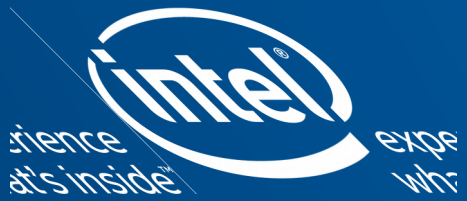


An Adventure of Analysis and Optimisation of the Linux Networking Stack

Marco Varlese, Kim-Marie Jones

28/02/16

Legal Disclaimer

Agenda

- Introduction
- Use cases
- BIOS, Kernel and system settings
- Benchmarks
- Next steps

Introduction

Technical Objectives

Enhance performance of the standard networking stack on Linux

- Improve *out-of-the-box* Linux performance
- Keep using existing software stacks (i.e. TCP, UDP, IPSEC, etc.)

Provide a detailed platform setup guidelines (Cookbook)

- BIOS
- Kernel configuration
- System-level settings (i.e. /proc/sys) & configuration (i.e. queues affinity, scheduling algorithm, TCP congestion control algorithms, etc.)

Bare metal, Virtualisation and **Containers**
will benefit from any performance enhancements

High-level approach

Network traffic to target L2 and L3 (UDP and TCP)

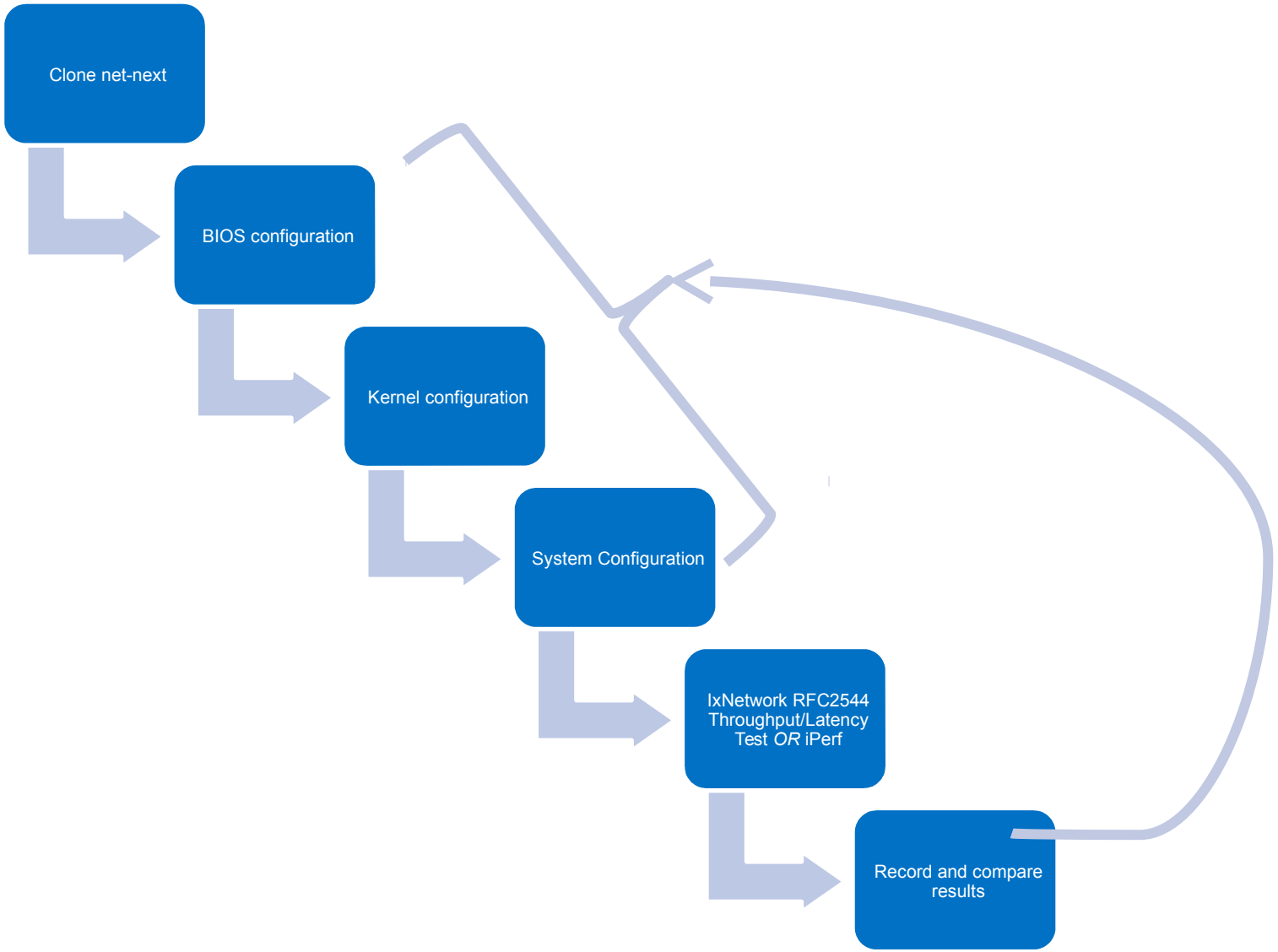
- To extend to SCTP in the future to cover some Telco use cases
- Packet sizes 64, 128, 256, 512, 1024, standard MTU size, jumbo frames and IMIX profile
  - IMIX Profile:

Packet size	Distribution
64	57%
570	7%
594	16%
1518	20%

Performance benchmarking

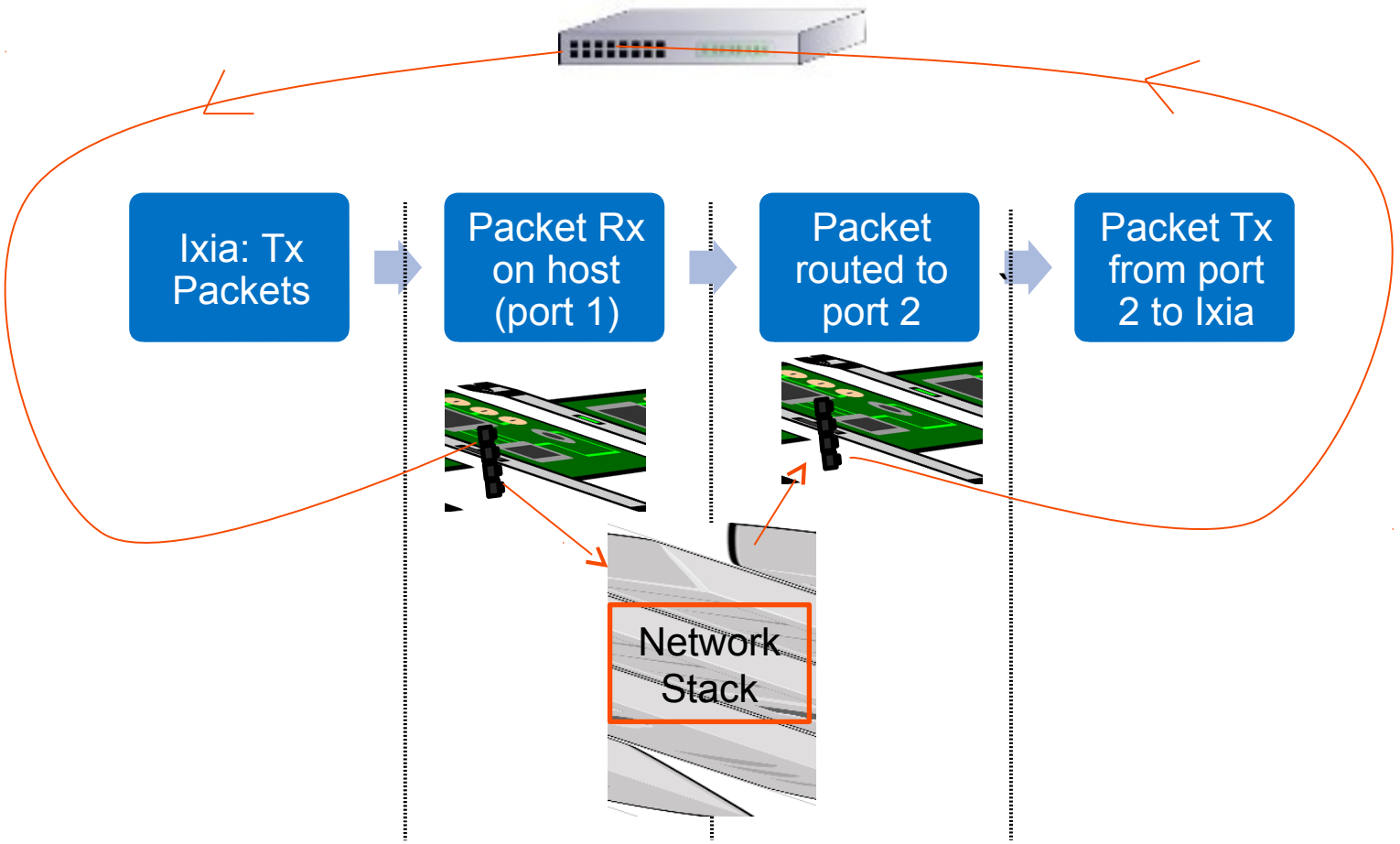
- Network side (throughput / latency / flows scalability)
- Platform side (statistics / counters i.e. CPU utilisation, memory utilisation, interrupts, etc.)

Iterative Testing Procedure

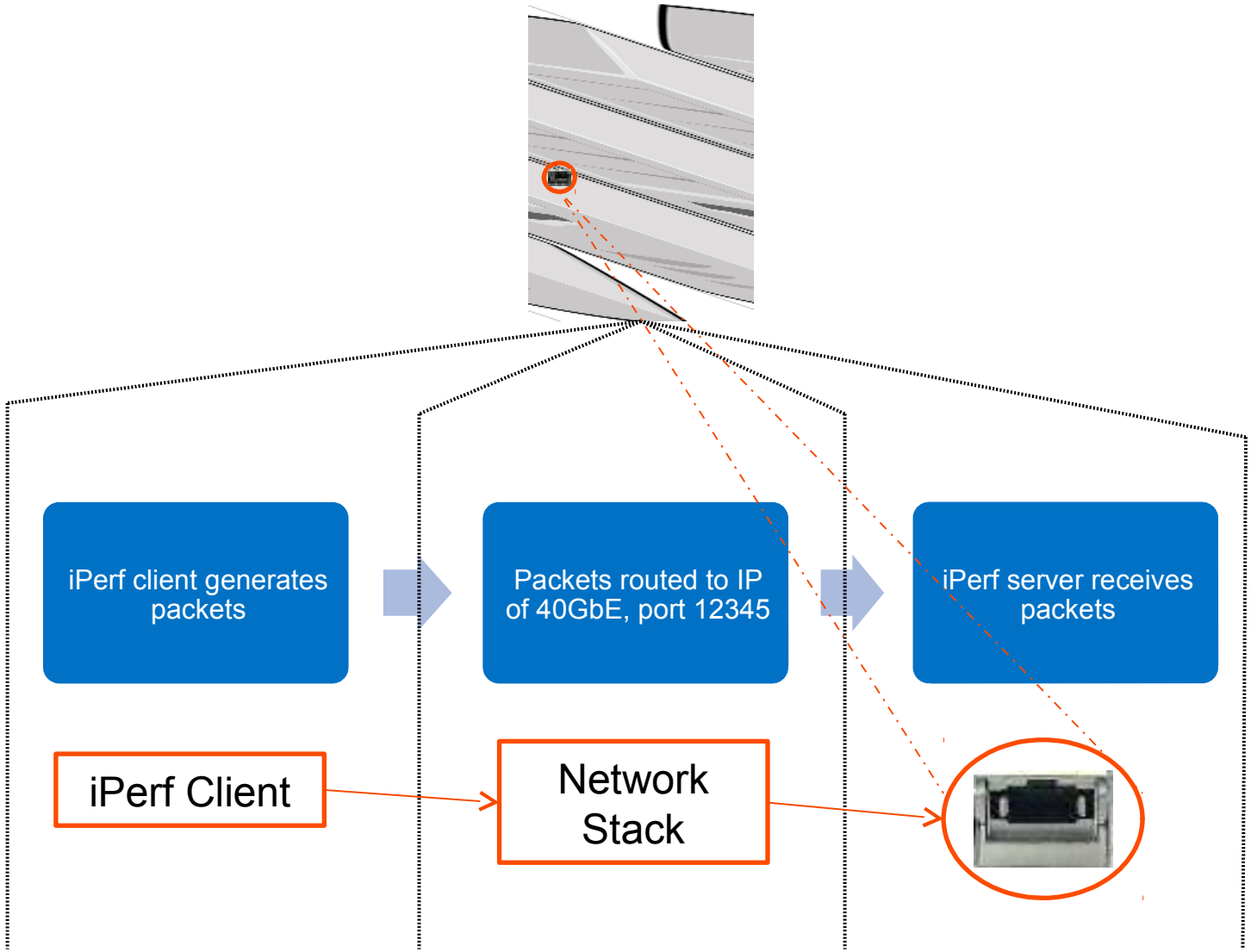


Use Cases

Forwarding Scenario



End-point Scenario



BIOS, Kernel and System Settings

BIOS, Kernel & System Settings

- Identified:
  - A few BIOS settings affecting throughput/latency
  - More than 20 system-level settings affecting throughput/latency and overall system performance and scalability
  - 1 change to Kernel settings which affects throughput
- Each setting was benchmarked to have a thorough understanding of its impact
- Finally, have a best-known configuration for both ***forwarding*** and ***end-point*** scenarios

Configuration of BIOS / P-States

Feature	Orig. Status	New Status	Justification
Hyper-Threading	Enabled	Disabled	Will impact IRQ affinity settings
Turbo Boost	Enabled	Disabled	Unstable performance results, higher jitter
C-States	Enabled	Disabled	Prevent CPUs from sleeping; causes higher latency
P-States	Enabled	Disabled	Run at maximum frequency & voltage at all times

These settings may affect other system characteristics (i.e. power utilisation)

Kernel & System-level Configuration

Kernel

Feature	Orig. value	New Value	Justification
CONFIG_PREEMPT_NONE	N (PREEMPT _RTB=y)	Y	Preemption geared towards throughput

System-Level

/proc/sys/net/core/…

/proc/sys/net/ipv4/…

/proc/irq/<IRQ #>/smp_affinity…

ethtool

Benchmarks

Legal Disclaimer

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at [intel.com].
Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.

§ Configurations: both system setup and test cases are described in the "Use Cases" section of this presentation
§ For more information go to http://www.intel.com/performance.

Hardware Specifications & Benchmarking methodology

NICs:

- Intel® Ethernet Converged Network Adapter
  - X710-DA4 (4 x 10 GbE)
  - XL710-QDA1 (1 x 40 GbE)
  - PCIe v3.0 (8.0 GT/s) x 8 Lane

Platform:

- Gigabyte GA-X99-UD4 Motherboard (Desktop)
  - CPU: Intel(R) Core(TM) i7-5960X CPU @ 3.00GHz
  - RAM: 64 GB

- Each test case runs for 30/60 secs
- Throughput used for each packet size is the average over the total runtime
- CPU idle % is read after the test started, once it stabilises

Some settings

GRUB_CMDLINE_LINUX
intel_pstate=disable ipv6.disable=1 transparent_hugepage=always default_hugepagesz=2M hugepagesz=2M hugepages=4096 isolcpus=0

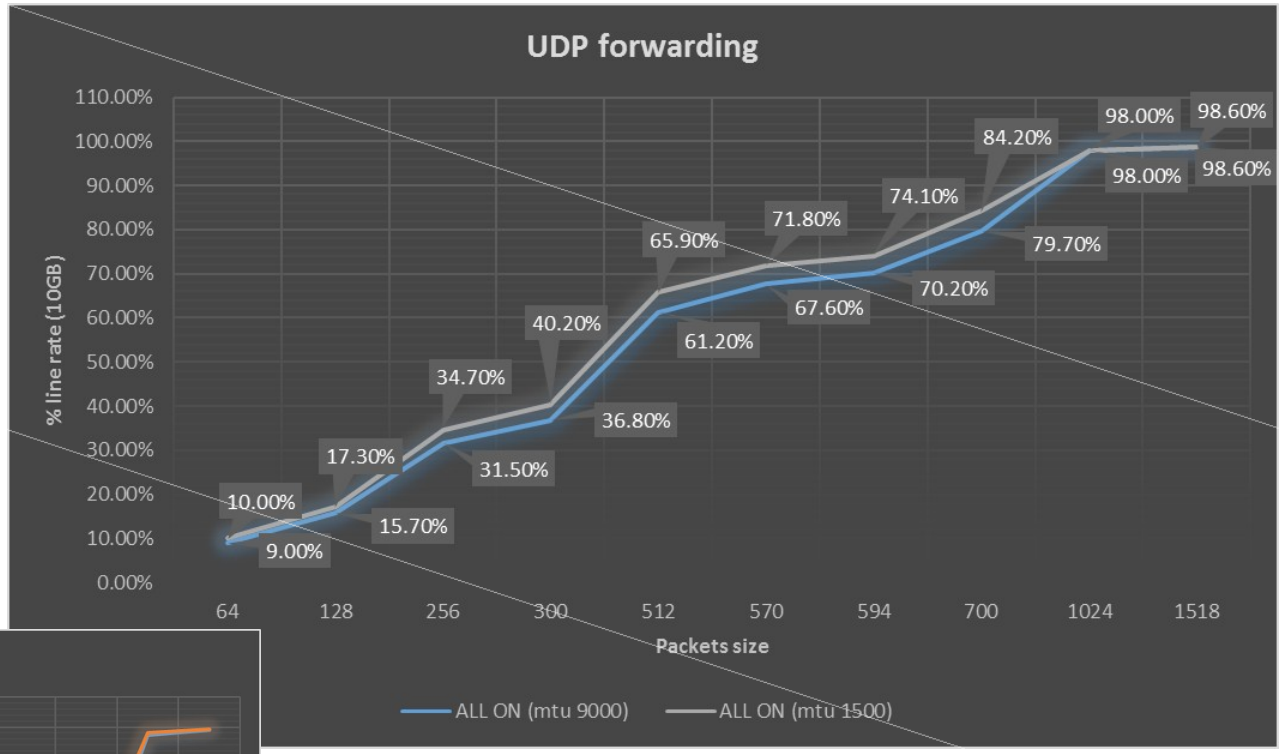
cpupower frequency-set --governor userspace
cpupower --cpu all frequency-set --freq 3.0Ghz

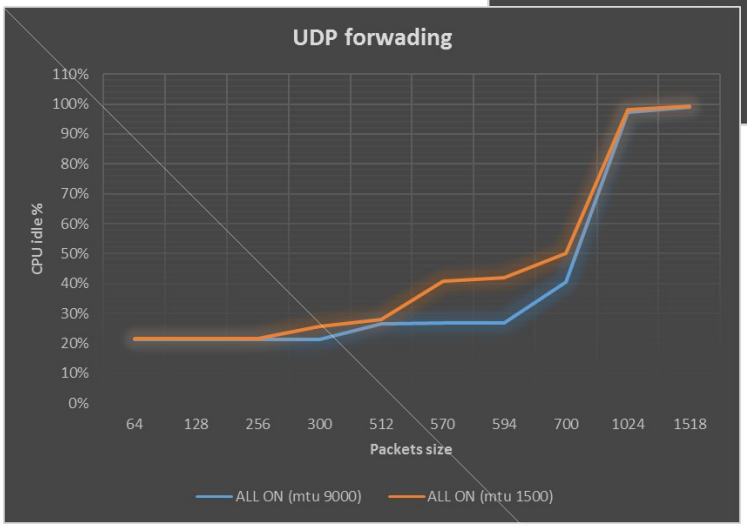
TCP_FULL_OFFLOAD
ethtool -K $dev rxhash on rx on tx on sg on tso on gso on gro on

TCP_NO_OFFLOAD
ethtool -K $dev rxhash off rx off tx off sg off tso off gso off gro off

* 3.0 Ghz is the max frequency allowed by the CPU used for the benchmarks

UDP Forwarding net-next vanilla

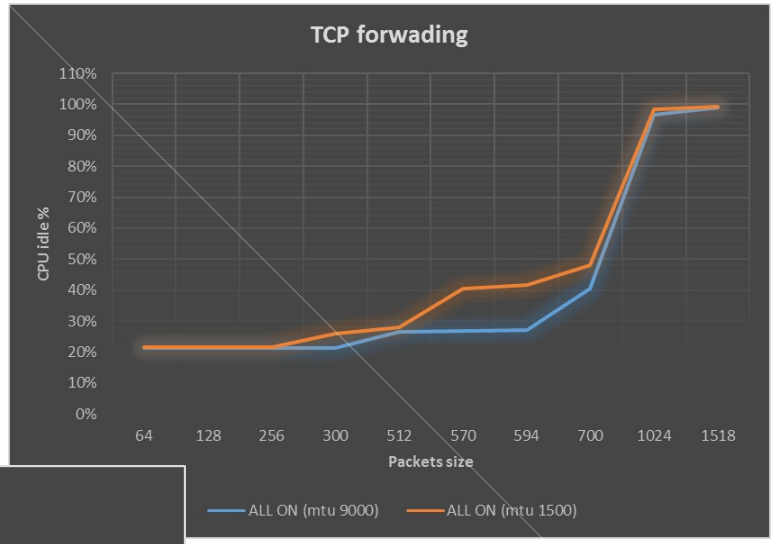


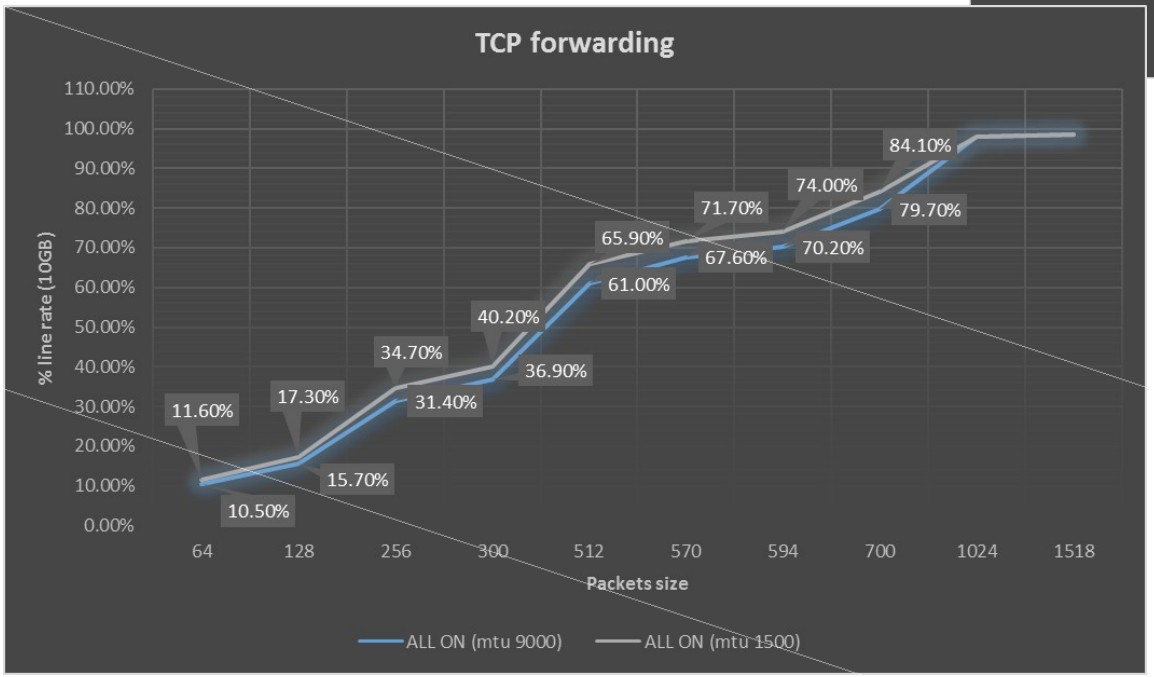


- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied

TCP Forwarding net-next vanilla

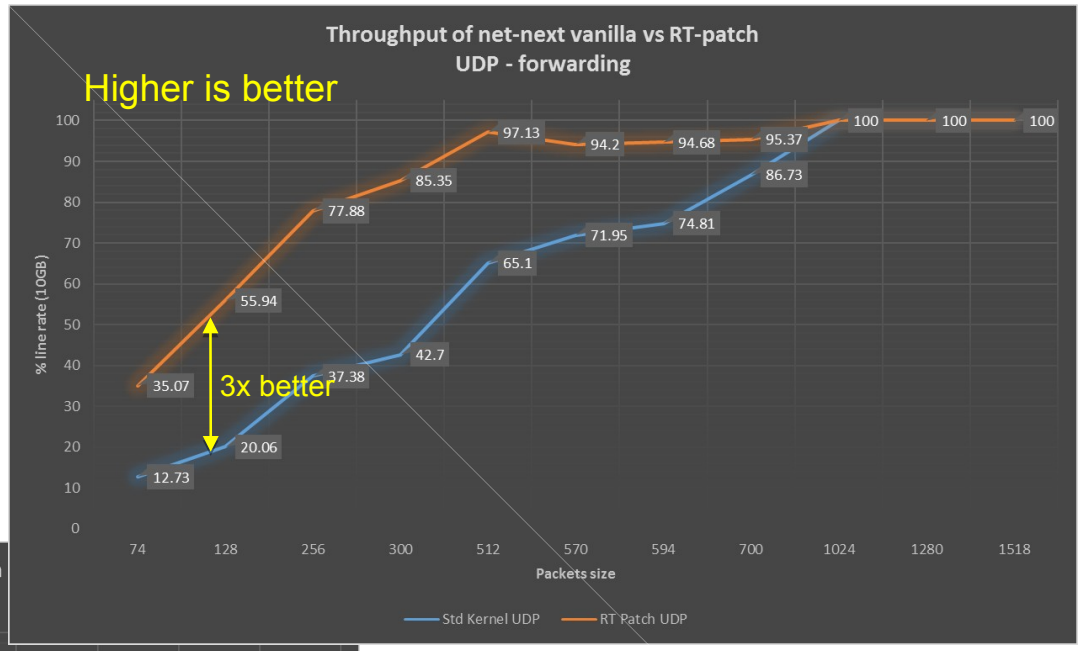
- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied

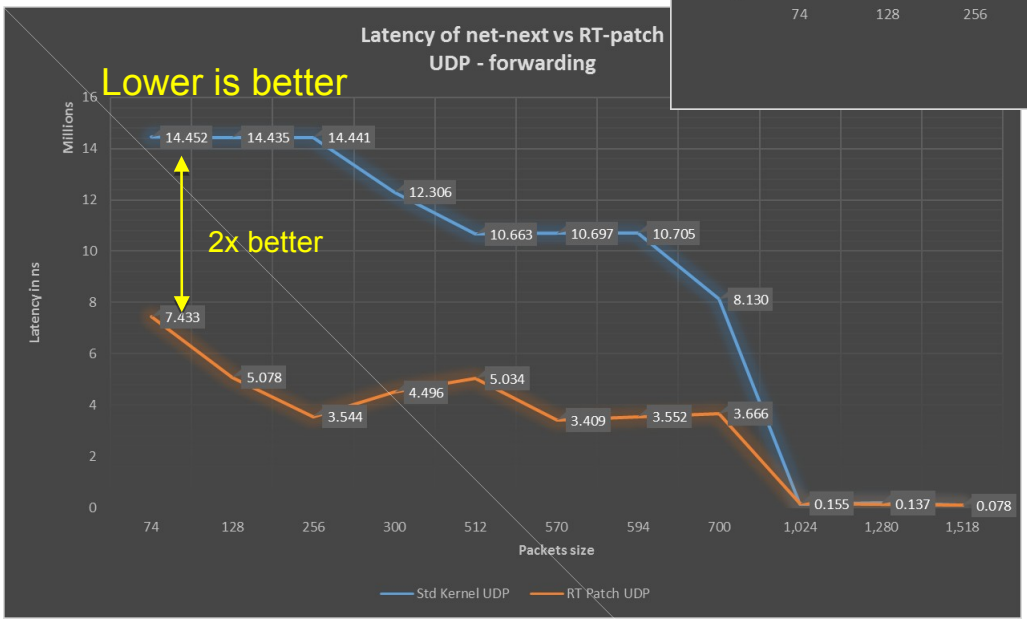


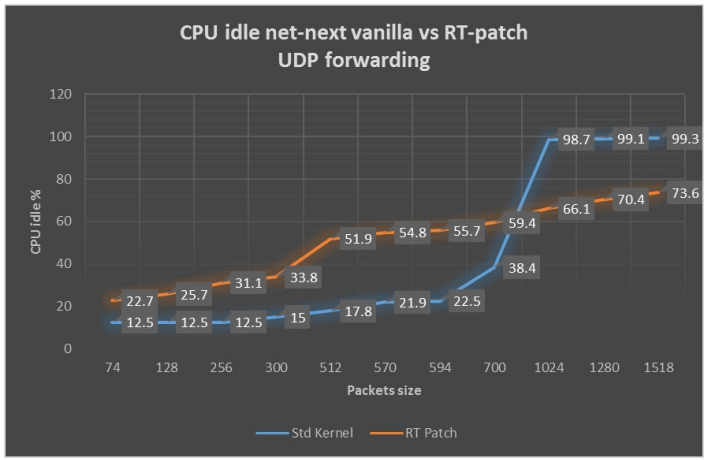


UDP Forwarding net-next vanilla vs RT-patch

- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied

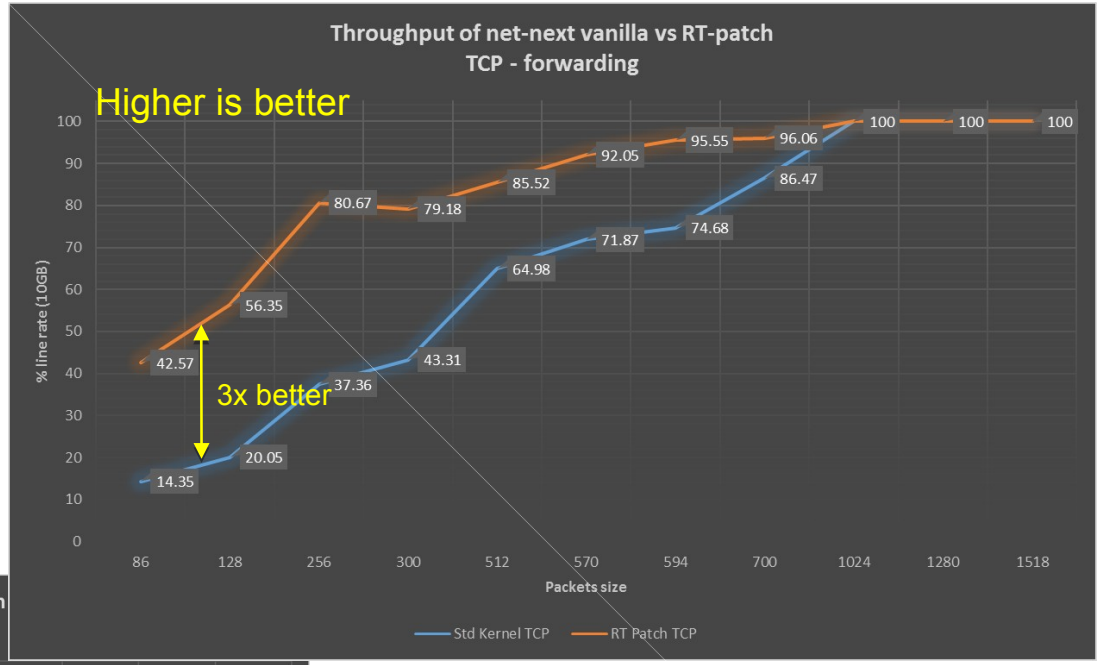


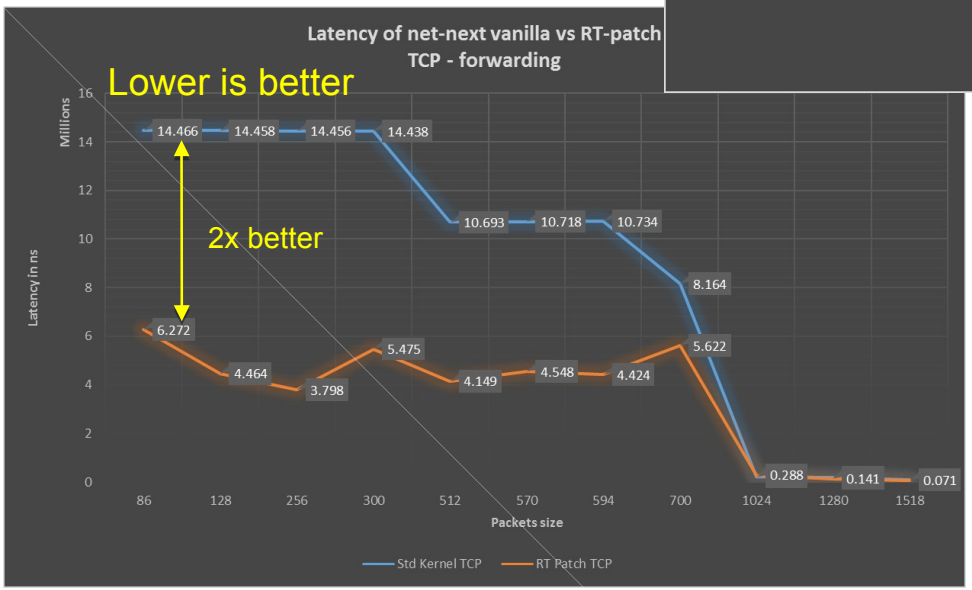


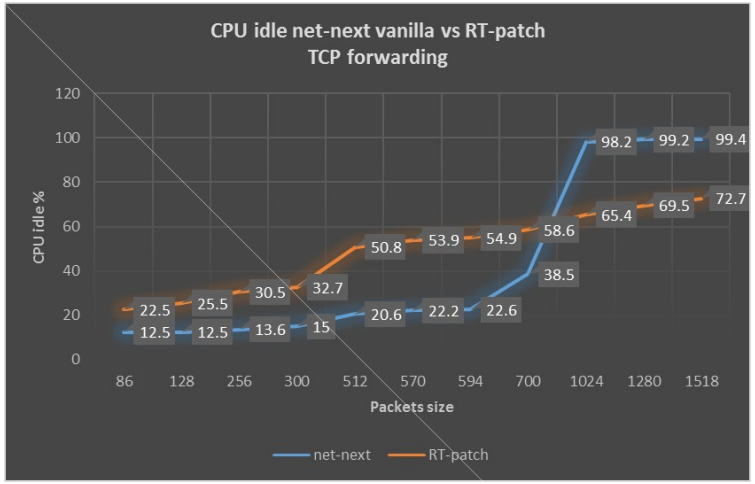


TCP Forwarding net-next vanilla vs RT-patch

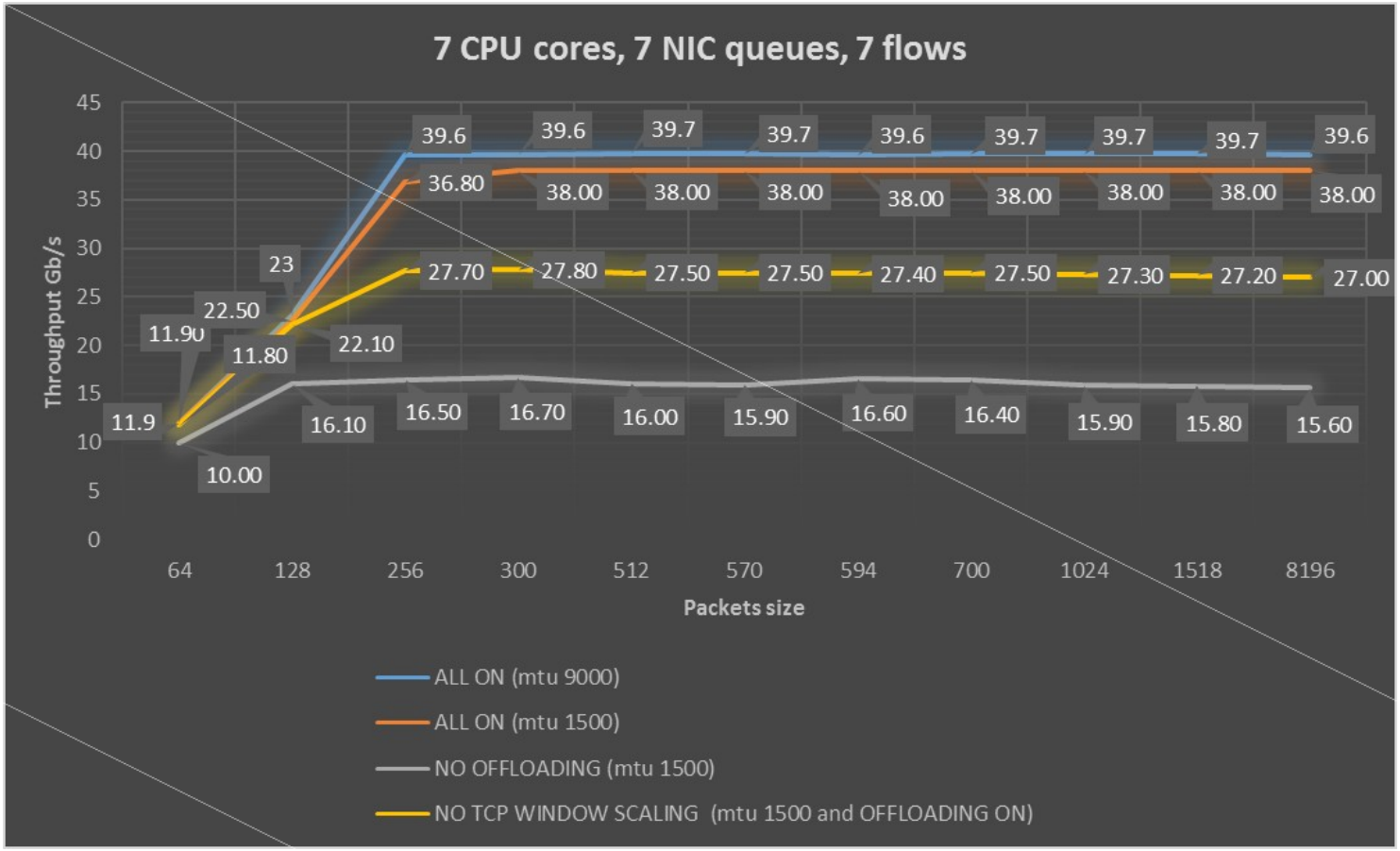
- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied





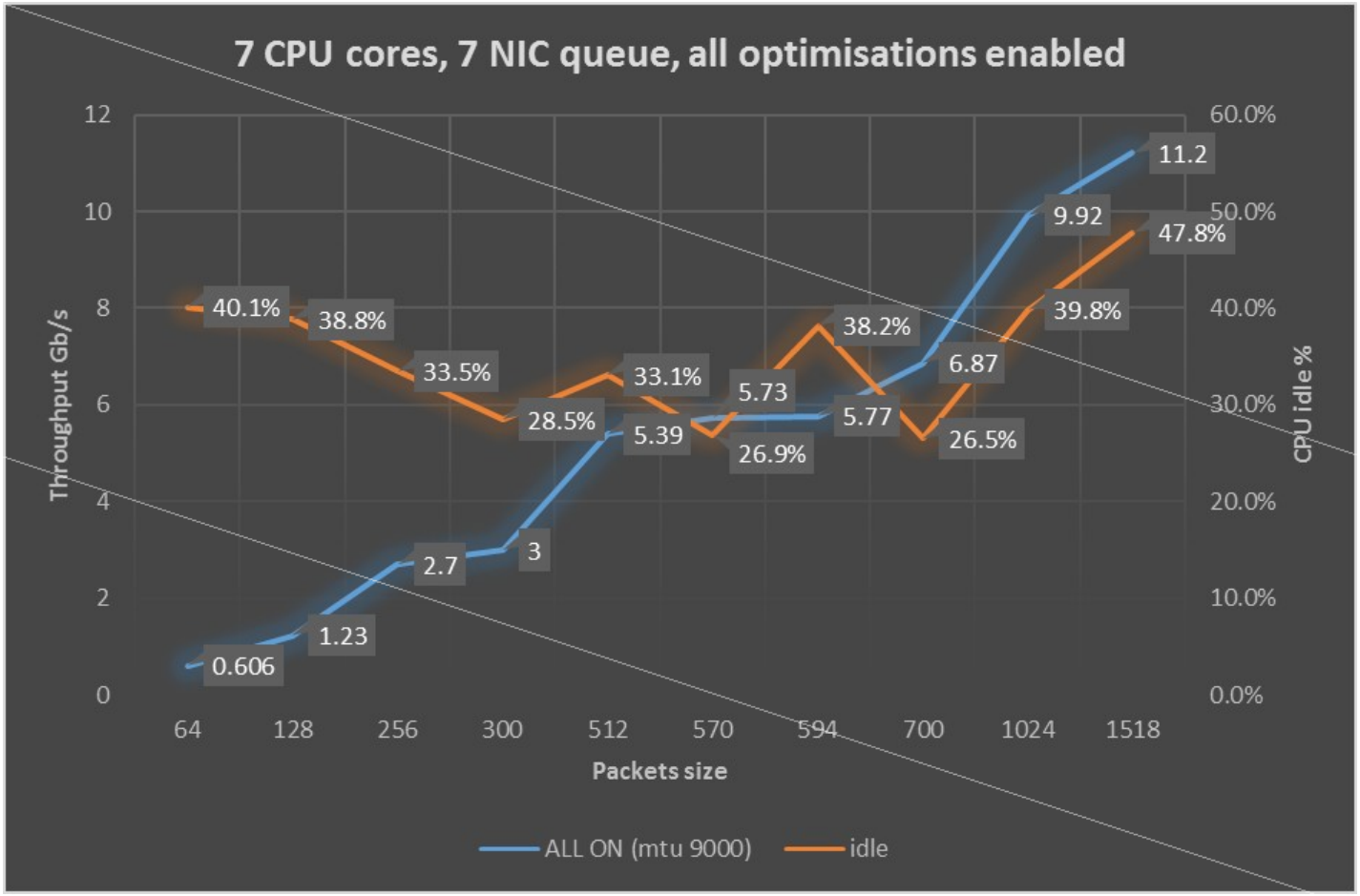


Benchmark results (TCP iperf)



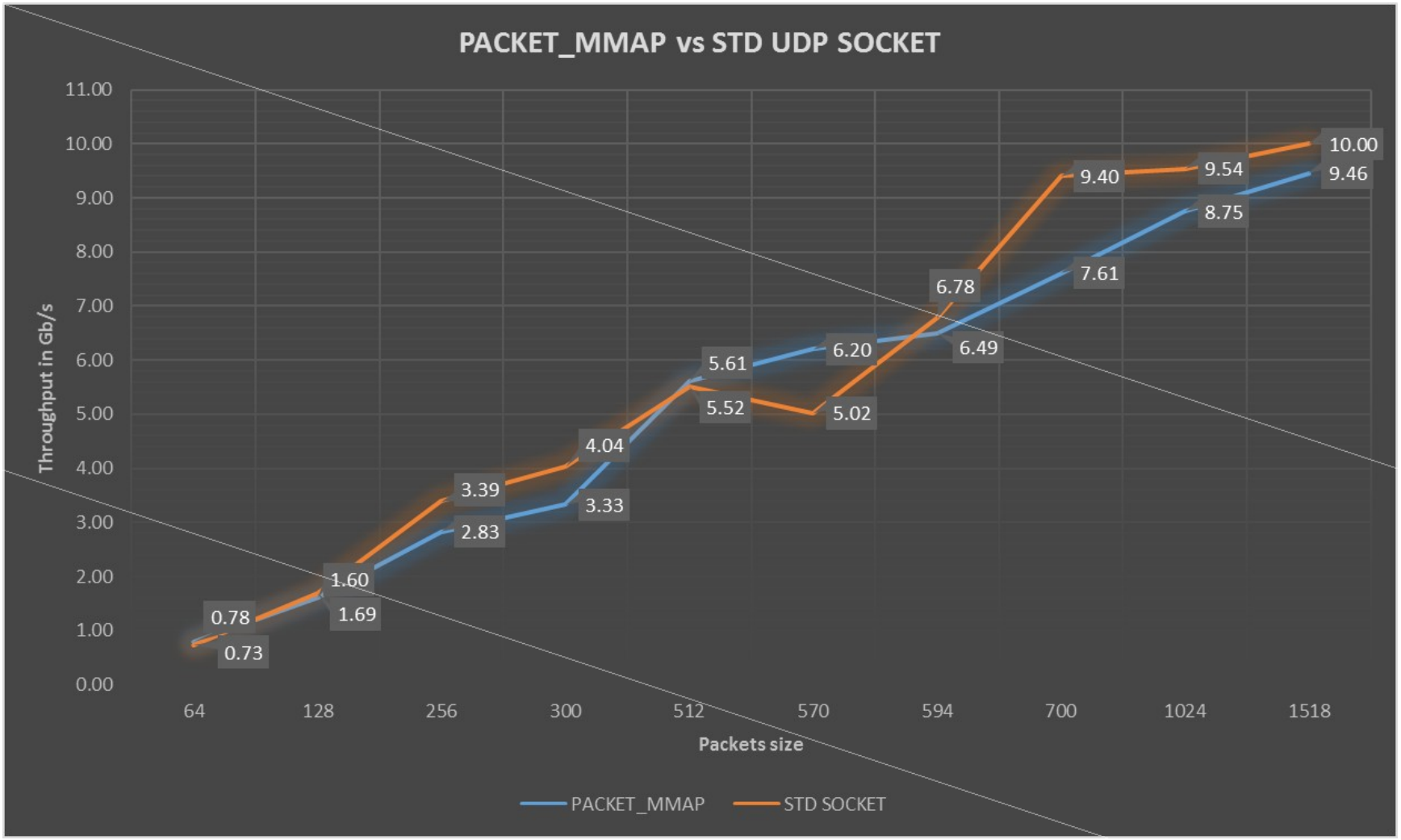
- Throughput scalability almost linear
  - 1 flow = 1.75 Gb/s
  - 7 flows = 11.9 Gb/s
- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied

Benchmark results (UDP iperf)



- BIOS tuning
- Kernel 4.4.0-rc3+
- All optimisations applied

Benchmark results (PACKET_MMAP)



Next steps

What's next

- Benchmark benefits of RT-patch in the end-point scenario
- Dig deeper into the MTU weirdness
- We had troubles using memory from huge pages allocation for the RX/TX rings in the PACKET_MMAP case;
  - We're going to investigate this further
- Create a standalone app for L3 forwarding using PACKET_MMAP
  - Investigate benefits of QDISC_BY_PASS
- Characterize performance optimisation using some real-life scenarios (i.e. customers' use cases, etc.)
- Start working on our 3rd phase and (hopefully) **collaborate** on code enhancements to the *netdev* mailing list soon
  - Identify bottlenecks
    - locks, memory copies, interrupt handlers, cache misses, TLB misses, etc.

Conclusions & Recommendations

Anyone thinking of embarking on a similar “adventure” should:

- Thoroughly optimise at BIOS / kernel / system level
  - Large performance boost from this alone
  - Some optimisations are use-case / hardware specific
- Allow for:
  - Extended ramp-up time on networking stack
  - Extended research time into perf results
    - # cycles / function AND
    - locks, memory copies, interrupt handlers, cache misses, TLB misses, etc.

Thank You

Backup

Introduction

Many Linux users are interested in boosting performance of the general purpose Kernel networking stack

- Different reasons (cost vs benefits, maintainability, manageability, scalability, flexibility, etc.)

Other Linux users are using *kernel-by-pass* technologies (i.e. DPDK) to boost packet processing throughput of software pipelines

- Packets polled directly from user-space “drivers”

We would like to have a best-in-class configuration for the Linux Kernel to have **high throughput**, **high packet density**, **low latency** and **better scalability**

Methodology

Three phase approach

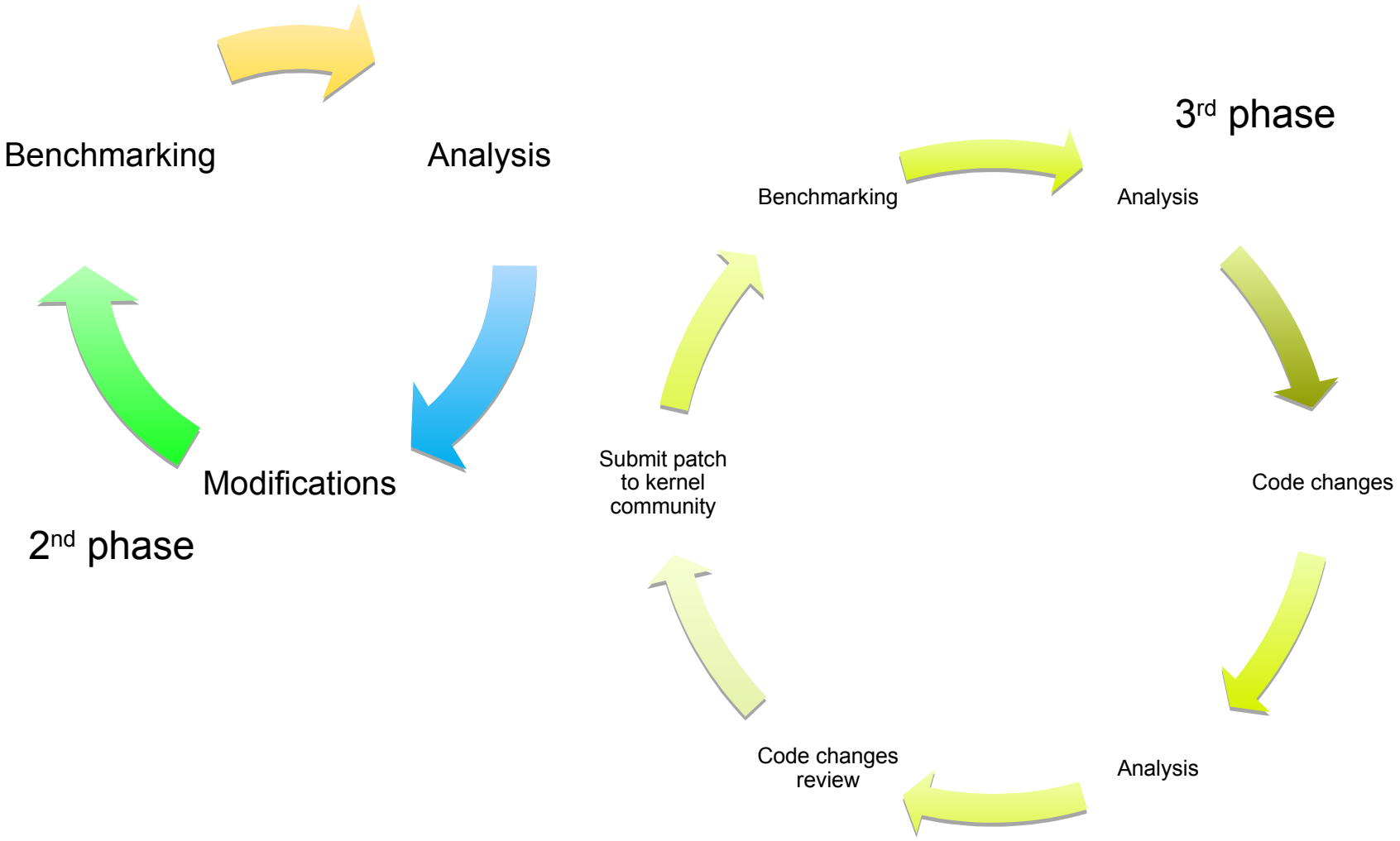
1. Measure performance of a Linux system (+net-next)
   - This represents what users will experience “soon”

2. Measure performance of a Linux system (+net-next) with our BIOS/Kernel/System settings
   - This represents what we can achieve without code changes

3. Measure performance of a Linux system (+net-next) with modified Kernel
   - This represents what we could achieve with our code enhancements being part of Linux kernel

3rd phase yet to be started

Cycle


Benchmarking
Analysis
Modifications
2nd phase
3rd phase
Benchmarking
Analysis
Code changes
Analysis
Code changes review
Submit patch to kernel community

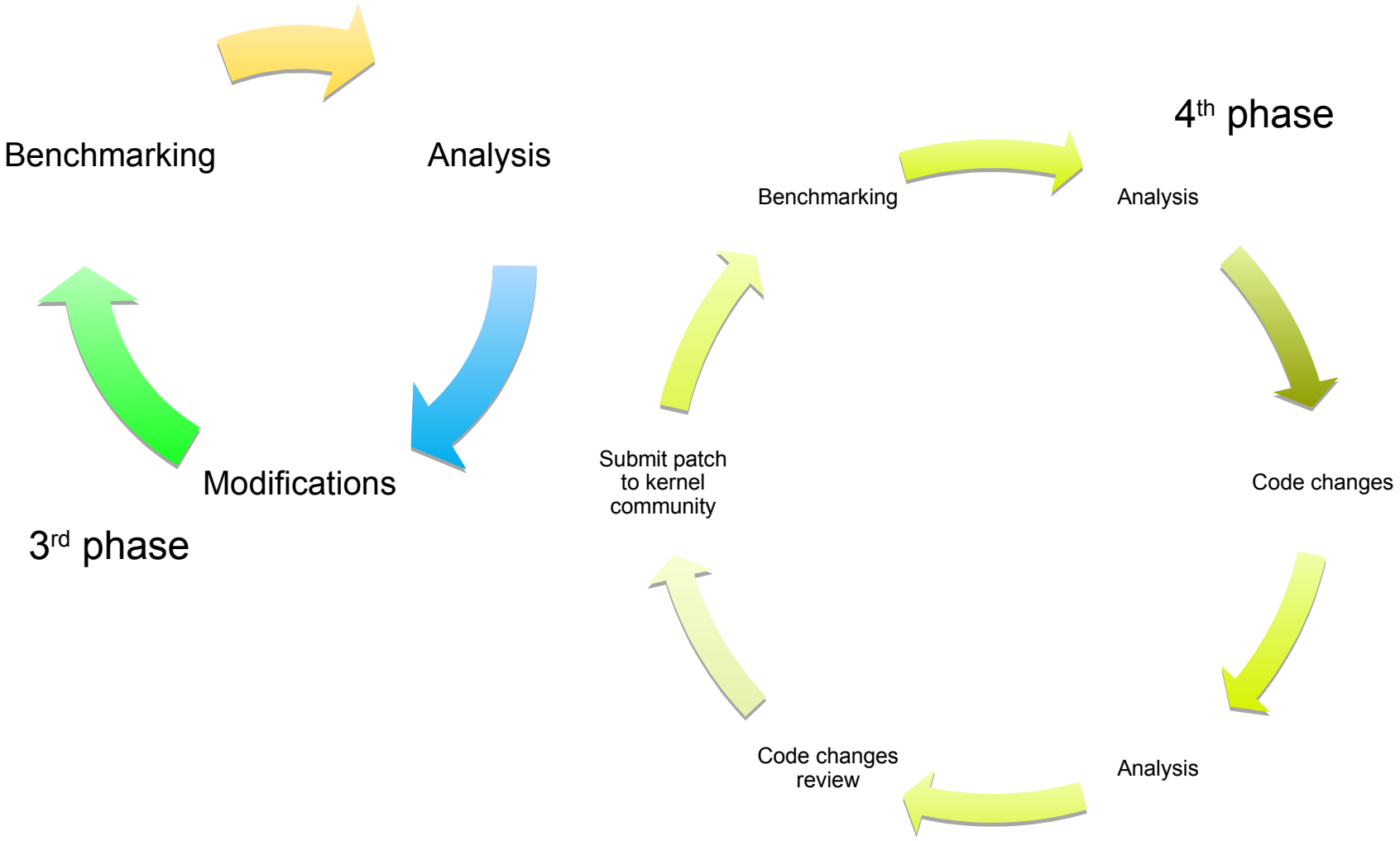
Methodology

Four phases approach

1. Measure performance of a standard Linux distribution (Fedora)
   - This represents what users experience out-of-the-box
2. Measure performance of a Linux system (+net-next)
   - This represents what users will experience “soon”
3. Measure performance of a Linux system (+net-next) with our BIOS/Kernel/System settings
   - This represents what we can achieve without code changes
4. Measure performance of a Linux system (+net-next) with modified Kernel
   - This represents what we could achieve with our code enhancements being part of Linux kernel

4th phase yet to be started

Cycle

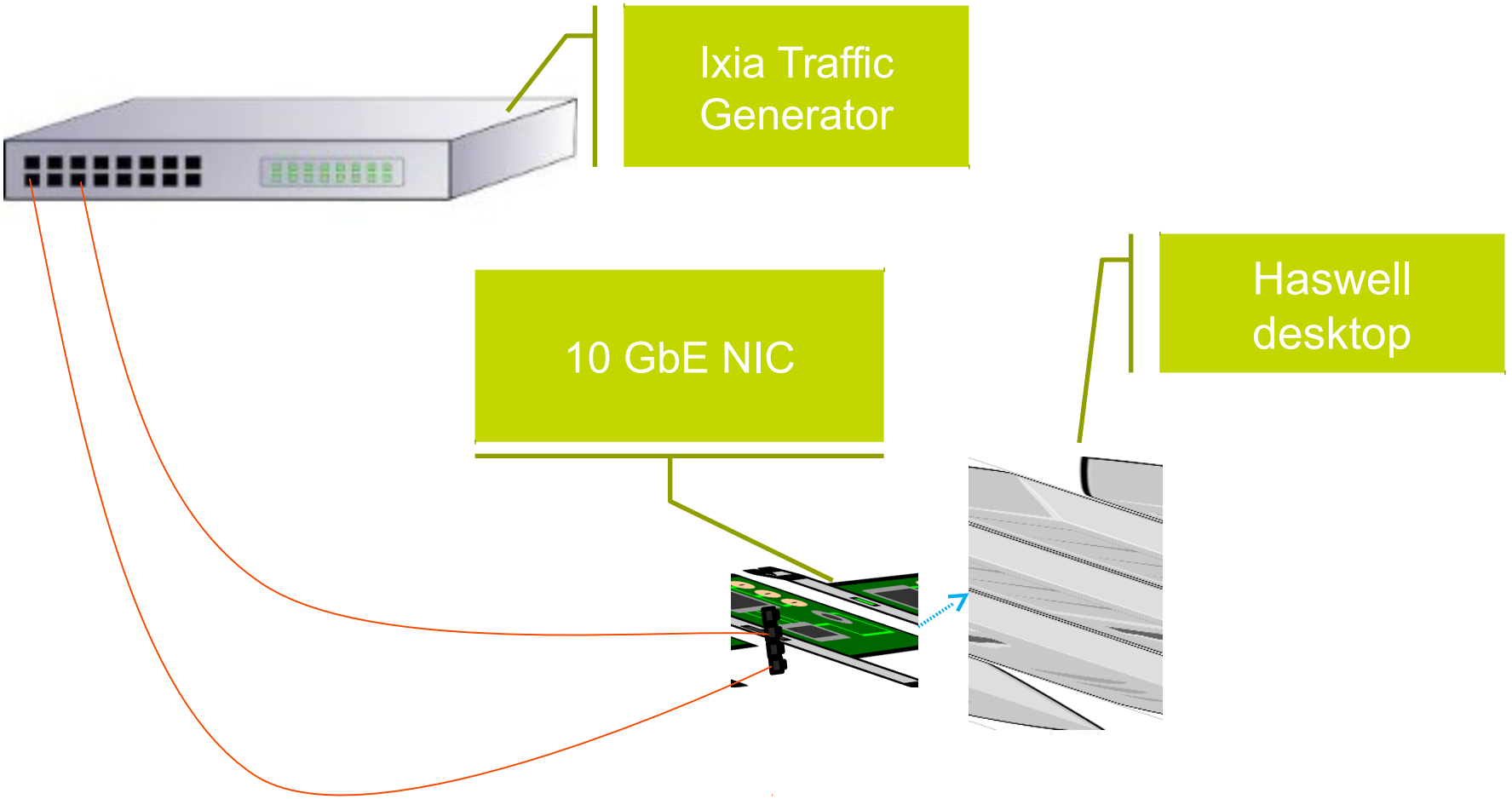

Benchmarking
Analysis
Modifications
3rd phase
4th phase
Benchmarking
Analysis
Code changes
Analysis
Code changes review
Submit patch to kernel community

Optimisation

Two methods:

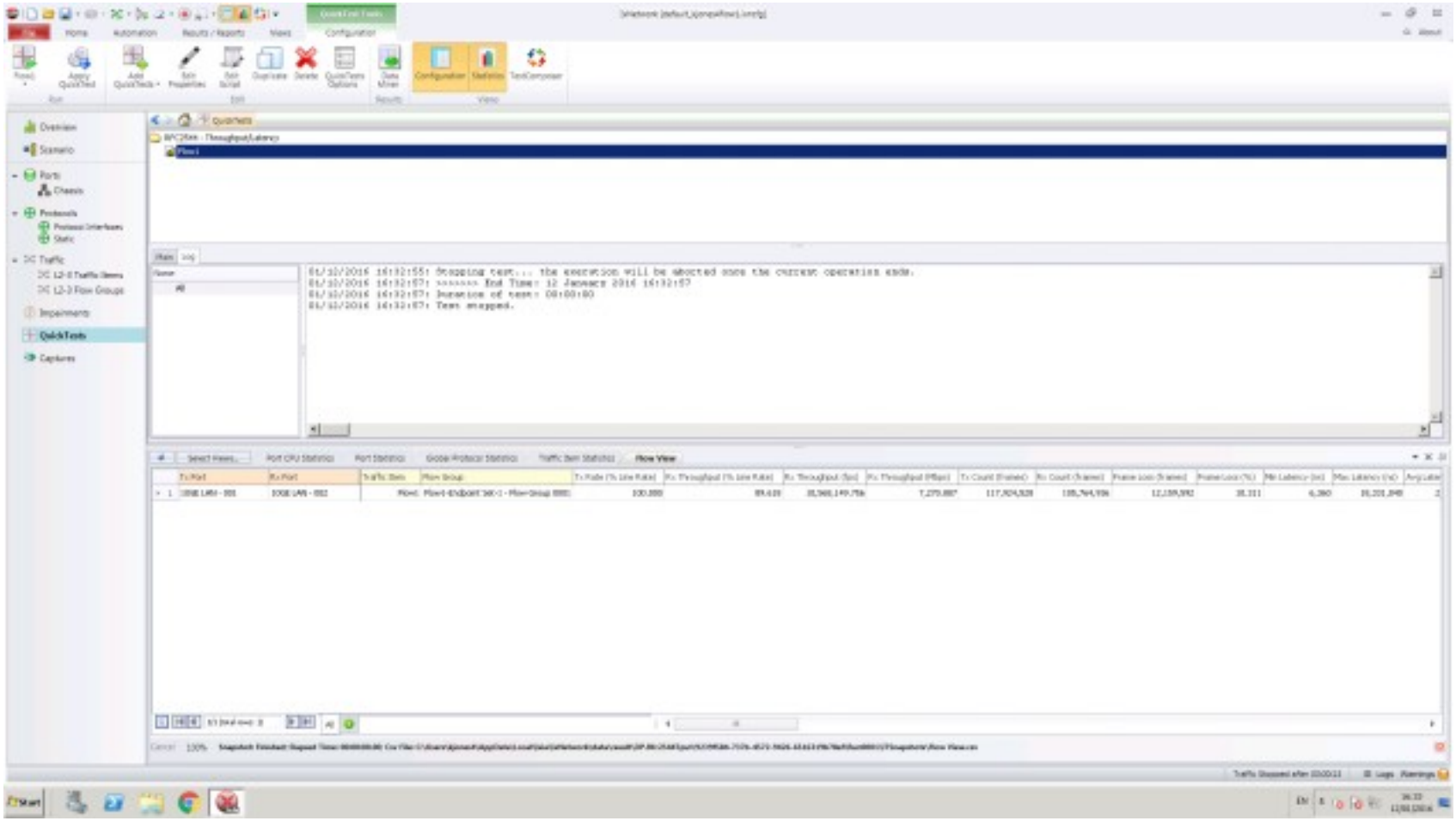
- Extensive research into network performance optimisation
- Iterative testing procedure

Setup



IxNetwork

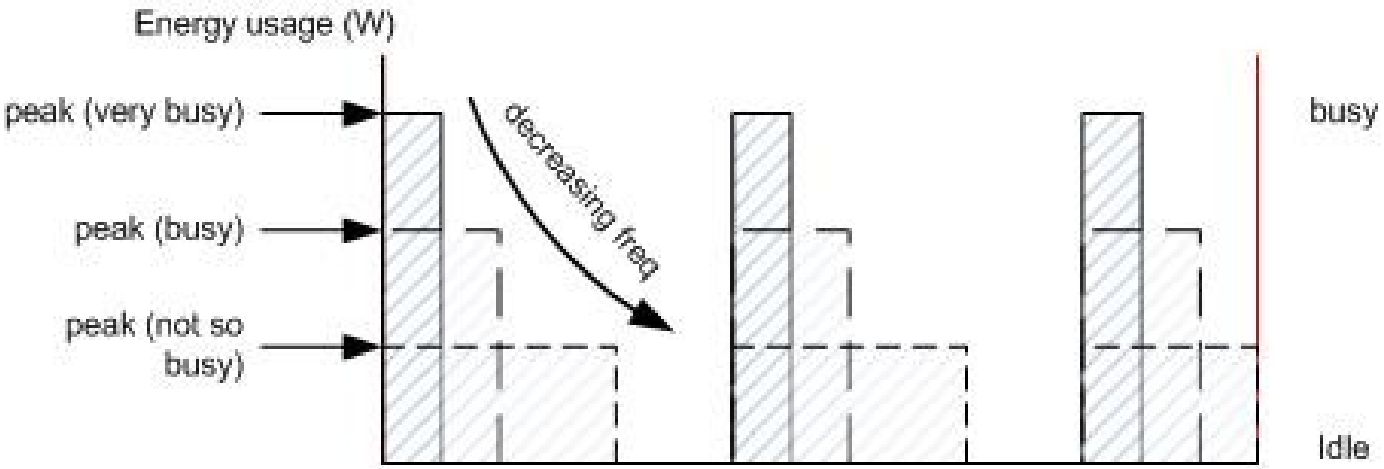
Flexible, customisable, wire-rate traffic generation



BIOS Options

Feature	Description
Hyper-Threading	Intel® Hyper-Threading Technology uses processor resources more efficiently, enabling multiple threads to run on each core. As a performance feature, it also increases processor throughput, improving overall performance on threaded software.
Turbo Boost	Intel® Turbo Boost Technology accelerates processor performance for peak loads, automatically allowing processor cores to run faster than the rated frequency if they're operating below power, current, and temperature specification limits.
C-States	C-states are idle states (except C0).

P-States



A P-state is an operational state; the core can be doing useful work in any P-state.
P0 has highest operating frequency and voltage.

Kernel Configuration

Feature	Orig. value	New Value	Justification
CONFIG_HUGETLBFS	Y	Y	Fewer TLB misses
CONFIG_HZ_1000	Y	Y	Higher timer interrupt resolution
CONFIG_INTEL_IOATDMA	Y	Y	DMA engine; allows the kernel to offload network data copying from the CPU to the DMA engine
CONFIG_DMA_ENGINE	Y	Y	Direct system memory access, see above
CONFIG_ASYNC_TX_DMA	Y	Y	async_tx API can utilise offload engines for memcpy, etc
CONFIG_DMADEVICES	Y	Y	Presents DMA Device drivers supported by the configured arch
CONFIG_DCA	Y	Y	Allows network driver to warm up CPU cache
CONFIG_PREEMPT_NONE	N (PREEMPT_RTB=y)	Y	Preemption geared towards throughput
CONFIG_HZ_PERIODIC	Y	Y	Timer tick running at all times

System-Level Configuration

/proc/sys/net/core/…

/proc/sys/net/ipv4/…

/proc/irq/<IRQ #>/smp_affinity…

ethtool

System Level Configuration

/proc/sys/…

- Allows reading and setting of system runtime information

/proc/irq/$IRQ/smp_affinity

Move management interface interrupt to be handled by CPU0

```
echo 01 > /proc/irq/$MGMT_IFACE_INTERRUPT/smp_affinity
```

Queues are affinitised to CPU1 - CPU7:

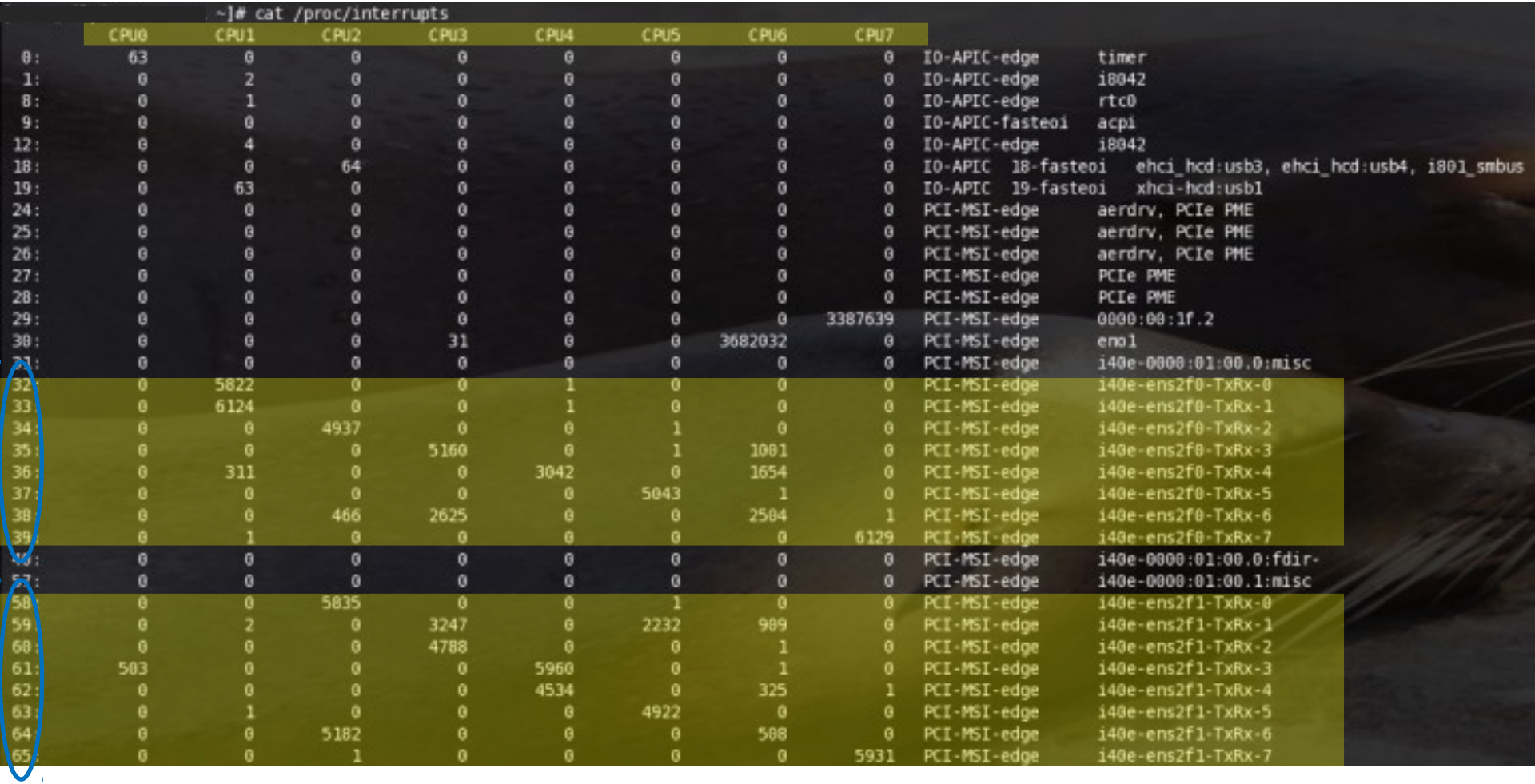
```
echo 02 > /proc/irq/$QUEUE-TXRX-0/smp_affinity
echo 04 > /proc/irq/$QUEUE-TXRX-1/smp_affinity
…
echo 80 > /proc/irq/$QUEUE-TXRX-6/smp_affinity
echo 02 > /proc/irq/$QUEUE-TXRX-7/smp_affinity
```

Forwarding Scenario: 8 flows

Endpoint Scenario: 7 flows

Disable irqbalance

/proc/interrupts



```
~]# cat /proc/interrupts
           CPU0       CPU1       CPU2       CPU3       CPU4       CPU5       CPU6       CPU7
  0:         63          0          0          0          0          0          0          0   IO-APIC-edge      timer
  1:          0          2          0          0          0          0          0          0   IO-APIC-edge      i8042
  8:          0          1          0          0          0          0          0          0   IO-APIC-edge      rtc0
  9:          0          0          0          0          0          0          0          0   IO-APIC-fasteoi   acpi
 12:          0          4          0          0          0          0          0          0   IO-APIC-edge      i8042
 18:          0          0         64          0          0          0          0          0   IO-APIC  18-fasteoi   ehci_hcd:usb3, ehci_hcd:usb4, i801_smbus
 19:          0         63          0          0          0          0          0          0   IO-APIC  19-fasteoi   xhci-hcd:usb1
 24:          0          0          0          0          0          0          0          0   PCI-MSI-edge      aerdrv, PCIe PME
 25:          0          0          0          0          0          0          0          0   PCI-MSI-edge      aerdrv, PCIe PME
 26:          0          0          0          0          0          0          0          0   PCI-MSI-edge      aerdrv, PCIe PME
 27:          0          0          0          0          0          0          0          0   PCI-MSI-edge      PCIe PME
 28:          0          0          0          0          0          0          0          0   PCI-MSI-edge      PCIe PME
 29:          0          0          0          0          0          0          0    3387639   PCI-MSI-edge      0000:00:1f.2
 30:          0          0          0         31          0          0    3682032          0   PCI-MSI-edge      eno1
 31:          0          0          0          0          0          0          0          0   PCI-MSI-edge      i40e-0000:01:00.0:misc
 32:          0       5822          0          0          1          0          0          0   PCI-MSI-edge      i40e-ens2f0-TxRx-0
 33:          0       6124          0          0          1          0          0          0   PCI-MSI-edge      i40e-ens2f0-TxRx-1
 34:          0          0       4937          0          0          1          0          0   PCI-MSI-edge      i40e-ens2f0-TxRx-2
 35:          0          0          0       5160          0          1       1001          0   PCI-MSI-edge      i40e-ens2f0-TxRx-3
 36:          0        311          0          0       3042          0       1654          0   PCI-MSI-edge      i40e-ens2f0-TxRx-4
 37:          0          0          0          0          0       5043          1          0   PCI-MSI-edge      i40e-ens2f0-TxRx-5
 38:          0          0        466       2625          0          0       2504          1   PCI-MSI-edge      i40e-ens2f0-TxRx-6
 39:          0          1          0          0          0          0          0       6129   PCI-MSI-edge      i40e-ens2f0-TxRx-7
 56:          0          0          0          0          0          0          0          0   PCI-MSI-edge      i40e-0000:01:00.0:fdir-
 57:          0          0          0          0          0          0          0          0   PCI-MSI-edge      i40e-0000:01:00.1:misc
 58:          0          0       5835          0          0          1          0          0   PCI-MSI-edge      i40e-ens2f1-TxRx-0
 59:          0          2          0       3247          0       2232        909          0   PCI-MSI-edge      i40e-ens2f1-TxRx-1
 60:          0          0          0       4788          0          0          1          0   PCI-MSI-edge      i40e-ens2f1-TxRx-2
 61:        503          0          0          0       5960          0          1          0   PCI-MSI-edge      i40e-ens2f1-TxRx-3
 62:          0          0          0          0       4534          0        325          1   PCI-MSI-edge      i40e-ens2f1-TxRx-4
 63:          0          1          0          0          0       4922          0          0   PCI-MSI-edge      i40e-ens2f1-TxRx-5
 64:          0          0       5182          0          0          0        508          0   PCI-MSI-edge      i40e-ens2f1-TxRx-6
 65:          0          0          1          0          0          0          0       5931   PCI-MSI-edge      i40e-ens2f1-TxRx-7
```

/proc/sys/fs/…

Setting	Orig. Value(s)	New Value(s)	Justification
file-max	6563009	65535	maximum number of file-handles that the Linux kernel will allocate.

/proc/sys/net/core/…

Setting	Orig. Value(s)	New Value(s)	Justification
netdev_max_backlog	1000	300000	Maximum number of packets, queued on INPUT side, when interface receives packets faster than kernel can process them.
somaxconn	128	1024	Limit of socket listen() backlog. Should be raised substantially to support bursts of request.
rmem_max	212992	134217728	Maximum receive socket buffer size (UDP).
wmem_max	212992	134217728	Maximum send socket buffer size (UDP).
rmem_default	212992	134217728	Default setting of the socket receive buffer (UDP).
wmem_default	212992	134217728	Default setting of the socket send buffer (UDP).

/proc/sys/net/core/…

Setting	Orig. Value(s)	New Value(s)	Justification
busy_read	0	0	Low latency busy poll timeout for socket reads
busy_poll	0	0	Low latency busy poll timeout for poll and select
dev_weight	64	4096	Maximum number of packets that kernel can handle on a NAPI interrupt. Per-CPU variable.
netdev_budget	300	4096	Maximum number of packets taken from all interfaces in one polling cycle (NAPI poll).
optmem_max	20480	134217728	Maximum ancillary buffer size allowed per socket.

/proc/sys/net/ipv4/…

Setting	Orig. Value(s)	New Value(s)	Description
ip_local_port_range	32768 60999	1024 65000	Min/Max local port range
tcp_max_syn_backlog	2048	300000	Maximal number of remembered connection requests, which have not received an acknowledgment from connecting client.
tcp_rmem	4096 87380 6291456	4096 87380 67108864	Min/Default/Max size of receive buffer used by TCP sockets.
tcp_wmem	4096 16384 4194304	4096 87380 67108864	Min/Default/Max amount of memory reserved for send buffers for TCP sockets.
udp_mem	1539354 2052472 3078708	4096 87380 67108864	Min/Default/Max pages allowed for queueing by all UDP sockets.

/proc/sys/net/ipv4/…

Setting	Orig. Value(s)	New Value(s)	Description
tcp_sack	1	0	Disable select acknowledgments
tcp_timestamps	1	0	Disable timestamps
tcp_mtu_probing	0	1	TCP Packetization-Layer Path MTU Discovery (only enabled when ICMP black hole detected)
tcp_no_metrics_save	0	0	TCP will cache metrics on closing connections
tcp_fin_timeout	60	30	Length of time an orphaned connection will remain in FIN_WAIT_2 state before it's aborted at local end.
tcp_keepalive_time	7200	60000	How often TCP sends out keepalive messages.

/proc/sys/net/ipv4/…

Setting	Orig. Value(s)	New Value(s)	Description
tcp_keepalive_intvl	75	15000	How frequently the probes are send out.
tcp_window_scaling	1	1	Enable window scaling
tcp_syncookies	1	1	Send out syncookies when syn backlog queue of socket overflows. Prevents 'SYN flood attack'
ip_forward	0	1	Forward Packets between interfaces.
tcp_congestion_control	cubic	tcp_htcp*	optimized congestion control algorithm for high speed networks with high latency

* Tested congestion control algorithms; as of yet we have not discovered which is the most performant. This needs further investigation

ethtool System Configuration

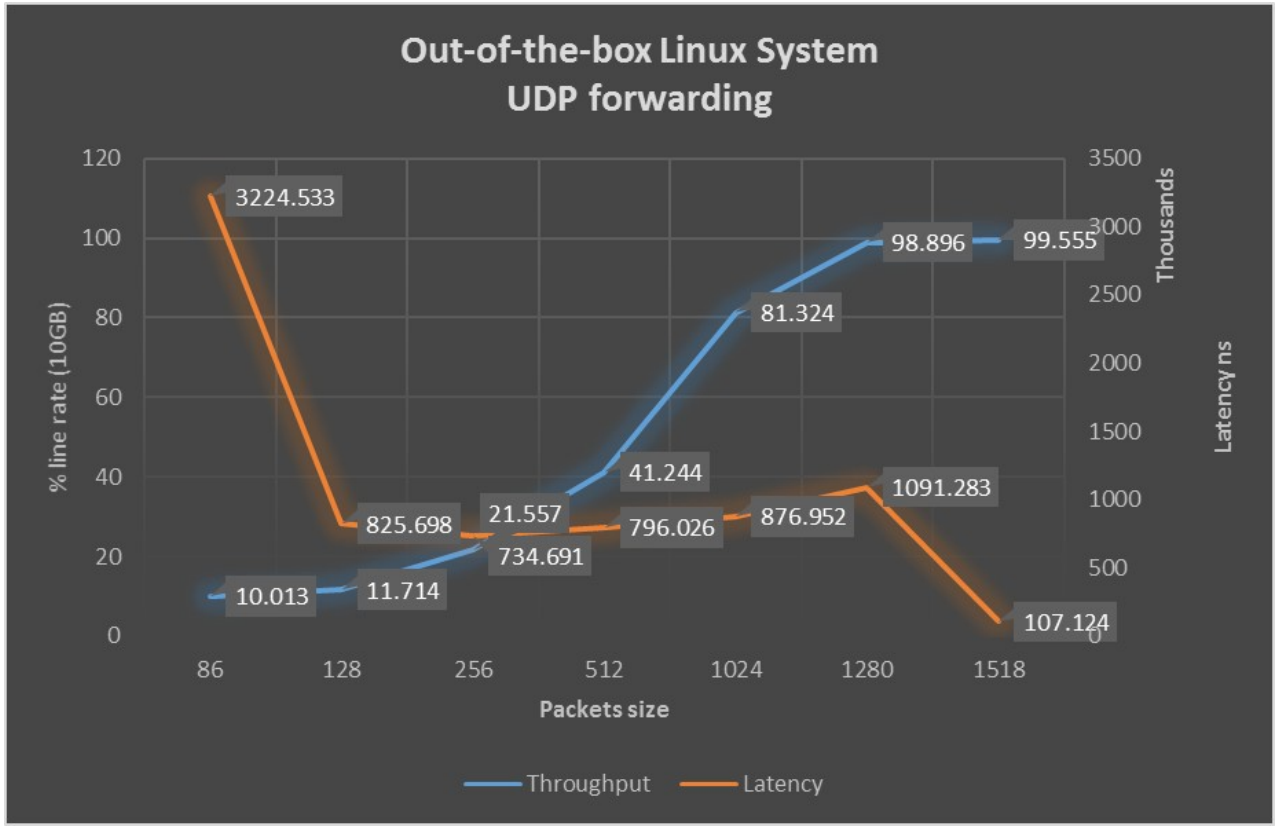
Feature	Orig. Value(s)	New Value(s)	Description
txqueuelen	1000	10000	Modify transmit queue length
-C adaptive-rx -C adaptive-tx	on on	off off	Dynamic control to decrease latency at low packet rates and increase throughput at high packet rates.
-C rx-usecs -C tx-usecs	25 25	25 75	Number of microsecs to wait before raising an interrupt after a packet has been sent.

ethtool System Configuration

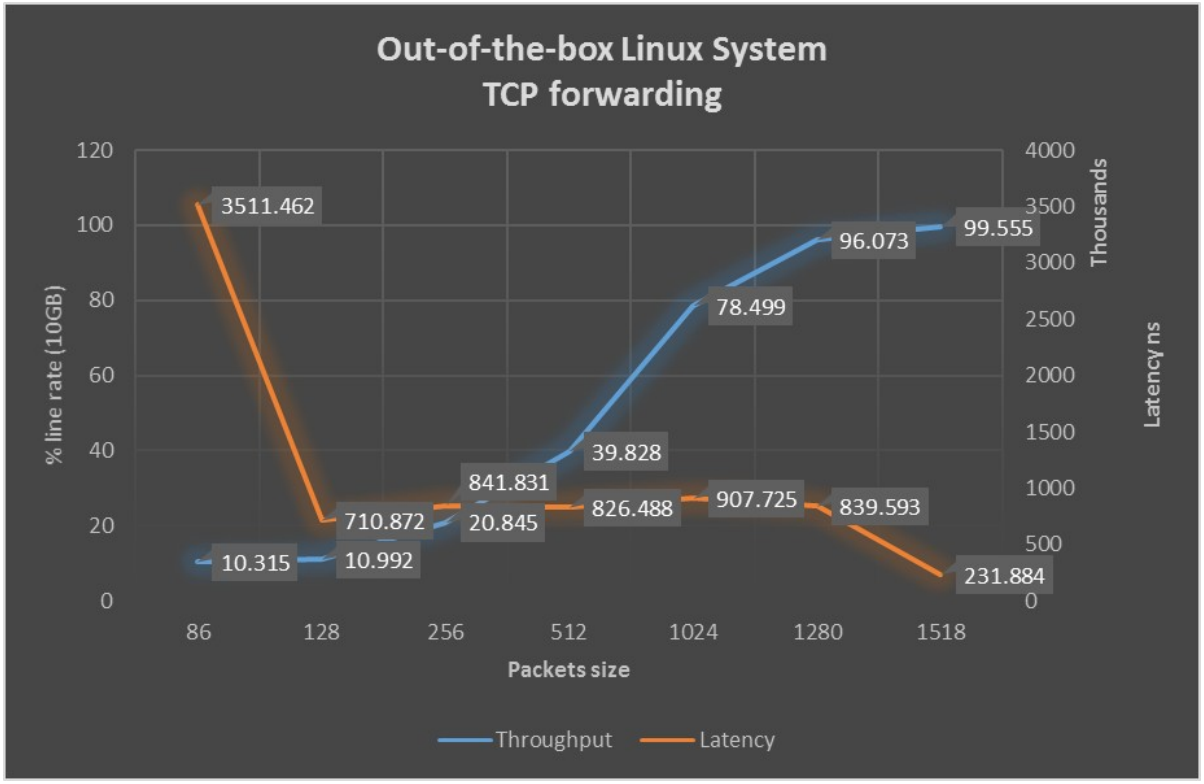
Feature	Orig. Value(s)	New Value(s)	Description
-G rx -G tx	4096 4096	4096 4096	Changes number of ring entries
-K [offloading]	on	on	Enable Rx hashing offload, TSO, GRO, GSO, scatter-gather, Checksumming
-K ntuple	on	off	Disable Rx ntuple filters and actions

UDP Forwarding – Out Of the Box

- No BIOS tuning
- Kernel 3.17.4-301.fc21.x86_64
- Fedora 21 default system settings



TCP Forwarding – Out Of the Box



- No BIOS tuning
- Kernel 3.17.4-301.fc21.x86_64
- Fedora 21 default system settings

perf record / report: Cycle-based Analysis

```
perf record -a -F 1000 sleep 60
```

```
perf report -M intel
```

```
Samples: 1M of event 'cycles', Event count (approx.): 2692315417468
Overhead  Command          Shared Object     Symbol
  11.94%  swapper          [kernel.vmlinux]  [k] _raw_spin_lock
   4.69%  swapper          [i40e]            [k] i40e_napi_poll
   4.02%  swapper          [kernel.vmlinux]  [k] __netif_receive_skb_core
   3.28%  swapper          [kernel.vmlinux]  [k] fib_table_lookup
   2.61%  swapper          [kernel.vmlinux]  [k] __slab_free
   2.12%  swapper          [kernel.vmlinux]  [k] consume_skb
   2.08%  swapper          [kernel.vmlinux]  [k] skb_release_data
   2.08%  swapper          [i40e]            [k] i40e_lan_xmit_frame
   1.90%  swapper          [kernel.vmlinux]  [k] kmem_cache_alloc
   1.66%  swapper          [i40e]            [k] i40e_alloc_rx_buffers_1buf
```

perf report: annotation