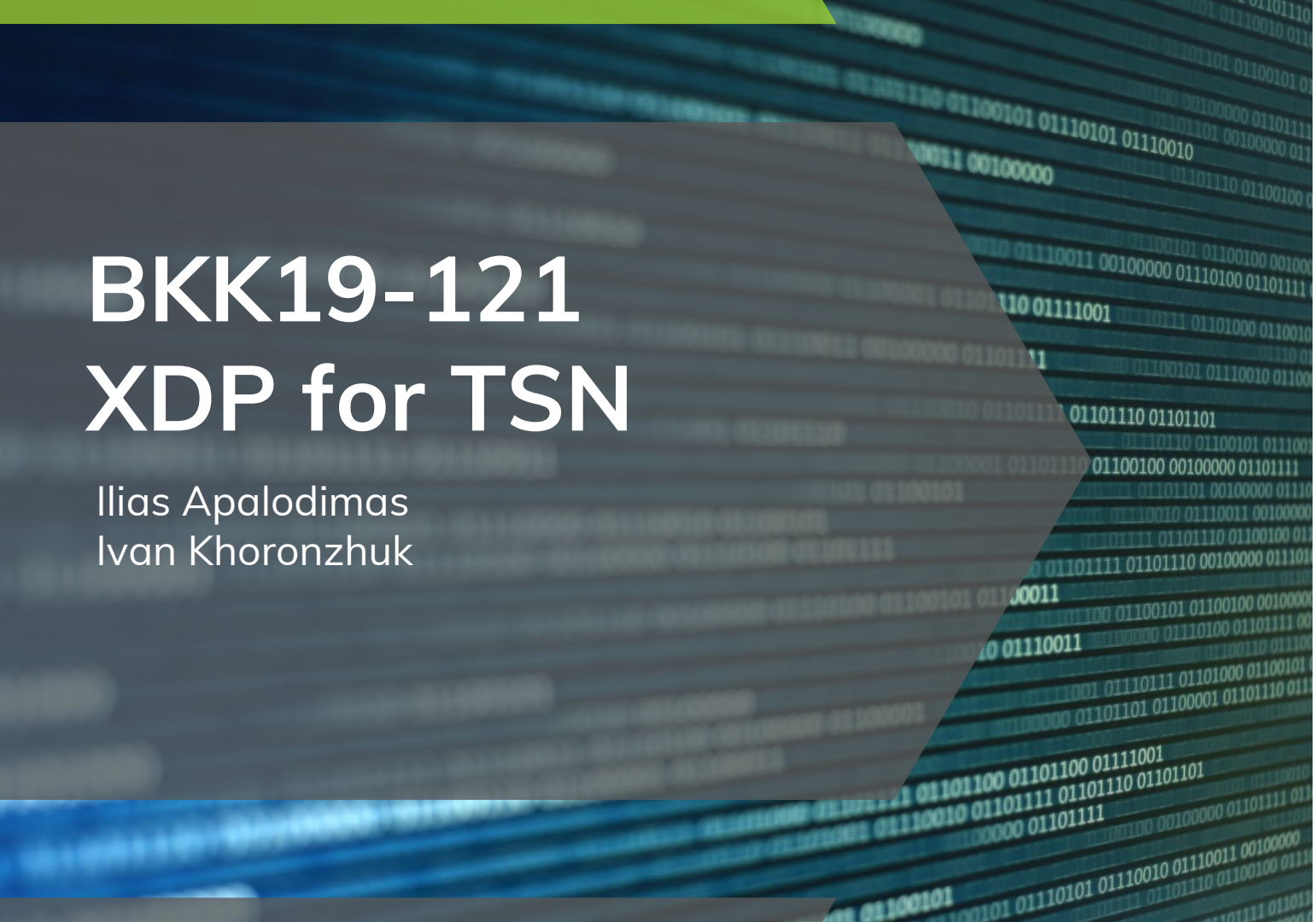
BKK19-121
XDP for TSN

Ilias Apalodimas
Ivan Khoronzhuk



Linaro

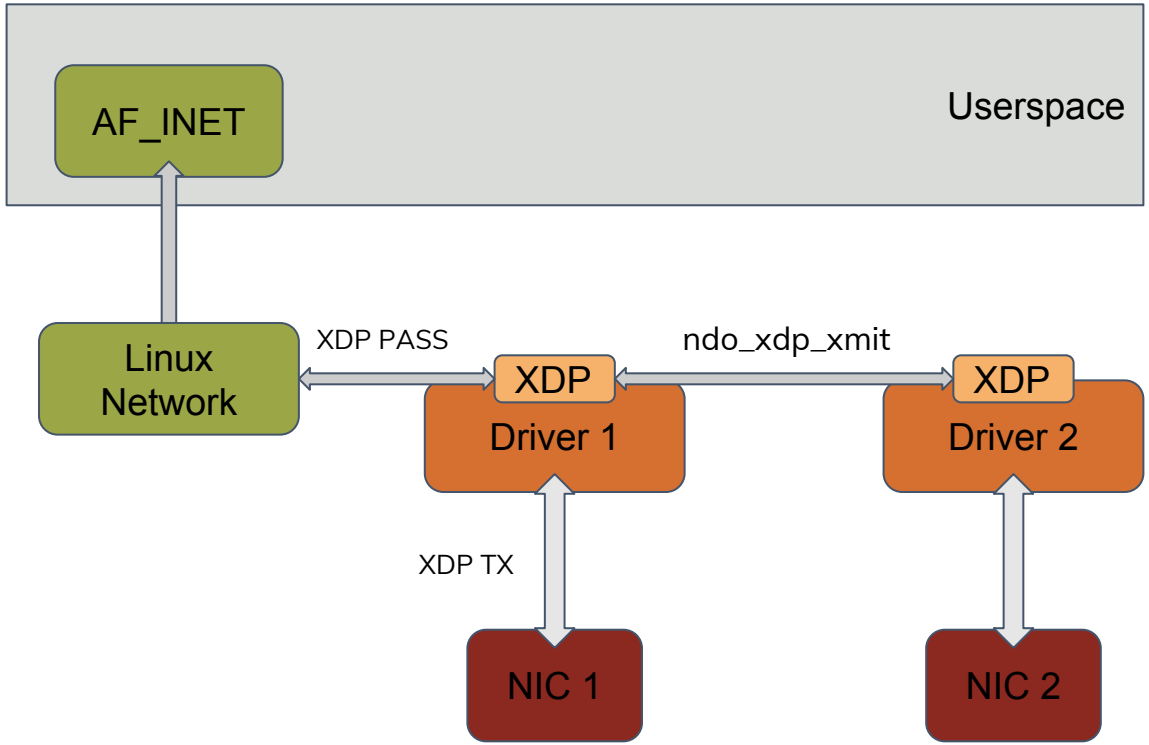
What is XDP

- XDP (eXpress Data Path) is a Linux kernel fast-path
- Operates at L2-L3
- Not a bypass, but in-kernel fast-path
- Operates on driver basis. Support must be added on each driver
- Native support for a limited set of drivers, but generic-XDP is available for all network drivers
- Think of XDP as building block for a solution. Not a solution by itself

XDP Actions

- XDP_PASS
  - Send packet to linux network stack for further processing
- XDP_DROP
  - Drop packets
- XDP_REDIRECT
  - Redirect packets
- XDP_TX
  - Transmit the packet out of the interface it was received

XDP Overview

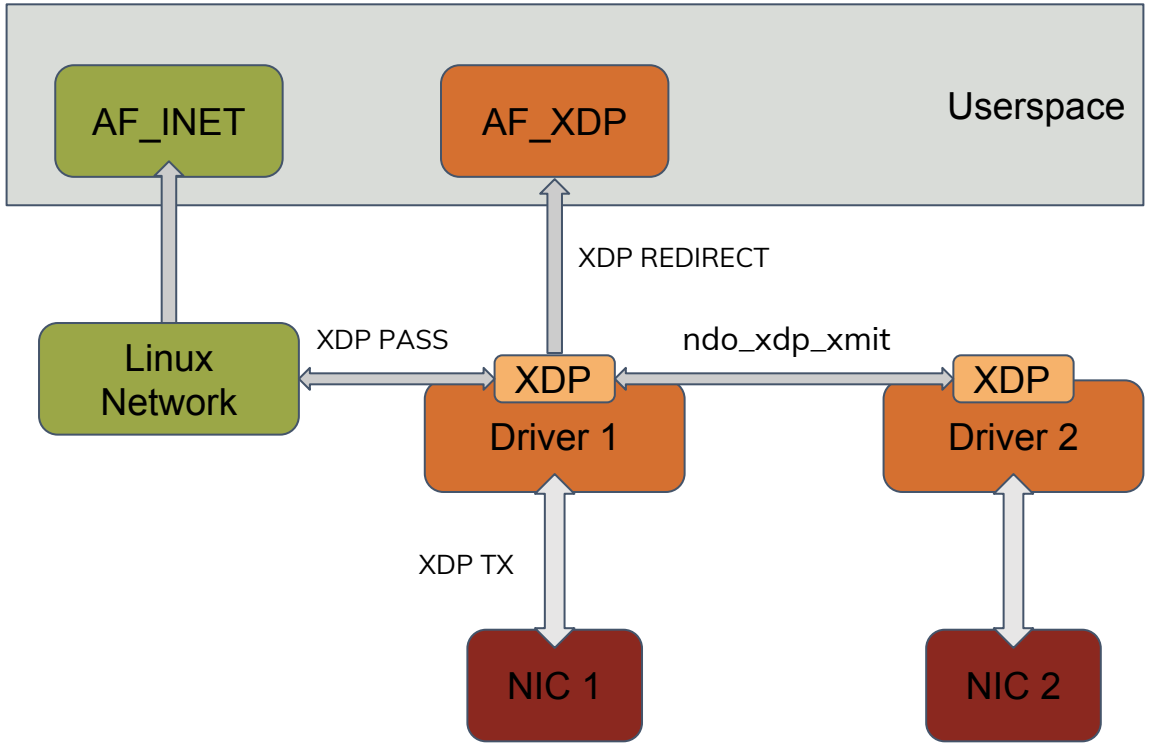


Linaro

What is AF_XDP

- Introduced in v4.18
- New ‘sinkhole’ for XDP_REDIRECT
- Offloads traffic to userspace
- Generic XDP
  - No code needed
  - Works for all drivers
  - Small performance gains
- Native XDP
  - Good performance gains on Rx
  - Tx still goes through SKBs
- Zero-copy
  - Significant performance gains
  - Offloaded Tx path

What is AF_XDP



Linaro

Why XDP for TSN

- TSN requires bounded latency and low bounded jitter
- XDP can’t guarantee jitter. It can offer significantly lower latency compared to the default network stack
- Due to it’s design can work well with ‘mixed’ packet scenarios. TSN packets can be offloaded to user-space while non-critical traffic is handled by the kernel
- AF_XDP is designed to operate as a socket
- AF_XDP L2 packets can work really well with existing user-space L2 solutions (VPP, LWIP etc)

Work in progress

- Features are still getting merged. The whole project is new
- Poll mode will definitely be needed for sub 1µs application
  - Interrupts and mmio access to (uncached) memory hurt
- Zero copy will be needed for drivers to take full advantage of it. At the moment adding zero-copy support on drivers is hard (Intel i40e/ixgbe only)

Latency measurement

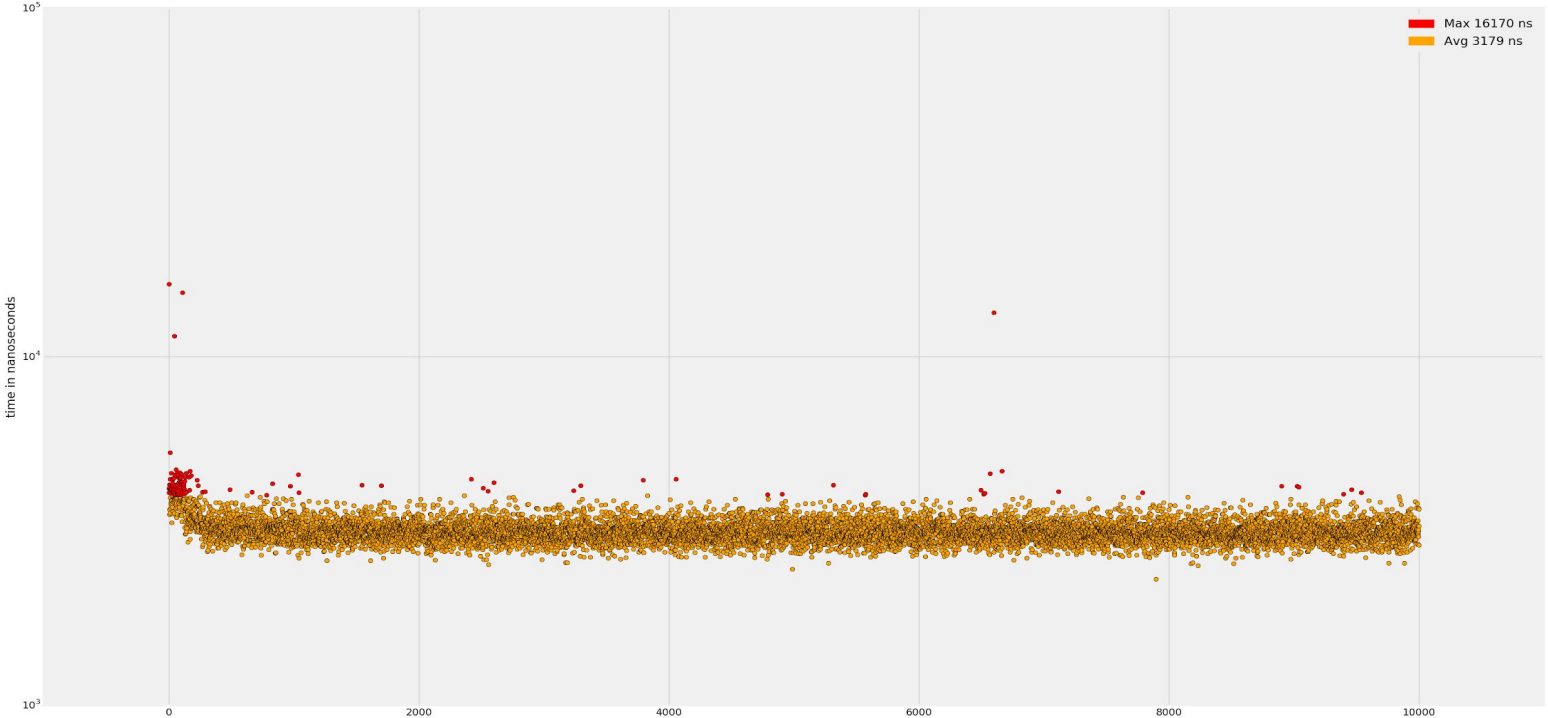
- XDP has no timestamp infrastructure similar to SKB's
- An easy way of measuring latency from the NAPI Rx poller down to user-space is piggyback on XDP data_meta of the xdp_buff struct (32 bytes)
- Store the timestamp and compare it to the timestamp on packet arrival in user-space
- If the hardware supports timestamping insert the hardware one and sync user-space with PHC clock
- If not insert a software timestamp
  - It will only measure stack latency [1]
  - No IRQ (hardware or NAPI softirq) included

[1] By 'stack latency' we refer to latency needed from entering napi_poll down to packet reaching users-space

Measurements on cortex-A53

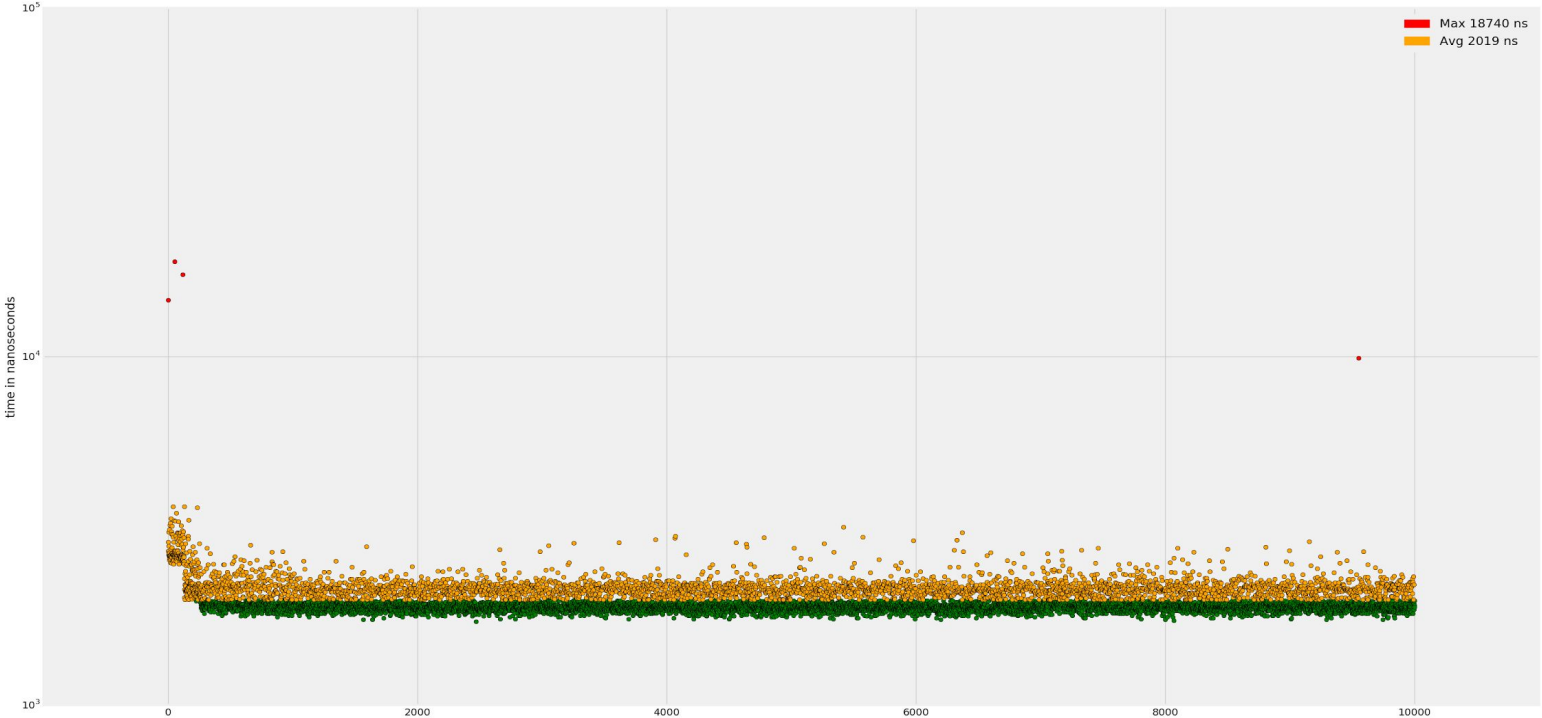
- The application run on CPU 20. CPU 21 is serving the NIC Rx-IRQ. Both are isolated with ‘isolcpus=’.
  - Stack-only since we have no access to hardware timestamps
- 10k packets 1kpps
  - Avg: 3179 ns
  - Peak: 16170 ns
- 10k packets 1kpps forced ‘polling’ (no mmio reads)
  - Avg: 2019 ns
  - Peak: 18740 ns
- 30k packets 5kpps
  - Avg: 3104 ns
  - Peak: 28940 ns
- 30k packets 5kpps forced ‘polling’ (no mmio reads)
  - Avg: 1921 ns
  - Peak: 18390 ns

10k packets 1kpps



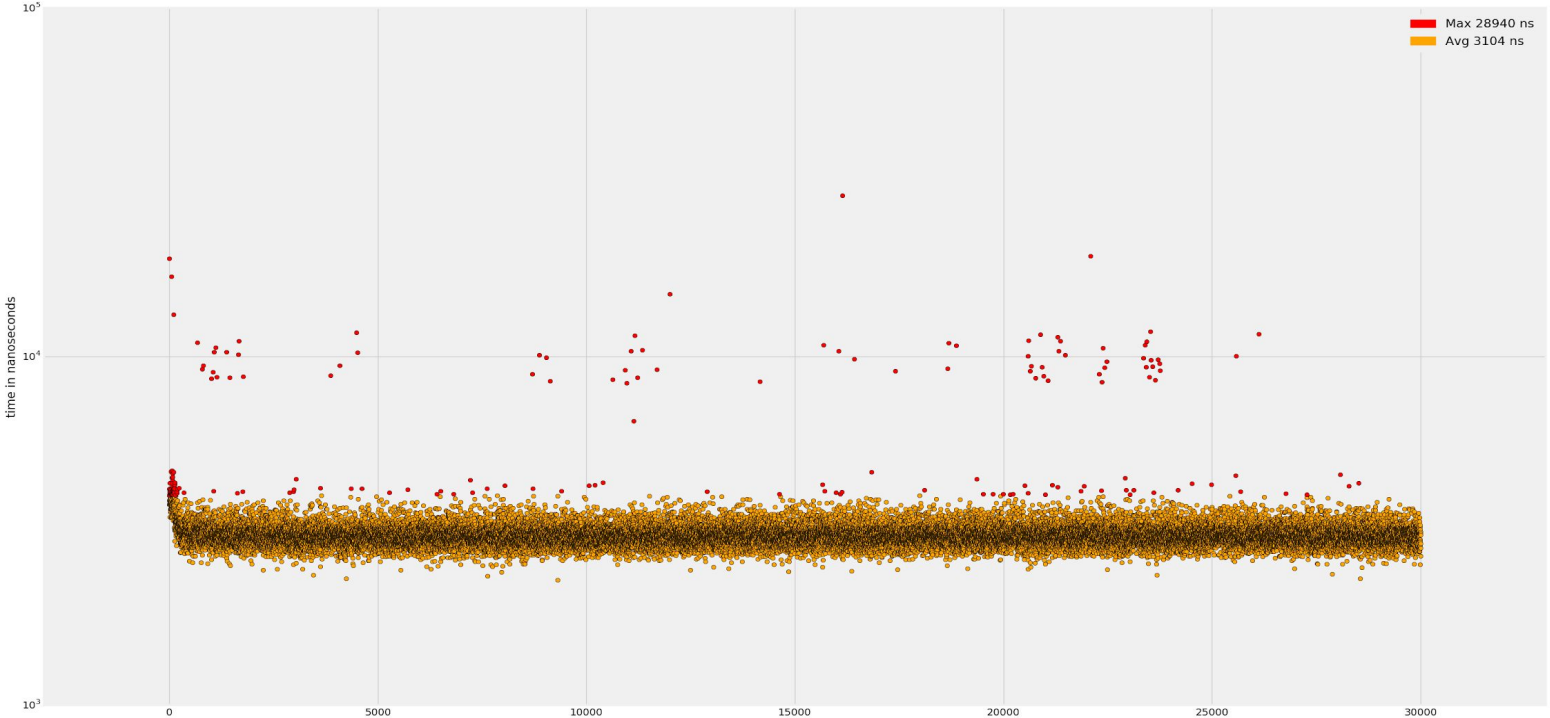
Linaro

10k packets 1kpps mmio access disabled



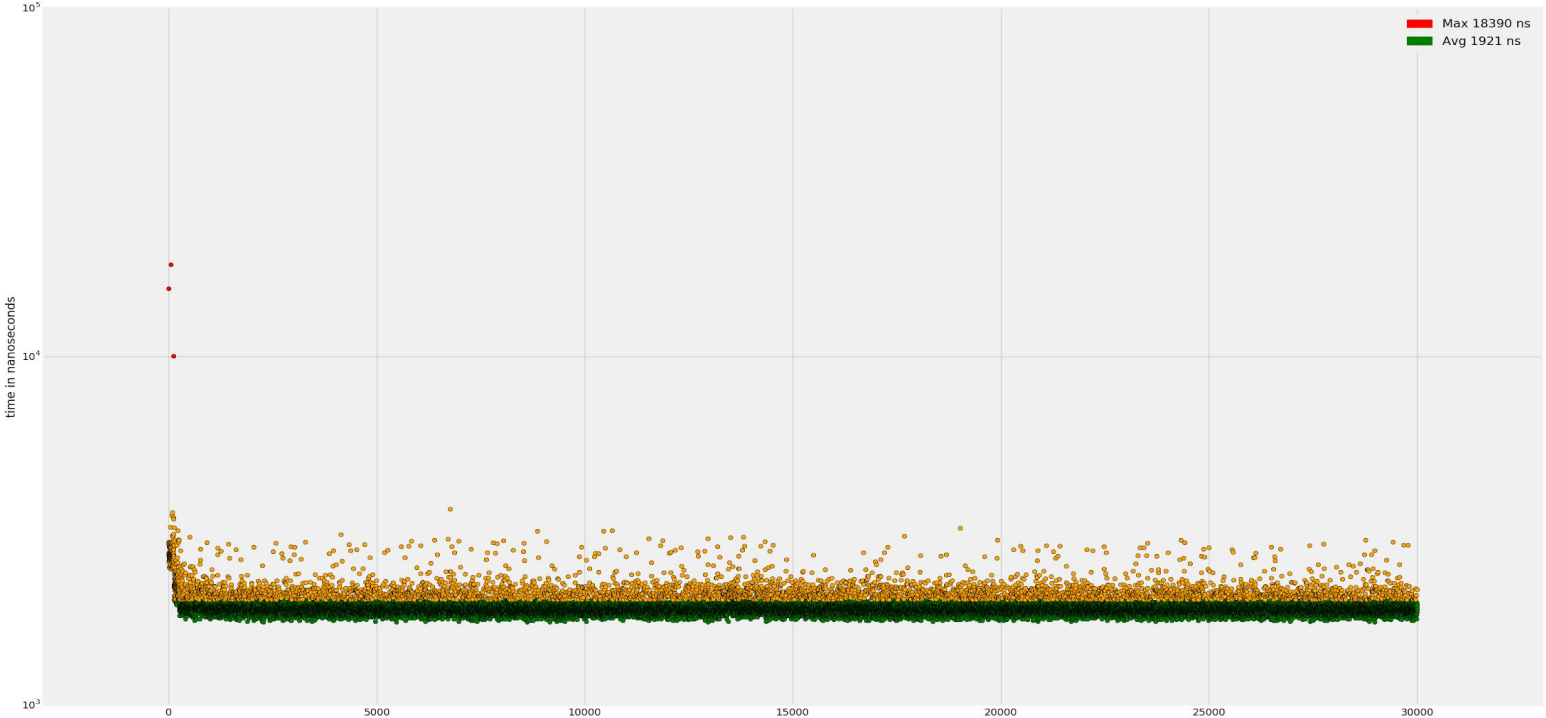
Linaro

30k packets 5kpps



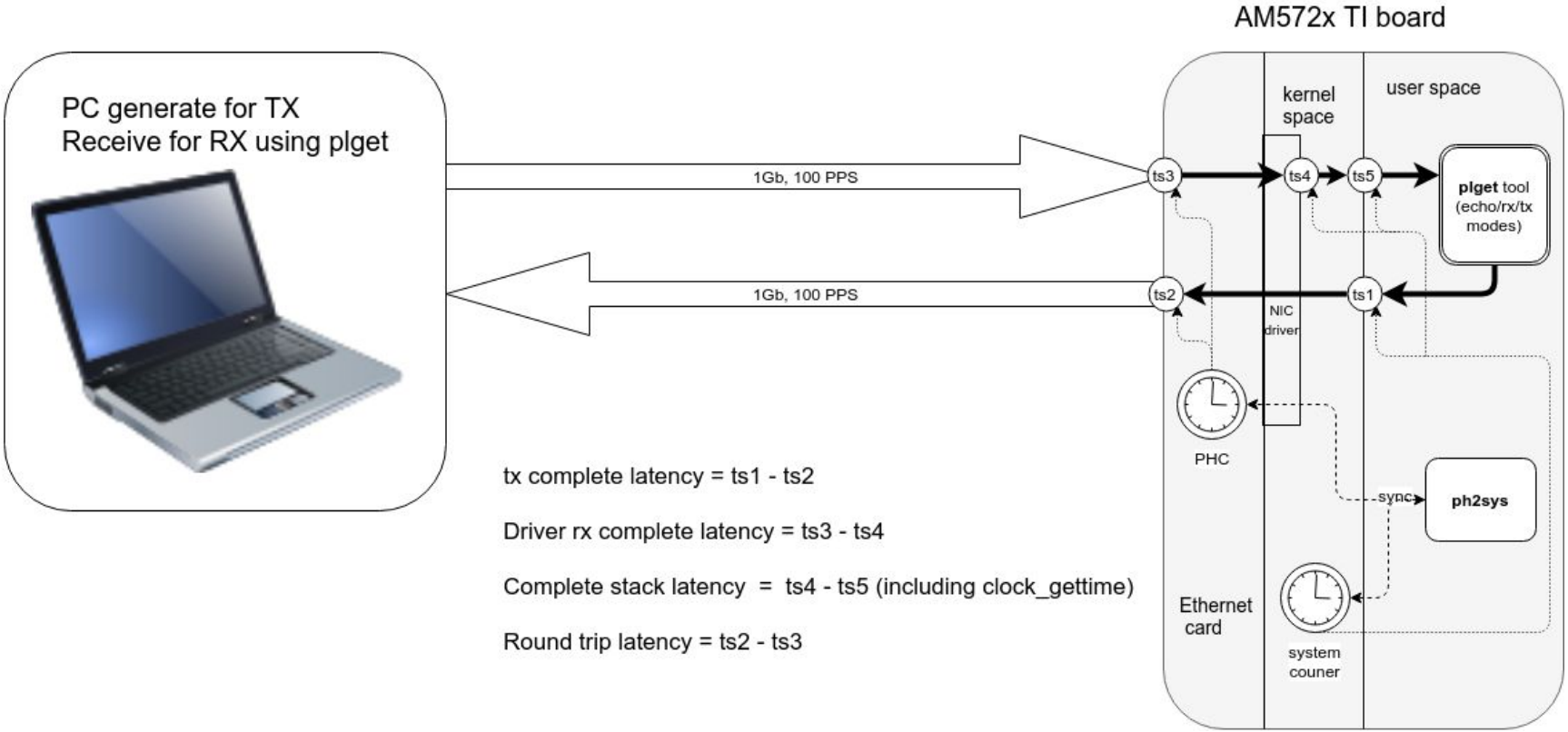
Linaro

30k packets 5kpps mmio access disabled



Linaro

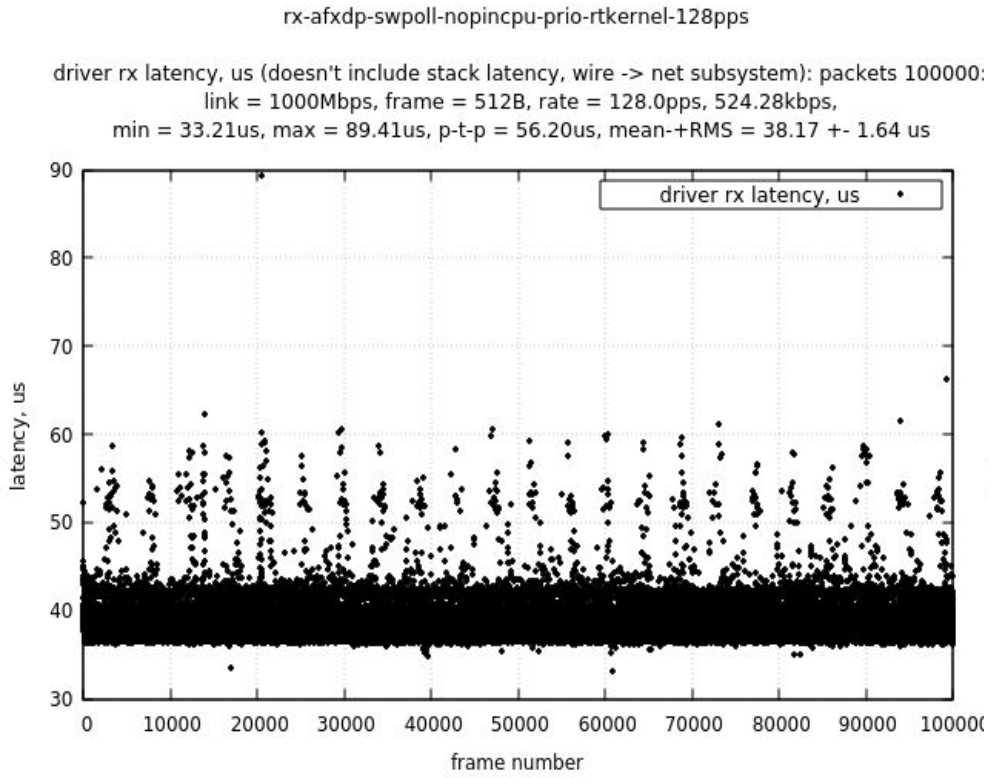
Test model

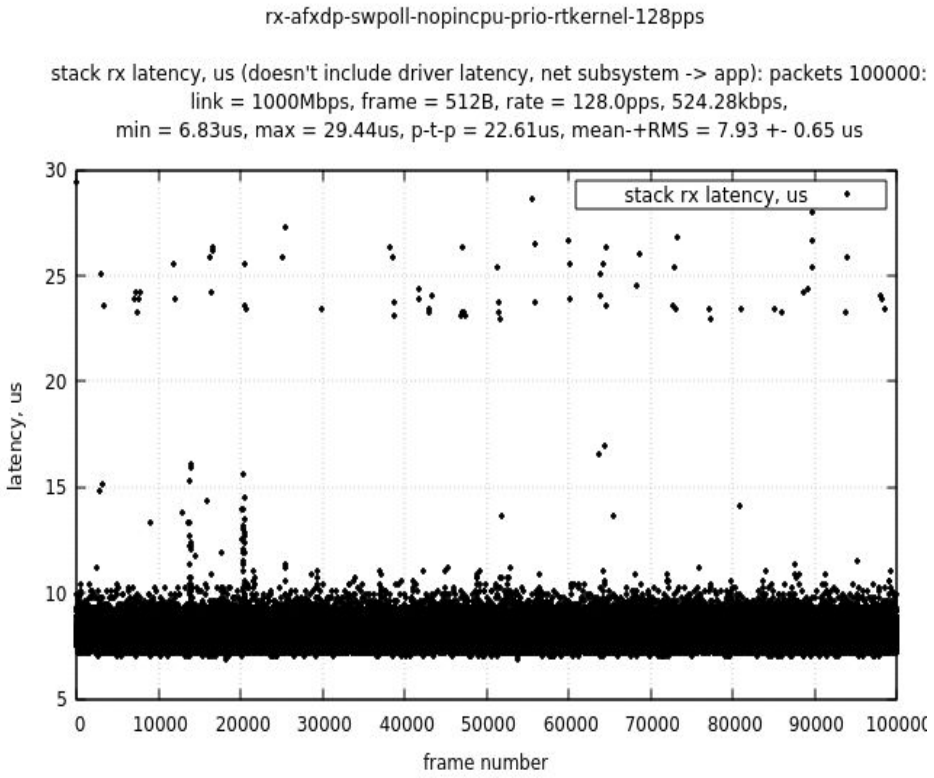

PC generate for TX
Receive for RX using plget
1Gb, 100 PPS
1Gb, 100 PPS
AM572x TI board
kernel space
user space
ts3
ts4
ts5
plget tool (echo/rx/tx modes)
ts2
ts1
NIC driver
PHC
sync
ph2sys
Ethernet card
system couner
tx complete latency = ts1 - ts2
Driver rx complete latency = ts3 - ts4
Complete stack latency = ts4 - ts5 (including clock_gettime)
Round trip latency = ts2 - ts3

Summary: RX, RT plget, **sw poll**

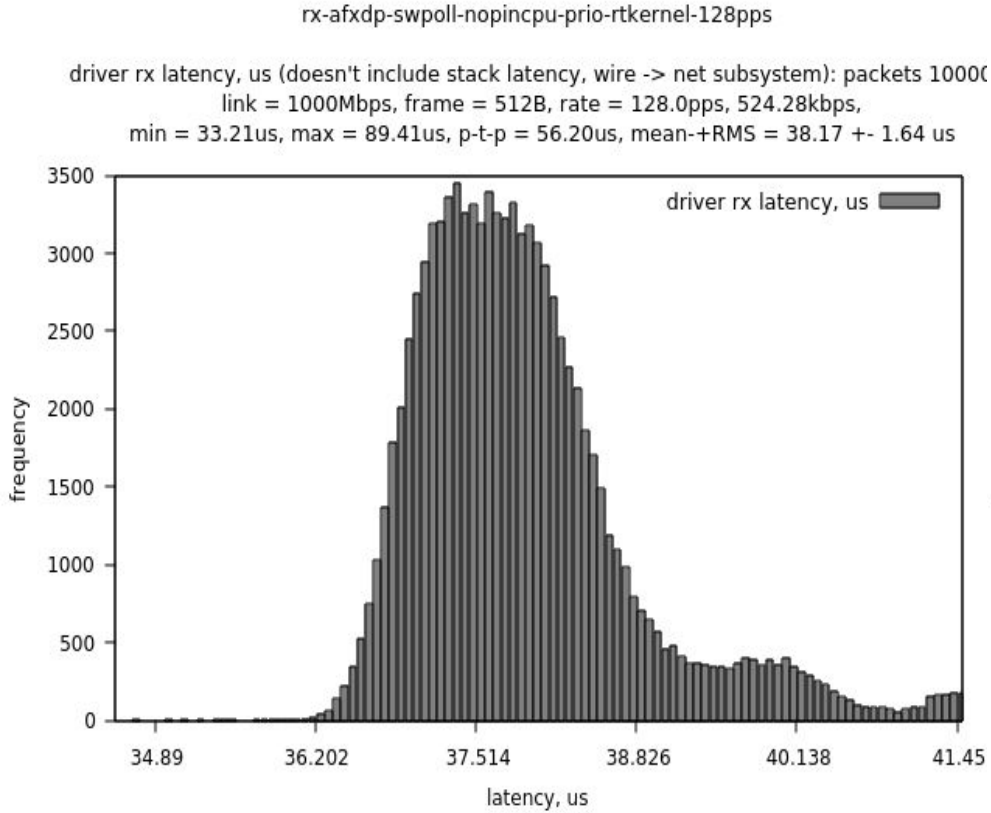
	Socket (AF_XDP, SOCK_RAW, 0) 128PPS	Socket (AF_XDP, SOCK_RAW, 0) 1PPS	Socket (AF_PACKET, SOCK_RAW, 0) 128PPS
Driver latency	max val(#20478) = 89.41us min val(#60848) = 33.21us peak-to-peak = 56.20us mean +- RMS = 38.17 +- 1.64 us	max val(#41061) = 65.70us min val(#83649) = 36.87us peak-to-peak = 28.84us mean +- RMS = 44.28 +- 2.17 us	max val(#24517) = 73.11us min val(#40897) = 37.38us peak-to-peak = 35.74us mean +- RMS = 42.98 +- 2.31 us
Stack latency	max val(#0) = 29.44us min val(#18296) = 6.83us peak-to-peak = 22.61us mean +- RMS = 7.93 +- 0.65 us	max val(#44251) = 28.79us min val(#24620) = 7.97us peak-to-peak = 20.82us mean +- RMS = 10.26 +- 0.92 us	max val(#74524) = 40.50us min val(#69757) = 9.11us peak-to-peak = 31.39us mean +- RMS = 11.33 +- 1.25 us
Complete latency	max val(#20478) = 98.84us min val(#60848) = 41.02us peak-to-peak = 57.83us mean +- RMS = 46.10 +- 2.02 us	max val(#41061) = 80.51us min val(#83649) = 46.79us peak-to-peak = 33.72us mean +- RMS = 54.53 +- 2.55 us	max val(#93955) = 94.27us min val(#40897) = 47.95us peak-to-peak = 46.32us mean +- RMS = 54.32 +- 2.73 us

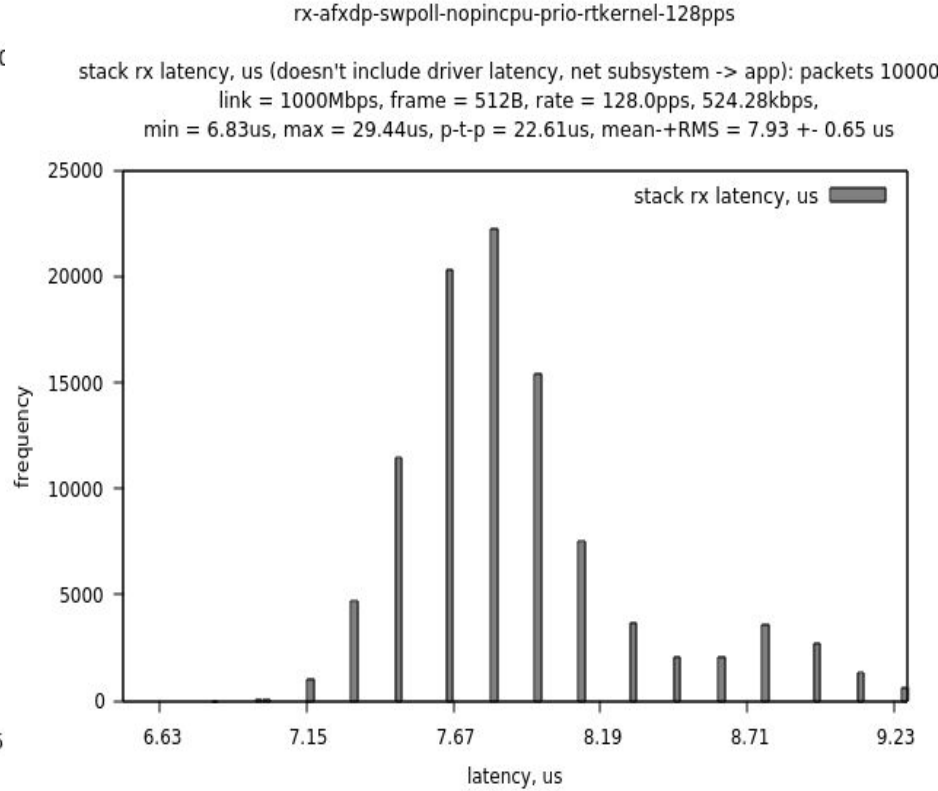
af_xdp RT driver/stack latency, sw poll, 128PPS



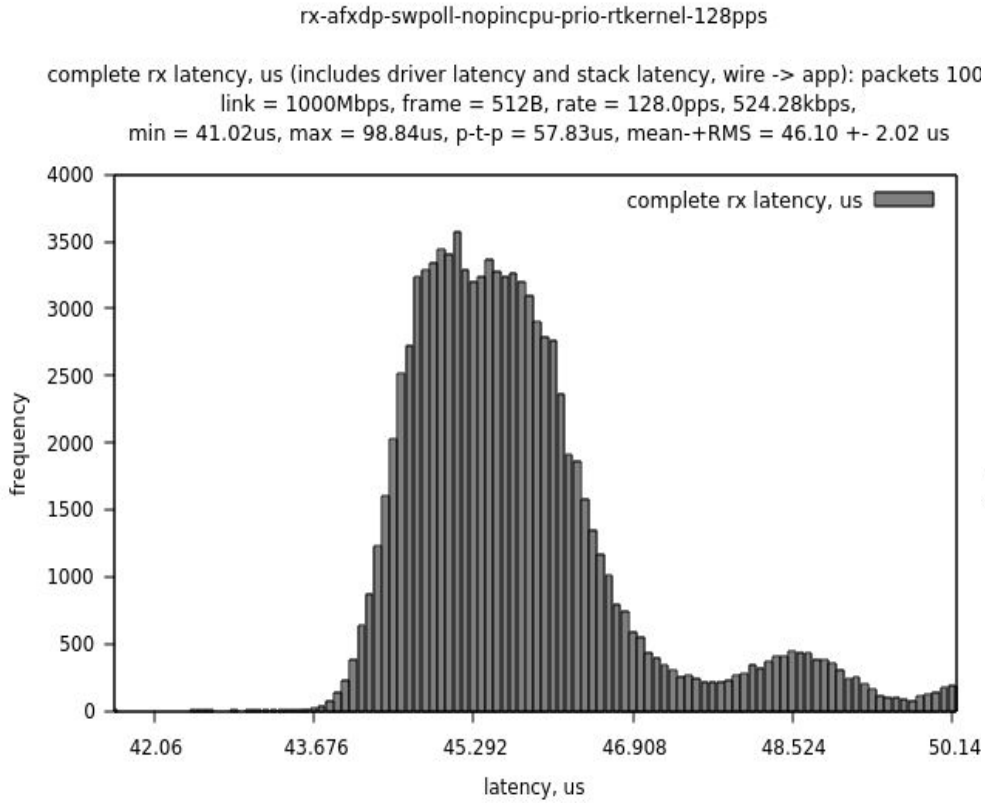


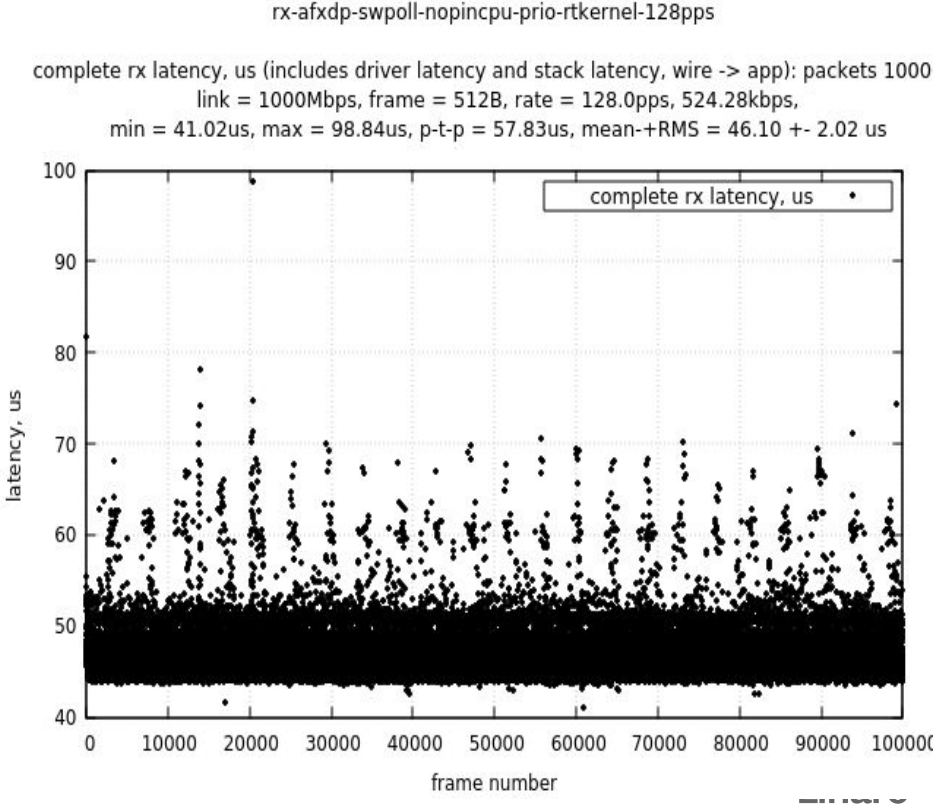
af_xdp RT driver/stack latency, sw poll, 128PPS





af_xdp RT complete latency, sw poll, 128PPS





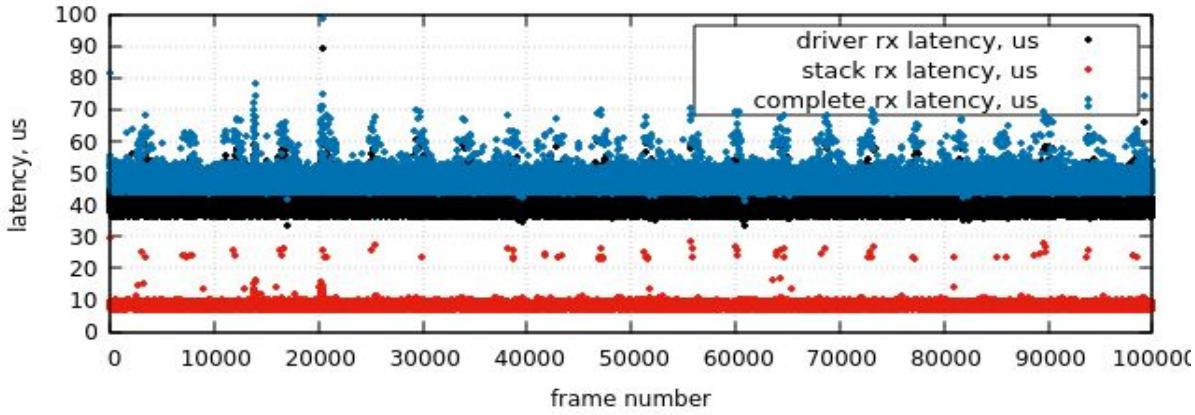
af_xdp RT driver/stack/complete, sw poll, 128PPS

rx-afxdp-swpoll-nopincpu-prio-rtkernel-128pps

driver rx latency, us (doesn't include stack latency, wire -> net subsystem): packets 100000
link = 1000Mbps, frame = 512B, rate = 128.0pps, 524.28kbps,
min = 33.21us, max = 89.41us, p-t-p = 56.20us, mean-+RMS = 38.17 +- 1.64 us

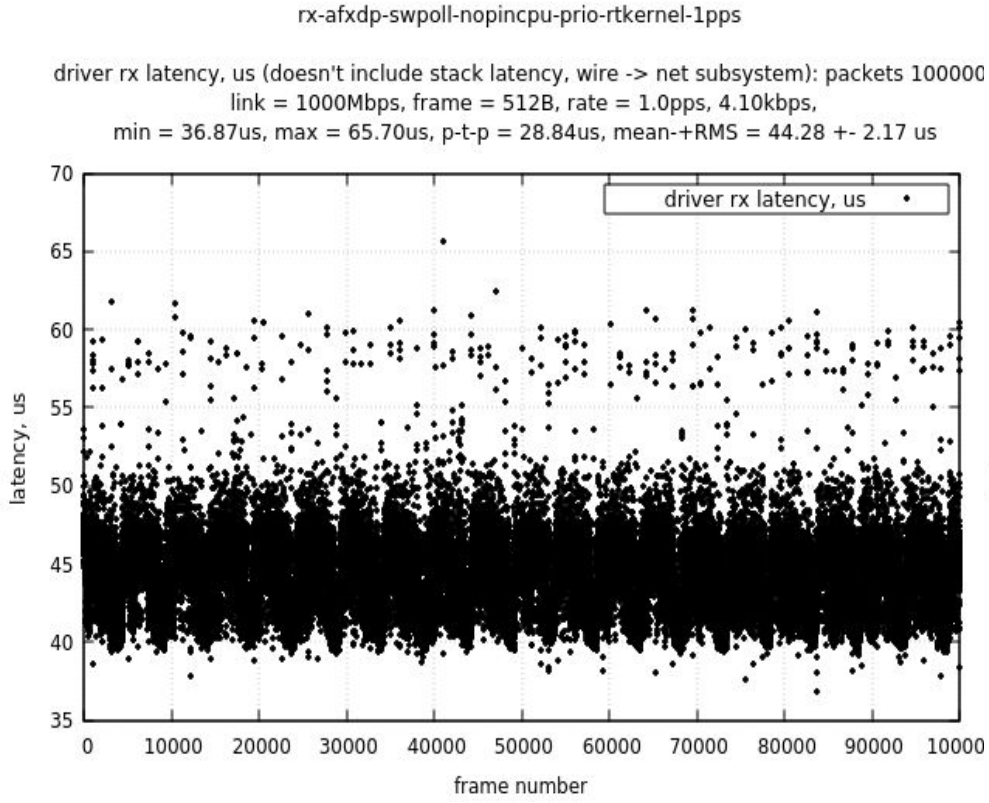
stack rx latency, us (doesn't include driver latency, net subsystem -> app): packets 100000
link = 1000Mbps, frame = 512B, rate = 128.0pps, 524.28kbps,
min = 6.83us, max = 29.44us, p-t-p = 22.61us, mean-+RMS = 7.93 +- 0.65 us

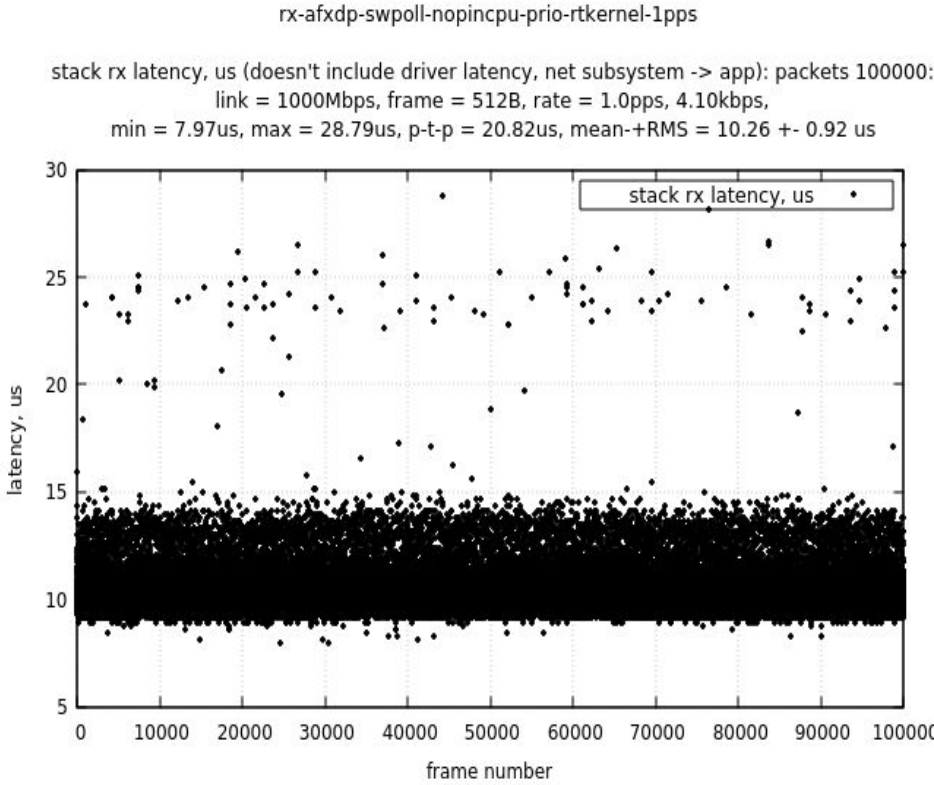
complete rx latency, us (includes driver latency and stack latency, wire -> app): packets 1000
link = 1000Mbps, frame = 512B, rate = 128.0pps, 524.28kbps,
min = 41.02us, max = 98.84us, p-t-p = 57.83us, mean-+RMS = 46.10 +- 2.02 us



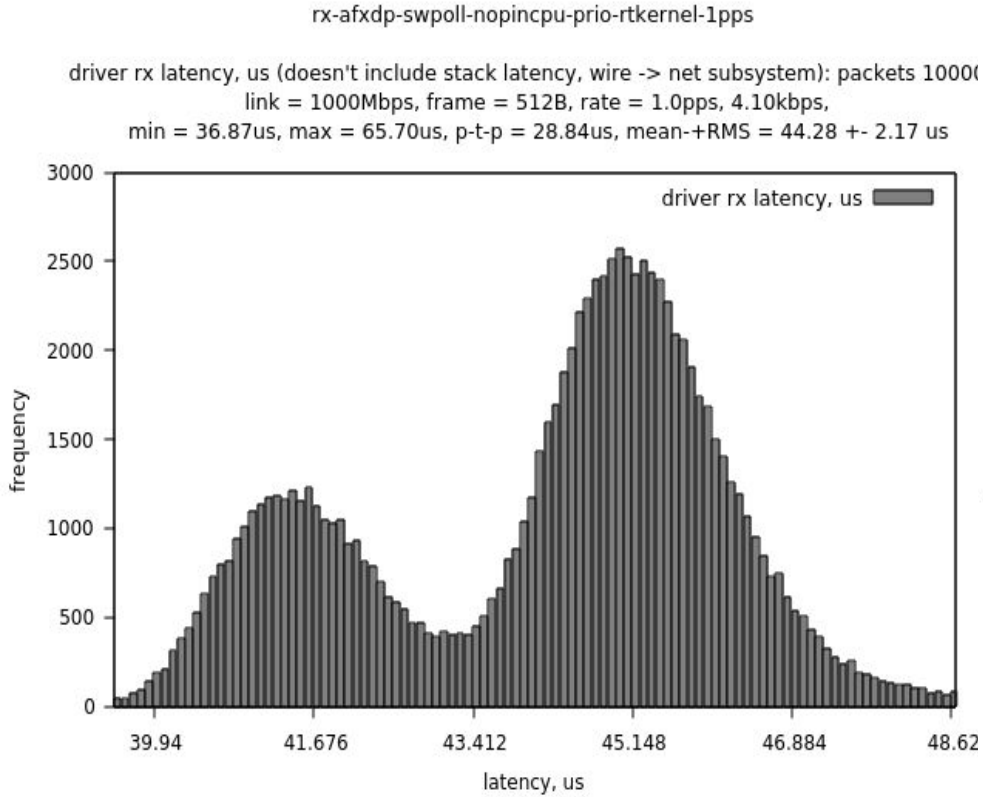
Linaro

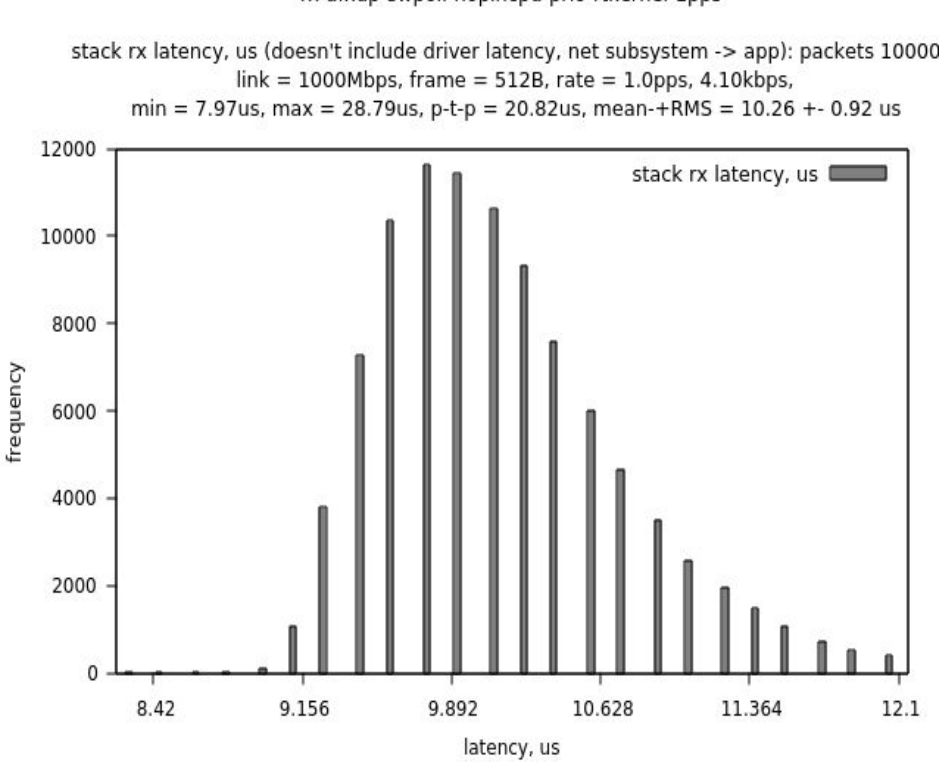
af_xdp RT driver/stack latency, sw poll, 1PPS



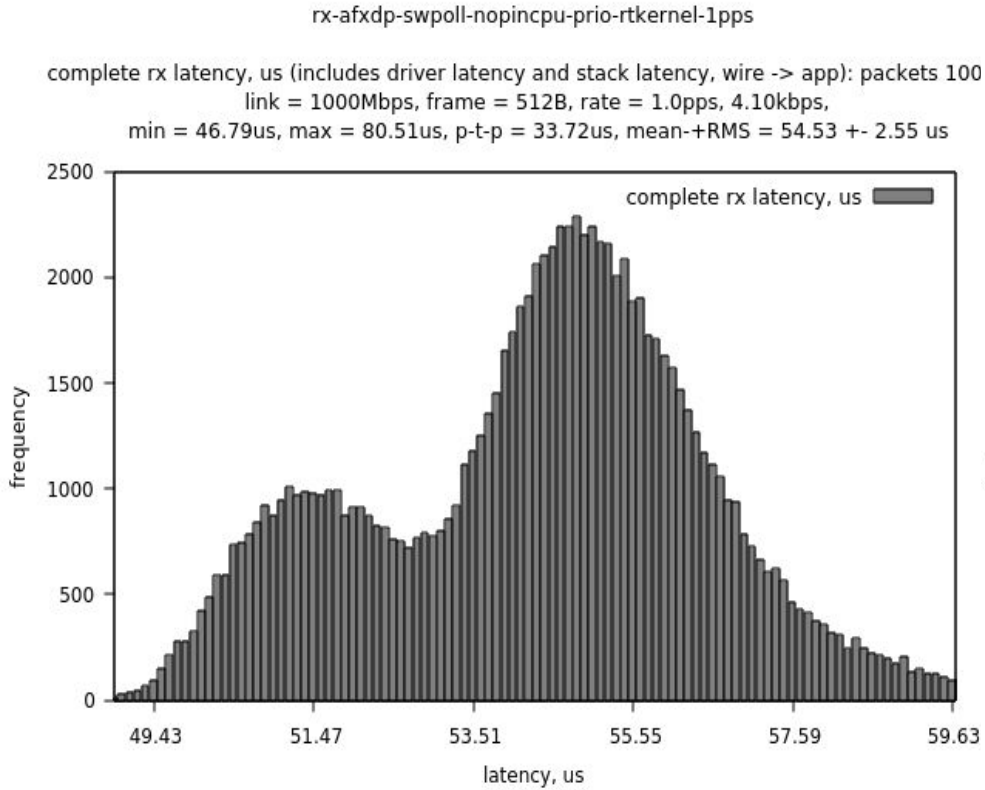


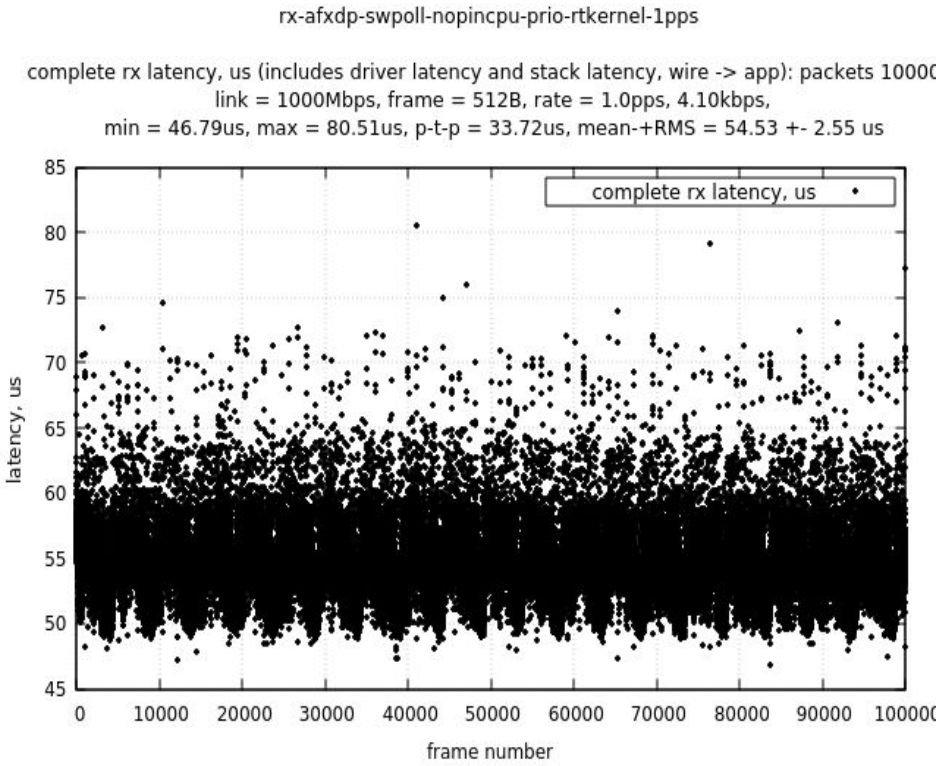
af_xdp RT driver/stack latency, sw poll, 1PPS





af_xdp RT complete latency, sw poll, 1PPS





Linaro

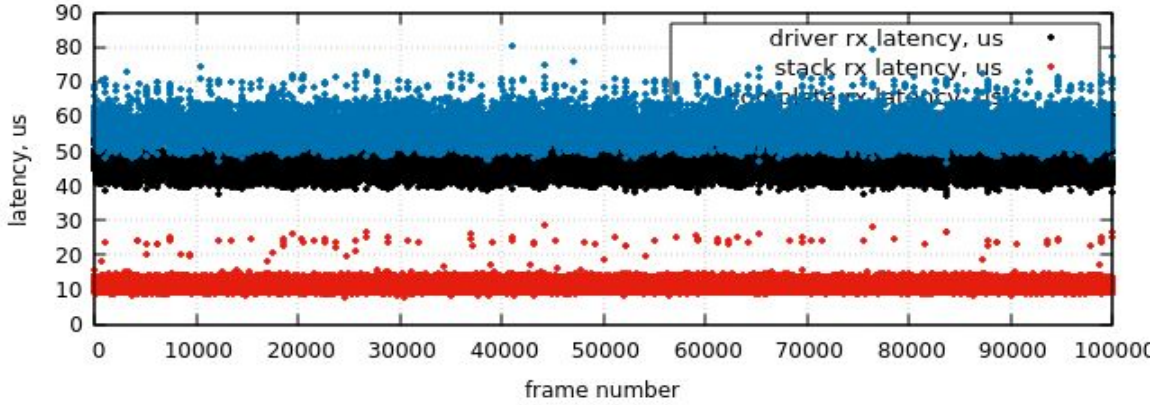
af_xdp RT driver/stack/complete, sw poll, 1PPS

rx-afxdp-swpoll-nopincpu-prio-rtkernel-1pps

driver rx latency, us (doesn't include stack latency, wire -> net subsystem): packets 100000:
link = 1000Mbps, frame = 512B, rate = 1.0pps, 4.10kbps,
min = 36.87us, max = 65.70us, p-t-p = 28.84us, mean-+RMS = 44.28 +- 2.17 us

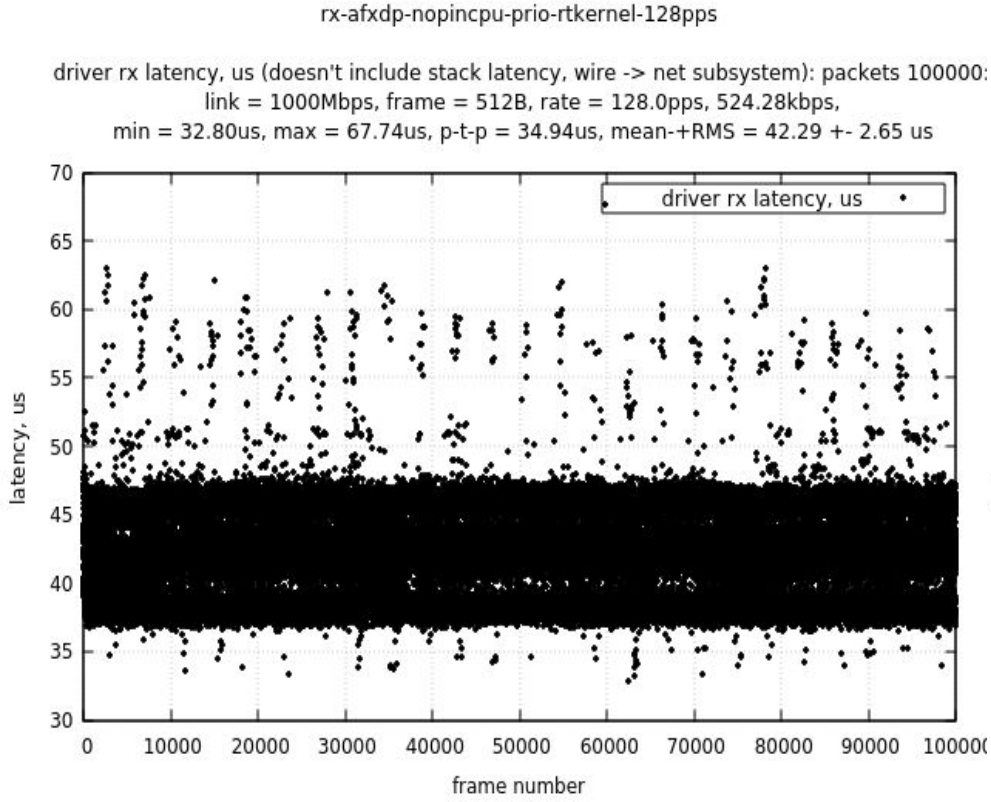
stack rx latency, us (doesn't include driver latency, net subsystem -> app): packets 100000:
link = 1000Mbps, frame = 512B, rate = 1.0pps, 4.10kbps,
min = 7.97us, max = 28.79us, p-t-p = 20.82us, mean-+RMS = 10.26 +- 0.92 us

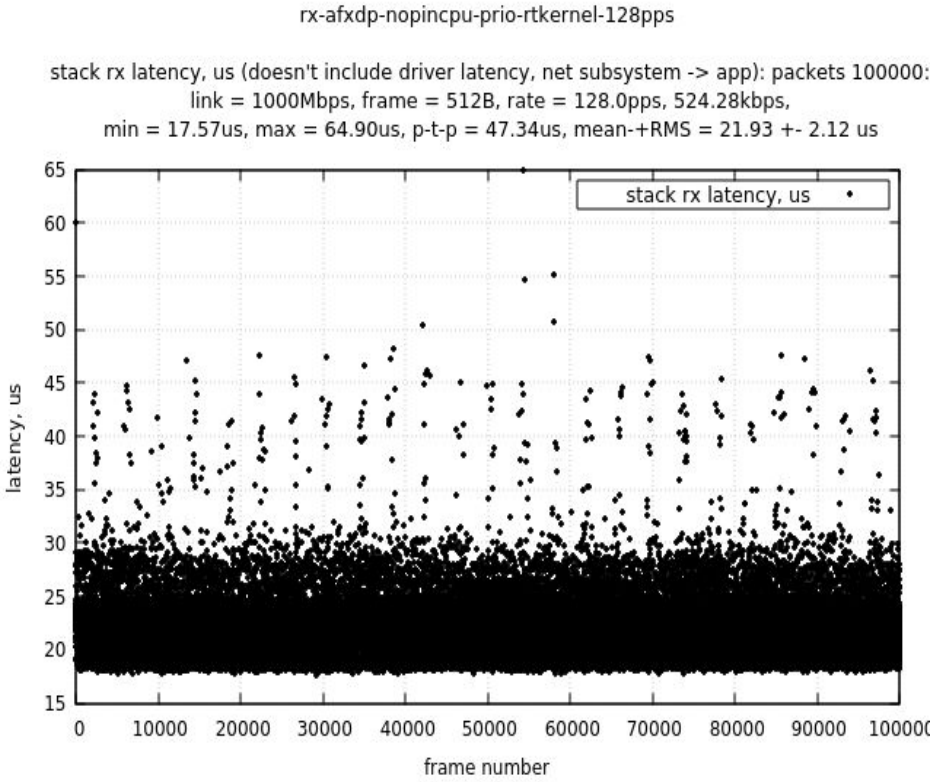
complete rx latency, us (includes driver latency and stack latency, wire -> app): packets 10000
link = 1000Mbps, frame = 512B, rate = 1.0pps, 4.10kbps,
min = 46.79us, max = 80.51us, p-t-p = 33.72us, mean-+RMS = 54.53 +- 2.55 us



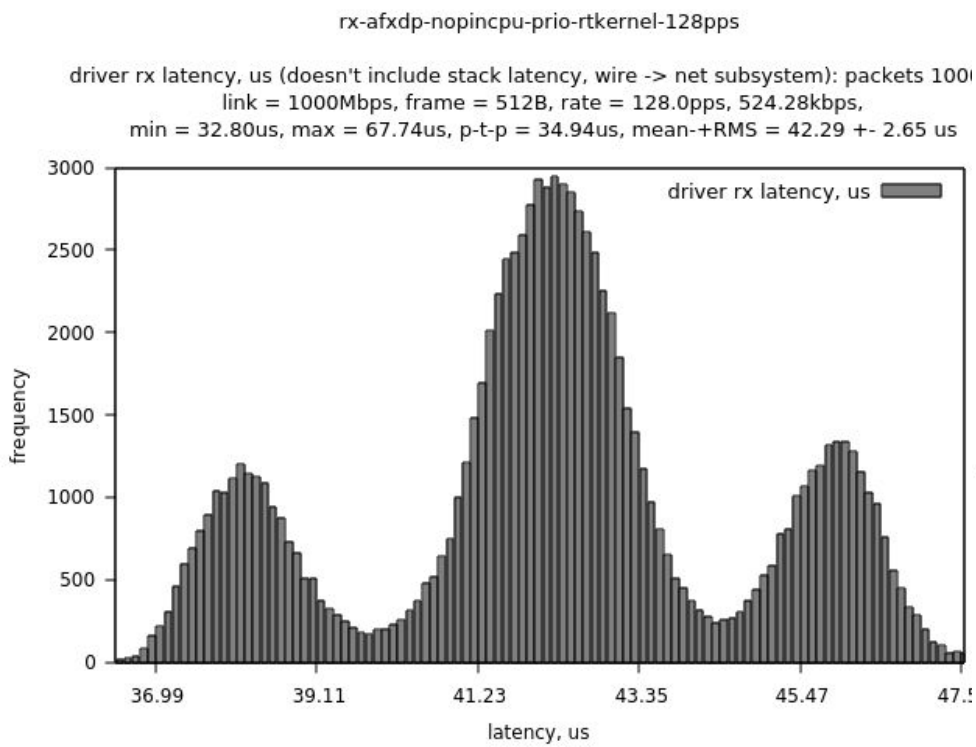
Linaro

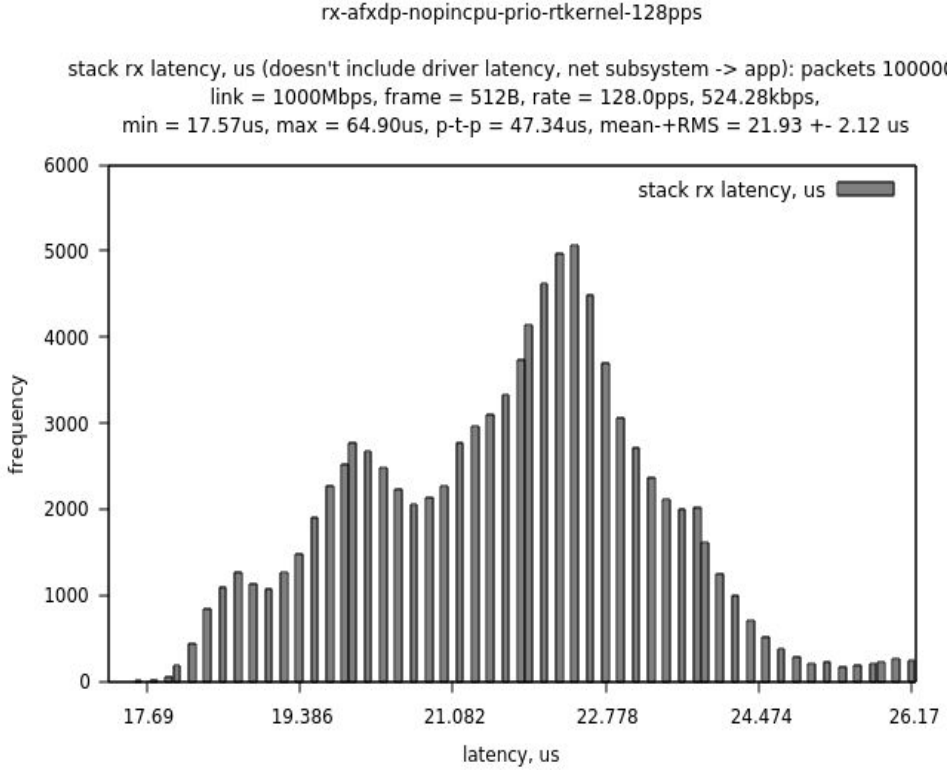
af_xdp RT driver/stack latency, sys poll, 128PPS



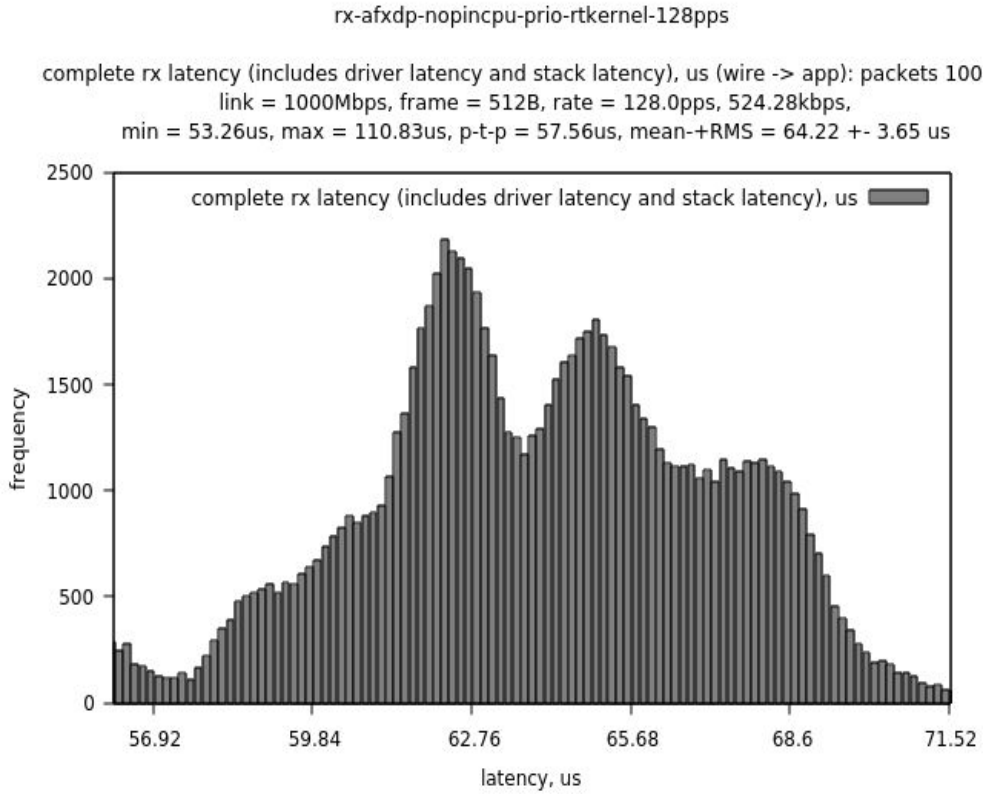


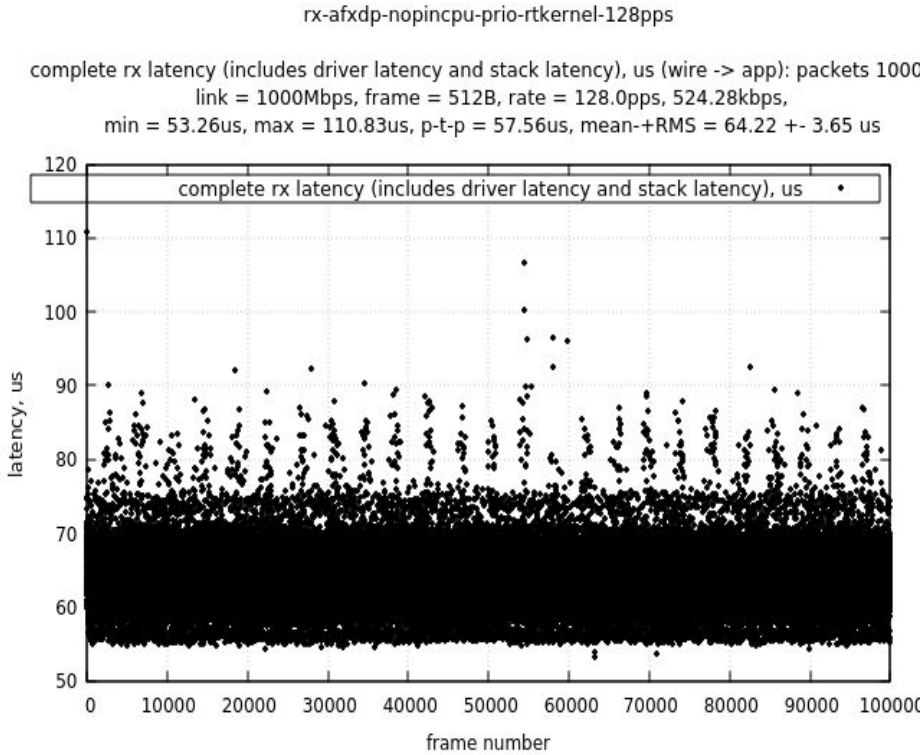
af_xdp RT driver/stack latency, sys poll, 128PPS



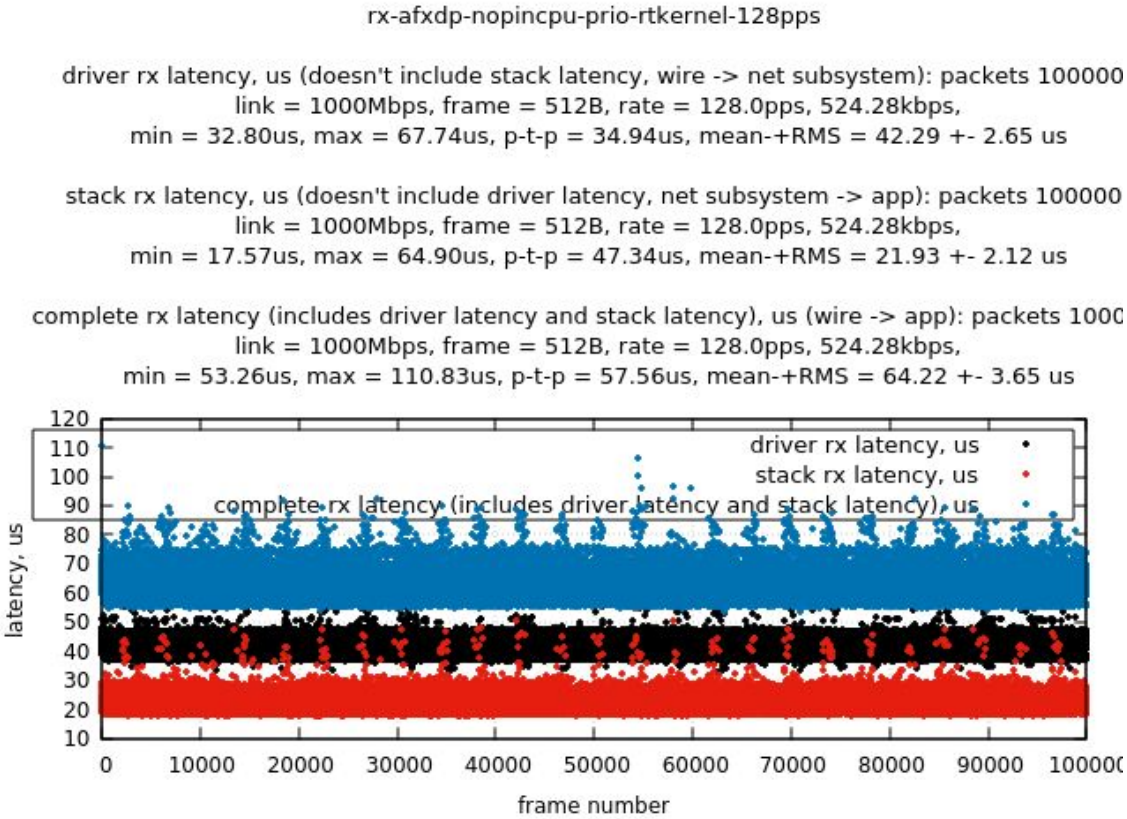


af_xdp RT complete latency, sys poll, 128PPS





af_xdp RT driver/stack/complete, sys poll, 128PPS



Thank you

Join Linaro to accelerate deployment of your Arm-based solutions through collaboration

contact@linaro.org