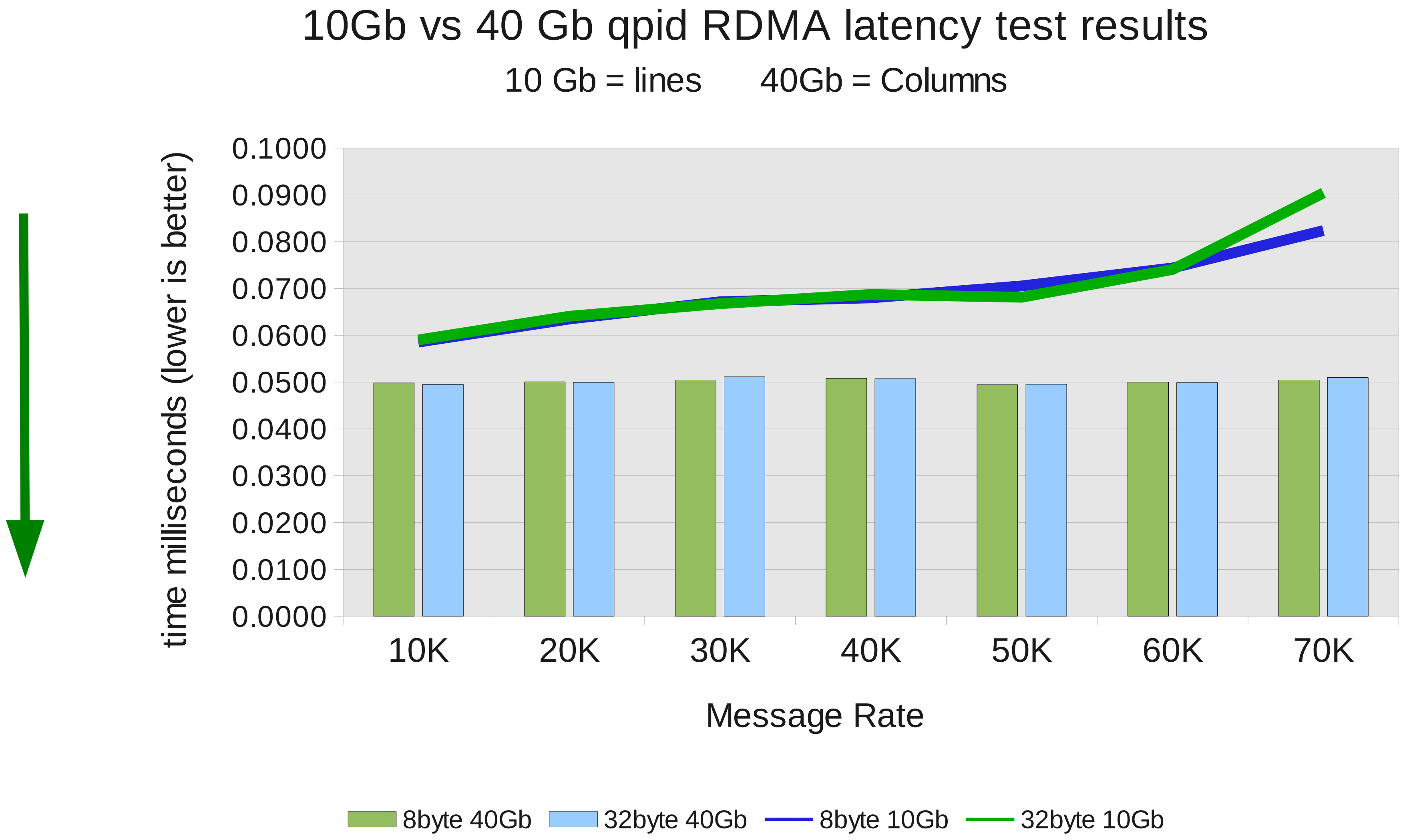
PCI Bus – and related issues

- Slot speed
- Multiple buses / Affinity
- Tools

PCI Bus – and related issues

- Make sure that you know the slot speed !
  - 10 Gbit needs 8X
    - At least with Gen2
  - 40 Gbit speeds need PCI-e 3
- Find if the slot is tied to a specific NUMA node
  - Know the bindings
  - Spread the load
- Can you change any of the parameters ?
  - setpci can change some of the parameters
    - It is tricky and dangerous

40 Gbit Gen3 vs 10 Gbit PCI Gen2 latency



CPU Characteristics – Basics

- Cache layout
- Hyperthreads
- cstates

CPU Characteristics – Basics

- Understand cache layout
  - It changes with different chip generations
  - Try to keep cache lines hot
- To use hyperthreads or not
  - No one stop answer
    - For latency sensitive probably not
    - For applications that block a lot probably yes
- cstates
  -

CPU – Power related characteristics

- Variable frequencies
- Multiple cores
- Power saving modes (cpuspeed governors)
  - performance
  - ondemand
  - Powersave

CPU – Performance Governors

- echo "performance" > \
  /sys/devices/system/cpu/cpu0/cpufreq/scaling_governor

- Best of both worlds – cron jobs to configure the governor mode using tuned-adm
  - tuned-adm profile latency-performance

Power Management– not always your friend

- Each HW generation power control evolves
  - Trend is towards power saving
  - Is good for the world
- BIOS / OS Control
  - Pstates in BIOS
  - Cstates in OS

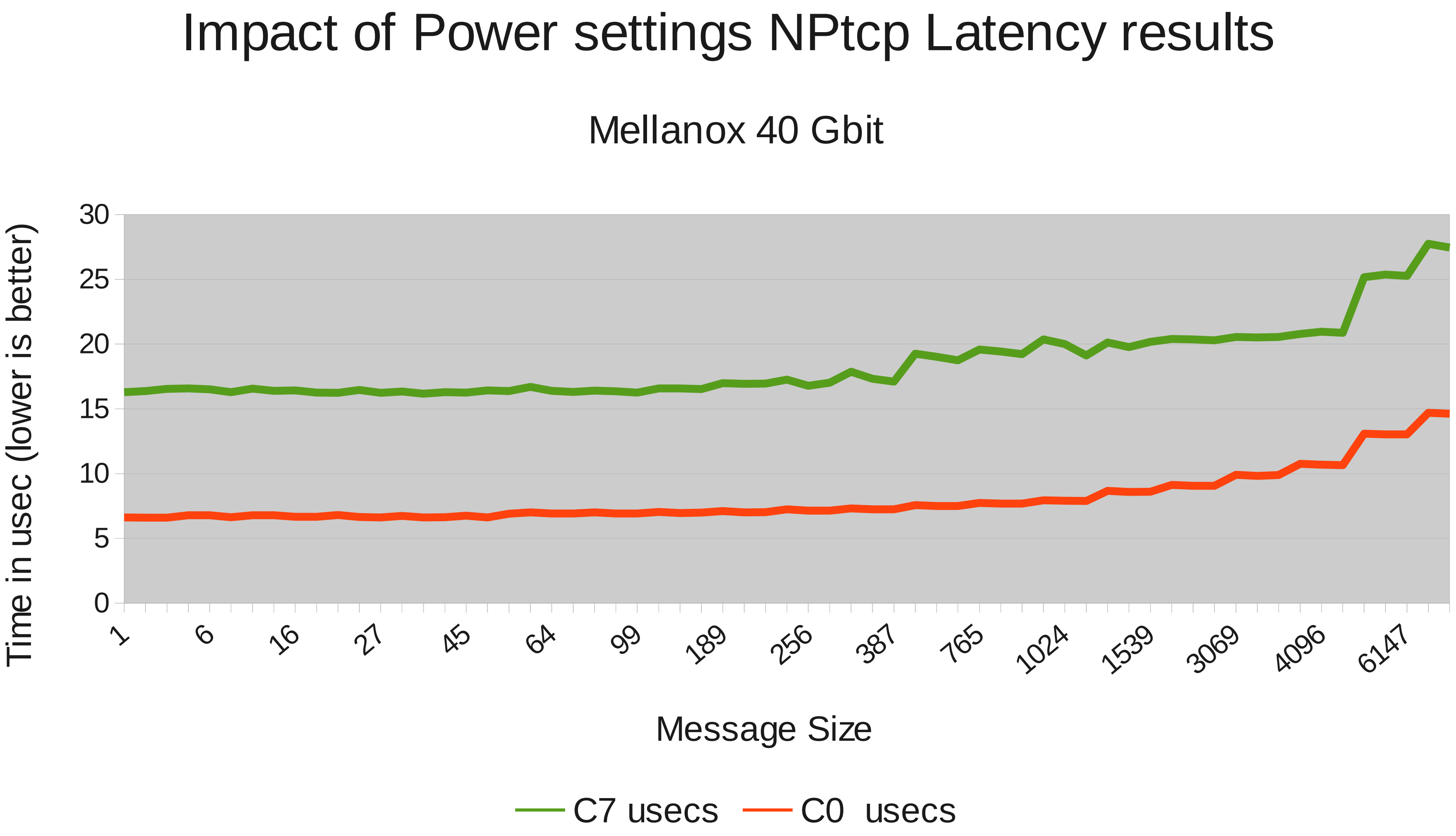
CSTATE default – C7 on this config

```
pk cor CPU    %c0  GHz  TSC    %c1    %c3    %c6    %c7  %pc2   %pc3  %pc6   %pc7   SMIs
             0.04 1.43 2.19   0.08   0.00   0.00  99.89  4.46   0.00  93.94  0.00      0
 0   0   0   0.41 1.28 2.19   0.93   0.01   0.00  98.66  3.13   0.01  93.91  0.00      0
 0   1   1   0.04 1.66 2.19   0.06   0.00   0.00  99.91  3.13   0.01  93.91  0.00      0
 0   2   2   0.01 1.73 2.19   0.01   0.00   0.00  99.98  3.13   0.01  93.92  0.00      0
 0   3   3   0.01 1.72 2.19   0.02   0.01   0.00  99.96  3.13   0.01  93.92  0.00      0
 0   4   4   0.01 1.85 2.19   0.01   0.00   0.00  99.98  3.13   0.01  93.92  0.00      0
 0   5   5   0.01 1.94 2.19   0.01   0.00   0.00  99.98  3.13   0.01  93.91  0.00      0
 0   6   6   0.01 1.92 2.19   0.02   0.00   0.00  99.98  3.13   0.01  93.91  0.00      0
 0   7   7   0.01 1.76 2.19   0.01   0.00   0.00  99.98  3.13   0.01  93.91  0.00      0
 1   0   8   0.01 1.71 2.19   0.02   0.01   0.00  99.96  5.80   0.00  93.96  0.00      0
 1   1   9   0.01 1.69 2.19   0.02   0.01   0.00  99.97  5.80   0.00  93.96  0.00      0
 1   2  10   0.01 1.75 2.19   0.02   0.00   0.00  99.97  5.80   0.00  93.96  0.00      0
 1   3  11   0.01 1.83 2.19   0.02   0.00   0.00  99.97  5.80   0.00  93.96  0.00      0
 1   4  12   0.01 1.84 2.19   0.02   0.00   0.00  99.97  5.80   0.00  93.96  0.00      0
 1   5  13   0.01 1.91 2.19   0.02   0.00   0.00  99.98  5.80   0.00  93.96  0.00      0
 1   6  14   0.01 1.96 2.19   0.02   0.00   0.00  99.98  5.80   0.00  93.96  0.00      0
 1   7  15   0.01 2.38 2.19   0.03   0.00   0.00  99.96  5.80   0.00  93.96  0.00      0
```

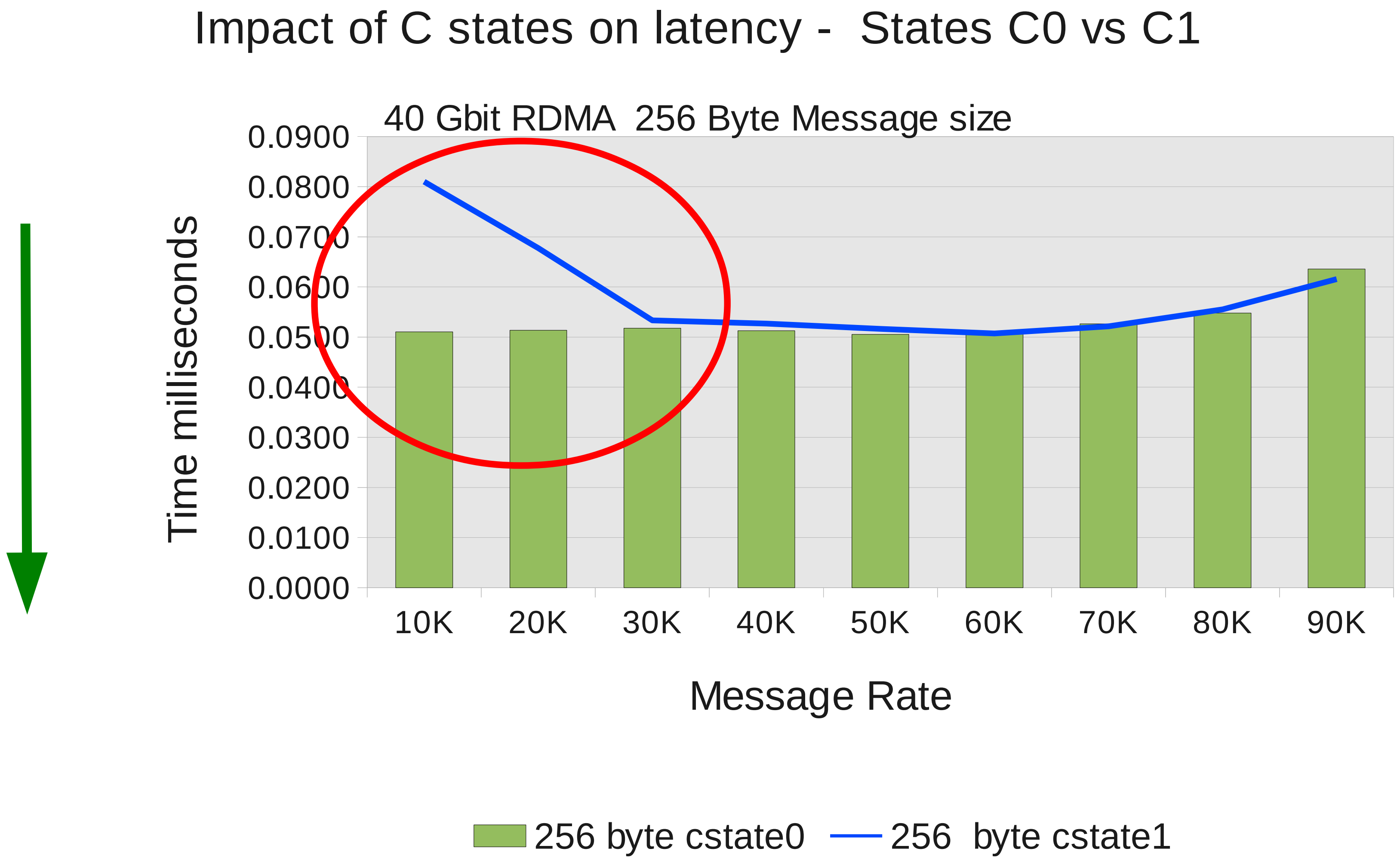
CSTATE disabled – Note speed

```
 pk cor CPU    %c0  GHz  TSC     %c1     %c3     %c6     %c7    %pc2    %pc3    %pc6    %pc7   SMIs
            100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   0   0 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   1   1 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   2   2 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   3   3 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   4   4 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   5   5 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   6   6 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  0   7   7 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   0   8 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   1   9 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   2  10 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   3  11 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   4  12 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   5  13 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   6  14 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
  1   7  15 100.00 2.69 2.19    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00      0
```

NPtcp latency vs cstates – c7 vs c0



Cstates impact on Latency



RHEL6 “tuned-adm” profiles

```
# tuned-adm list
Available profiles:
  - default
- latency-performance
- throughput-performance
- enterprise-storage
- virtual-host, virtual-guest *
```

Example

```
# tuned-adm profile enterprise-storage
```

* NEW for RHEL6.3

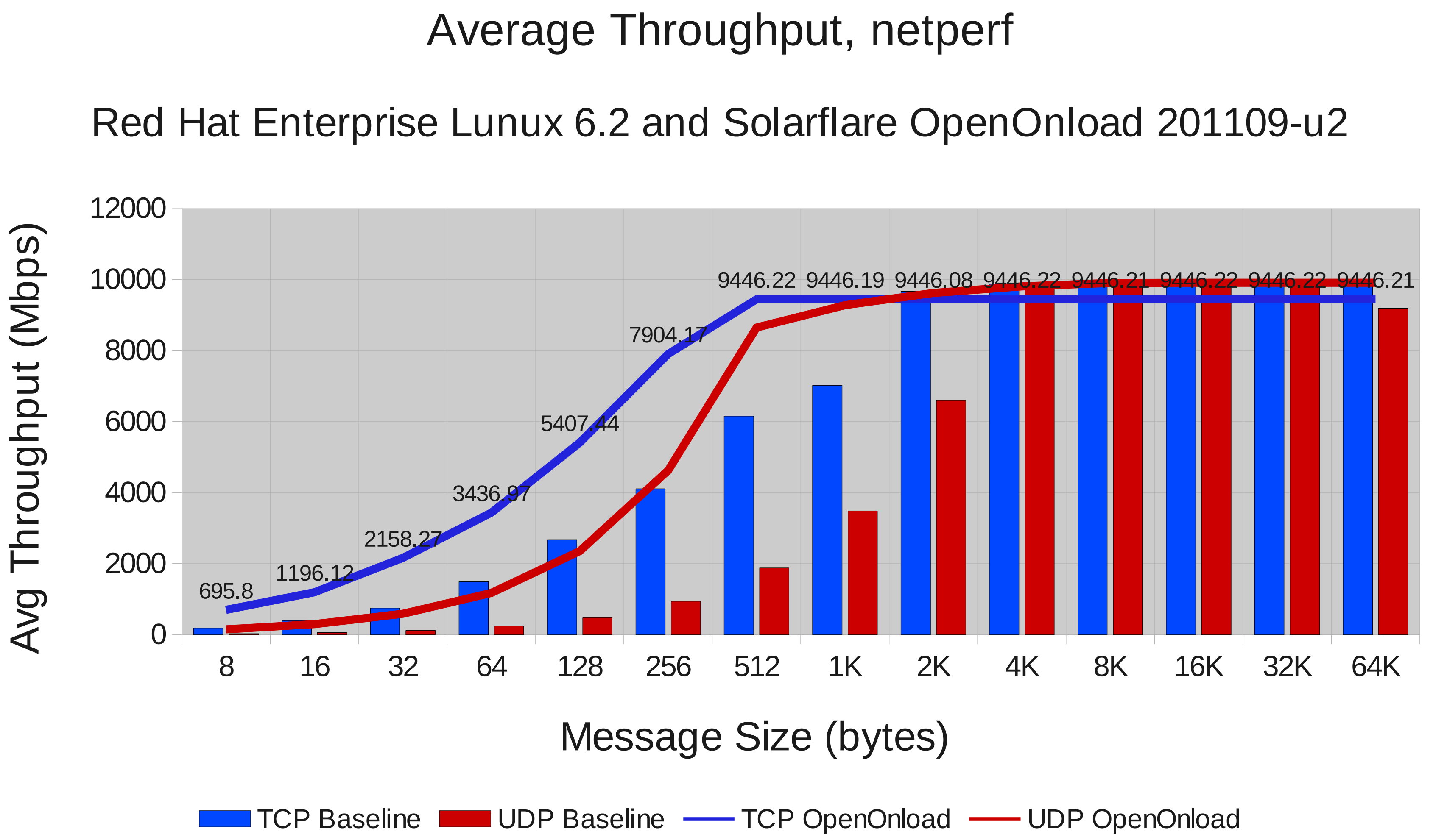
tuned profiles – virtual-host/guest new RHEL6.3

Tunable	default	latency-performance	throughput-performance	enterprise-storage	virtual-host	virtual-guest
kernel.sched_min_granularity_ns	4ms		10ms	10ms	10ms	10ms
kernel.sched_wakeup_granularity_ns	4ms		15ms	15ms	15ms	15ms
vm.dirty_ratio	20% RAM		40%	40%	10%	40%
vm.dirty_background_ratio	10% RAM				5%	
vm.swappiness	60				10	30
I/O Scheduler (Elevator)	CFQ	deadline	deadline	deadline	deadline	deadline
Filesystem Barriers	On			Off	Off	Off
CPU Governor	ondemand	performance	performance	performance	performance	performance
Disk Read-ahead				4x	4x	4x

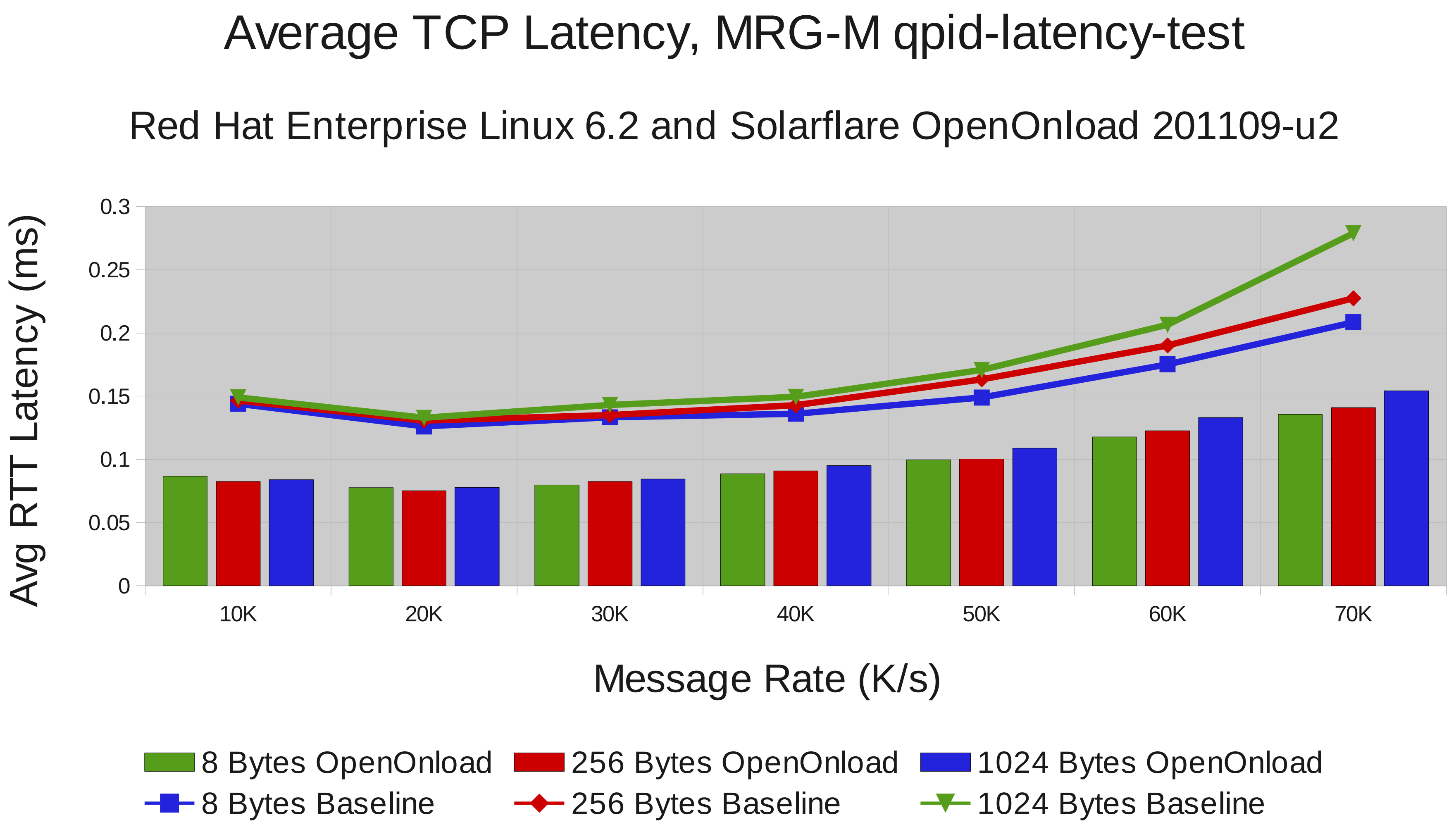
Kernel Bypass Technologies – Pros and Cons

- Multiple Technologies
- Some Proprietary
  - SolarFlare OpenOnload
- Coding Changes needed
  - RDMA

OpenOnload – Throughput

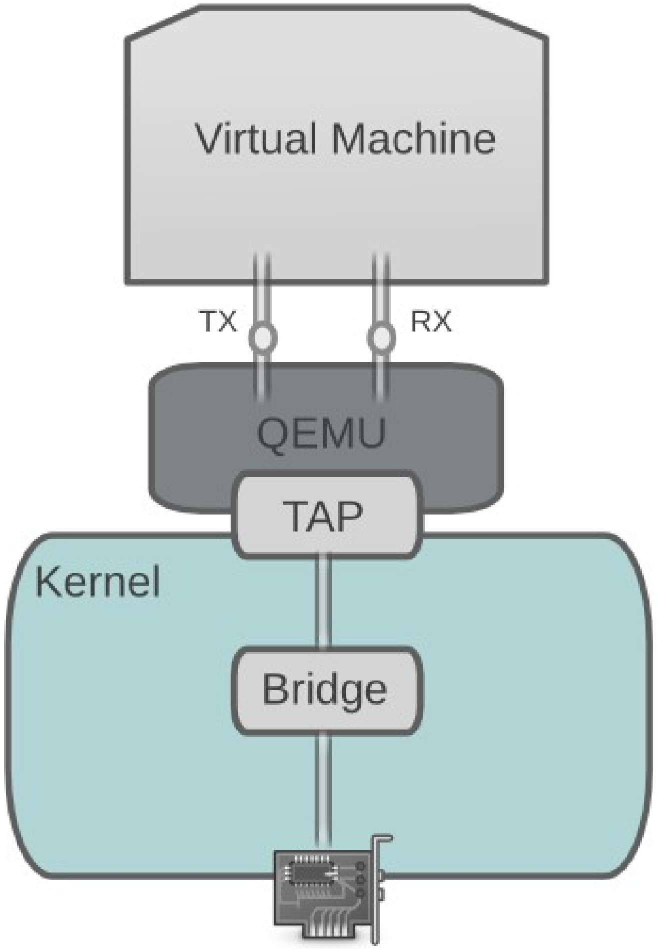


Offload – Solarflare OpenOnload



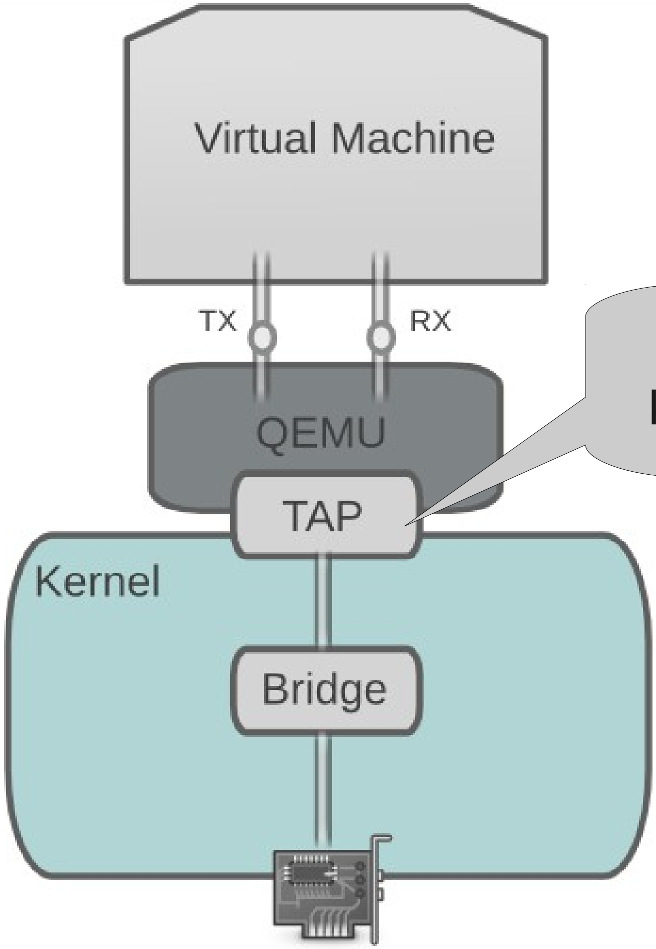
KVM Network Architecture - VirtIO

- Virtual Machine sees paravirtualized network device – VirtIO
  - VirtIO drivers included in Linux Kernel
  - VirtIO drivers available for Windows
- Network stack implemented in userspace



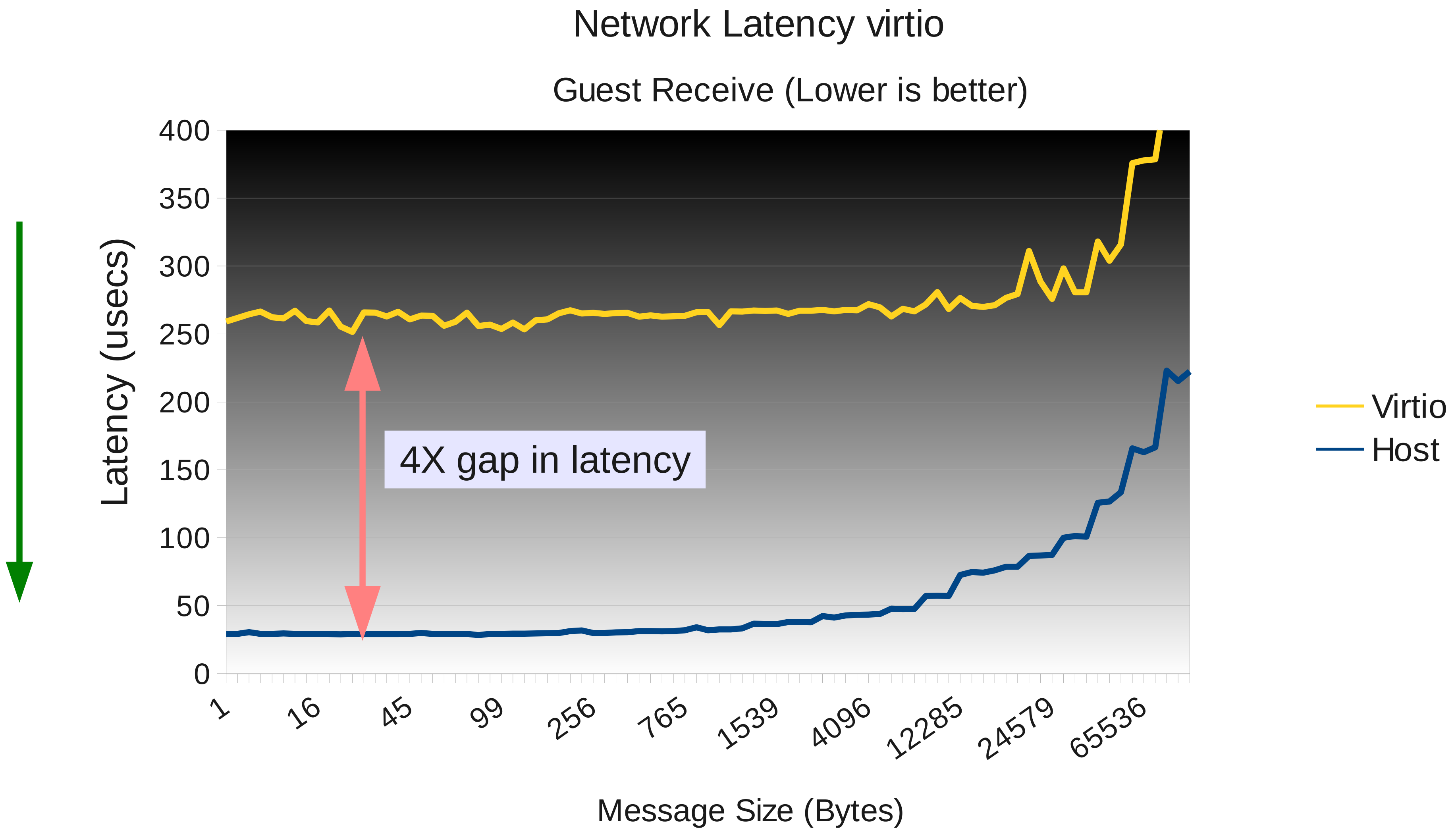
KVM Network Architecture

Virtio



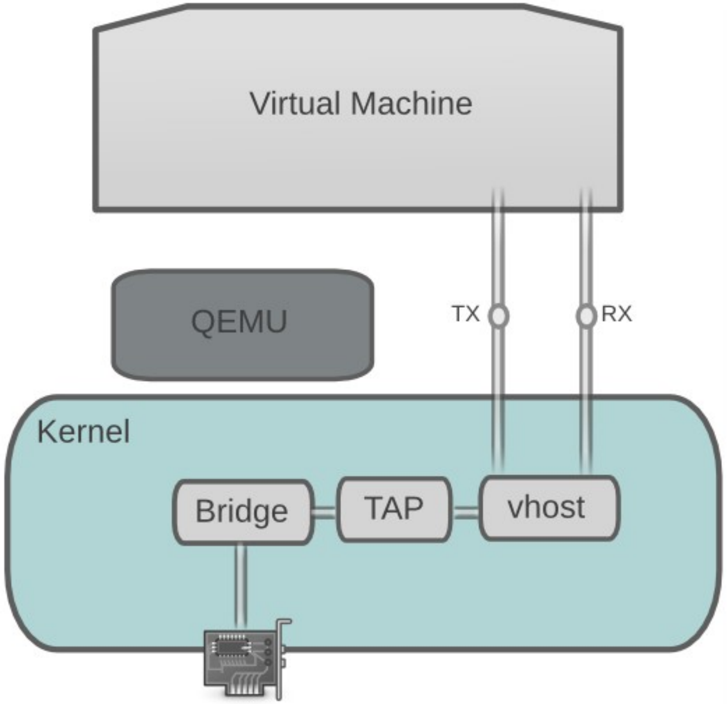
Context switch host kernel <-> userspace

Latency comparison – RHEL 6



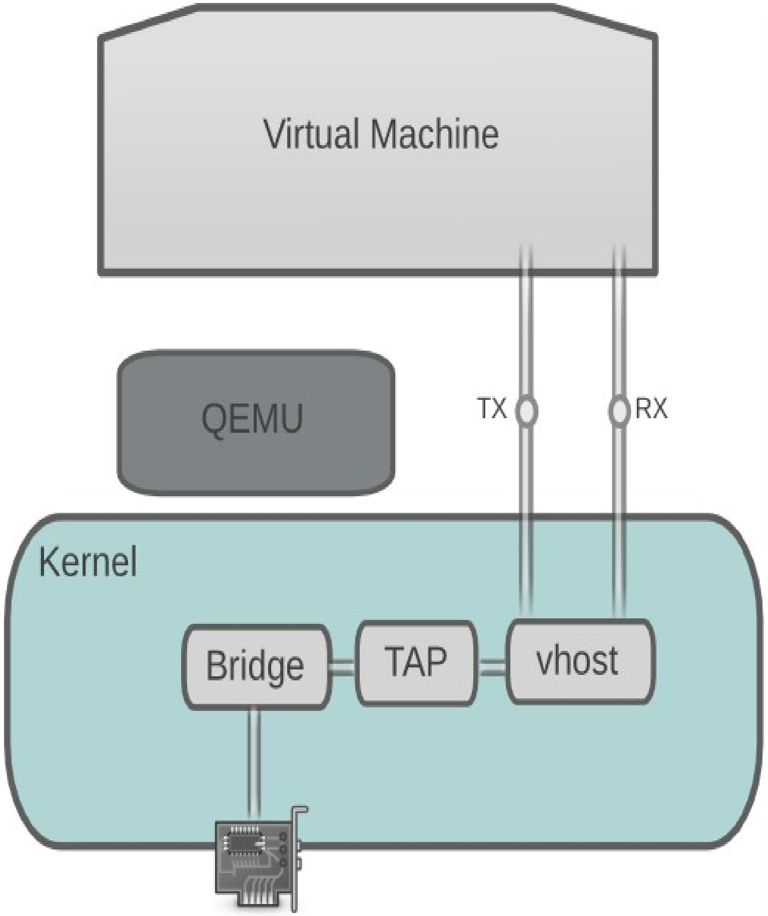
KVM Network Architecture – vhost_net

- New in RHEL6.1
- Moves QEMU network stack from userspace to kernel
- Improved performance
- Lower Latency
- Reduced context switching
- One less copy

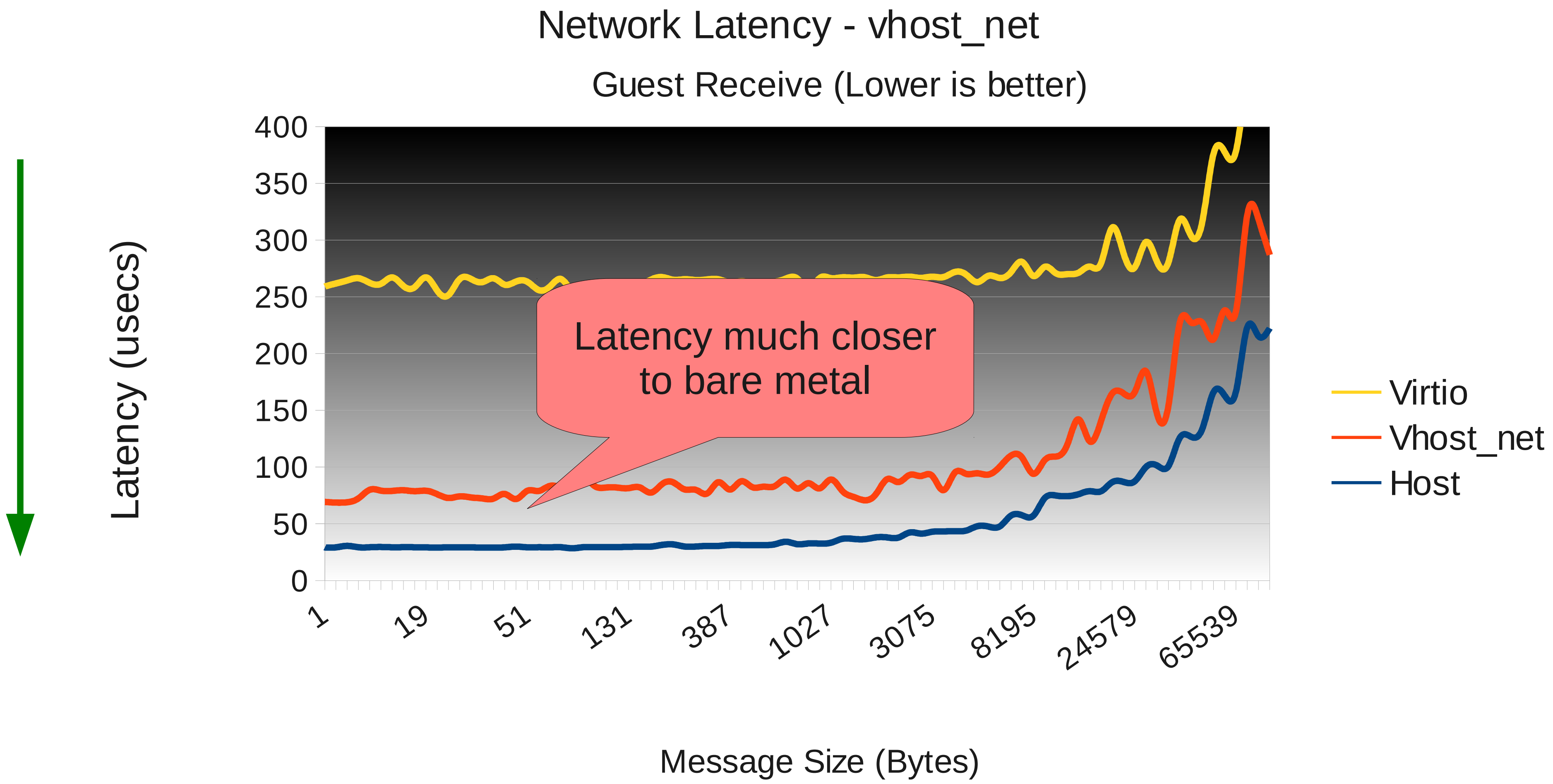


KVM Network Architecture – vhost_net

vhost_net



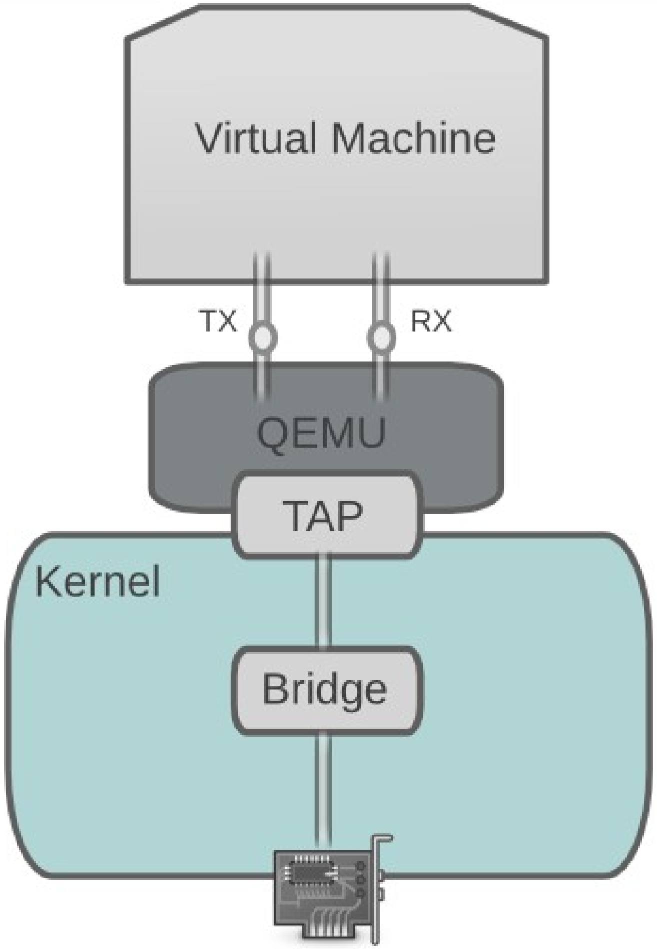
Latency comparison – RHEL 6



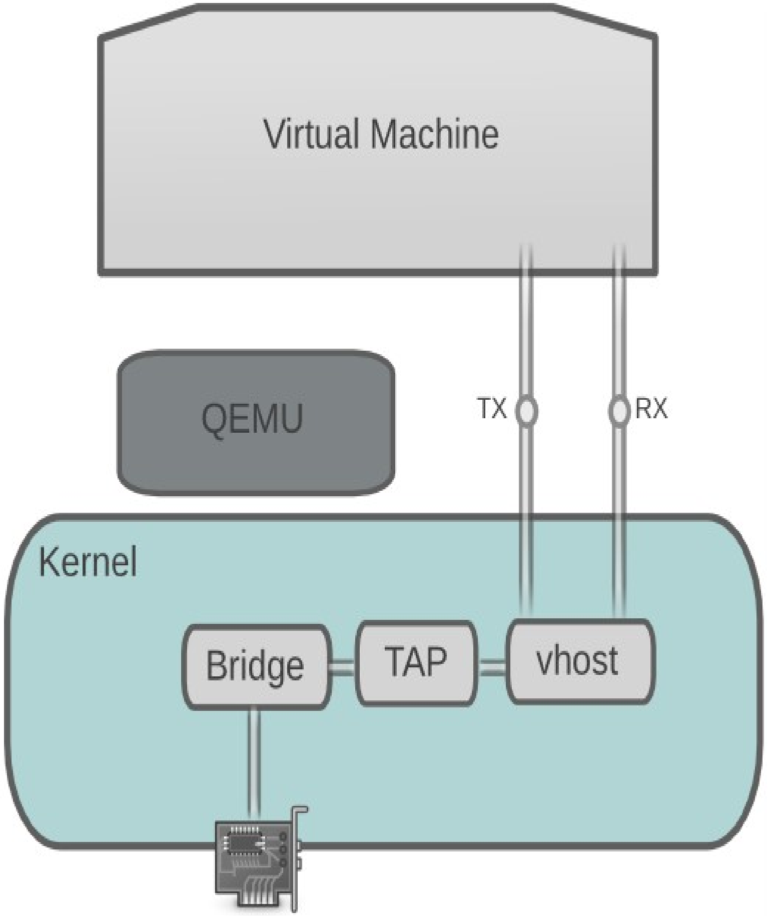
Latency much closer to bare metal

KVM Network Architecture – VirtIO vs vhost_net

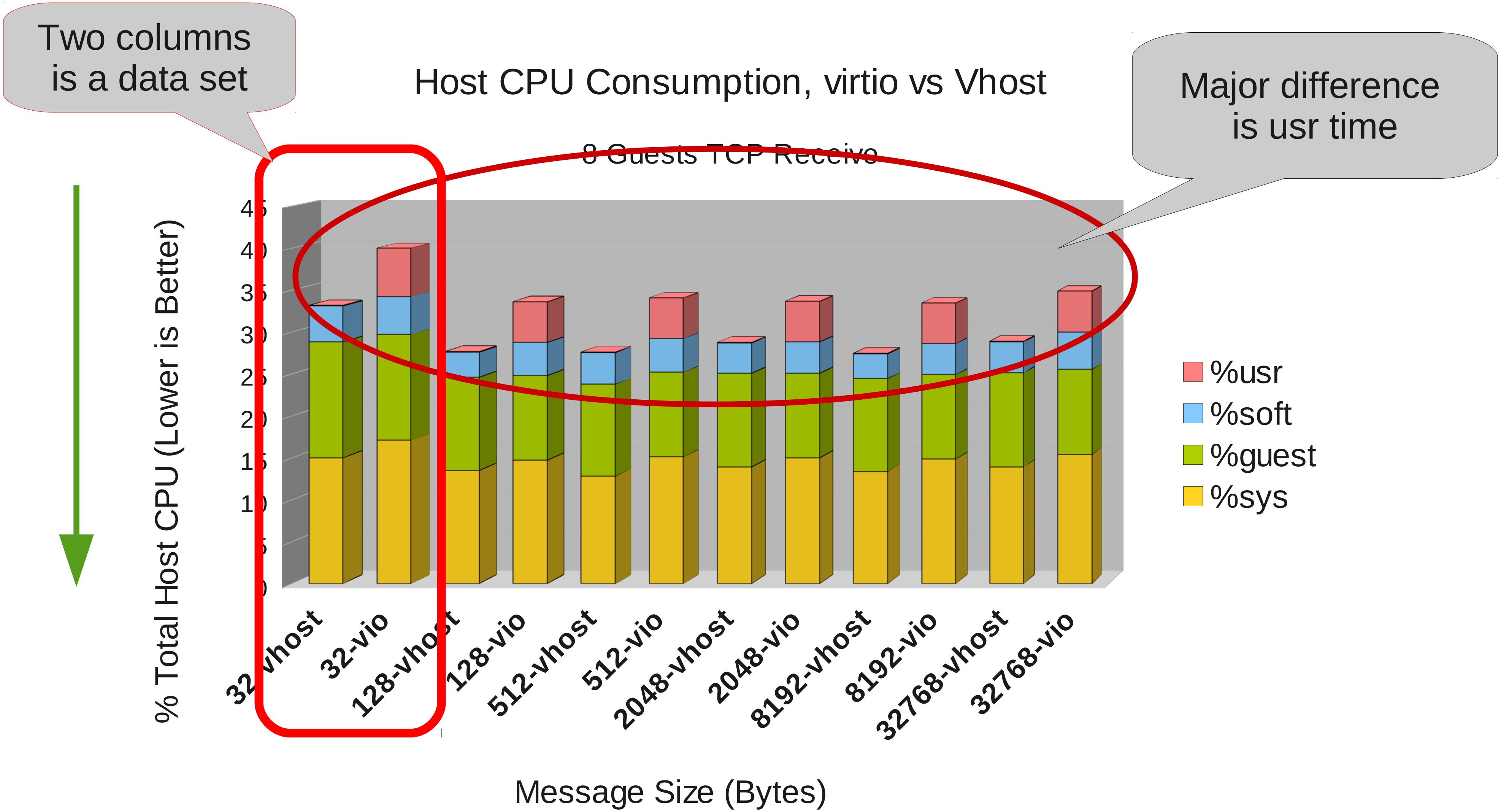
Virtio



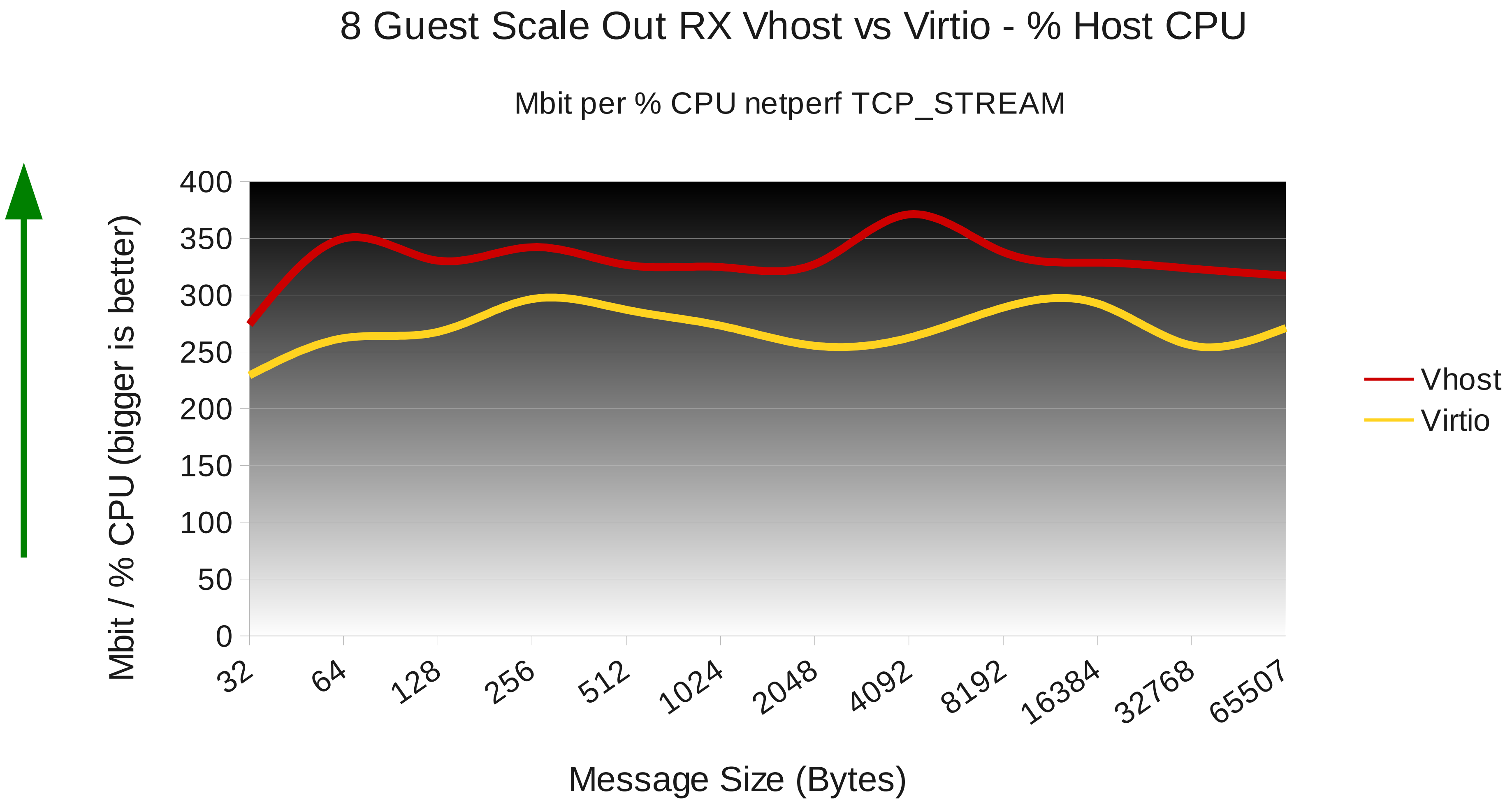
vhost_net



Host CPU Consumption virtio vs vhost_net



vhost_net Efficiency



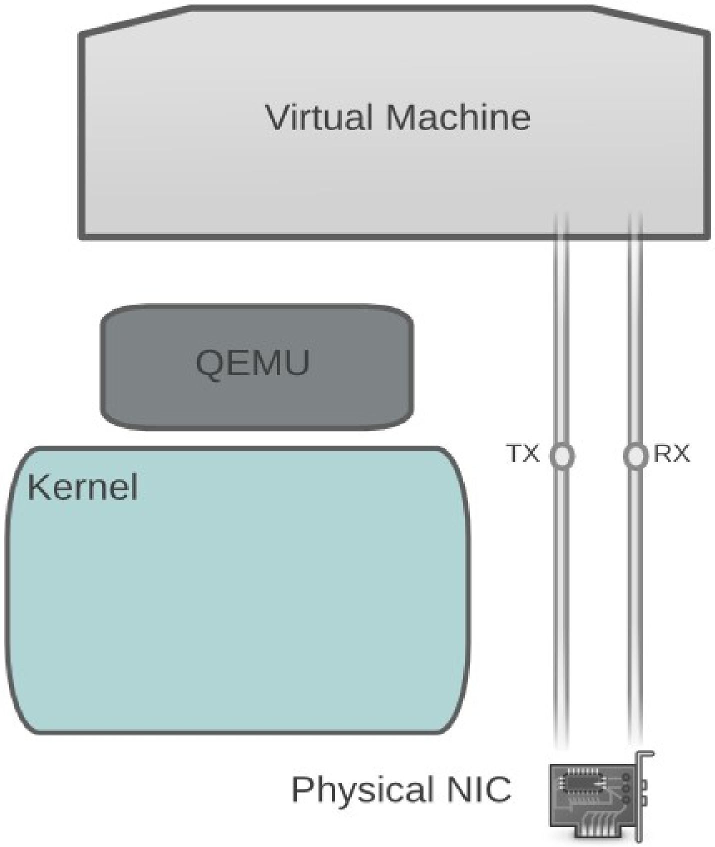
KVM Network Architecture – PCI Device Assignment

- Physical NIC is passed directly to guest
- Guest sees real physical device
  - Needs physical device driver
- Requires hardware support

  Intel VT-D or AMD IOMMU
- Lose hardware independence
- 1:1 mapping of NIC to Guest
- BTW - This also works on some I/O controllers

KVM Network Architecture – Device Assignment

Device Assignment



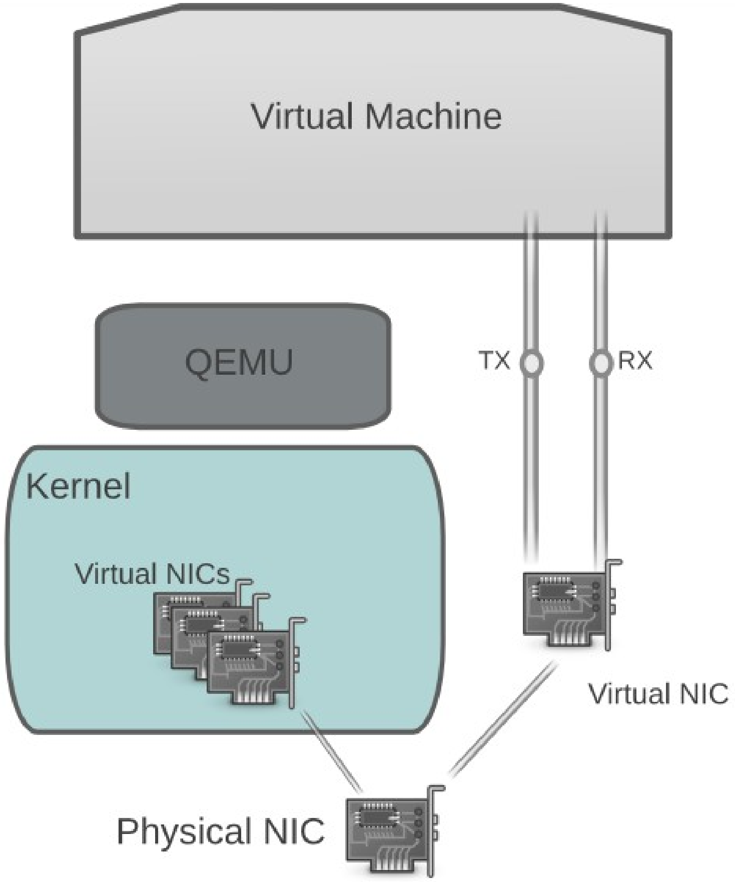
KVM Network Architecture – SR-IOV

- Single Root I/O Virtualization

  New class of PCI devices that present multiple virtual devices that appear as regular PCI devices
- Guest sees real physical device
  - Needs physical device driver
- Requires hardware support
- Low overhead, high throughput
- No live migration
- Lose hardware independence

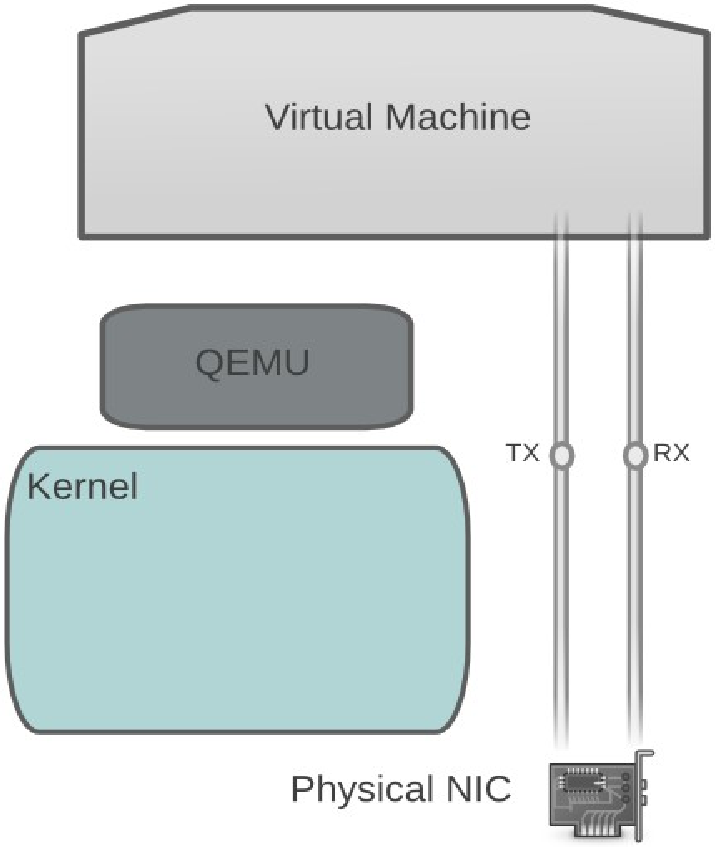
KVM Architecture – SR-IOV

SR-IOV

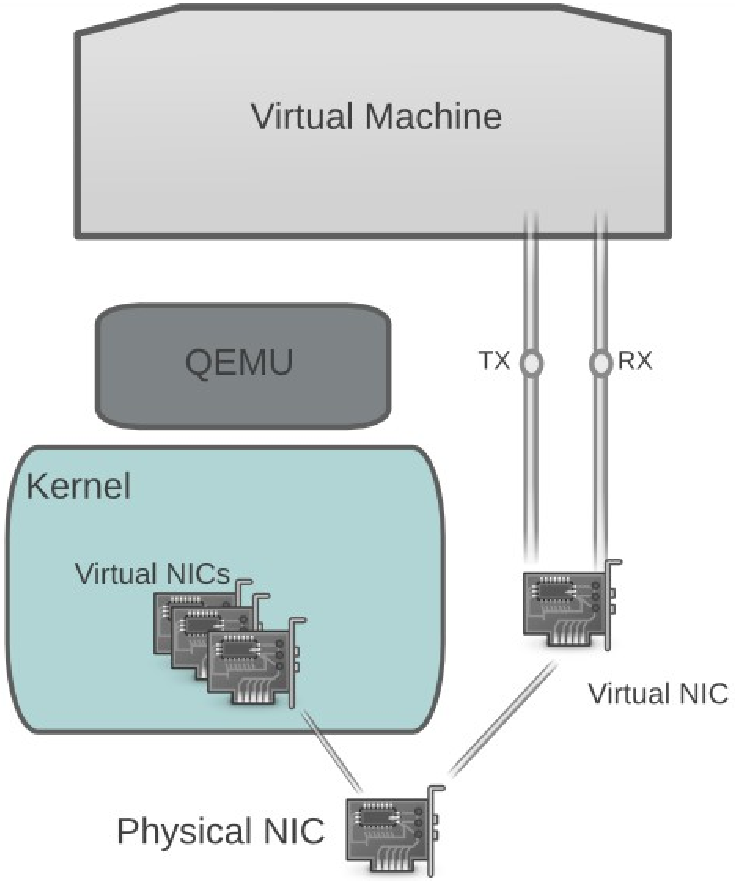


KVM Architecture – Device Assignment vs SR/IOV

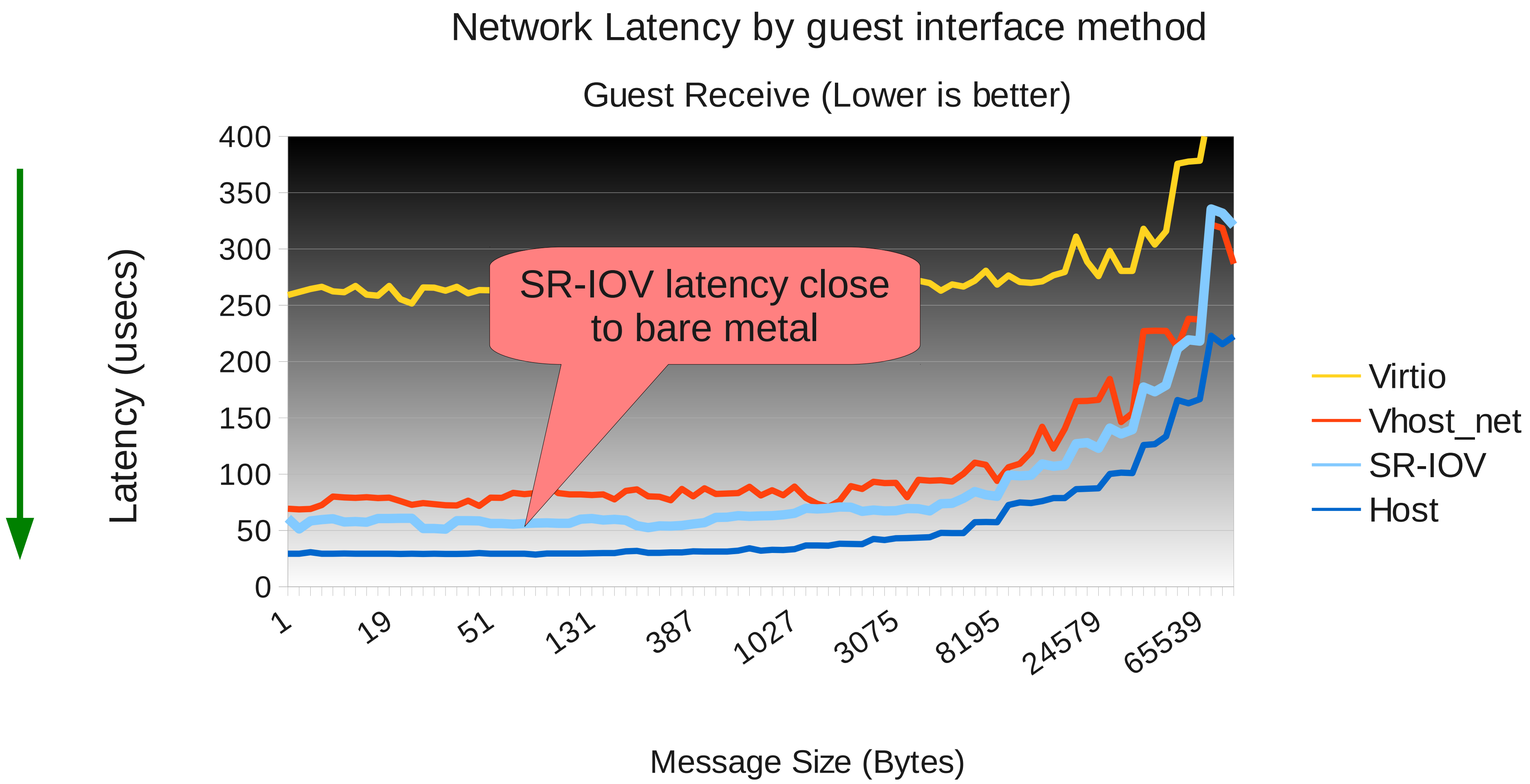
Device Assignment



SR-IOV



Latency comparison – RHEL 6 based methods



SR-IOV latency close to bare metal

RHEL6 – new features

- Multi-queue Transmit
- Tools to monitor dropped packets
- Traffic Steering
- Flow control
- Driver improvements
- Data center bridging DCB
  - FCoE performance improvements

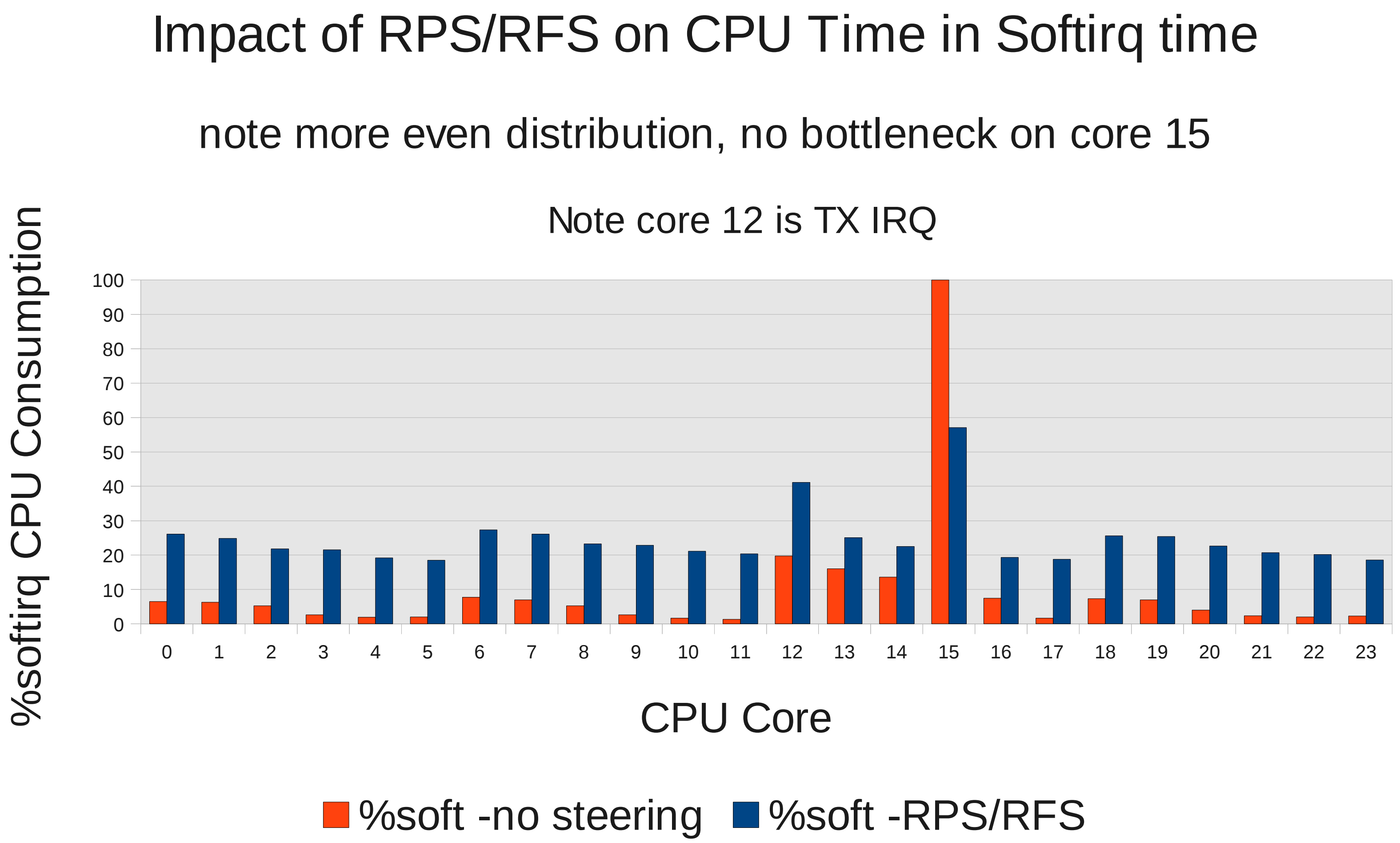
RHEL6 – new features

- Receive Packet Steering (RPS)
  - breaks the bottleneck of having to receive network traffic for a NIC on one CPU
- Receive Flow Steering (RFS)
  - allows the optimal CPU to receive network data intended for a specific application

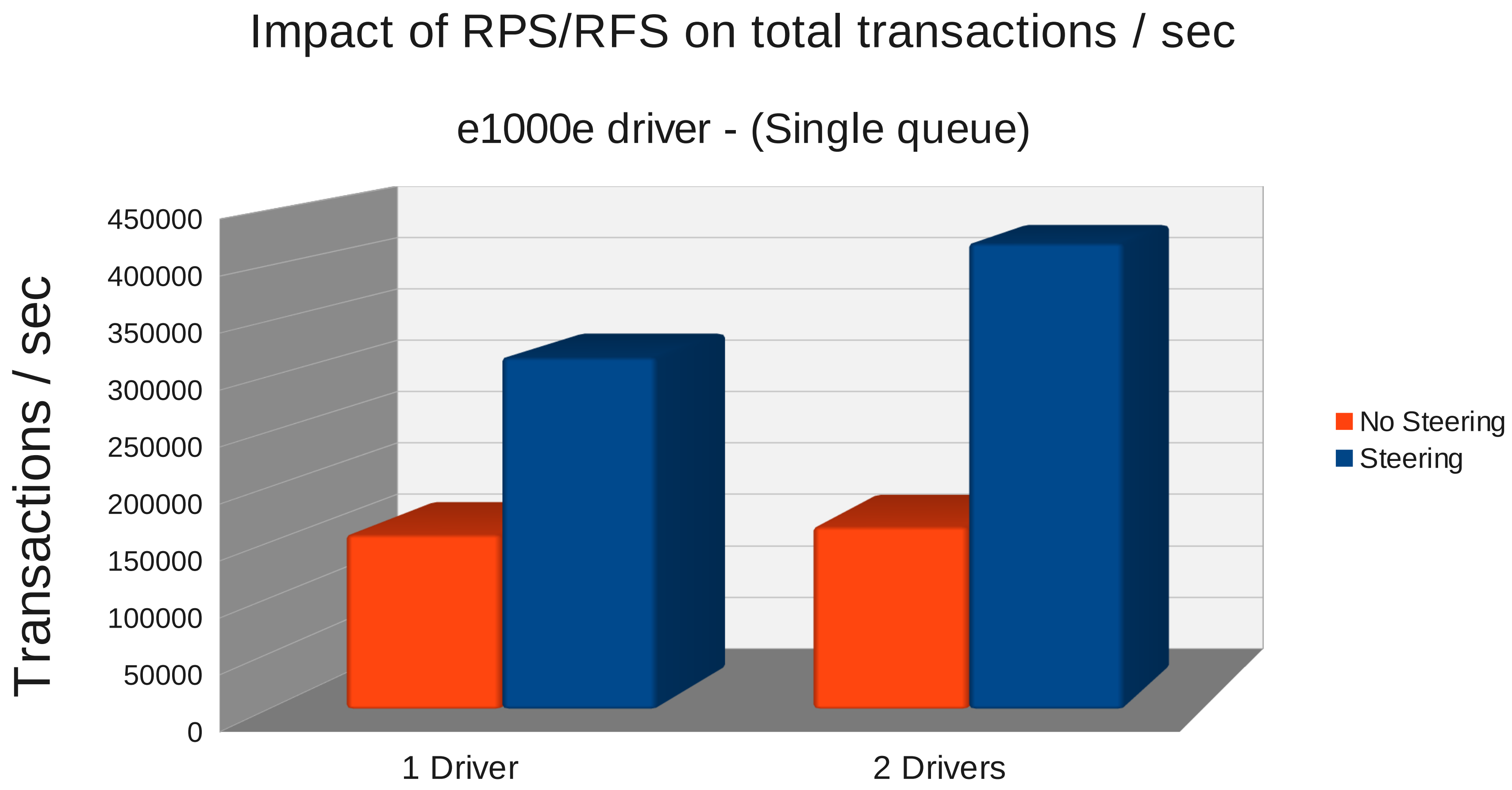
RHEL6 – new features

- Add getsockopt support for TCP thin-streams
  - reduce latency from retransmission of lost packets in time-sensitive applications
- Add Transparent Proxy (TProxy) support for non-locally bound IPv4 TCP and UDP sockets
  - similar to Linux 2.2
  - Allows packet interception and serving of response without client reconfiguration (transparent to client)

Impact of using RPS/RFS



Receive Steering – improved message rates



each driver running 100 concurrent netperf TCP_RR tests

Tuning Knobs – Overview

- Linux networking tuned for reliability
- Linux “autotunes” buffers for connections
- Watch BufferBloat !
- Don't forget UDP !
- Look at documentation in kernel tree

Tuning Knobs – Overview

- By default, Linux networking not tuned for max performance, more for reliability
  - Remember that Linux “autotunes” buffers for connections
  - Don't forget UDP !
- Try via command line
  - When you are happy with the results, add to /etc/sysctl.conf
- Look at documentation in `/usr/src`

sysctl – View and set */proc/sys* settings

- sysctl -a - lists all variables
- sysctl -q - queries a variable
- sysctl -w - writes a variable

sysctl – View and set */proc/sys* settings

- sysctl -w - writes a variable
  - When setting values, spaces are not allowed
    - sysctl -w net.ipv4.conf.lo.arp_filter=0
- Setting a variable via sysctl on the command line is **not persistent** The change is only valid until the next reboot
  - Write entries into the /etc/sysctl.conf file to have them applied at boot time

sysctl – popular settings

- These settings are often mentioned in tuning guides
- Experiment but don't take blindly!
  - net.ipv4.tcp_window_scaling
    - toggles window scaling
  - net.ipv4.tcp_timestamps
    - toggles TCP timestamp support
  - net.ipv4.tcp_sack
    - toggles SACK (Selective ACK) support

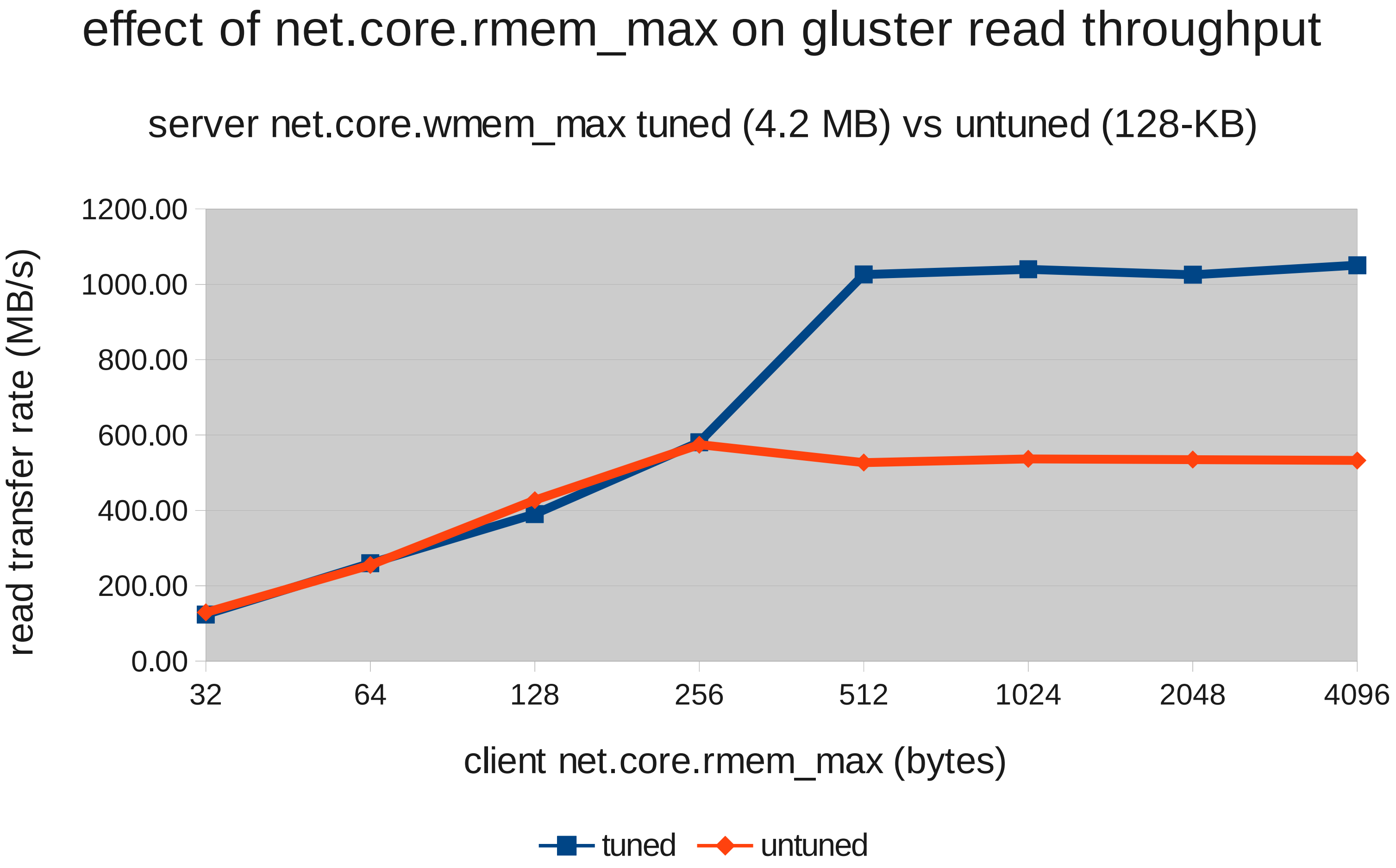
sysctl – TCP related settings

- TCP Memory Allocations - min/pressure/max
  - net.ipv4.tcp_rmem - TCP read buffer - in bytes
    - overriden by core.rmem_max
  - net.ipv4.tcp_wmem - TCP write buffer - in bytes
    - overridden by core/wmem_max
  - net.ipv4.tcp_mem   - TCP buffer space
    - measured in pages, not bytes !

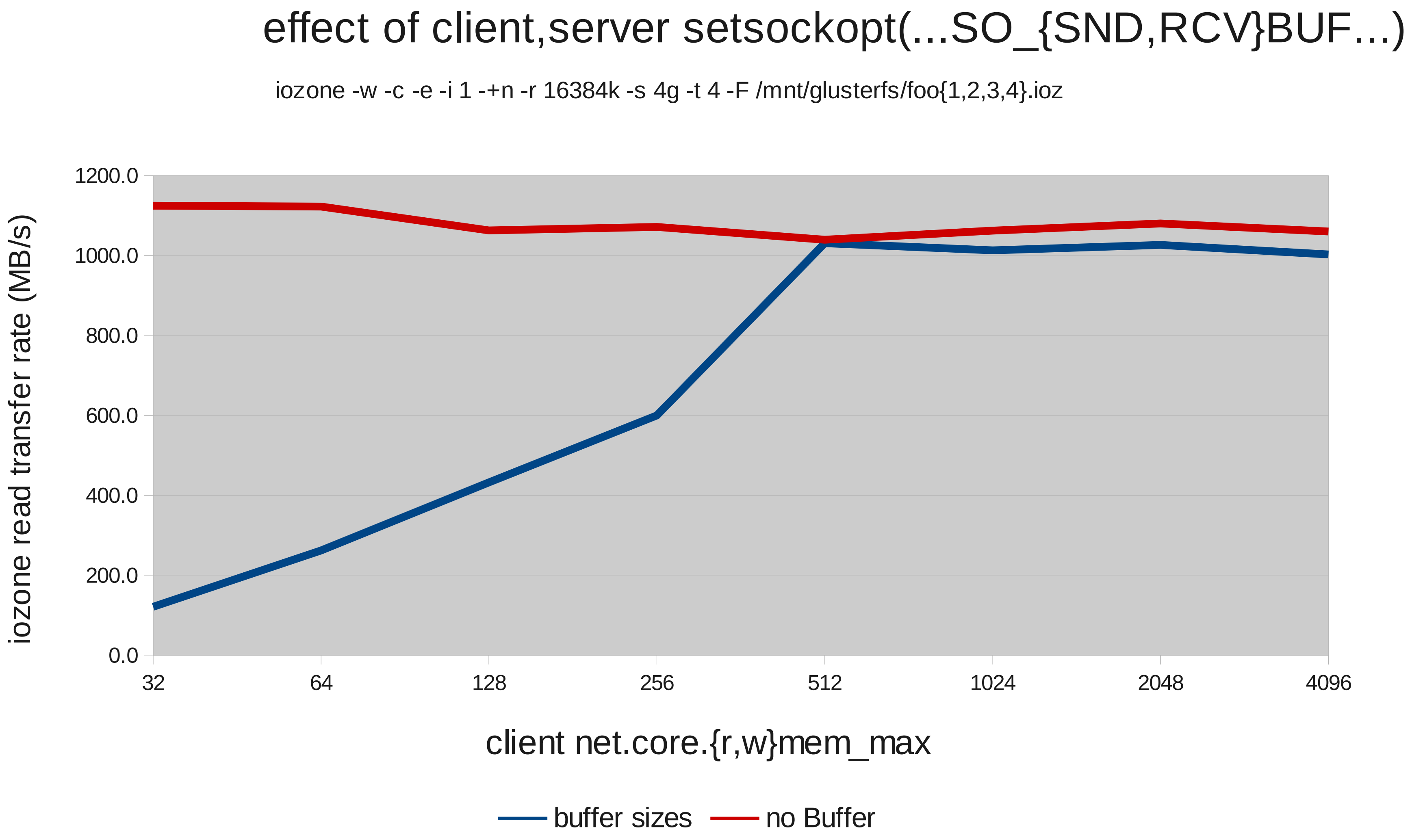
sysctl – ”core” memory settings

- CORE memory settings
  - net.core.(r/w)mem_max
    - max size of (r/w)x socket buffer
  - net.core.(r/w)mem_default
    - default (r/w)x size of socket buffer
  - net.core.optmem_max
    - maximum amount of option memory buffers
  - net.core.netdev_max_backlog
    - how many unprocessed rx packets before kernel starts to drop them
- These settings also impact UDP !

Why Bother ? – Teaser 1



Linux auto tuning – It ROCKS!



Why Bother – A quick teaser

- Two 40Gbit cards back to back (no switch).

```
# ./netperf -l 30 -H 172.17.200.82
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
172.17.200.82 (172.17.200.82) port 0 AF_INET : spin interval : demo
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00      8868.76
```

lspci – details

```
# lspci -vvvs 81:00.0
81:00.0 Ethernet controller: Mellanox Technologies MT27500 Family [ConnectX-3]
        Subsystem: Mellanox Technologies Device 0035
        Control: I/O- Mem+ BusMaster+ SpecCycle- MemWINV- VGASnoop- ParErr+ Stepping- SERR+ FastB2B- DisINTx+
        Status: Cap+ 66MHz- UDF- FastB2B- ParErr- DEVSEL=fast >TAbort- <TAbort- <MAbort- >SERR- <PERR- INTx-
        Latency: 0, Cache Line Size: 64 bytes
        Interrupt: pin A routed to IRQ 56
        Capabilities: [48] Vital Product Data
                Product Name: CX313A - ConnectX-3 QSFP
                Read-only fields:
                        [PN] Part number: MCX313A-BCBT
                        [V0] Vendor specific: PCIe Gen3 x8
                        [RV] Reserved: checksum good, 0 byte(s) reserved
        Capabilities: [60] Express (v2) Endpoint, MSI 00
                DevCap:         MaxPayload 256 bytes, PhantFunc 0, Latency L0s <64ns, L1 unlimited
                        ExtTag- AttnBtn- AttnInd- PwrInd- RBE+ FLReset+
                DevCtl:         Report errors: Correctable+ Non-Fatal+ Fatal+ Unsupported-
                        RlxdOrd- ExtTag- PhantFunc- AuxPwr- NoSnoop- FLReset-
                        MaxPayload 256 bytes, MaxReadReq 4096 bytes
                DevSta:         CorrErr- UncorrErr- FatalErr- UnsuppReq- AuxPwr- TransPend-
                LnkCap:         Port #8, Speed unknown, Width x8, ASPM L0s, Latency L0 unlimited, L1 unlimited
                        ClockPM- Surprise- LLActRep- BwNot-
                LnkCtl:         ASPM Disabled; RCB 64 bytes Disabled- Retrain- CommClk+
                        ExtSynch- ClockPM- AutWidDis- BWInt- AutBWInt-
                LnkSta:         Speed unknown, Width x8, TrErr- Train- SlotClk+ DLActive- BWMgmt- ABWMgmt-
        Capabilities: [148] Device Serial Number 00-02-c9-03-00-05-6a-a8
        Capabilities: [18c] #19
        Kernel driver in use: mlx4_core
        Kernel modules: mlx4_core
```

NOTE Lots of data truncated for brevity

Why Bother – A quick teaser

- Check MTU

```
# ifconfig eth4

eth4      Link encap:Ethernet  HWaddr 00:02:C9:36:79:80
          inet addr:172.17.200.50  Bcast:172.17.200.255
Mask:255.255.255.0
          inet6 addr: fe80::202:c9ff:fe36:7980/64 Scope:Link
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:2634628 errors:0 dropped:0 overruns:0 frame:0
          TX packets:31433648 errors:0 dropped:0 overruns:0
carrier:0
          collisions:0 txqueuelen:1000
          RX bytes:184742056 (176.1 MiB)  TX bytes:47590480340
(44.3 GiB)
```

Why Bother – A quick teaser

- *ifconfig eth0 mtu 9000*

```
# ./netperf -l 30 -H 172.17.200.82
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
172.17.200.82 (172.17.200.82) port 0 AF_INET : spin interval : demo
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00    23923.65
```

- Changing MTU 9 Gb/sec -> 24 Gbit /sec

Tuning – debug simple netperf TCP_STREAM test

- Found the bottleneck !
  - CPU bound on RX side

```
04:39:33 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
04:39:36 PM  all    0.02    0.00    2.88    0.00    0.00    3.38    0.00    0.00   93.73
04:39:36 PM    0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    4    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    5    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    6    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    7    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    8    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM    9    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   10    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   11    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   12    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   13    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   14    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
04:39:36 PM   15    0.33    0.00   45.67    0.00    0.00   54.00    0.00    0.00    0.00
```

Tuning– first pass bottleneck resolution

- Disable irqbalance
  - We will pin the interrupts where we want them
  - But where do they go ?
- Look in /sys to see if there are hints
  - A value of -1 could mean error or undefined
  - In this case we see that the pci slot is tied to NUMA node 1
  - Move the interrupts there
- Alternative is trial and error

Tuning– first pass bottleneck resolution

```
#dmesg | grep -i numa

NUMA: Allocated memnodemap from 9000 - 90c0
NUMA: Using 30 for the hash shift.
pci_bus 0000:00: on NUMA node 0 (pxm 0)
pci_bus 0000:80: on NUMA node 1 (pxm 1)

# lspci | grep Mellanox
81:00.0 Ethernet controller: Mellanox Technologies MT27500 Family
[ConnectX-3]

# find /sys  -name numa_node | grep 81:00.0
/sys/devices/pci0000:80/0000:80:02.0/0000:81:00.0/numa_node


# cat /sys/devices/pci0000:80/0000:80:02.0/0000:81:00.0/numa_node
1

# cat /sys/devices/pci0000:80/0000:80:02.0/0000:81:00.0/local_cpulist
8-15
```

Tuning – second pass setup

- Disable irqbalance
  - `irqbalance stop`
  - `chkconfig irqbalance off`
- Identify the interrupts
  - `grep eth4 /proc/interrupts`
- But wait, mlx also has an second driver!
  - `grep mlx /proc/interrupts`
- or

```
# ls /sys/devices/pci0000:80/0000:80:02.0/0000:81:00.0/msi_irqs
177  178  179  180  181  182  183  184  185  186  187  188  189
190  191  192  193  194  195  196  197
```

Tuning – move the interrupts

- Map the interrupts to the proper cores for the NUMA node
  - CPU cores designated by bitmap
  - Use `` `numactl --hardware` `` to check core mappings to numa nodes
  - Understand the layout of the cache in relationship to the cores
- Remember these values do not persistent across reboots!
- Set IRQ affinity
  - `echo 80 > /proc/irq/192/smp_affinity`
  - Use “tuna'

Tuning – irqbalance disabled, netperf pinning

- Rerun the tests, pin the netperf TX and RX to core 12

```
# ./netperf -l 30 -H 172.17.200.82 -T 12,12
TCP STREAM TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET
to 172.17.200.82 (172.17.200.82) port 0 AF_INET : spin
interval : demo : cpu bind
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00    25609.34
```

- Hmmm, not really much better

Tuning – second pass

- mpstat on the receiver

```
11:45:04 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
11:45:07 PM  all    0.02    0.00    5.02    0.00    0.00    0.02    0.00    0.00   94.94
11:45:07 PM    0    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    4    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    5    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    6    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    7    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    8    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM    9    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM   10    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM   11    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM   12    0.33    0.00   77.08    0.00    0.00    0.66    0.00    0.00   21.93
11:45:07 PM   13    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM   14    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:07 PM   15    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
```

Tuning – second pass

- mpstat on the transmit

```
11:45:03 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
11:45:06 PM  all    0.08    0.00    3.52    0.00    0.00    0.19    0.00    0.00   96.20
11:45:06 PM    0    0.33    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.67
11:45:06 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    2    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    4    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    5    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    6    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    7    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    8    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM    9    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM   10    0.00    0.00    0.00    0.00    0.00    0.43    0.00    0.00   99.57
11:45:06 PM   11    0.00    0.00    0.00    0.00    0.00    0.34    0.00    0.00   99.66
11:45:06 PM   12    0.70    0.00   57.49    0.00    0.00    2.44    0.00    0.00   39.37
11:45:06 PM   13    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:45:06 PM   14    0.00    0.00    0.33    0.00    0.00    0.00    0.00    0.00   99.67
11:45:06 PM   15    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
```

Tuning – step 2 not clear

- No apparent cpu bottleneck
- Lets try looking at process

```
# perf top -p 37590
Events: 14K cycles
 27.04%  [kernel]      [k] copy_user_generic_string
  6.01%  [kernel]      [k] alloc_pages_current
  5.61%  [kernel]      [k] __alloc_pages_nodemask
  4.87%  [kernel]      [k] get_page_from_freelist
  4.54%  [kernel]      [k] tcp_sendmsg
  2.36%  [kernel]      [k] put_page
  2.13%  [kernel]      [k] list_del
```

- netperf is spending a lot of time generating data

Tuning – step 3

- Try TCP_SENDFILE

```
# ./netperf -l 30 -H 172.17.200.82 -T 12,12 -t
TCP_SENDFILE
TCP SENDFILE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET
to 172.17.200.82 (172.17.200.82) port 0 AF_INET : spin
interval : demo : cpu bind
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00    34106.58
```

- Looking Better !

Tuning – are we done ?

- Look for bottlenecks
    - Transmit is CPU bound

```
11:54:54 PM  CPU    %usr   %nice    %sys %iowait    %irq   %soft  %steal  %guest   %idle
11:54:57 PM  all    0.08    0.00    6.16    0.00    0.00    0.11    0.00    0.00   93.65
11:54:57 PM    0    0.33    0.00    0.33    0.00    0.00    0.00    0.00    0.00   99.34
11:54:57 PM    1    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    2    0.33    0.00    0.00    0.00    0.00    0.00    0.00    0.00   99.67
11:54:57 PM    3    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    4    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    5    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    6    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    7    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    8    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM    9    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM   10    0.00    0.00    0.00    0.00    0.00    0.39    0.00    0.00   99.61
11:54:57 PM   11    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM   12    1.00    0.00   97.66    0.00    0.00    1.34    0.00    0.00    0.00
11:54:57 PM   13    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM   14    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
11:54:57 PM   15    0.00    0.00    0.00    0.00    0.00    0.00    0.00    0.00  100.00
```

Tuning – checking ethtool -S eth4

- Check for errors, pause frames, etc.
- Check nic on TX side

```
# ethtool -S eth4
NIC statistics:
     rx_packets: 135224755
     tx_packets: 1137704051
     rx_bytes: 8729946637
     tx_bytes: 9906371184752
     rx_errors: 0
     tx_errors: 0
     rx_dropped: 0
     tx_dropped: 0
     tso_packets: 20844101
     queue_stopped: 92899164
     wake_queue: 92899164
```

Tuning – sysctl settings

- We need more buffers
  - net.core.netdev_max_backlog = 250000
  - net.core.wmem_max = 16777216
  - net.core.rmem_default = 16777216
  - net.core.wmem_default = 16777216
  - net.core.optmem_max = 16777216
  - net.ipv4.tcp_mem = 16777216 16777216 16777216
  - net.ipv4.tcp_rmem = 4096 87380 16777216
  - net.ipv4.tcp_wmem = 4096 65536 16777216
  - net.core.rmem_max = 16777216

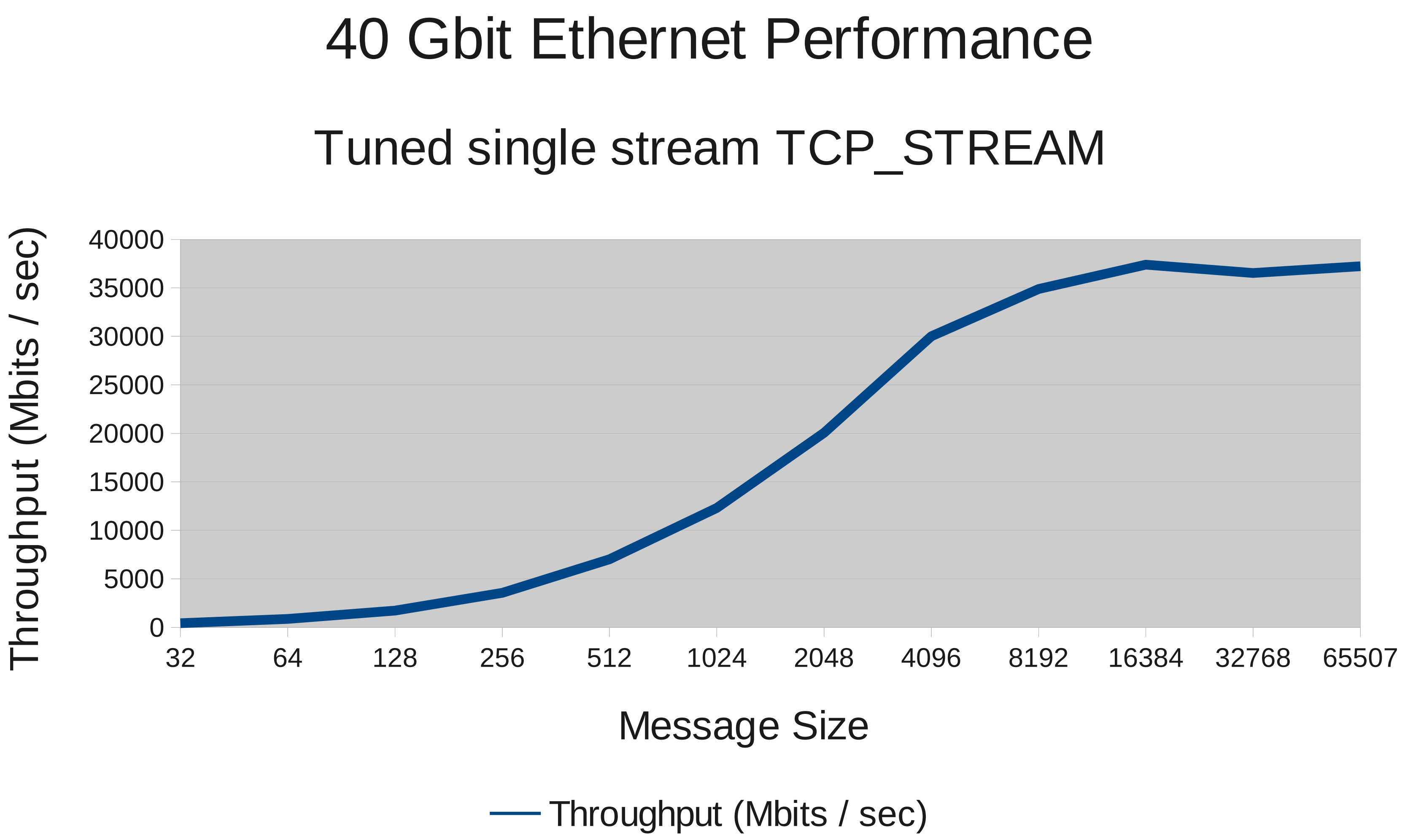
Tuning – step 4

- More buffers

```
# ./netperf -l 30 -H 172.17.200.82 -T 12,12 -t
TCP_SENDFILE
TCP SENDFILE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET
to 172.17.200.82 (172.17.200.82) port 0 AF_INET : spin
interval : demo : cpu bind
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec

 87380  16384  16384    30.00    37354.41
```

- We are done !

Tuning – throughput graph



Tuning – sanity check

- Sometimes mistuning can show that it is working

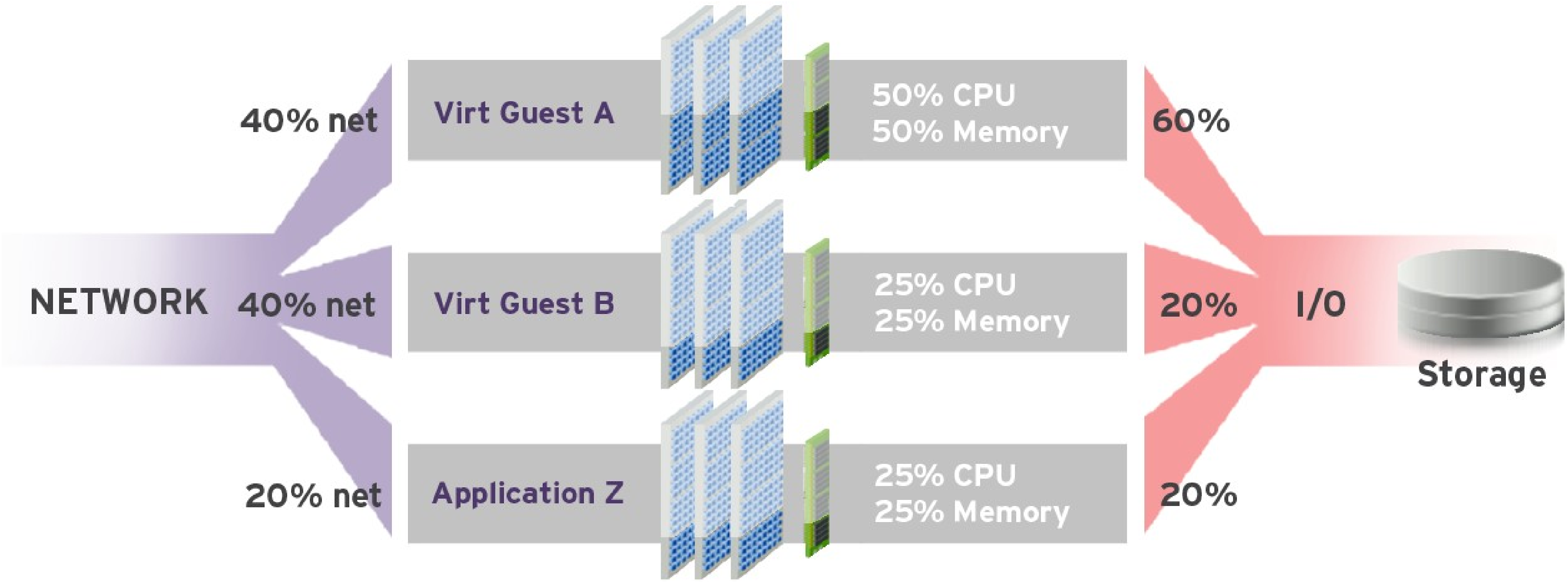
```
# ./netperf -l 30 -H 172.17.200.82 -T 12,2 -t TCP_SENDFILE
TCP SENDFILE TEST from 0.0.0.0 (0.0.0.0) port 0 AF_INET to
172.17.200.82 (172.17.200.82) port 0 AF_INET : spin interval : demo :
cpu bind
Recv   Send    Send
Socket Socket  Message  Elapsed
Size   Size    Size     Time     Throughput
bytes  bytes   bytes    secs.    10^6bits/sec


87380  16384  16384    30.00    13033.89
```

- 37 Gb -> 13 Gb due to crossing NUMA boundary
  - OUCH !

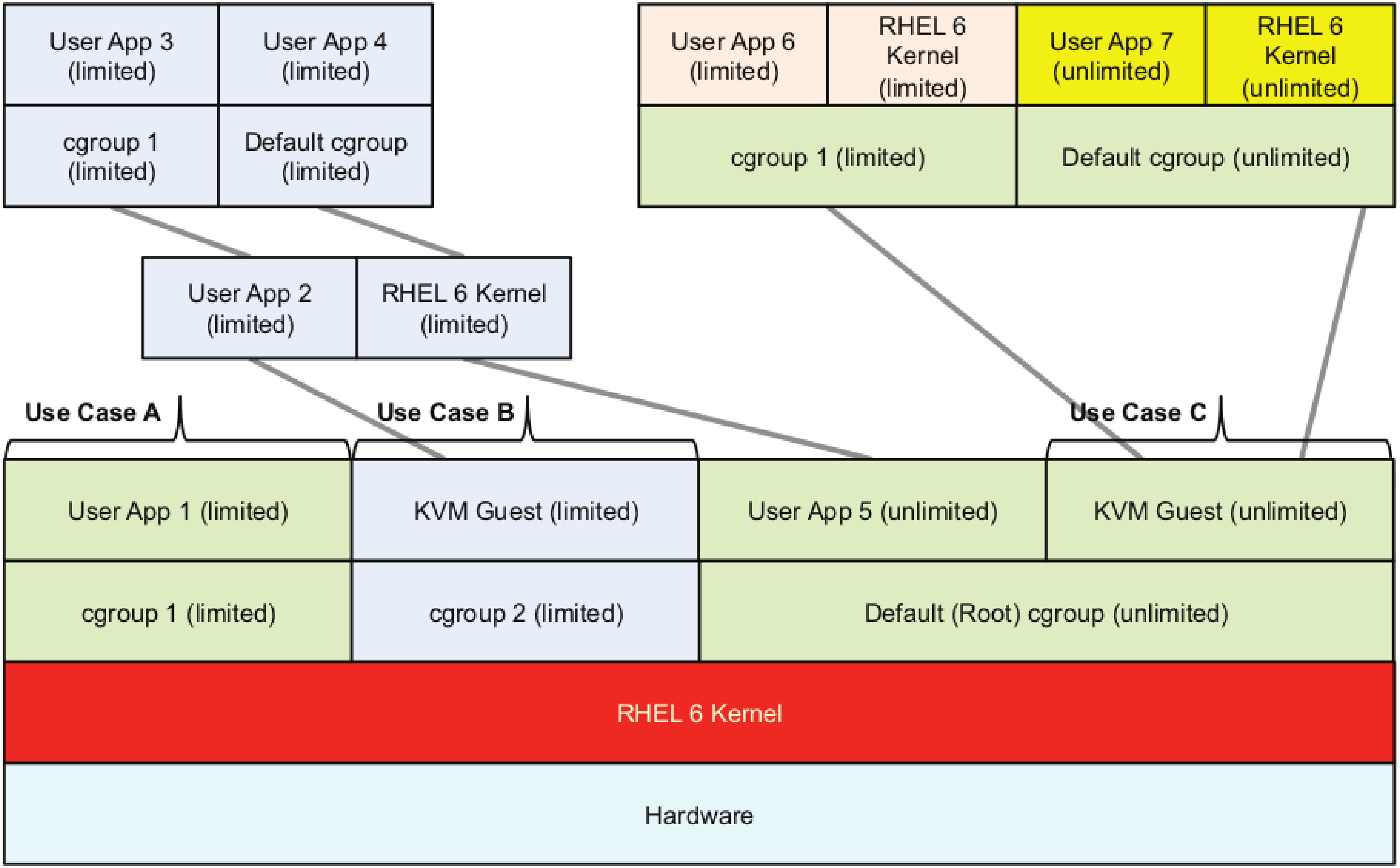
Throttling – cgroups

- Control Group (Cgroups) for
  - CPU/Memory/Network/Disk
- Benefit:
  - guarantee Quality of Service
  - dynamic resource allocation
- Ideal for managing any multi-application environment
- From back-ups to the Cloud

Throttling – cgroups in Action



cgroups Architecture



Cgroup default mount points

```
# cat /etc/cgconfig.conf

mount {
        cpuset      = /cgroup/cpuset;
        cpu   = /cgroup/cpu;
        cpuacct     = /cgroup/cpuacct;
        memory      = /cgroup/memory;
        devices     = /cgroup/devices;
        freezer     = /cgroup/freezer;
        net_cls     = /cgroup/net_cls;
        blkio = /cgroup/blkio;
}
```

```
# ls -l /cgroup

drwxr-xr-x 2 root root 0 Jun 21 13:33 blkio
drwxr-xr-x 3 root root 0 Jun 21 13:33 cpu
drwxr-xr-x 3 root root 0 Jun 21 13:33 cpuacct
drwxr-xr-x 3 root root 0 Jun 21 13:33 cpuset
drwxr-xr-x 3 root root 0 Jun 21 13:33 devices
drwxr-xr-x 3 root root 0 Jun 21 13:33 freezer
drwxr-xr-x 3 root root 0 Jun 21 13:33 memory
drwxr-xr-x 2 root root 0 Jun 21 13:33 net_cls
```

Cgroup how-to

1GB/2CPU subset of a 16GB/8CPU system

```
#numactl --hardware
#mount -t cgroup xxx /cgroups
#mkdir -p /cgroups/test
#cd /cgroups/test
#echo 1 > cpuset.mems
#echo 2-3 > cpuset.cpus
#echo 1G > memory.limit_in_bytes
#echo $$ > tasks
```

cgroups

```
[root@dhcp-100-19-50 ~]# forkoff 20MB 100procs &

[root@dhcp-100-19-50 ~]# top -d 5

top - 12:24:13 up  1:36,  4 users,  load average: 22.70, 5.32, 1.79

Tasks: 315 total,  93 running, 222 sleeping,   0 stopped,   0 zombie

Cpu0  :  0.0%us,  0.2%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st

Cpu1  :  0.0%us,  0.2%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st

Cpu2  :100.0%us,  0.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st

Cpu3  : 89.6%us, 10.0%sy,  0.0%ni,  0.0%id,  0.0%wa,  0.2%hi,  0.2%si,  0.0%st

Cpu4  :  0.4%us,  0.6%sy,  0.0%ni, 98.8%id,  0.0%wa,  0.0%hi,  0.2%si,  0.0%st

Cpu5  :  0.4%us,  0.0%sy,  0.0%ni, 99.2%id,  0.0%wa,  0.0%hi,  0.4%si,  0.0%st

Cpu6  :  0.0%us,  0.0%sy,  0.0%ni,100.0%id,  0.0%wa,  0.0%hi,  0.0%si,  0.0%st

Cpu7  :  0.0%us,  0.0%sy,  0.0%ni, 99.8%id,  0.0%wa,  0.0%hi,  0.2%si,  0.0%st

Mem:  16469476k total,  1993064k used, 14476412k free,    33740k buffers

Swap:  2031608k total,   185404k used,  1846204k free,   459644k cached
```

Verify correct bindings

```
[root@dhcp47-183 test]# echo 0 > cpuset.mems
[root@dhcp47-183 test]# echo 0-3 > cpuset.cpus
[root@dhcp47-183 test]# numastat
                       node0       node1
numa_hit             1648772      438778
numa_miss              23459     2134520
local_node           1648648      423162
other_node             23583     2150136

[root@dhcp47-183 test]# /common/lwoodman/code/memory 4
faulting took 1.616062s
touching took 0.364937s

[root@dhcp47-183 test]# numastat
                       node0       node1
numa_hit             2700423      439550
numa_miss              23459     2134520
local_node           2700299      423934
other_node             23583     2150136
```

incorrect bindings!

```
[root@dhcp47-183 test]# echo 1 > cpuset.mems
[root@dhcp47-183 test]# echo 0-3 > cpuset.cpus
[root@dhcp47-183 test]# numastat
                           node0         node1
numa_hit                 1623318        434106
numa_miss                  23459       1082458
local_node               1623194        418490
other_node                 23583       1098074

[root@dhcp47-183 test]# /common/lwoodman/code/memory 4
faulting took 1.976627s
touching took 0.454322s

[root@dhcp47-183 test]# numastat
                           node0         node1
numa_hit                 1623341        434147
numa_miss                  23459       2133738
local_node               1623217        418531
other_node                 23583       2149354
```

Throttle with cgroups

- Example:
  - Set a 9 Gbit / sec limit on the cgroup

```
# tc qdisc add dev eth1 root handle 10: htb default 10

# tc class add dev eth1 parent 10:10 classid 10:10 htb rate 9gbit ceil 9gbit

# tc filter add dev eth1 parent 10:0 protocol all prio 1 handle 1 cgroup

# echo 0x100010 > /cgroup/net_cls/net_cls.classid
```

Throttle with cgroups

- memory
  - associate a cgroup with a classid that 'tc' utility creates/manages
  - Set upper-bounds
- Example:
  - Set a 9 Gbit / sec limit on the cgroup

```
# tc qdisc add dev eth1 root handle 10: htb default 10
# tc class add dev eth1 parent 10:10 classid 10:10 htb rate 9gbit ceil 9gbit
# tc filter add dev eth1 parent 10:0 protocol all prio 1 handle 1 cgroup
# echo 0x100010 > /cgroup/net_cls/net_cls.classid
```

Network Tuning Tips

- Packet size - MTU
- Buffers
- IRQ affinity
- CPU affinity

Network Tuning Tips

- Separate networks for different functions
  - Use arp_filter to prevent ARP Flux
    - echo 1 > /proc/sys/net/ipv4/conf/all/arp_filter
    - Use /etc/sysctl.conf for permanent

Wrap UP

- Use this talk as suggestions of things to try
  - Our work is based on a private, local network – wide area network will be different
  - Do not assume “my” setting will work for you without some tweaks
  - Your environment is probably different then mine.
    - Experiment ! (but be careful)
- I should be around the Summit for the remainder of the week.
  - Feel free to stop me and ask questions, provide feedback, etc
- There will be members of the Performance team in the booth

For More Information – Other talks

- Performance Analysis & Tuning of Red Hat Enterprise Linux – Shak and Larry
  - Part 1 - Thurs 2:30
  - Part 2 - Thurs 3:40
- Tuning Red Hat Systems for Databases - Sanjay Rao
  - Thurs 4:50
- Red Hat Storage Performance - Ben England
  - Fri 9:45

For More Information

- Reference Architecture Website
  - https://access.redhat.com/knowledge/refarch/TBD
- Principled Technologies
  - http://www.principledtechnologies.com/clients/reports/Red%20Hat/Red%20Hat.htm
- New edition of the “Performance Tuning Guide”
  - http://docs.redhat.com/docs/en-US/Red_Hat_Enterprise_Linux/index.html
- IRQ Balance paper
  - https://access.redhat.com/knowledge/techbriefs/optimizing-red-hat-enterprise-linux-performance-tuning-irq-affinity

Stay connected through the Red Hat Customer Portal

RHEL 6 Network Performance

Watch video

Performance Issues in Red Hat Enterprise Linux (Part 3)

Review Tech brief

Join Red Hat Enterprise Linux

Join Group

access.redhat.com

THE ASSOCIATION OF SUPPORT PROFESSIONALS
Award Winner 2012
The Year's Ten Best Web Support Sites

LIKE US ON FACEBOOK

www.facebook.com/redhatinc

FOLLOW US ON TWITTER

www.twitter.com/redhatsummit

TWEET ABOUT IT

#redhat

READ THE BLOG

summitblog.redhat.com

GIVE US FEEDBACK

www.redhat.com/summit/survey

Tools – Hardware / Driver Focus

- lspci
- ethtool
- modinfo
- hwloc

Configuration Tools – System Level

- numactl
- tuna
- ifconfig / ip
- tc
- cgroups
- sysctl
- **man**

Monitoring Tools – System Level

- numstat
- mpstat
- vmstat
- watch
- tcpdump / wireshark
- netstat
- oprofile / perf
- sar
- iptraf

sar – some common flags

- Some common flags for sar
    - Adding E gets failure stats
  - # sar -n EDEV - View failure statistics for interfaces
  - # sar -n NFS - View NFS client activity for interfaces
  - # sar -n NFSD - View NFS server activity for interfaces
  - # sar -n (E)IP - View IPv4 activity for interfaces
  - # sar -n (E)ICMP - View ICMPv4 activity for interfaces
  - # sar -n (E)TCP - View TCPv4 activity for interfaces

ethtool – View and change Ethernet card settings

- Works mostly at the HW level
  - ethtool -S – provides HW level stats
    - Counters since boot time, create scripts to calculate diffs
  - ethtool -c - Interrupt coalescing
  - ethtool -g - provides ring buffer information
  - ethtool -k - provides hw assist information
  - ethtool -i - provides the driver information