The Battle Of The ZCs

Who Is The Prettiest of Them All?

Jamal Hadi Salim | Nabil Bitar | Victor Nogueira | Pedro Tammela

Gemini: draw an image of two DRAM sticks playing a role in the theme of "mirror mirror on the wall, who is the prettiest of them all?"

Work Done At Bloomberg

Goals: RX+TX TCP ZC

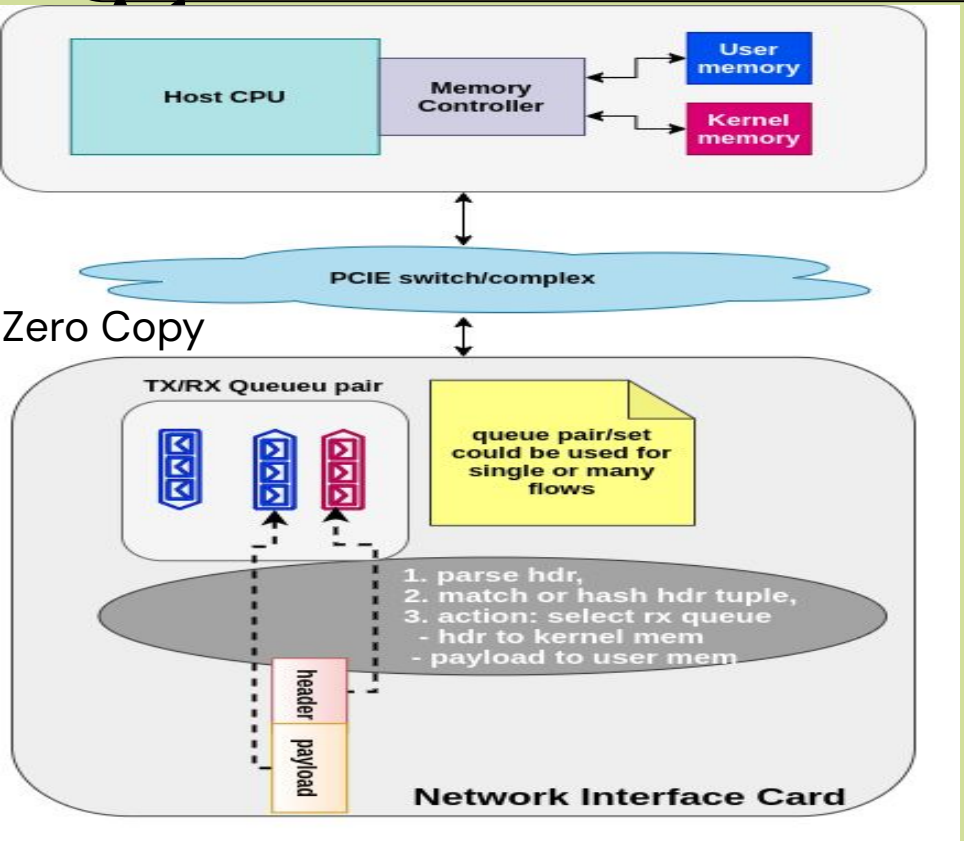
Investigate two new RX+TX TCP ZC techniques from an application perspective

- Baseline: traditional socket API app without any zero copy
- TCP Devmem and io_uring ZC: development in motion at time of preparing this talk

Note: There are other established ZC techniques we are not going to look at

- Total Kernel bypass: DPDK and friends
- Semi-kernel bypass: AF_XDP
- Sendfile (TX only)

Zero Copy High Level



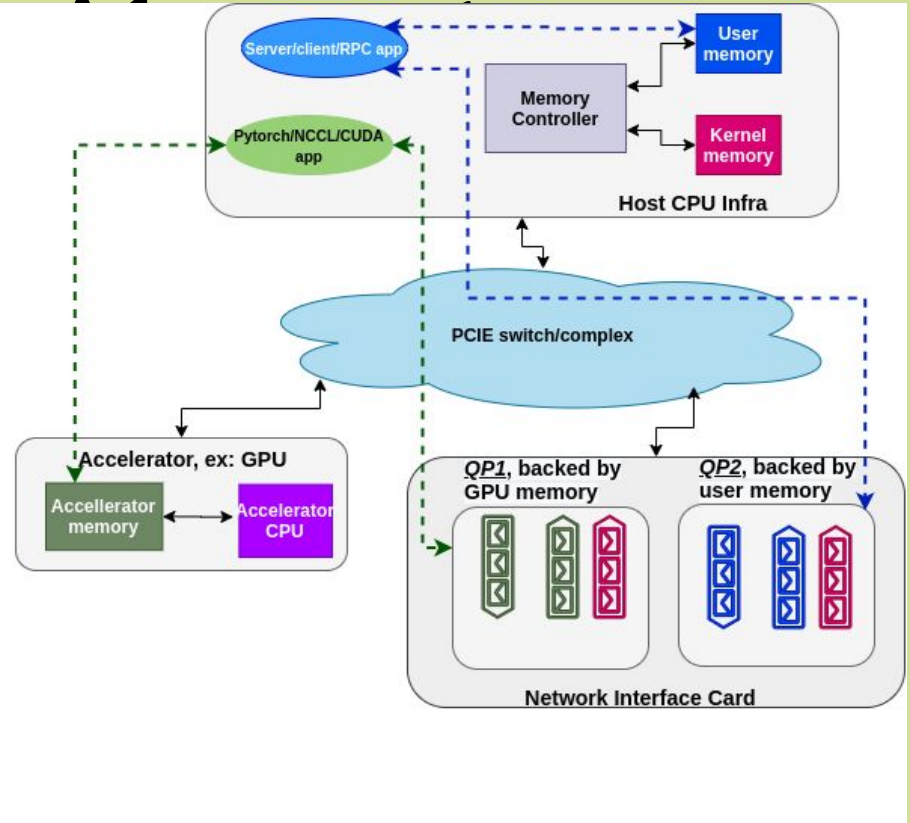
RX Control:

User memory(Blue) is bound to NIC buffers on rx of a queue pair

RX Fast path:

1. Incoming frame parsed for headers
2. header matched or hashed
3. Action: Header placed onto kernel memory(red) and payload placed on user mapped memory(blue)
4. Header goes over standard net stack and user app gets notified using either devmem or io_uring

Zero Copy High Level



User-space memory bound to NIC using udmabuf (Devmem)

io_uring memory mapped ring

- Accelerator (e.g., GPU) memory bound to NIC using dmabuf
  - Devmem
  - Not currently supported by io_uring

Setup

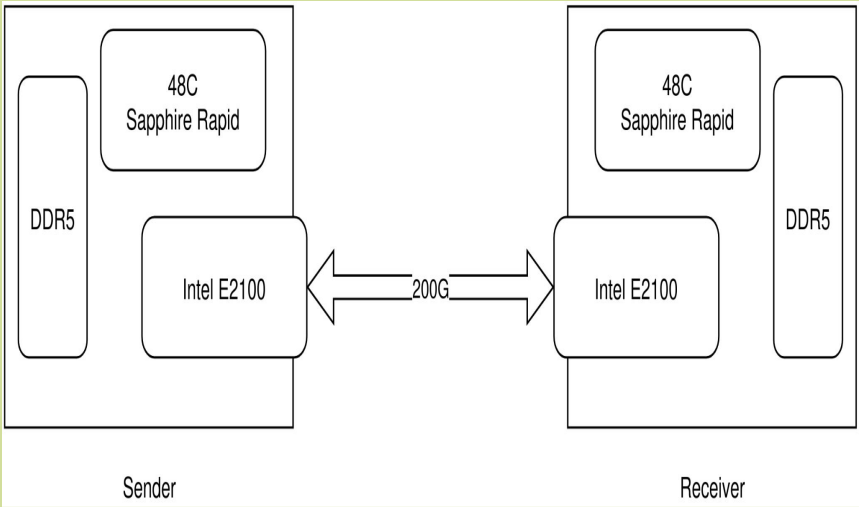
Traffic Patterns Used

To quote David Ahern @netdevconf 0x18:

"The ultimate test is to fill ~x00 gbps with one flow"

- Asymmetric Client-Server
  - MTU large enough to cover MSS++
  - IPv4
  - TCP Advertised mss: 4108B (4096B + 12B TCP options)
  - Very long flow analogous to Netperf TCP_STREAM- single flow runs for 100s
  - 64MB app message
  - udmabuf used for devmem (both sides)

Setup: CPU



- Motherboard: Intel Ruby Pass
- CPU: Single Intel(R) Sapphire Rapids CPU
  - Eagle Stream platform TDP 350W
  - 48C, Hyper-Threading off
- RAM: DDR5 128GB 4800 MT/s
- Application
  - We use our own home-brewed test program
  - CPU binding
    - Networking bound to 1 Core
    - User app bound to a different Core
  - Traffic: Single flow lands on queue 0

Setup: NIC

Intel IPU E2100:
(https://www.intel.com/content/www/us/en/products/details/network-io/ipu/adapter-e2100.html)

Used Single port connection back to back (2x100G Intel IPU E2100)

- using PCIE4x16 slot (should be able to handle 200Gbps)
- 2x100G with Port 0 at 200G config and Port 1 not in use
- Header split, SG, csum offload, TSO, HW GRO capability
- Ring size:
  - RX:4096
  - TX:4096

```
driver: idpf
version: 6.14.0-rc1+
firmware-version:
expansion-rom-version:
bus-info: 0000:6a:00.0
supports-statistics: yes
supports-test: no
supports-eeprom-access: no
supports-register-dump: no
supports-priv-flags: no
```

General settings

```
sysctl -w net.core.rmem_max=536870912
sysctl -w net.core.wmem_max=536870912
sysctl -w net.core.rmem_default=16777216
sysctl -w net.core.wmem_default=16777216
sysctl -w net.ipv4.tcp_rmem="4096 87380 536870912"
sysctl -w net.ipv4.tcp_wmem="4096 87380 536870912"

sysctl -w net.core.optmem_max=4194304

echo 16384 > /sys/devices/system/node/node0/hugepages/hugepages-2048kB/nr_hugepages

sysctl -w vm.max_map_count=524240
```

Receiver

```
sysctl -w net.core.optmem_max=4194304
```

Sender

General settings

- Kernel: 6.14.0_rc1
- Extra patches for driver
  - Adapt idpf driver to devmem tcp (From Sridhar)
  - Fix issue in RSC flow (From Sridhar)
  - Enable BIG TCP
    - Driver change to 131072B
    - Kernel config: CONFIG_MAX_SKB_FRAGS to 45
- io_uring: zcrx v13 (from Pavel's repo)
- devmem: RFC V1 (December 2024 from Mina)
  - Bug fixes on devmem TX

Setup: Measurement

- Interested in 3 metrics
  - Throughput, Power Consumption and CPU Usage
  - Results computed based on these 3 metrics
- Power consumption measured by an external device
  - Server as a ‘black box’
  - Baseline power around 230 Watts
- Test duration: 100 seconds (> 1TB of data exchanged, so glitches even out)
  - Repeated 5 times, throw 1st and last and middle 3 averaged
- Governor: starts in power-save mode but ramps up within 100 seconds

Experiments

Experiment Matrix

- MTU Fixed: 4148B (4K MSS)
- Receiver side setup: permutation of SW GRO, HW GRO, BigTCP, no BigTCP
  - Baseline (no ZC)
  - ZC
    - Devmem
    - io_uring
- Sender side: BigTCP and no BigTCP
  - Baseline (no ZC)
  - ZC
    - Devmem
    - io_uring

Here Be Dragons

Gemini prompt: " Draw an old map with dragons and other creatures "

Some Bugs Were Fixed!

- Rewind the global binding tx_iter by the dmabuf offset after tcp_sendmsg_locked finished
- Only return -EOPNOTSUPP if (!pool->dma_map || !pool->dma_sync) in page_pool_init (as it was before commit b400f4b8743)
- Add page pool stats to idpf driver

io_uring - iowait accounting

- io-uring trips the iowait cpu meta state - gets worse with more CPUs
  - On RX 2 cpus this accounts for up to ~35% but for 1 cpu it accounts for ~5%
  - On TX (2 cpus) it accounts for ~87%
  - Pavel provided us a patch (back in 0x18) but it was not used in these tests

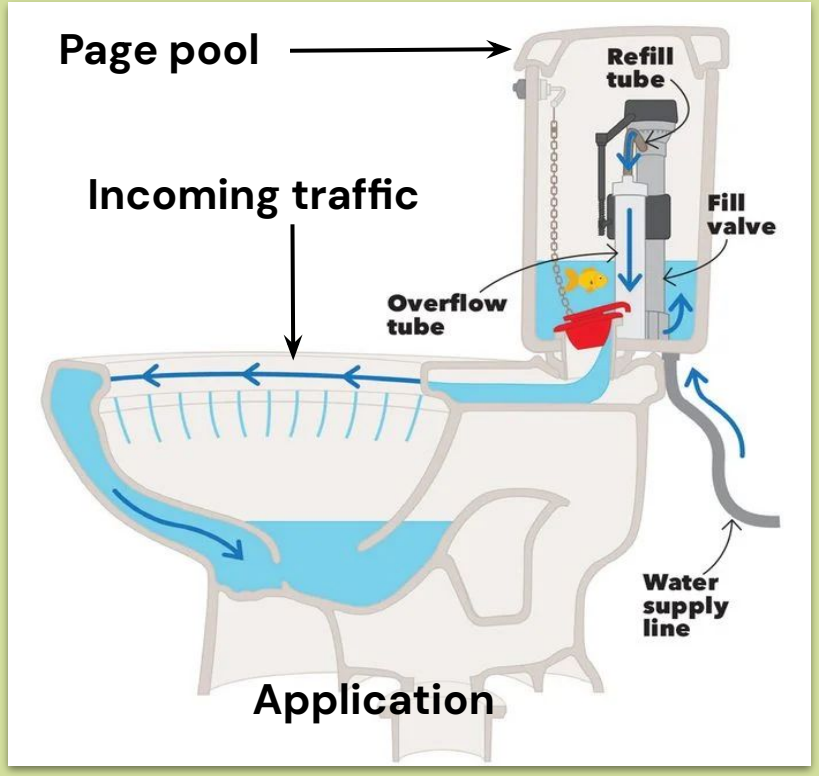
IO_URing - iowait accounting

CPU	POLL	C1	C1E	C6
0	**0**	0.01	57.06	26.14
1	**13.09**	6.68	41.16	19.2
2	**11.47**	5.41	43.19	45.38
3	**1.01**	1.23	58.13	20.56
4	**7.07**	4.23	40.04	37.66
5	**0**	0.01	51.67	30.73
6	**20.91**	9.43	27.83	13.34
7	**12.81**	3.16	40.48	42.64
8	**0.02**	0	38.25	73.54
9	**21.96**	10.02	23.68	46.14
10	**15.1**	8.6	33.94	45.59
11	**0**	0.01	33.22	47.47
12	**11.98**	5.41	33.17	30.66
13	**22.29**	9.16	23.86	44.8
14	**14.34**	6.04	35.44	44.07
15	**11.31**	5.67	32.77	57.62

Improved idle states
Reduced power consumption

CPU	POLL	C1	C1E	C6
0	**0**	0	0.58	67.21
1	**0.47**	1.56	49.8	27.06
2	**0.98**	1.4	42.03	29.86
3	**1.56**	1.66	49.61	22.85
4	**0.82**	1.61	30.51	33.81
5	**0.02**	0.04	19.82	63.87
6	**0**	0	0.67	99.29
7	**0**	0	0.6	82.31
8	**0.99**	1.61	50.61	36.51
9	**0**	0	0.32	83.25
10	**0.06**	0.11	25.89	52.4
11	**0.47**	1.59	40.61	53.88
12	**0.9**	1.72	46.9	44.29
13	**1.11**	1.16	42.77	42.01
14	**0.79**	1.67	42.32	52.22
15	**0**	0	0.58	89.64

Getting The Plumbing Right



200Gbps @0.1ms RTT implies we need to fill a tank of ~2MB to cover BDP (Bandwidth delay product)

Assuming ~2MB inflight to the application, use 2xBDP (~4MB)

Fix RX ring: 4096 (per qp) x page size(4096B) = 16MB.

Compensate for app sitting on accepted data (48MB): overprovision to 64MB per qp for udmabuf

Devmem RX: Receive Size

- We allocated Udmabuf size of 64MB
  - 4x the DMA RX size to factor in that the app may sit on the buffers for a while
- We observed varying recvmsg() read size also affects performance
  - Started with app read size of 512KB
  - In this scenario, buffers are not recycled fast enough, which triggers the page allocator more frequently (see “Bad”)
- Read as much as 64MB improved our throughput by 30%
  - On average we are seeing ~875 tokens (recall each payload is 4KB)

Bad:

```
Samples: 139K of event 'cycles', Event count (approx.): 89799604839
  Overhead  Command          Shared Object      Symbol
+    9.40%  ksoftirqd/0      [kernel.kallsyms]  [k] gen_pool_alloc_algo_owner
+    8.52%  :33681           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8.11%  ksoftirqd/0      [kernel.kallsyms]  [k] gen_pool_has_addr
+    7.98%  ksoftirqd/0      [kernel.kallsyms]  [k] gen_pool_free_owner
+    3.24%  swapper          [kernel.kallsyms]  [k] gen_pool_alloc_algo_owner
+    3.11%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    3.10%  swapper          [kernel.kallsyms]  [k] gen_pool_has_addr
+    2.93%  swapper          [kernel.kallsyms]  [k] gen_pool_free_owner
+    2.30%  :33681           [kernel.kallsyms]  [k] put_cmsg
```

Good:

```
Samples: 99K of event 'cpu-cycles', Event count (approx.): 90031202737
  Overhead  Command          Shared Object      Symbol
+   31.21%  :22366           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    6.25%  :22366           [kernel.kallsyms]  [k] napi_pp_put_page
+    6.09%  :22366           [kernel.kallsyms]  [k] xas_store
+    5.62%  :22366           [kernel.kallsyms]  [k] put_cmsg
+    2.87%  :22366           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    2.51%  :22366           [kernel.kallsyms]  [k] _raw_spin_lock
+    2.19%  :22366           [kernel.kallsyms]  [k] xas_find_marked
+    1.90%  :22366           [kernel.kallsyms]  [k] tcp_rcv_established
```

io_uring ring resizing

- In our tests <u>we may have</u> encountered cq events drops when testing on 1 CPU
  - This should not be possible, the default *IORING_FEAT_NODROP* is always on
  - Likely a bug, did not have time to investigate. Pavel gave us a tip:

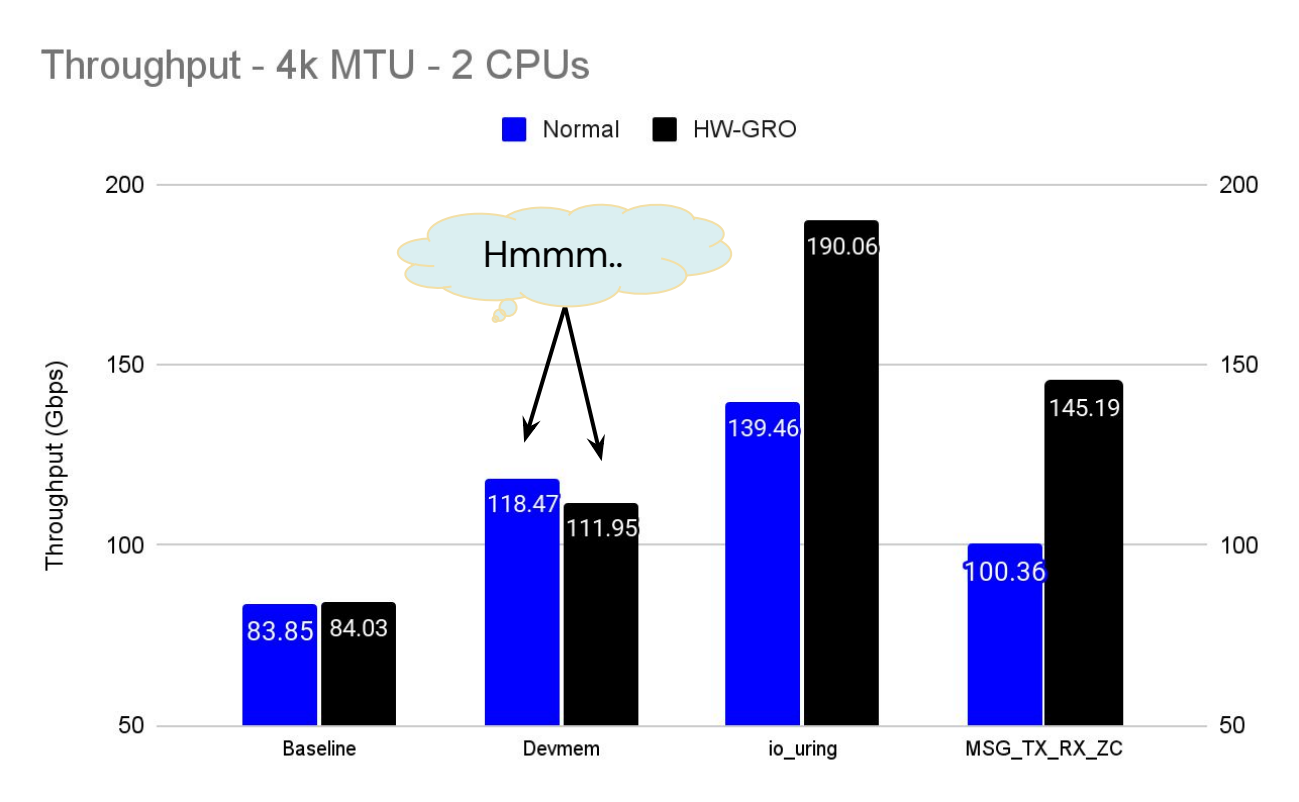
```
In extreme memory starvation cases a CQE might get dropped, the syscall
should eventually result in -EBADR.
And can be checked with liburing ring->cq.koverflow.
```

Note: Anytime we exhaust the page pool, the incoming data is wrapped in skbs and stored in socket queue then posted on CQ when space becomes available

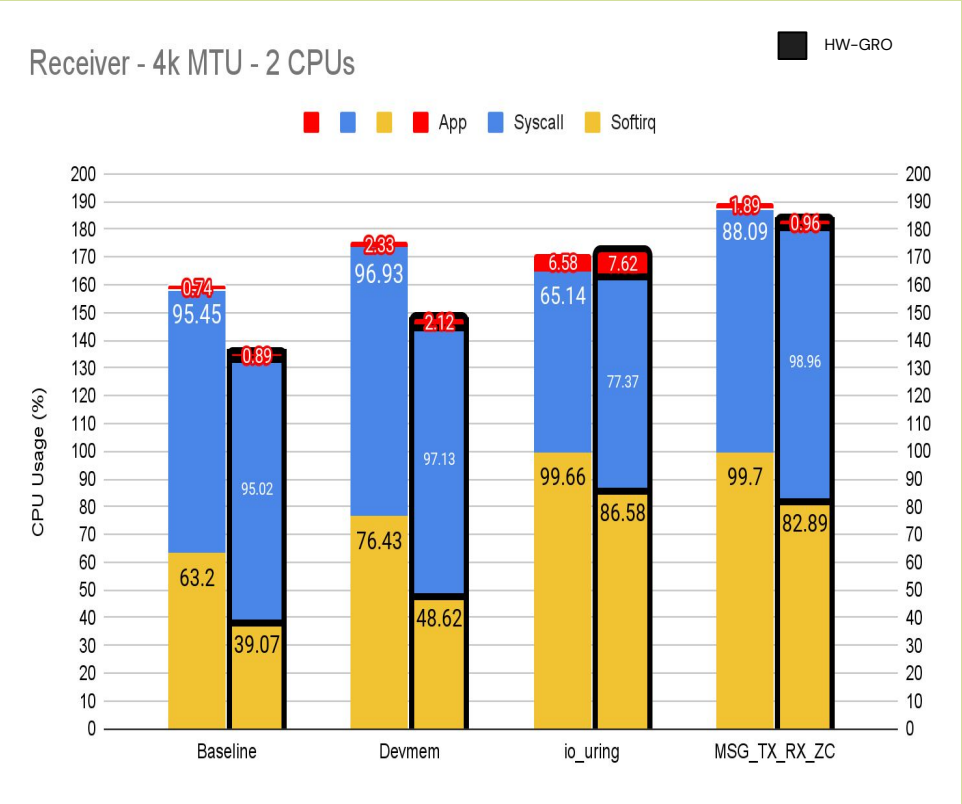
  - This approach is taken by both io_uring and devmem

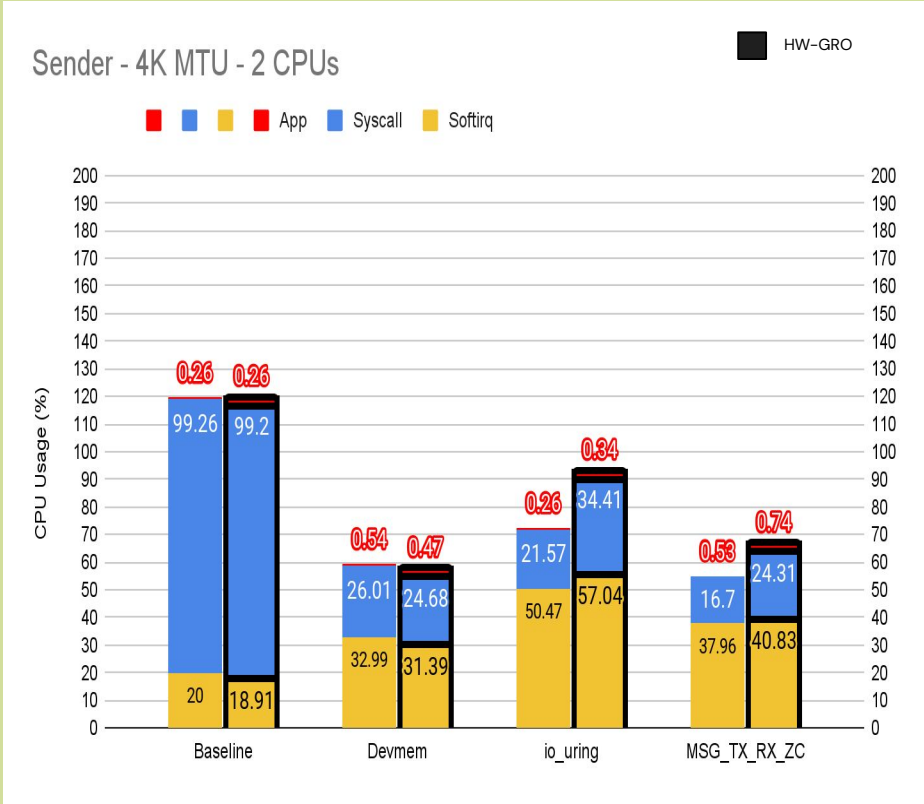
However <u>best results</u> are always achieved with proper provisioning

First Impressions: Throughput

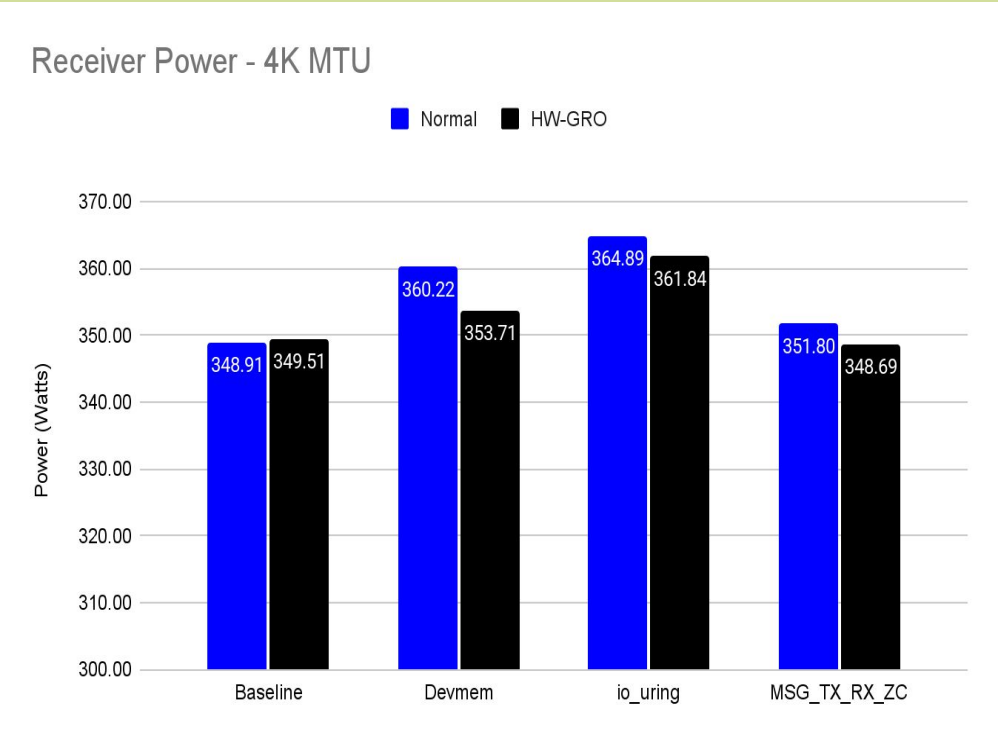


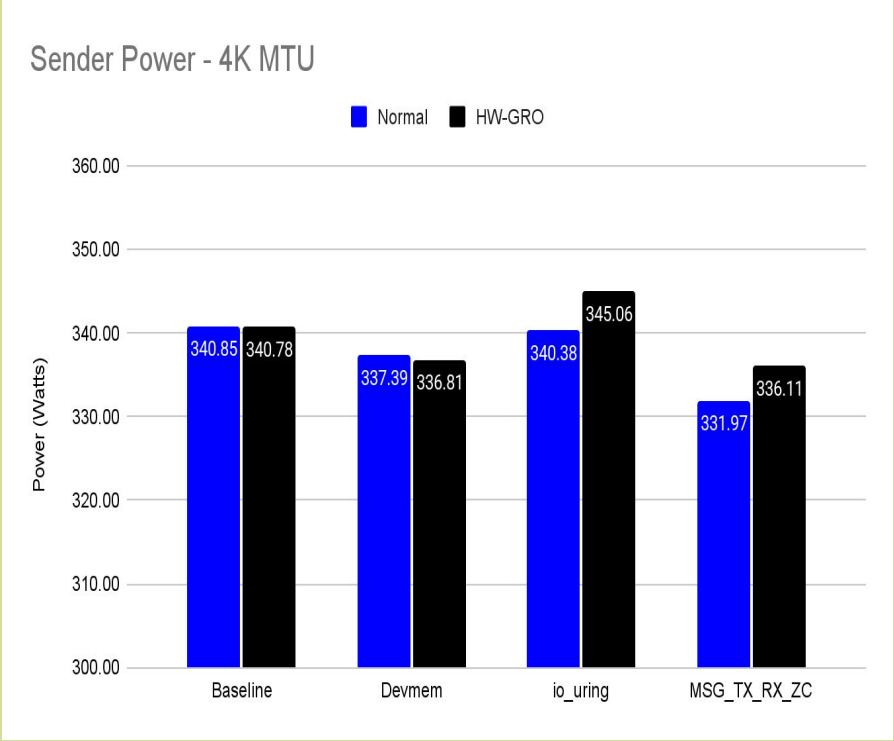
CPU Usage


Receiver - 4k MTU - 2 CPUs
HW-GRO
App
Syscall
Softirq
CPU Usage (%)
0.74
95.45
63.2
0.89
95.02
39.07
2.33
96.93
76.43
2.12
97.13
48.62
6.58
65.14
99.66
7.62
77.37
86.58
1.89
88.09
99.7
0.96
98.96
82.89
Baseline
Devmem
io_uring
MSG_TX_RX_ZC


Sender - 4K MTU - 2 CPUs
HW-GRO
App
Syscall
Softirq
CPU Usage (%)
0.26
99.26
20
0.26
99.2
18.91
0.54
26.01
32.99
0.47
24.68
31.39
0.26
21.57
50.47
0.34
34.41
57.04
0.53
16.7
37.96
0.74
24.31
40.83
Baseline
Devmem
io_uring
MSG_TX_RX_ZC

Power


Receiver Power - 4K MTU
Normal
HW-GRO
Power (Watts)
370.00
360.00
350.00
340.00
330.00
320.00
310.00
300.00
348.91
349.51
360.22
353.71
364.89
361.84
351.80
348.69
Baseline
Devmem
io_uring
MSG_TX_RX_ZC


Sender Power - 4K MTU
Normal
HW-GRO
Power (Watts)
360.00
350.00
340.00
330.00
320.00
310.00
300.00
340.85
340.78
337.39
336.81
340.38
345.06
331.97
336.11
Baseline
Devmem
io_uring
MSG_TX_RX_ZC

Devmem vs io_uring: S/W vs H/W GRO

Devmem

Devmem - 2 CPUs (118.47 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 168906025843
  Overhead  Command          Shared Object      Symbol
+   18,61%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    6,29%  :18102           [kernel.kallsyms]  [k] napi_pp_put_page
+    5,71%  swapper          [kernel.kallsyms]  [k] tcp_gro_receive
+    5,21%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,63%  :18102           [kernel.kallsyms]  [k] put_cmsg
+    2,93%  :18102           [kernel.kallsyms]  [k] xas_store
+    2,12%  :18102           [kernel.kallsyms]  [k] xas_find_marked
+    1,82%  :18102           [kernel.kallsyms]  [k] xas_load
+    1,68%  :18102           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,53%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,32%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,31%  swapper          [kernel.kallsyms]  [k] dev_gro_receive
+    1,26%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,24%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18102           [kernel.kallsyms]  [k] xas_clear_mark
+    1,11%  :18102           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,09%  swapper          [kernel.kallsyms]  [k] skb_gro_receive
+    0,87%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,85%  :18102           [kernel.kallsyms]  [k] tcp_rcv_space_adjust
+    0,84%  swapper          [kernel.kallsyms]  [k] __napi_build_skb
+    0,82%  swapper          [kernel.kallsyms]  [k] intel_idle
+    0,82%  :18102           [kernel.kallsyms]  [k] skb_try_coalesce
Tip: To browse sample contexts use perf report --sample 10 and select in context menu
```

Devmem - 2 CPUs - HW GRO (111.95 Gbps)

```
Samples: 198K of event 'cpu-cycles', Event count (approx.): 142019659159
  Overhead  Command          Shared Object      Symbol
+   22,57%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  :18805           [kernel.kallsyms]  [k] napi_pp_put_page
+    7,05%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    4,31%  swapper          [libeth]           [k] libeth_rx_recycle_slow
+    4,12%  :18805           [kernel.kallsyms]  [k] put_cmsg
+    3,87%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,41%  :18805           [kernel.kallsyms]  [k] xas_store
+    2,64%  swapper          [kernel.kallsyms]  [k] intel_idle
+    2,37%  :18805           [kernel.kallsyms]  [k] xas_find_marked
+    2,05%  :18805           [kernel.kallsyms]  [k] xas_load
+    1,87%  :18805           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,46%  :18805           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,37%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,25%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,25%  :18805           [kernel.kallsyms]  [k] xas_clear_mark
+    1,23%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18805           [kernel.kallsyms]  [k] skb_try_coalesce
+    1,07%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,01%  :18805           [kernel.kallsyms]  [k] tcp_v4_do_rcv
+    0,79%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,72%  :18805           [kernel.kallsyms]  [k] check_preemption_disabled
+    0,69%  :18805           [kernel.kallsyms]  [k] sock_rfree
+    0,69%  swapper          [kernel.kallsyms]  [k] net_rx_action
Tip: To see call chains by final symbol taking CPU time (bottom up) use perf report -G
```

io_uring

io_uring - 2 CPUs (139.46 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 172695707526
  Overhead  Command          Shared Object      Symbol
+    8,25%  :25567           [kernel.kallsyms]  [k] io_zcrx_recv_skb
+    6,66%  ksoftirqd/0      [idpf]             [k] idpf_vport_splitq_napi_poll
+    6,40%  ksoftirqd/0      [kernel.kallsyms]  [k] tcp_gro_receive
+    5,45%  ksoftirqd/0      [kernel.kallsyms]  [k] napi_pp_put_page
+    4,04%  :25567           [kernel.kallsyms]  [k] _raw_spin_lock
+    3,22%  swapper          [kernel.kallsyms]  [k] intel_idle
+    2,64%  ksoftirqd/0      [kernel.kallsyms]  [k] io_pp_zc_alloc_netmems
+    2,39%  ksoftirqd/0      [kernel.kallsyms]  [k] skb_release_data
+    1,85%  ksoftirqd/0      [kernel.kallsyms]  [k] dev_gro_receive
+    1,84%  ksoftirqd/0      [kernel.kallsyms]  [k] napi_build_skb
+    1,78%  ksoftirqd/0      [kernel.kallsyms]  [k] skb_gro_receive
+    1,57%  ksoftirqd/0      [kernel.kallsyms]  [k] page_pool_alloc_frag_netmem
+    1,49%  ksoftirqd/0      [kernel.kallsyms]  [k] __inet_lookup_establhsed
+    1,44%  ksoftirqd/0      [idpf]             [k] idpf_tx_clean_complq
+    1,05%  ksoftirqd/0      [kernel.kallsyms]  [k] check_preemption_disabled
+    1,02%  ksoftirqd/0      [kernel.kallsyms]  [k] inet_gro_receive
+    1,01%  :25567           [kernel.kallsyms]  [k] tcp_rcv_established
+    0,99%  ksoftirqd/0      [kernel.kallsyms]  [k] __napi_build_skb
+    0,99%  ksoftirqd/0      [kernel.kallsyms]  [k] napi_consume_skb
+    0,95%  ksoftirqd/0      [kernel.kallsyms]  [k] _raw_spin_lock
+    0,91%  ksoftirqd/0      [kernel.kallsyms]  [k] tcp_v4_rcv
+    0,88%  :25567           [kernel.kallsyms]  [k] tcp_recv_skb
+    0,87%  :25567           [kernel.kallsyms]  [k] __tcp_transmit_skb
```

io_uring - 2 CPUs - HW GRO (190.06 Gbps)

```
Samples: 191K of event 'cpu-cycles', Event count (approx.): 172013179812
  Overhead  Command          Shared Object      Symbol
+   10,78%  :30719           [kernel.kallsyms]  [k] io_zcrx_recv_skb
+    4,47%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,97%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    3,66%  :30719           [kernel.kallsyms]  [k] _raw_spin_lock
+    3,61%  swapper          [kernel.kallsyms]  [k] poll_idle
+    2,98%  swapper          [kernel.kallsyms]  [k] io_pp_zc_alloc_netmems
+    2,85%  swapper          [libeth]           [k] libeth_rx_recycle_slow
+    2,12%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    2,08%  swapper          [kernel.kallsyms]  [k] page_pool_alloc_frag_netmem
+    1,94%  swapper          [kernel.kallsyms]  [k] net_rx_action
+    1,78%  swapper          [idpf]             [k] idpf_tx_clean_complq
+    1,69%  swapper          [kernel.kallsyms]  [k] _raw_spin_lock
+    1,62%  swapper          [kernel.kallsyms]  [k] intel_idle
+    1,54%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    1,40%  swapper          [kernel.kallsyms]  [k] intel_idle_xstate
+    1,40%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,30%  :30719           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,22%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    0,96%  :30719           [kernel.kallsyms]  [k] sock_rfree
+    0,96%  :30719           [kernel.kallsyms]  [k] skb_attempt_defer_free
+    0,94%  :30719           [kernel.kallsyms]  [k] native_queued_spin_lock_slowpath
+    0,91%  :30719           [kernel.kallsyms]  [k] tcp_recv_skb
+    0,77%  :30719           [kernel.kallsyms]  [k] check_preemption_disabled
```

Syscall Costs

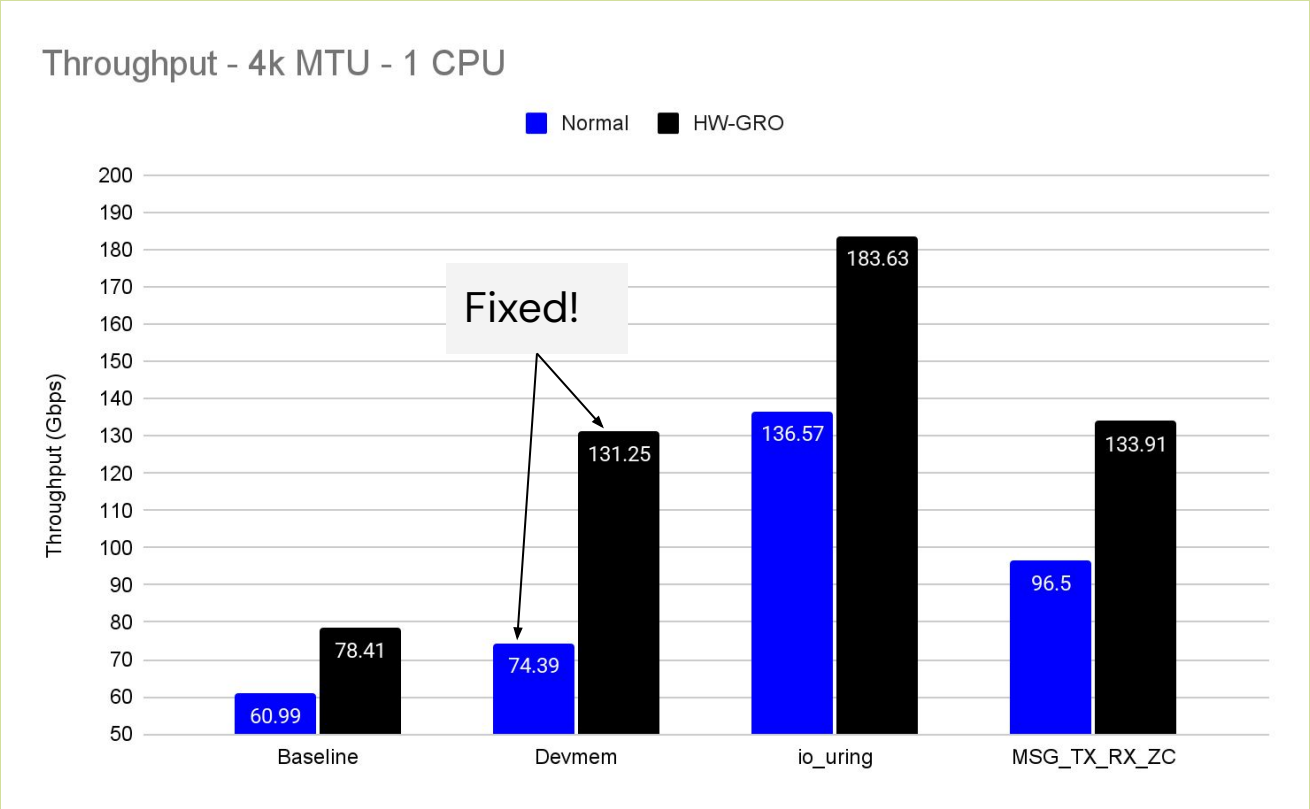
Devmem: 2 CPUs

```
:14342 (14342), 4745666 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  recvmsg           615359    1830 68831.493     0.001     0.112     0.371      0.03%
  setsockopt        530116       0 22841.143     0.001     0.043     0.284      0.03%
  ioctl            1224901       0  1232.993     0.001     0.001     0.604      0.09%
  epoll_wait          1835       0   628.492     0.000     0.343   100.162     28.36%
```

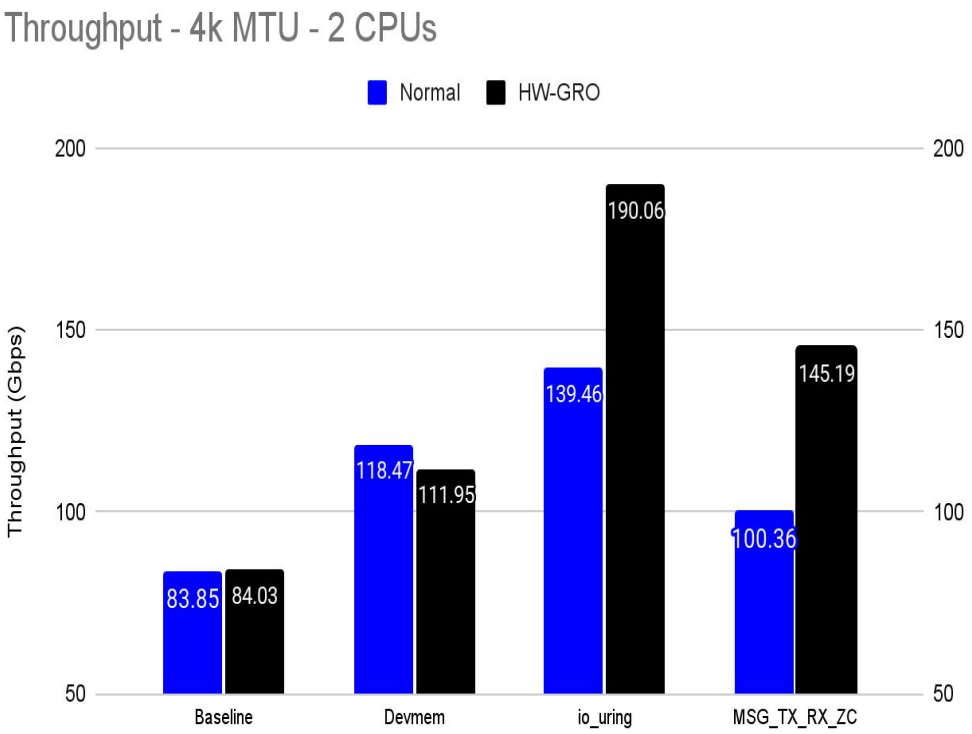
io_uring: 2 CPUs

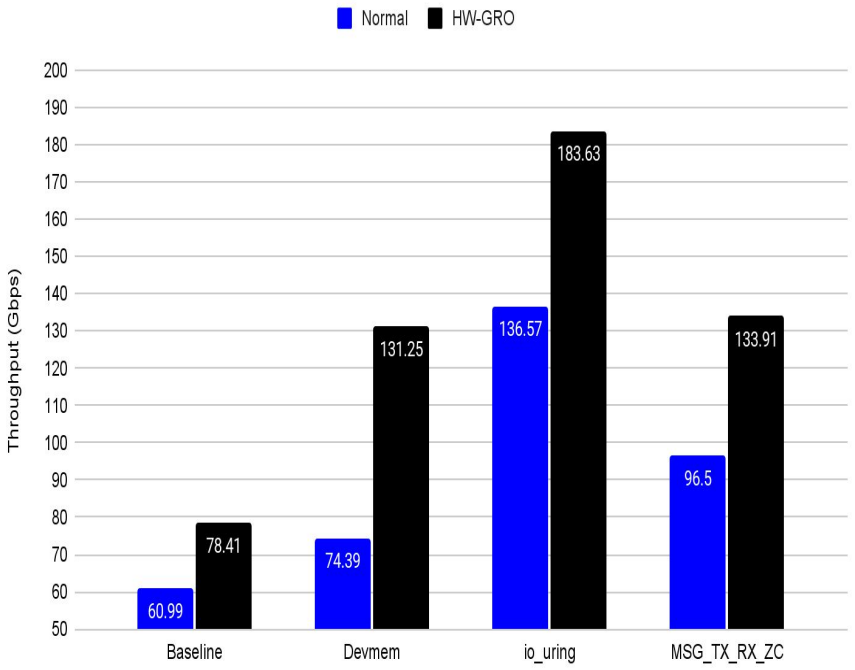
```
tcpzc2 (10500), 21032201 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  io_uring_enter  10519910       0 76548.951     0.000     0.007     0.450      0.02%
```

Throughput (1 CPU)



Throughput (1 vs 2 CPU)





Perf Devmem GRO (1 vs 2 CPUs)

Devmem - 2 CPUs (118.47 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 168906025843
  Overhead  Command          Shared Object      Symbol
+   10,61%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    6,29%  :18102           [kernel.kallsyms]  [k] napi_pp_put_page
+    5,71%  swapper          [kernel.kallsyms]  [k] tcp_gro_receive
+    5,21%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,63%  :18102           [kernel.kallsyms]  [k] put_cmsg
+    2,93%  :18102           [kernel.kallsyms]  [k] xas_store
+    2,12%  :18102           [kernel.kallsyms]  [k] xas_find_marked
+    1,82%  :18102           [kernel.kallsyms]  [k] xas_load
+    1,68%  :18102           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,53%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,32%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,31%  swapper          [kernel.kallsyms]  [k] dev_gro_receive
+    1,26%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,24%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18102           [kernel.kallsyms]  [k] xas_clear_mark
+    1,11%  :18102           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,09%  swapper          [kernel.kallsyms]  [k] skb_gro_receive
+    0,87%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,85%  :18102           [kernel.kallsyms]  [k] tcp_rcv_space_adjust
+    0,84%  swapper          [kernel.kallsyms]  [k] __napi_build_skb
+    0,82%  swapper          [kernel.kallsyms]  [k] intel_idle
+    0,82%  :18102           [kernel.kallsyms]  [k] skb_try_coalesce
Tip: When collecting LBR backtraces use --stitch-lbr to handle more than 32 deep entries: perf record --call-graph lbr ; perf report --stitc
```

Devmem - 2 CPUs - HW GRO (111.95 Gbps)

```
Samples: 198K of event 'cpu-cycles', Event count (approx.): 142019659159
  Overhead  Command          Shared Object      Symbol
+   22,57%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  :18805           [kernel.kallsyms]  [k] napi_pp_put_page
+    7,05%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    4,31%  swapper          [libeth]           [k] libeth_rx_recycle_slow
+    4,12%  :18805           [kernel.kallsyms]  [k] put_cmsg
+    3,87%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,41%  :18805           [kernel.kallsyms]  [k] xas_store
+    2,64%  swapper          [kernel.kallsyms]  [k] intel_idle
+    2,37%  :18805           [kernel.kallsyms]  [k] xas_find_marked
+    2,05%  :18805           [kernel.kallsyms]  [k] xas_load
+    1,87%  :18805           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,46%  :18805           [kernel.kallsyms]  [k] tcp_rcv_establlshed
+    1,37%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,25%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,25%  :18805           [kernel.kallsyms]  [k] xas_clear_mark
+    1,23%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18805           [kernel.kallsyms]  [k] skb_try_coalesce
+    1,07%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,01%  :18805           [kernel.kallsyms]  [k] tcp_v4_do_rcv
+    0,79%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,72%  :18805           [kernel.kallsyms]  [k] check_preemption_disabled
+    0,69%  :18805           [kernel.kallsyms]  [k] sock_rfree
+    0,69%  swapper          [kernel.kallsyms]  [k] net_rx_action
Tip: Compare performance results with: perf diff [<old file> <new file>]
```

Devmem - 1 CPU (74.39 Gbps)

```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 93207866858
  Overhead  Command          Shared Object      Symbol
+    8,02%  :8076            [kernel.vmlinux]   [k] gen_pool_has_addr
+    7,88%  :8076            [kernel.vmlinux]   [k] gen_pool_free_owner
+    7,70%  :8076            [idpf]             [k] idpf_vport_splitq_napi_poll
+    7,04%  :8076            [kernel.vmlinux]   [k] tcp_recvmsg_dmabuf
+    6,52%  :8076            [kernel.vmlinux]   [k] tcp_gro_receive
+    6,40%  :8076            [kernel.vmlinux]   [k] gen_pool_alloc_algo_owner
+    4,25%  :8076            [kernel.vmlinux]   [k] put_cmsg
+    3,65%  :8076            [kernel.vmlinux]   [k] xas_store
+    3,27%  :8076            [kernel.vmlinux]   [k] napi_pp_put_page
+    2,34%  :8076            [kernel.vmlinux]   [k] xas_find_marked
+    2,28%  :8076            [kernel.vmlinux]   [k] xas_load
+    1,75%  :8076            [kernel.vmlinux]   [k] _raw_spin_lock_bh
+    1,65%  :8076            [kernel.vmlinux]   [k] check_preemption_disabled
+    1,56%  :8076            [kernel.vmlinux]   [k] dev_gro_receive
+    1,30%  :8076            [kernel.vmlinux]   [k] xas_clear_mark
+    1,26%  :8076            [kernel.vmlinux]   [k] net_is_devmem_iov
+    1,20%  :8076            [kernel.vmlinux]   [k] page_pool_alloc_frag_netmem
+    1,09%  :8076            [kernel.vmlinux]   [k] inet_gro_receive
+    1,00%  :8076            [kernel.vmlinux]   [k] skb_gro_receive
+    0,89%  :8076            [kernel.vmlinux]   [k] __napi_build_skb
+    0,88%  :8076            [kernel.vmlinux]   [k] eth_type_trans
+    0,85%  :8076            [kernel.vmlinux]   [k] page_pool_put_unrefed_netmem
+    0,74%  :8076            [kernel.vmlinux]   [k] napi_build_skb
```

Devmem - 1 CPU - HW GRO (131.25 Gbps)

```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 93160176214
  Overhead  Command          Shared Object      Symbol
+    9,35%  :8892            [kernel.vmlinux]   [k] tcp_recvmsg_dmabuf
+    7,52%  :8892            [kernel.vmlinux]   [k] put_cmsg
+    6,57%  :8892            [kernel.vmlinux]   [k] xas_store
+    4,74%  :8892            [idpf]             [k] idpf_vport_splitq_napi_poll
+    4,49%  :8892            [kernel.vmlinux]   [k] xas_find_marked
+    3,97%  :8892            [kernel.vmlinux]   [k] xas_load
+    3,81%  :8892            [kernel.vmlinux]   [k] napi_pp_put_page
+    3,30%  :8892            [libeth]           [k] libeth_rx_recycle_slow
+    2,95%  :8892            [kernel.vmlinux]   [k] _raw_spin_lock_bh
+    2,27%  :8892            [kernel.vmlinux]   [k] page_pool_alloc_frag_netmem
+    2,26%  :8892            [kernel.vmlinux]   [k] xas_clear_mark
+    1,74%  :8892            [kernel.vmlinux]   [k] check_preemption_disabled
+    1,42%  :8892            [kernel.vmlinux]   [k] net_is_devmem_iov
+    1,42%  :8892            [kernel.vmlinux]   [k] __xa_cmpxchg
+    1,26%  :8892            [kernel.vmlinux]   [k] xas_start
+    1,24%  :8892            [kernel.vmlinux]   [k] page_pool_put_unrefed_netmem
+    1,05%  :8892            [kernel.vmlinux]   [k] preempt_count_add
+    1,01%  :8892            [kernel.vmlinux]   [k] tcp_gro_receive
+    0,96%  :8892            [kernel.vmlinux]   [k] xas_create
+    0,94%  :8892            [kernel.vmlinux]   [k] __xa_alloc
+    0,89%  :8892            [kernel.vmlinux]   [k] xas_set_mark
+    0,86%  :8892            [kernel.vmlinux]   [k] __local_bh_enable_ip
+    0,82%  :8892            [kernel.vmlinux]   [k] page_pool_refill_alloc_cache
```

Syscall Costs (All In One Core)

```
:10435 (10435), 1627807 events, 100.0%

  syscall            calls  errors  total       min       avg       max       stddev
                                    (msec)    (msec)    (msec)    (msec)        (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  setsockopt        313019       0 63395.250     0.001     0.203   460.799      0.84%
  recvmsg           168459    1391 26726.717     0.001     0.159    84.382      0.32%
  ioctl             334018       0  1547.941     0.001     0.005    26.047      3.12%
  epoll_wait          1398       0   727.045     0.000     0.520   100.199     25.99%
```

```
tcpzc2 (15989), 2172455 events, 100.0%

  syscall            calls  errors  total       min       avg       max       stddev
                                    (msec)    (msec)    (msec)    (msec)        (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  io_uring_enter    1087324       0 46435.945     0.001     0.043    42.240      1.09%
```

Devmem Syscall Costs (2 vs 1 CPU)

2 CPUs

```
:14342 (14342), 4745666 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  recvmsg            615359    1830 68831.493     0.001     0.112     0.371      0.03%
  setsockopt         530116       0 22841.143     0.001     0.043     0.284      0.03%
  ioctl             1224901       0  1232.993     0.001     0.001     0.604      0.09%
  epoll_wait           1835       0   628.492     0.000     0.343   100.162     28.36%
```

1 CPU

```
:10435 (10435), 1627807 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  setsockopt         313019       0 63395.250     0.001     0.203   460.799      0.84%
  recvmsg            168459    1391 26726.717     0.001     0.159    84.382      0.32%
  ioctl              334018       0  1547.941     0.001     0.005    26.047      3.12%
  epoll_wait           1398       0   727.045     0.000     0.520   100.199     25.99%
```

Devmem Recycling Cost

Devmem - 2 CPUs (118.47 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 168906025843
  Overhead  Command          Shared Object      Symbol
+   18,61%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    6,29%  :18102           [kernel.kallsyms]  [k] napi_pp_put_page
+    5,71%  swapper          [kernel.kallsyms]  [k] tcp_gro_receive
+    5,21%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,63%  :18102           [kernel.kallsyms]  [k] put_cmsg
+    2,93%  :18102           [kernel.kallsyms]  [k] xas_store
+    2,12%  :18102           [kernel.kallsyms]  [k] xas_find_marked
+    1,82%  :18102           [kernel.kallsyms]  [k] xas_load
+    1,68%  :18102           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,53%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,32%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,31%  swapper          [kernel.kallsyms]  [k] dev_gro_receive
+    1,26%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,24%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18102           [kernel.kallsyms]  [k] xas_clear_mark
+    1,11%  :18102           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,09%  swapper          [kernel.kallsyms]  [k] skb_gro_receive
+    0,87%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,85%  :18102           [kernel.kallsyms]  [k] tcp_rcv_space_adjust
+    0,84%  swapper          [kernel.kallsyms]  [k] __napi_build_skb
+    0,82%  swapper          [kernel.kallsyms]  [k] intel_idle
+    0,82%  :18102           [kernel.kallsyms]  [k] skb_try_coalesce
Tip: When collecting LBR backtraces use --stitch-lbr to handle more than 32 deep entries: perf record --call-graph lbr ; perf report --stitc
```

Devmem - 2 CPUs - HW GRO (111.95 Gbps)

```
Samples: 198K of event 'cpu-cycles', Event count (approx.): 142019659159
  Overhead  Command          Shared Object      Symbol
+   22,57%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  :18805           [kernel.kallsyms]  [k] napi_pp_put_page
+    7,85%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    4,31%  swapper          [libeth]           [k] libeth_rx_recycle_slow
+    4,12%  :18805           [kernel.kallsyms]  [k] put_cmsg
+    3,87%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,41%  :18805           [kernel.kallsyms]  [k] xas_store
+    2,64%  swapper          [kernel.kallsyms]  [k] intel_idle
+    2,37%  :18805           [kernel.kallsyms]  [k] xas_find_marked
+    2,05%  :18805           [kernel.kallsyms]  [k] xas_load
+    1,87%  :18805           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,46%  :18805           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,37%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,25%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,25%  :18805           [kernel.kallsyms]  [k] xas_clear_mark
+    1,23%  :18805           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18805           [kernel.kallsyms]  [k] skb_try_coalesce
+    1,07%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,01%  :18805           [kernel.kallsyms]  [k] tcp_v4_do_rcv
+    0,79%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,72%  :18805           [kernel.kallsyms]  [k] check_preemption_disabled
+    0,69%  :18805           [kernel.kallsyms]  [k] sock_rfree
+    0,69%  swapper          [kernel.kallsyms]  [k] net_rx_action
Tip: Compare performance results with: perf diff [<old file> <new file>]
```

Devmem - 1 CPU (74.39 Gbps)

```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 93207866858
  Overhead  Command          Shared Object      Symbol
+    8,02%  :8076            [kernel.vmlinux]   [k] gen_pool_has_addr
+    7,86%  :8076            [kernel.vmlinux]   [k] gen_pool_free_owner
+    7,70%  :8076            [idpf]             [k] idpf_vport_splitq_napi_poll
+    7,04%  :8076            [kernel.vmlinux]   [k] tcp_recvmsg_dmabuf
+    6,52%  :8076            [kernel.vmlinux]   [k] tcp_gro_receive
+    6,40%  :8076            [kernel.vmlinux]   [k] gen_pool_alloc_algo_owner
+    4,25%  :8076            [kernel.vmlinux]   [k] put_cmsg
+    3,65%  :8076            [kernel.vmlinux]   [k] xas_store
+    3,27%  :8076            [kernel.vmlinux]   [k] napi_pp_put_page
+    2,34%  :8076            [kernel.vmlinux]   [k] xas_find_marked
+    2,28%  :8076            [kernel.vmlinux]   [k] xas_load
+    1,75%  :8076            [kernel.vmlinux]   [k] _raw_spin_lock_bh
+    1,65%  :8076            [kernel.vmlinux]   [k] check_preemption_disabled
+    1,56%  :8076            [kernel.vmlinux]   [k] dev_gro_receive
+    1,30%  :8076            [kernel.vmlinux]   [k] xas_clear_mark
+    1,26%  :8076            [kernel.vmlinux]   [k] net_is_devmem_iov
+    1,20%  :8076            [kernel.vmlinux]   [k] page_pool_alloc_frag_netmem
+    1,09%  :8076            [kernel.vmlinux]   [k] inet_gro_receive
+    1,00%  :8076            [kernel.vmlinux]   [k] skb_gro_receive
+    0,89%  :8076            [kernel.vmlinux]   [k] __napi_build_skb
+    0,88%  :8076            [kernel.vmlinux]   [k] eth_type_trans
+    0,85%  :8076            [kernel.vmlinux]   [k] page_pool_put_unrefed_netmem
+    0,74%  :8076            [kernel.vmlinux]   [k] napi_build_skb
```

Devmem - 1 CPU - HW GRO (131.25 Gbps)

```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 93160176214
  Overhead  Command          Shared Object      Symbol
+    9,35%  :8892            [kernel.vmlinux]   [k] tcp_recvmsg_dmabuf
+    7,52%  :8892            [kernel.vmlinux]   [k] put_cmsg
+    6,57%  :8892            [kernel.vmlinux]   [k] xas_store
+    4,74%  :8892            [idpf]             [k] idpf_vport_splitq_napi_poll
+    4,49%  :8892            [kernel.vmlinux]   [k] xas_find_marked
+    3,97%  :8892            [kernel.vmlinux]   [k] xas_load
+    3,81%  :8892            [kernel.vmlinux]   [k] napi_pp_put_page
+    3,30%  :8892            [libeth]           [k] libeth_rx_recycle_slow
+    2,95%  :8892            [kernel.vmlinux]   [k] _raw_spin_lock_bh
+    2,27%  :8892            [kernel.vmlinux]   [k] page_pool_alloc_frag_netmem
+    2,26%  :8892            [kernel.vmlinux]   [k] xas_clear_mark
+    1,74%  :8892            [kernel.vmlinux]   [k] check_preemption_disabled
+    1,42%  :8892            [kernel.vmlinux]   [k] net_is_devmem_iov
+    1,42%  :8892            [kernel.vmlinux]   [k] __xa_cmpxchg
+    1,26%  :8892            [kernel.vmlinux]   [k] xas_start
+    1,24%  :8892            [kernel.vmlinux]   [k] page_pool_put_unrefed_netmem
+    1,05%  :8892            [kernel.vmlinux]   [k] preempt_count_add
+    1,01%  :8892            [kernel.vmlinux]   [k] tcp_gro_receive
+    0,96%  :8892            [kernel.vmlinux]   [k] xas_create
+    0,94%  :8892            [kernel.vmlinux]   [k] __xa_alloc
+    0,89%  :8892            [kernel.vmlinux]   [k] xas_set_mark
+    0,86%  :8892            [kernel.vmlinux]   [k] __local_bh_enable_ip
+    0,82%  :8892            [kernel.vmlinux]   [k] page_pool_refill_alloc_cache
```

The IDPF HW GRO Dragon …

HDS implies two page pools: **Header** and payload pool

- Simple lifecycle management: 1-1 mapping
  - IDPF uses 1 page per header page per payload
- Unfortunately when you have HW GRO, you only need 1 header for X payloads..

For sake of discussion, say we receive a GRO size of 16(*4K payload)

- it means only the first header is relevant i.e the others are dummies
  - *libeth_rx_recycle_slow()* recycles these dummy (15) header pages back to the header page pool at softirq context
    - *foreach page {hdr producer lock, recycle, release hdr producer lock}*
    - The page with header is only recycled when *recvmsg()* completes and *skb_release* kicks in at user context (contends for hdr producer lock)*
    - In the meantime more and more GRO'ed packets are coming in and contend for that same lock..

Slaying The IDPF HW GRO Dragon …

- We did not have time to slay this dragon

Potential approach:

- Keep single page for X headers and use refcounts
- For IPv4, no more than 64B is really needed
  - so 4096/64 should be much higher than max GRO size…

The Payload Recycle Dragon

On single CPU we observed that the cost of setsockopt was higher both in the syscall trace as well as in perf

➔ Recycling of the payload happens with setsockopt

After looking at the kernel code, we observed that although the uapi allows for a batch of buffers from user space to be sent to the kernel, the kernel would still do:

foreach page {grab payload producer lock, recycle, release payload producer lock}

We made two changes:

1. Application collects as many as X consumed buffers for recycling (max 1024)
2. Changed the kernel code to amortize the cost of the lock:

grab payload producer lock,
for each page: recycle
release payload producer lock

Devmem RX Fix 1: App buffer recycle batching

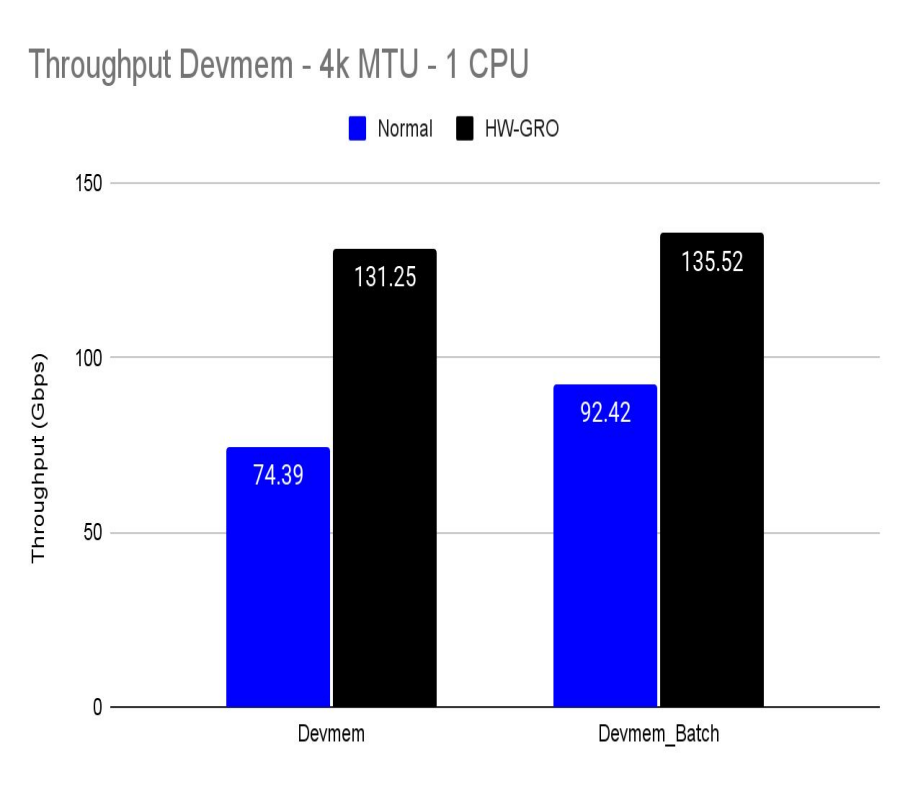
- The kernel allows up to 128 token containers with a max of up to 1024 tokens to free at once
  - The common case is only 1 token container with up to 1024 tokens
- recvmsg() will likely receive more than one token per call, up to the provided control space
  - The number of tokens will vary depending on the bandwidth and tcp buffer conditions
  - At 200G, our average token was at ~875 per recvmsg
- In our tests we accumulate small recvmsgs up to 512 tokens and then issue a setsockopt to release them
  - It reduces our setsockopt usage by 15% but did not show visible improvements on throughput a cpu utilization

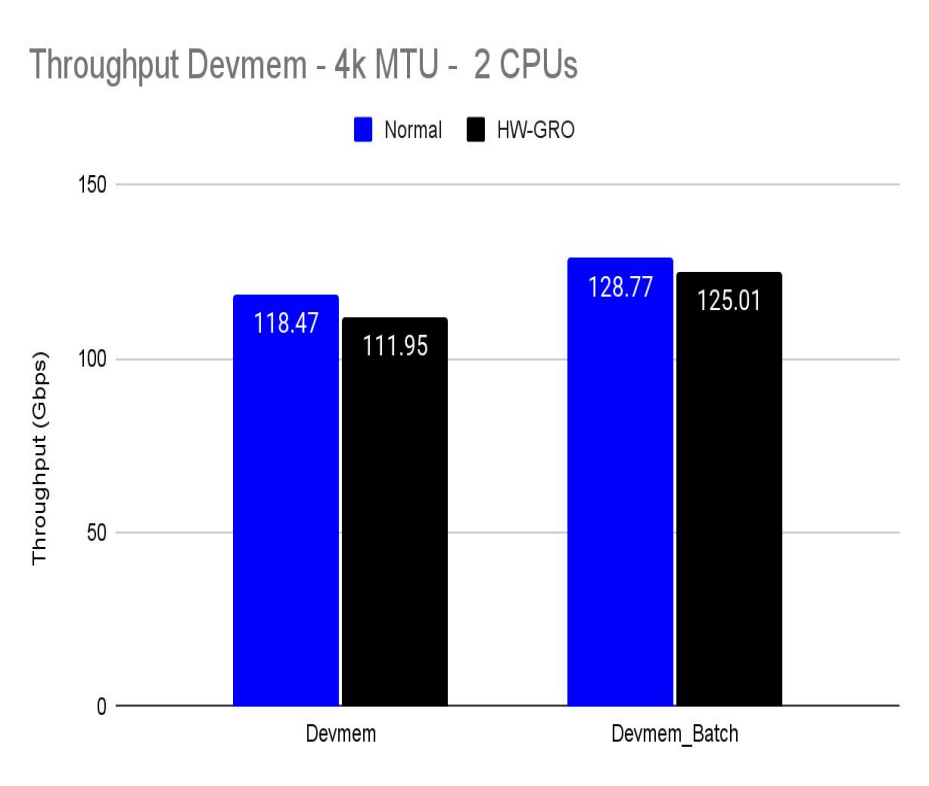
Devmem recv read size distribution

--- System Call Analysis ---
Number of recvmsg calls: 253592
Number of setsockopt calls: 315547
--- Frags Analysis ---
Average frags per recvmsg: 874.644
Minimum frags received: 1
Maximum frags received: 1499
Number of times accumulated frags > 1024: 111714
Number of recvmsg with frags > 1024: 97418
Number of recvmsg with frags >= 512 && < 1024: 102476
Number of recvmsg with frags < 512: 53698
Number of setsockopt avoided (accumulated up to 512): 42242
Percentage >1024/recvmsg: 38.42%
Percentage >1024/acc(>1024): 87.20%
Percentage <512/recvmsg: 21.17%

On average we are getting more than 512 pages per recvmsg

Throughput Devmem - (Batch vs No Batch)





Syscall Costs Before Batching

2 CPUs

```
:14342 (14342), 4745666 events, 100.0%

  syscall            calls  errors  total       min       avg       max       stddev
                                    (msec)    (msec)    (msec)    (msec)        (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  recvmsg            615359    1830 68831.493     0.001     0.112     0.371      0.03%
  setsockopt         530116       0 22841.143     0.001     0.043     0.284      0.03%
  ioctl             1224901       0  1232.993     0.001     0.001     0.604      0.09%
  epoll_wait           1835       0   628.492     0.000     0.343   100.162     28.36%
```

Syscall Costs After Batching

2 CPUs

```
:4139 (4139), 5904400 events, 100.0%

  syscall            calls  errors  total       min       avg       max       stddev
                                    (msec)    (msec)    (msec)    (msec)        (%)
  --------------- --------  ------ -------- --------- --------- ---------     ------
  recvmsg            809214    2595 76170.078     0.001     0.094     0.272      0.03%
  setsockopt         530128       0 14209.108     0.001     0.027     0.214      0.03%
  ioctl             1609592       0  1544.356     0.001     0.001     0.643      0.07%
  epoll_wait           2599       0   725.770     0.001     0.279   100.171     28.20%
```

Syscall Costs Before Batching

1 CPU

```
:3525 (3525), 1817743 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- -------- ------ -------- --------- --------- ---------     ------
  setsockopt        294394      0 60365.027     0.001     0.205   324.928      0.98%
  recvmsg           206570   4885 30241.574     0.001     0.146   155.632      0.54%
  ioctl             403339      0  1305.559     0.001     0.003    21.414      6.40%
  epoll_wait          4889      0  1095.391     0.001     0.224   100.170     19.81%
```

Syscall Costs After Batching

1 CPU

```
tcpzc2 (6557), 3564035 events, 100.0%

  syscall            calls  errors  total       min       avg       max      stddev
                                    (msec)    (msec)    (msec)    (msec)       (%)
  --------------- -------- ------ -------- --------- --------- ---------     ------
  recvmsg           486578  24534 55429.204     0.001     0.114   148.957      0.37%
  setsockopt        347744      0 27035.294     0.001     0.078   210.768      1.68%
  epoll_wait         24541      0  3486.672     0.000     0.142   100.172      5.29%
  ioctl             923799      0  2447.129     0.001     0.003    26.322      2.67%
```

Perf Devmem (Batch x No Batch)

Devmem - 2 CPUs - Batch (128.77 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 179161519297
  Overhead  Command          Shared Object     Symbol
+   19,10%  :57661           [kernel.vmlinux]  [k] tcp_recvmsg_dmabuf
+    6,10%  swapper          [idpf]            [k] idpf_vport_splitq_napi_poll
+    3,97%  swapper          [kernel.vmlinux]  [k] napi_pp_put_page
+    3,86%  :57661           [kernel.vmlinux]  [k] put_cmsg
+    3,42%  swapper          [kernel.vmlinux]  [k] tcp_gro_receive
+    3,32%  ksoftirqd/0      [idpf]            [k] idpf_vport_splitq_napi_poll
+    3,03%  :57661           [kernel.vmlinux]  [k] xas_store
+    3,00%  :57661           [kernel.vmlinux]  [k] page_pool_put_netmem_bulk
+    2,22%  :57661           [kernel.vmlinux]  [k] xas_find_marked
+    2,05%  ksoftirqd/0      [kernel.vmlinux]  [k] napi_pp_put_page
+    1,92%  :57661           [kernel.vmlinux]  [k] xas_load
+    1,87%  ksoftirqd/0      [kernel.vmlinux]  [k] tcp_gro_receive
+    1,48%  :57661           [kernel.vmlinux]  [k] tcp_recvmsg_locked
+    1,15%  :57661           [kernel.vmlinux]  [k] tcp_rcv_established
+    1,10%  swapper          [kernel.vmlinux]  [k] napi_build_skb
+    1,07%  :57661           [kernel.vmlinux]  [k] xas_clear_mark
+    0,98%  swapper          [kernel.vmlinux]  [k] dev_gro_receive
+    0,90%  swapper          [kernel.vmlinux]  [k] skb_gro_receive
+    0,90%  swapper          [kernel.vmlinux]  [k] skb_release_data
+    0,90%  swapper          [kernel.vmlinux]  [k] __inet_lookup_established
+    0,90%  :57661           [kernel.vmlinux]  [k] sock_rfree
+    0,86%  :57661           [kernel.vmlinux]  [k] skb_try_coalesce
+    0,80%  :57661           [kernel.vmlinux]  [k] tcp_v4_do_rcv
Tip: Save output of perf stat using: perf stat record <target workload>
```

Devmem - 2 CPUs (118.47 Gbps)

```
Samples: 199K of event 'cpu-cycles', Event count (approx.): 168906025843
  Overhead  Command          Shared Object     Symbol
+   18,01%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_dmabuf
+    8,54%  swapper          [idpf]             [k] idpf_vport_splitq_napi_poll
+    6,29%  :18102           [kernel.kallsyms]  [k] napi_pp_put_page
+    5,71%  swapper          [kernel.kallsyms]  [k] tcp_gro_receive
+    5,21%  swapper          [kernel.kallsyms]  [k] napi_pp_put_page
+    3,63%  :18102           [kernel.kallsyms]  [k] put_cmsg
+    2,93%  :18102           [kernel.kallsyms]  [k] xas_store
+    2,12%  :18102           [kernel.kallsyms]  [k] xas_find_marked
+    1,82%  :18102           [kernel.kallsyms]  [k] xas_load
+    1,68%  :18102           [kernel.kallsyms]  [k] _raw_spin_lock_bh
+    1,53%  swapper          [kernel.kallsyms]  [k] napi_build_skb
+    1,32%  swapper          [kernel.kallsyms]  [k] tcp_v4_rcv
+    1,31%  swapper          [kernel.kallsyms]  [k] dev_gro_receive
+    1,26%  swapper          [kernel.kallsyms]  [k] skb_release_data
+    1,24%  :18102           [kernel.kallsyms]  [k] tcp_recvmsg_locked
+    1,12%  :18102           [kernel.kallsyms]  [k] xas_clear_mark
+    1,11%  :18102           [kernel.kallsyms]  [k] tcp_rcv_established
+    1,09%  swapper          [kernel.kallsyms]  [k] skb_gro_receive
+    0,87%  swapper          [kernel.kallsyms]  [k] __inet_lookup_established
+    0,85%  :18102           [kernel.kallsyms]  [k] tcp_rcv_space_adjust
+    0,84%  swapper          [kernel.kallsyms]  [k] __napi_build_skb
+    0,82%  swapper          [kernel.kallsyms]  [k] intel_idle
+    0,82%  :18102           [kernel.kallsyms]  [k] skb_try_coalesce
Tip: If you have debuginfo enabled, try: perf report -s sym,srcline
```

Devmem - 1 CPU - Batch (92.42 Gbps)

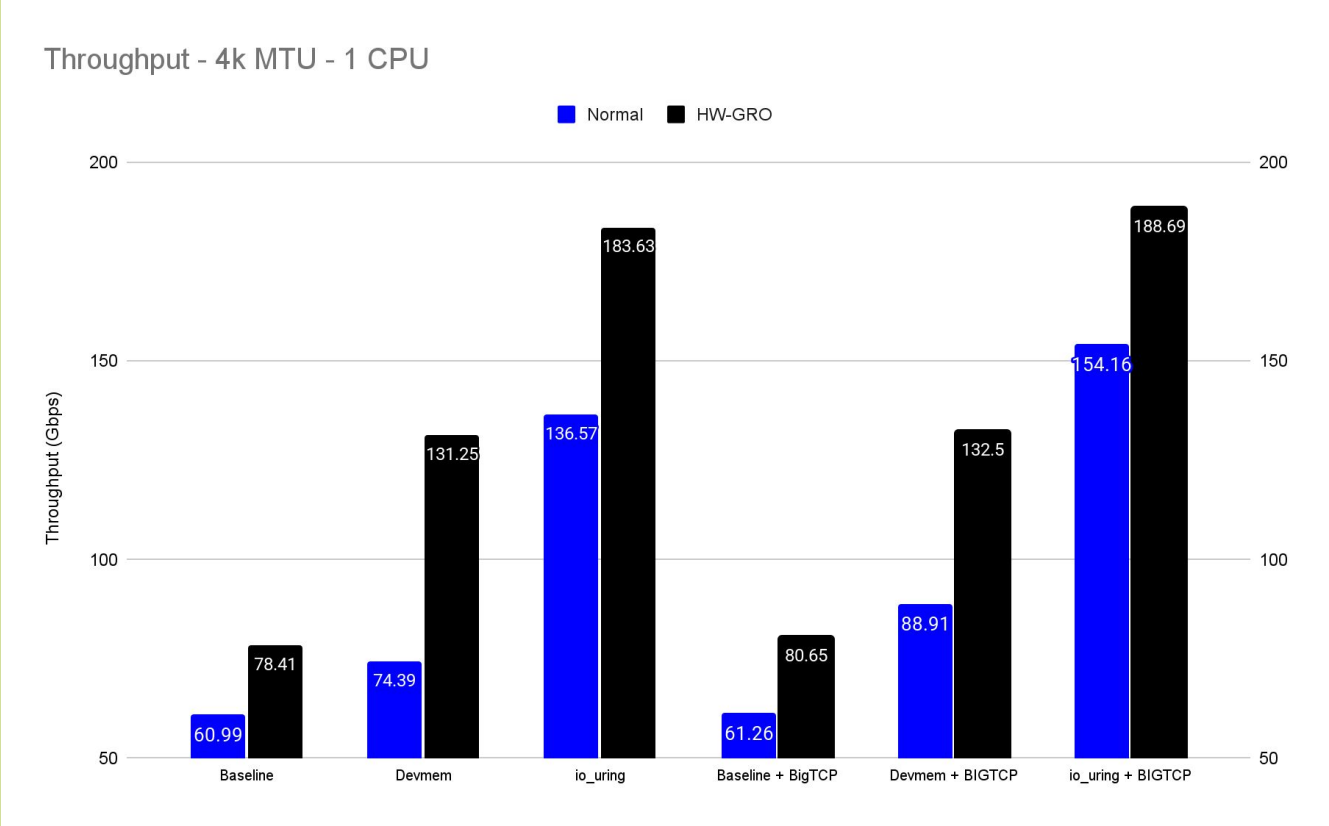
```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 92873468822
  Overhead  Command          Shared Object     Symbol
+    8,32%  :52790           [kernel.vmlinux]  [k] tcp_recvmsg_dmabuf
+    8,03%  :52790           [idpf]            [k] idpf_vport_splitq_napi_poll
+    5,66%  :52790           [kernel.vmlinux]  [k] tcp_gro_receive
+    5,33%  :52790           [kernel.vmlinux]  [k] put_cmsg
+    4,36%  :52790           [kernel.vmlinux]  [k] xas_store
+    3,23%  :52790           [kernel.vmlinux]  [k] napi_pp_put_page
+    2,98%  :52790           [kernel.vmlinux]  [k] xas_find_marked
+    2,97%  :52790           [kernel.vmlinux]  [k] gen_pool_free_owner
+    2,94%  :52790           [kernel.vmlinux]  [k] gen_pool_has_addr
+    2,85%  :52790           [kernel.vmlinux]  [k] xas_load
+    2,10%  :52790           [kernel.vmlinux]  [k] gen_pool_alloc_algo_owner
+    1,85%  :52790           [kernel.vmlinux]  [k] check_preemption_disabled
+    1,62%  :52790           [kernel.vmlinux]  [k] dev_gro_receive
+    1,50%  :52790           [kernel.vmlinux]  [k] xas_clear_mark
+    1,42%  :52790           [kernel.vmlinux]  [k] page_pool_put_netmem_bulk
+    1,36%  :52790           [kernel.vmlinux]  [k] page_pool_alloc_frag_netmem
+    1,32%  :52790           [kernel.vmlinux]  [k] net_is_devmem_iov
+    1,23%  ksoftirqd/0      [idpf]            [k] idpf_vport_splitq_napi_poll
+    1,08%  :52790           [kernel.vmlinux]  [k] skb_gro_receive
+    1,06%  :52790           [kernel.vmlinux]  [k] __xa_cmpxchg
+    1,02%  :52790           [kernel.vmlinux]  [k] __napi_build_skb
+    0,97%  :52790           [kernel.vmlinux]  [k] inet_gro_receive
+    0,96%  :52790           [kernel.vmlinux]  [k] eth_type_trans
```

Devmem - 1 CPU (74.39 Gbps)

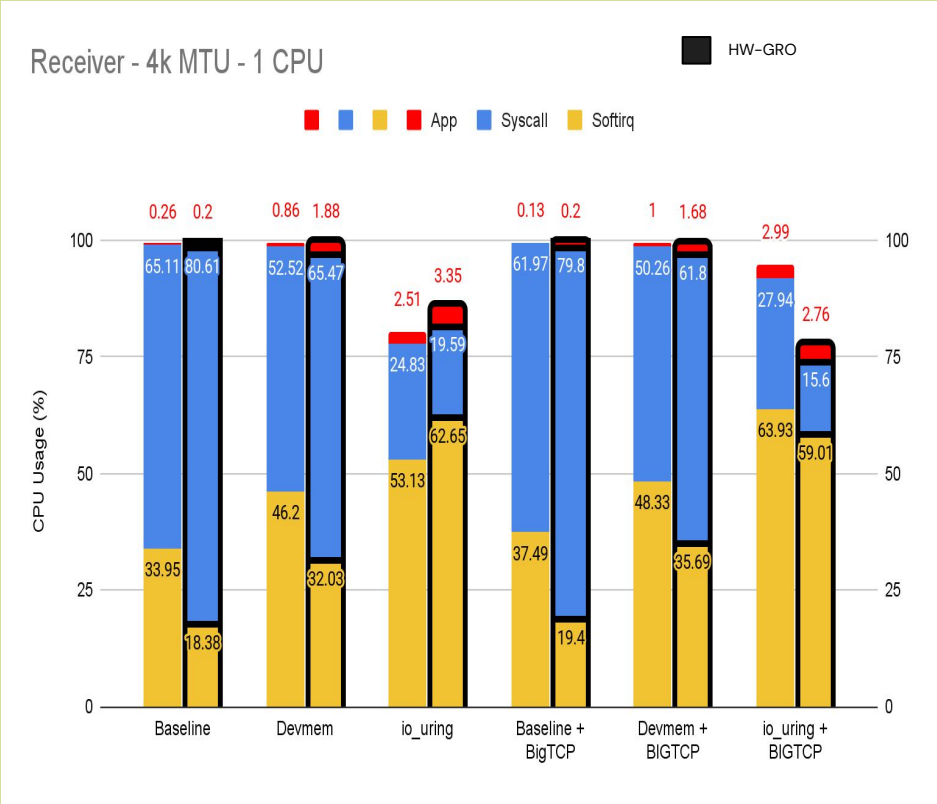
```
Samples: 100K of event 'cpu-cycles', Event count (approx.): 93207866858
  Overhead  Command          Shared Object     Symbol
+    8,02%  :8076            [kernel.vmlinux]  [k] gen_pool_has_addr
+    7,80%  :8076            [kernel.vmlinux]  [k] gen_pool_free_owner
+    7,70%  :8076            [idpf]            [k] idpf_vport_splitq_napi_poll
+    7,04%  :8076            [kernel.vmlinux]  [k] tcp_recvmsg_dmabuf
+    6,52%  :8076            [kernel.vmlinux]  [k] tcp_gro_receive
+    6,40%  :8076            [kernel.vmlinux]  [k] gen_pool_alloc_algo_owner
+    4,25%  :8076            [kernel.vmlinux]  [k] put_cmsg
+    3,65%  :8076            [kernel.vmlinux]  [k] xas_store
+    3,27%  :8076            [kernel.vmlinux]  [k] napi_pp_put_page
+    2,34%  :8076            [kernel.vmlinux]  [k] xas_find_marked
+    2,28%  :8076            [kernel.vmlinux]  [k] xas_load
+    1,75%  :8076            [kernel.vmlinux]  [k] _raw_spin_lock_bh
+    1,65%  :8076            [kernel.vmlinux]  [k] check_preemption_disabled
+    1,56%  :8076            [kernel.vmlinux]  [k] dev_gro_receive
+    1,30%  :8076            [kernel.vmlinux]  [k] xas_clear_mark
+    1,26%  :8076            [kernel.vmlinux]  [k] net_is_devmem_iov
+    1,20%  :8076            [kernel.vmlinux]  [k] page_pool_alloc_frag_netmem
+    1,09%  :8076            [kernel.vmlinux]  [k] inet_gro_receive
+    1,00%  :8076            [kernel.vmlinux]  [k] skb_gro_receive
+    0,89%  :8076            [kernel.vmlinux]  [k] __napi_build_skb
+    0,88%  :8076            [kernel.vmlinux]  [k] eth_type_trans
+    0,85%  :8076            [kernel.vmlinux]  [k] page_pool_put_unrefed_netmem
+    0,74%  :8076            [kernel.vmlinux]  [k] napi_build_skb
```

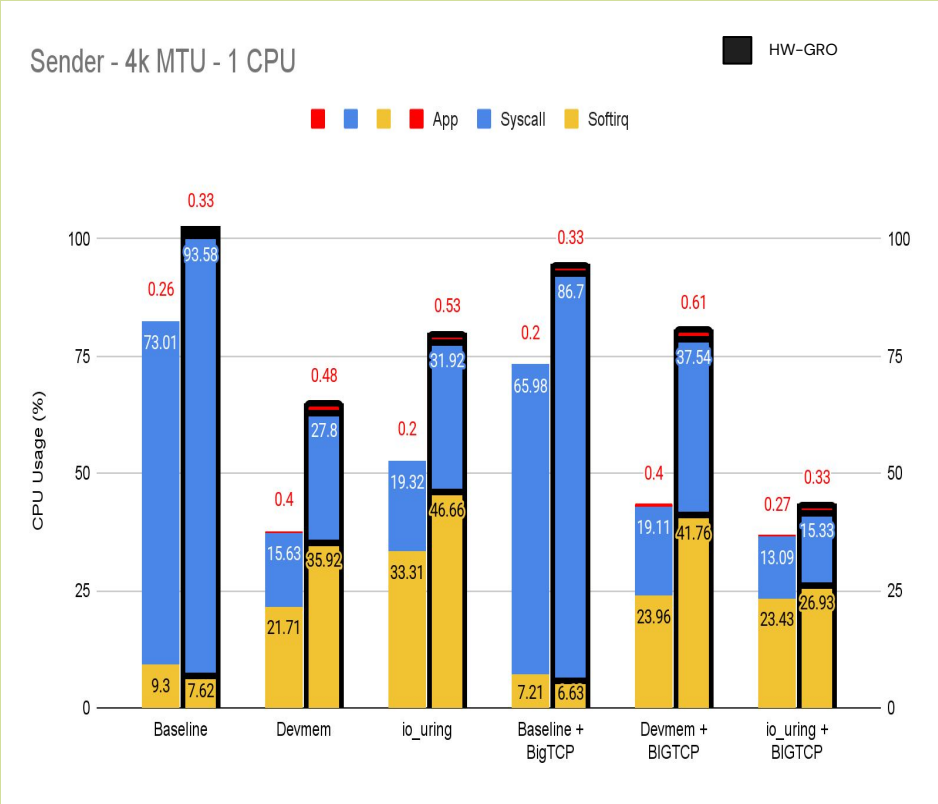
Results
Putting app + softirq on one core

Throughput - 1 CPU

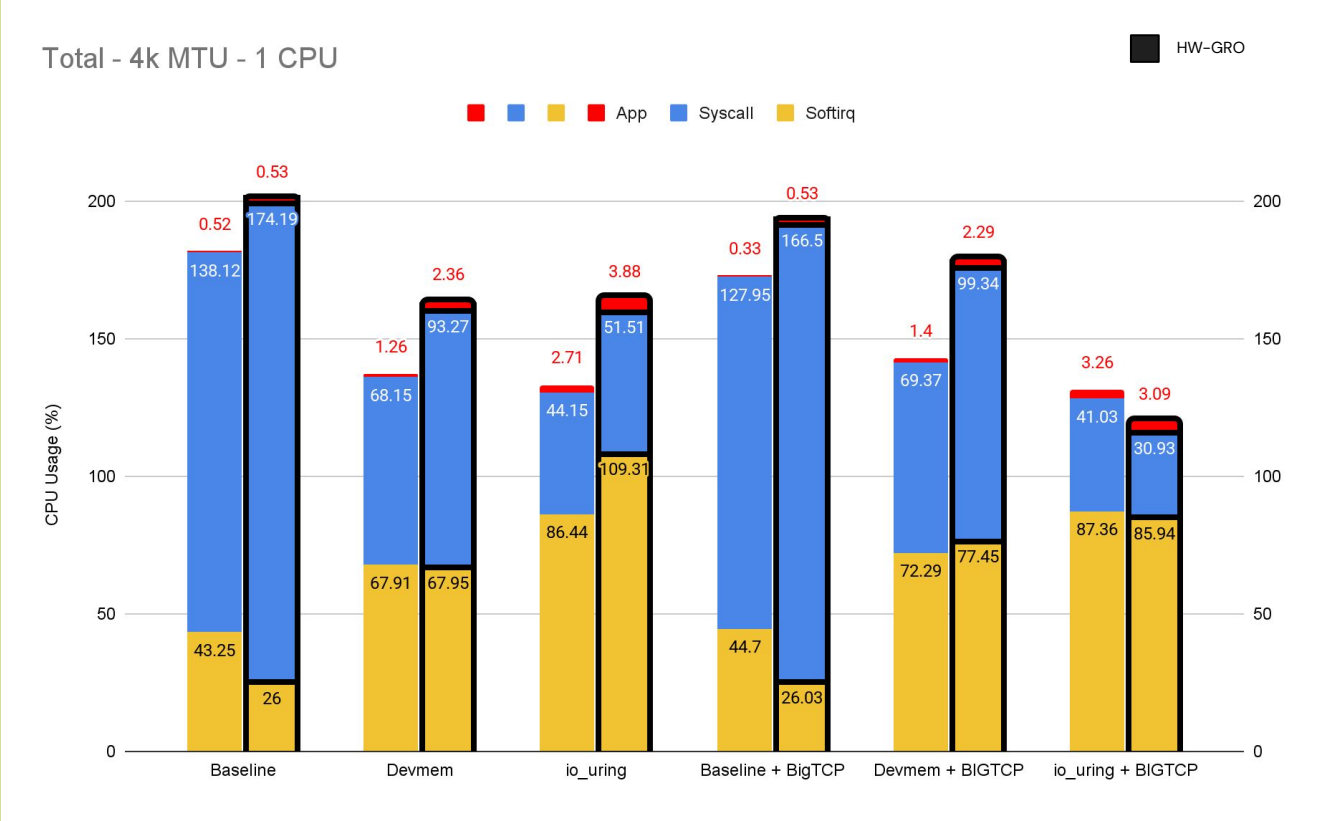


CPU Usage - 1 CPU - Receiver vs Sender

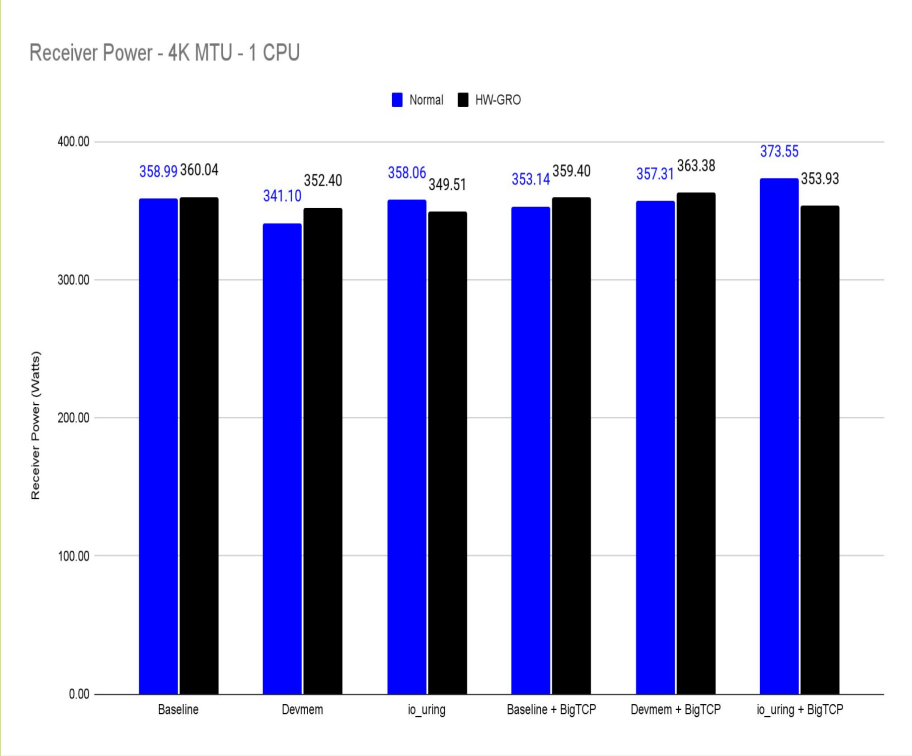


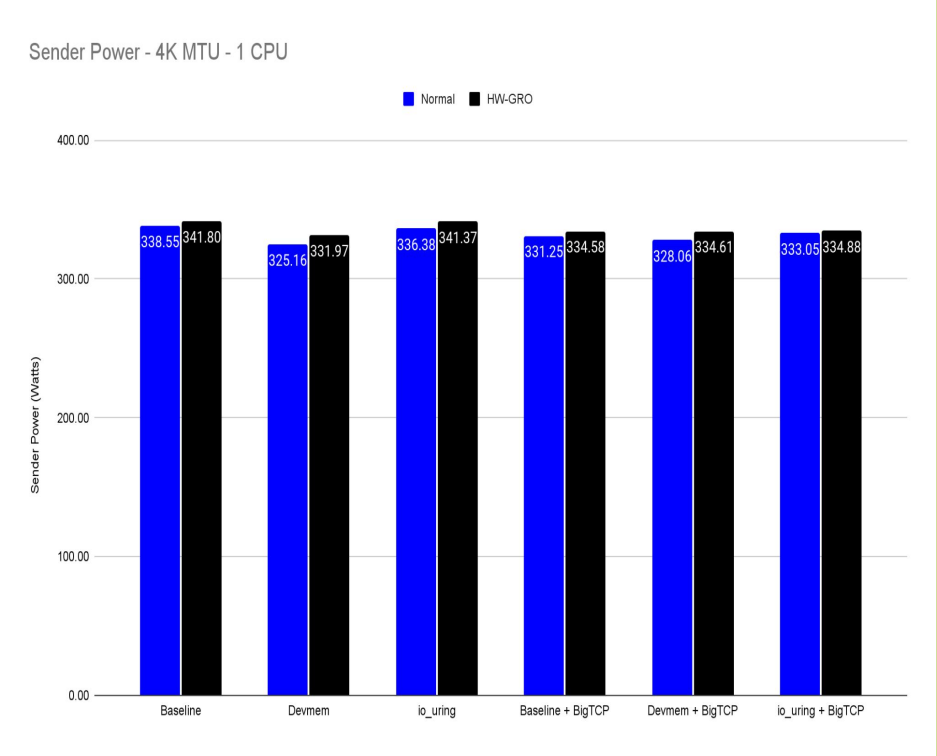


CPU Usage - 1 CPU

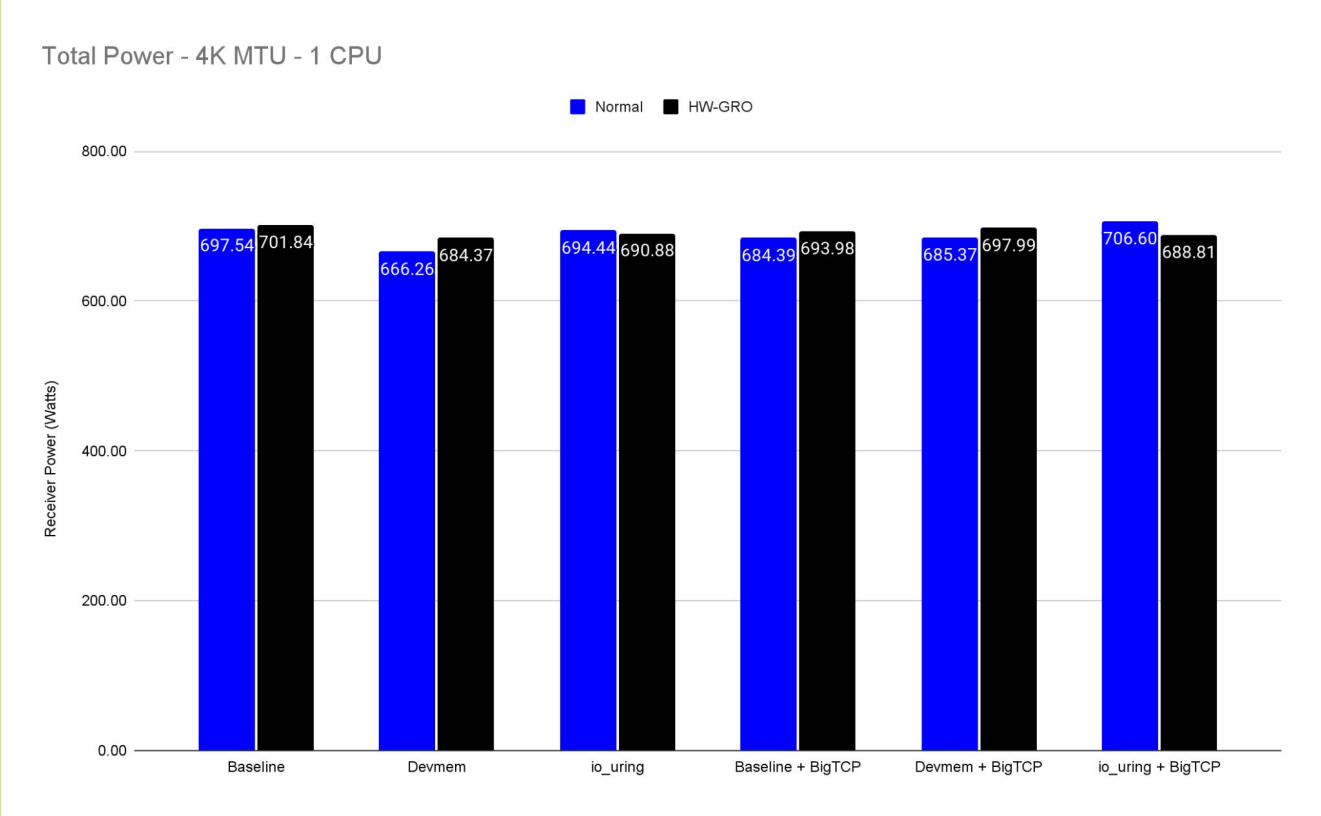


Power - 1 CPU


Receiver Power - 4K MTU - 1 CPU
Normal
HW-GRO
Receiver Power (Watts)
400.00
300.00
200.00
100.00
0.00
358.99
360.04
341.10
352.40
358.06
349.51
353.14
359.40
357.31
363.38
373.55
353.93
Baseline
Devmem
io_uring
Baseline + BigTCP
Devmem + BigTCP
io_uring + BigTCP


Sender Power - 4K MTU - 1 CPU
Normal
HW-GRO
Sender Power (Watts)
400.00
300.00
200.00
100.00
0.00
338.55
341.80
325.16
331.97
336.38
341.37
331.25
334.58
328.06
334.61
333.05
334.88
Baseline
Devmem
io_uring
Baseline + BigTCP
Devmem + BigTCP
io_uring + BigTCP

Power Total - 1 CPU



ROI Performance Metrics

When comparing different results for <u>throughput</u>, it is often hard to say which results gives you the best return on investmen(ROI). We came up with two ROI formulas:

$$\frac{T^n}{\sum C_i}$$

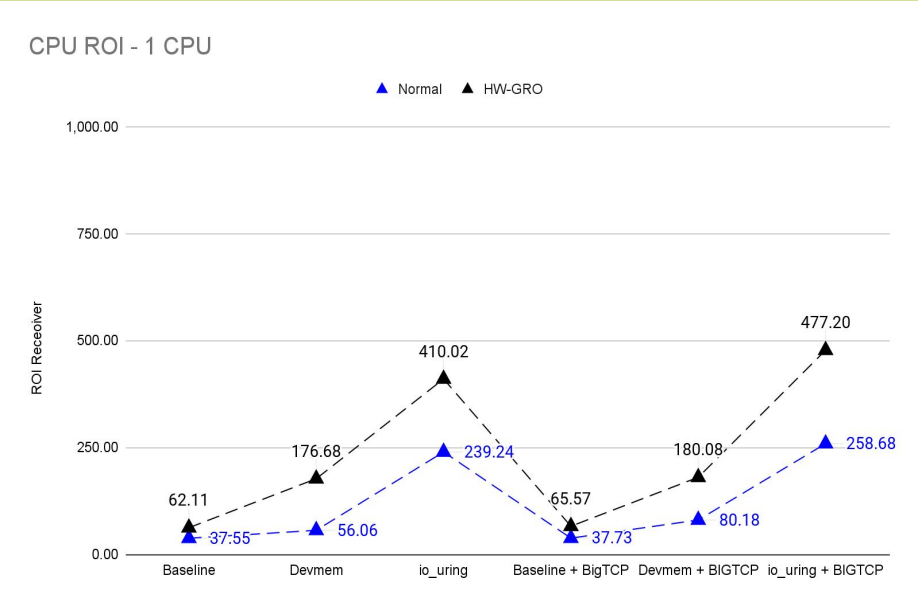
T=Throughput achieved in Gbp/s. n=2
C=compute cost to achieve the throughput. Summed across all CPUs

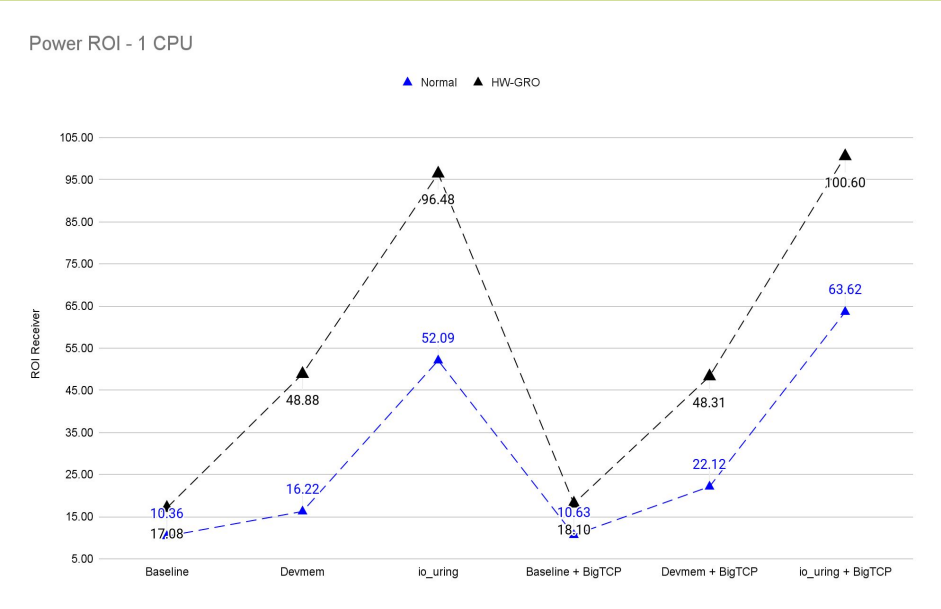
$$\frac{T^n}{P^m}$$

T=Throughput achieved in Gbp/s. n=2
P=Power consumed by in watts. m=1

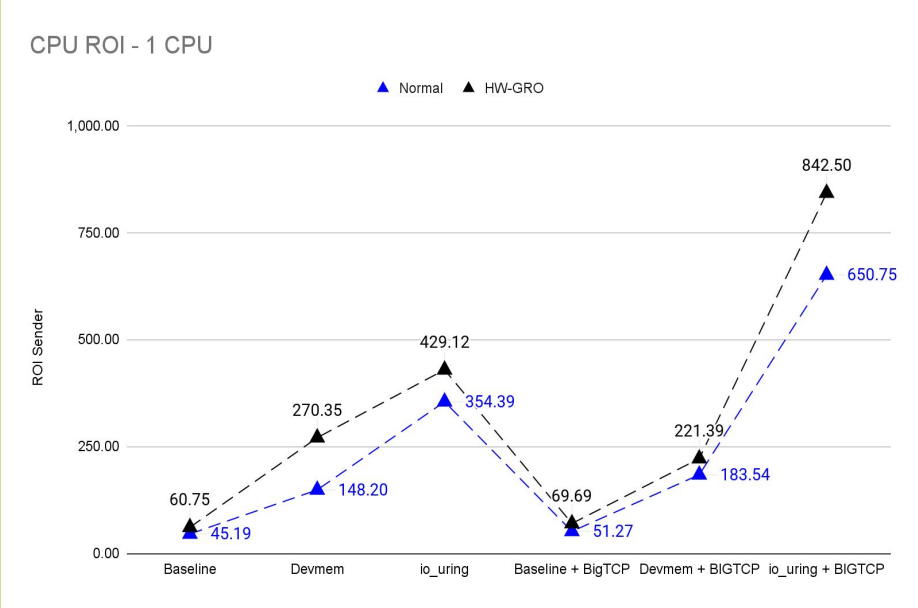
We pick n=2 to emphasize throughput as the goal. So 10 gbps using 200% cpu is considered to be better ROI than 5 gbps using 100% cpu

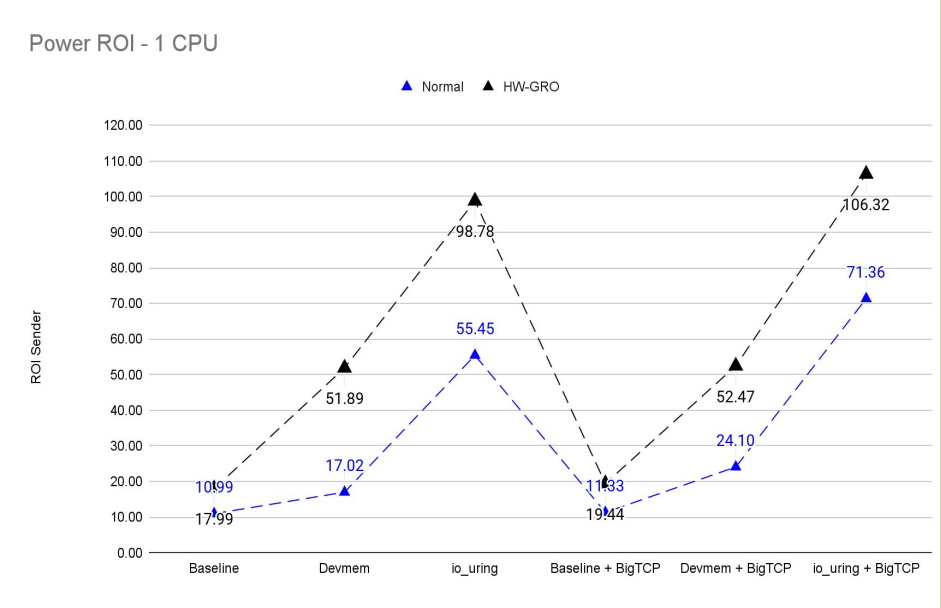
ROI Receiver - 1 CPU



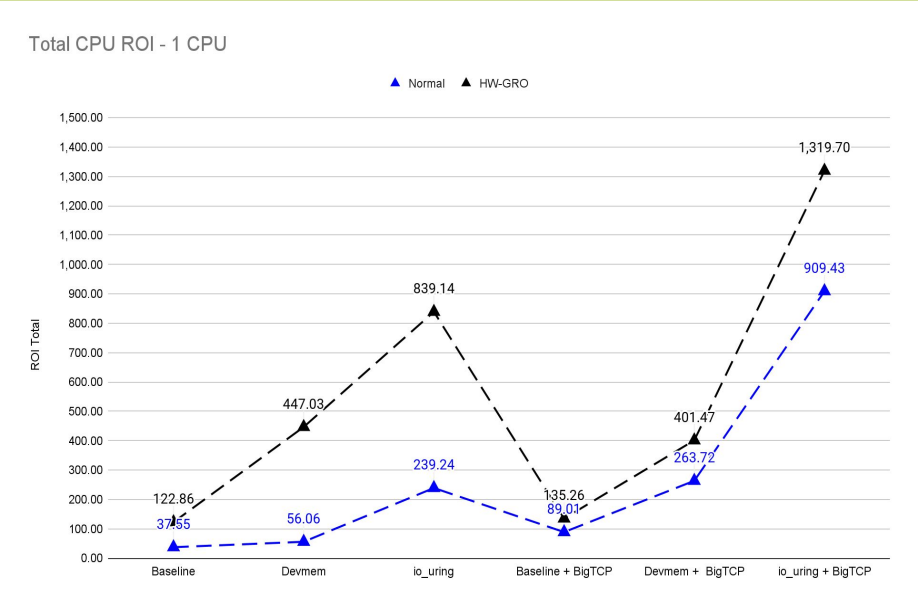


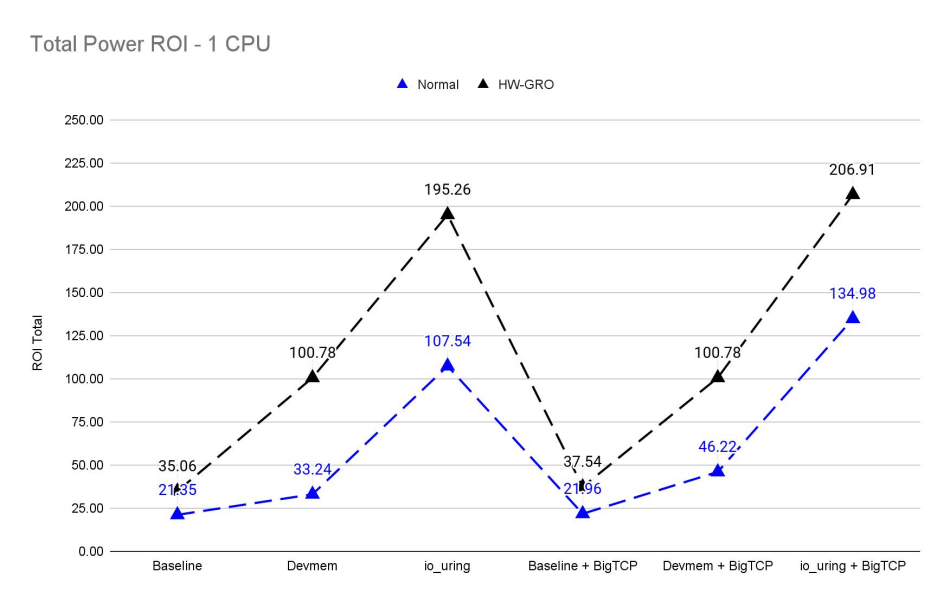
ROI Sender - 1 CPU





ROI Total - 1 CPU





Conclusions And Recommendations

- The two ZC interfaces are still fluid but nevertheless deliver on the message
- Devmem is syscall heavy while io_uring is kernel/softirq heavy
- io_uring demonstrates a clear advantage over devmem, however we need to test scaling to much higher rates (>200Gbps)
- Page pool API or perhaps driver use of the PP API could be improved
- HW GRO clearly provides a big advantage

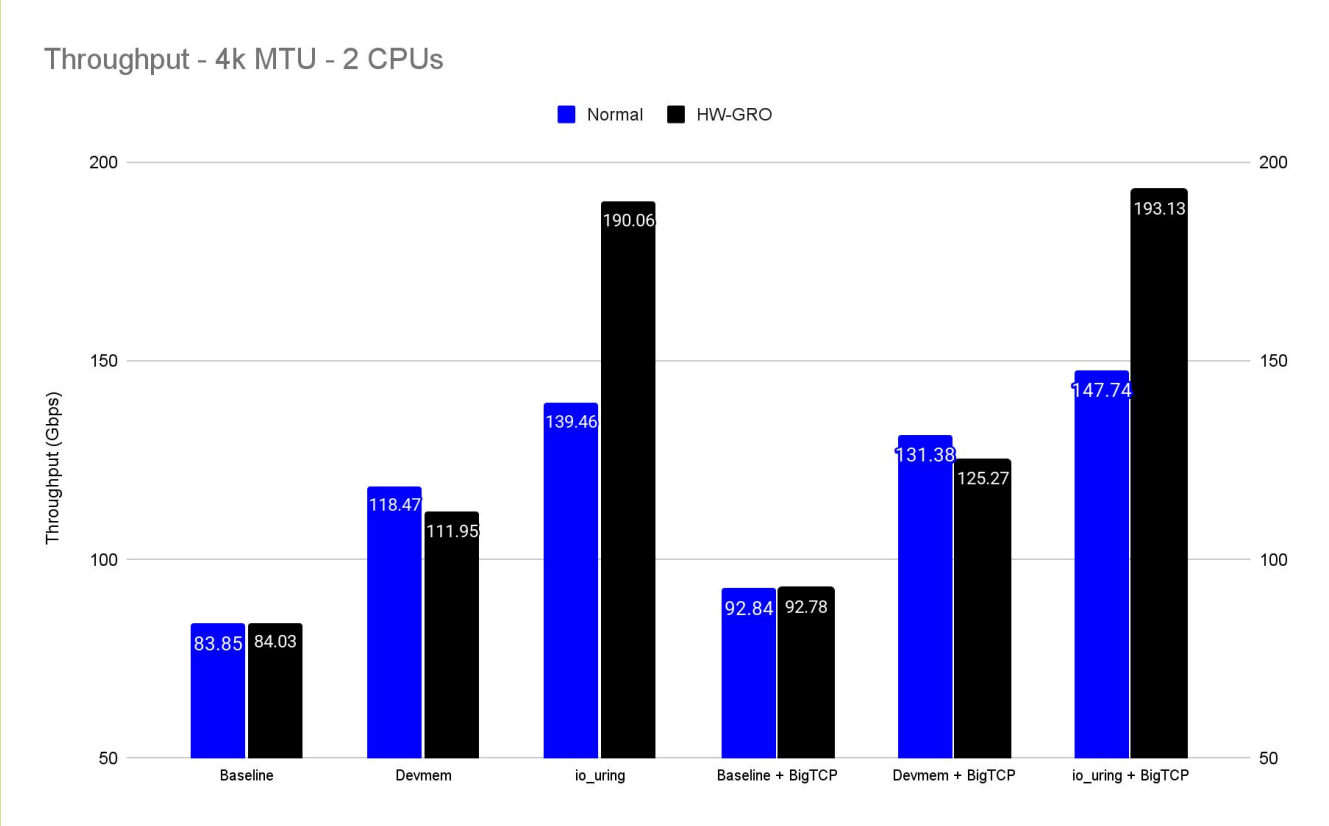
Acknowledgements

We could not have done it without their help and patience. Appreciated!

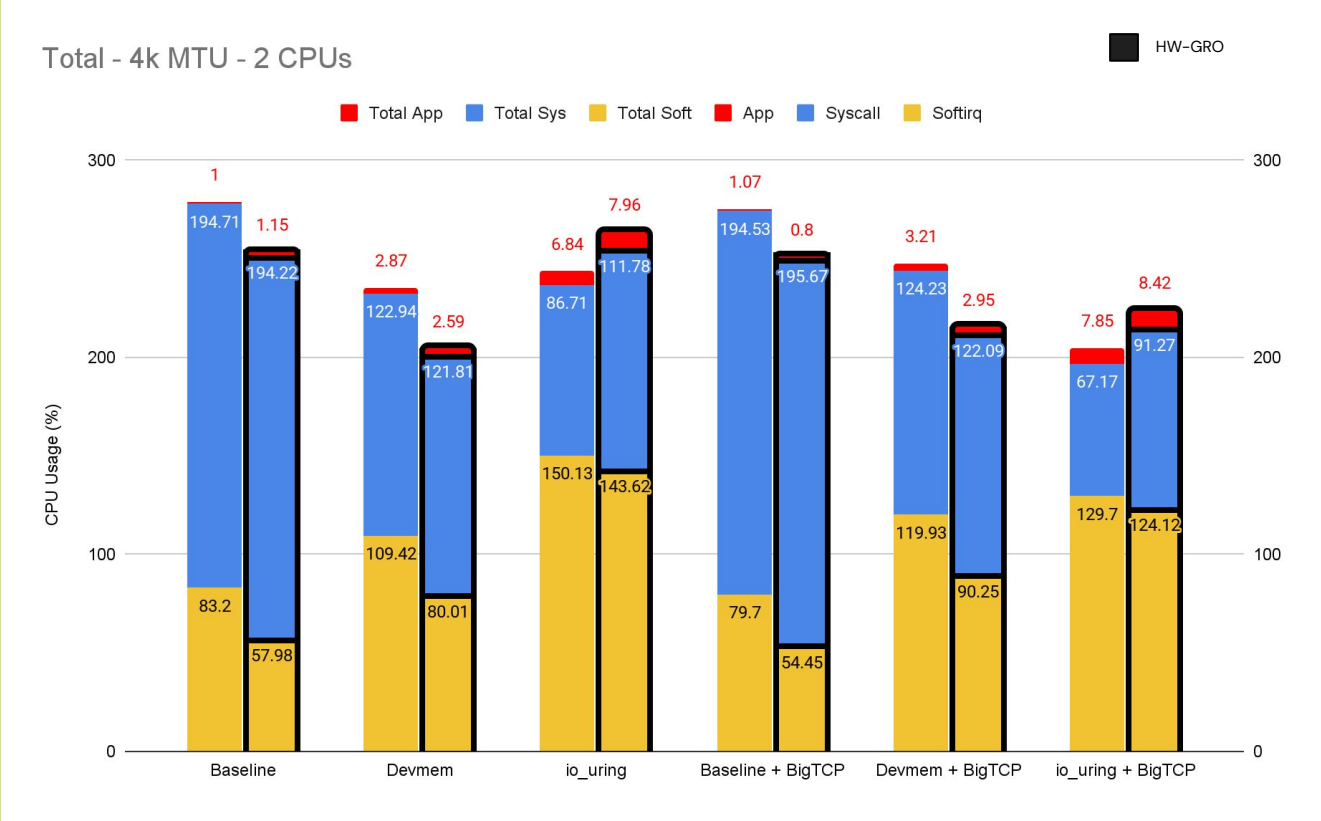
- Anjali S. Jain - Instrumental for anything Intel!
- Sridhar Samudrala for driver patches
- Mina Almasry for Devmem and guidance into gpud/nccl integration
- Pavel Bengukov for io_uring

Back Slides

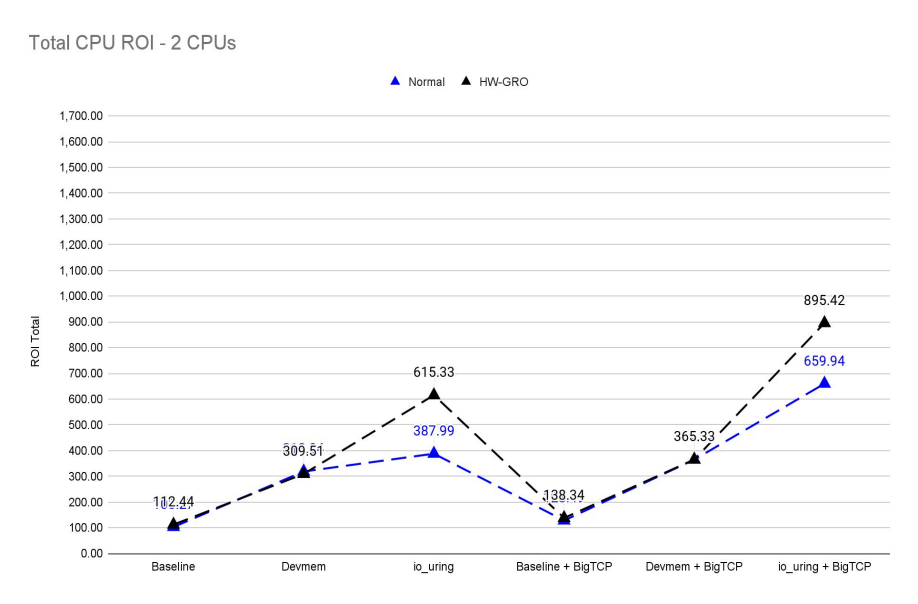
Throughput - 2 CPUs

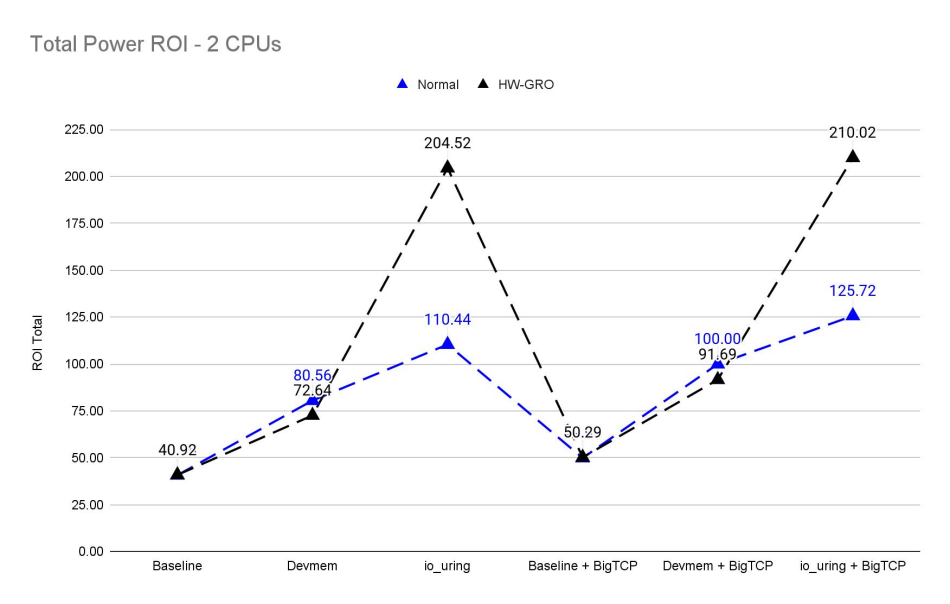


CPU Usage - 2 CPUs

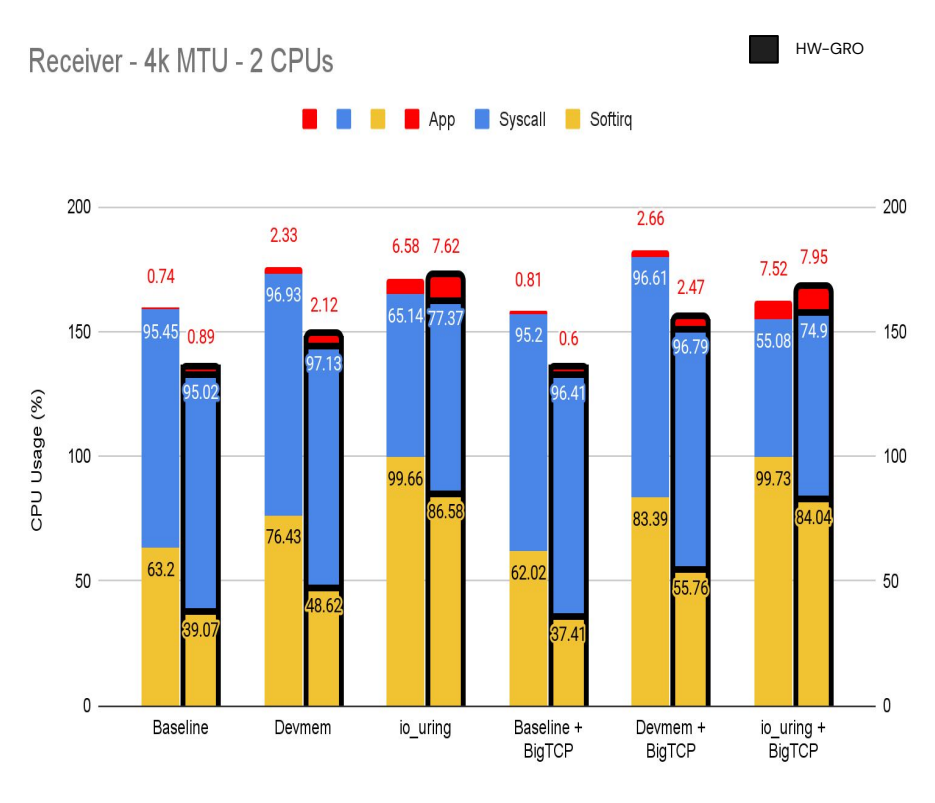


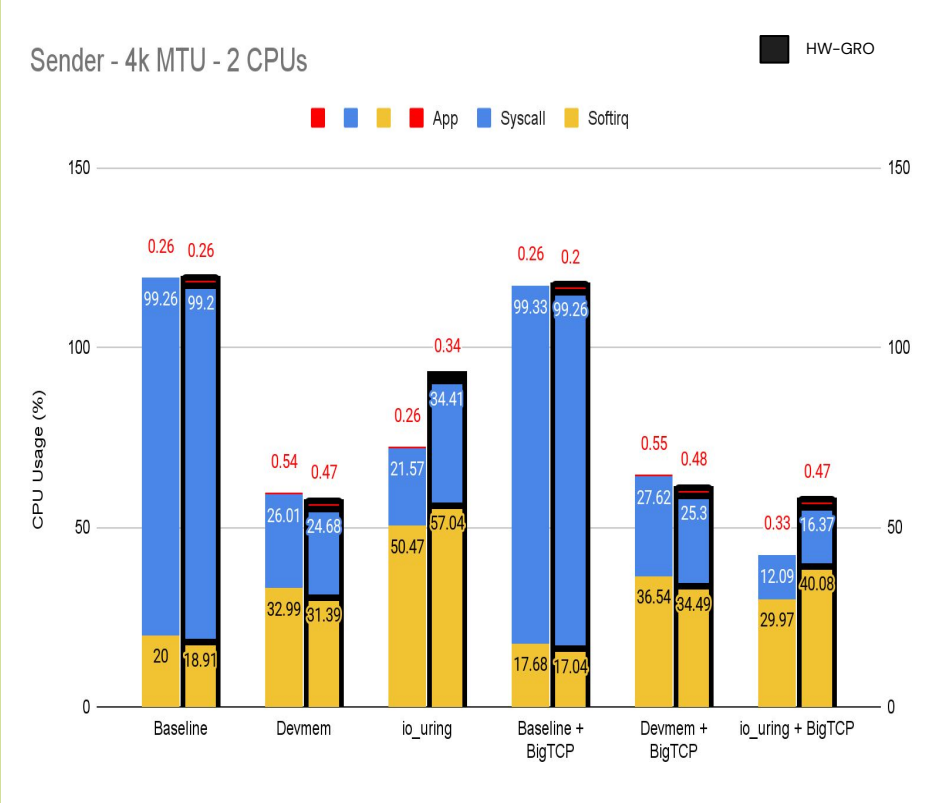
ROI Total - 2 CPUs



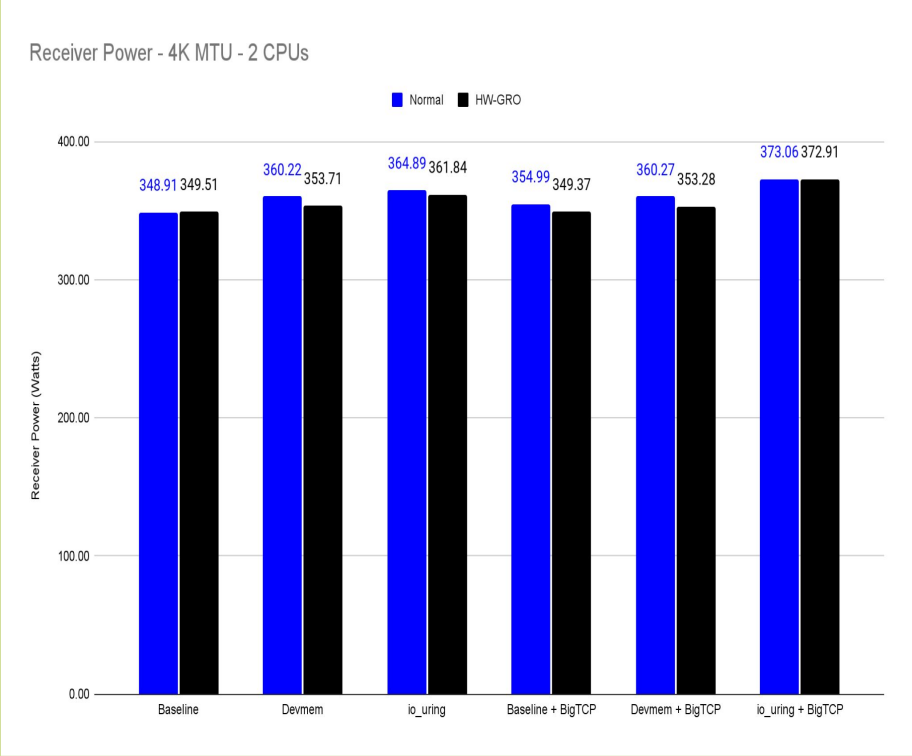


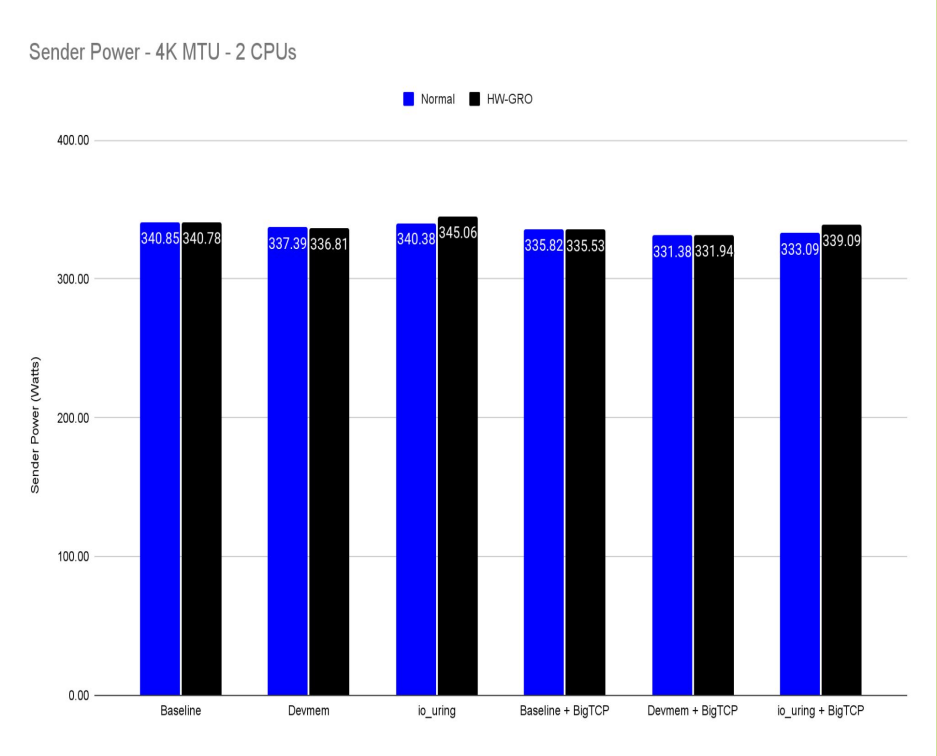
CPU Usage - 2 CPUs - Receiver vs Sender



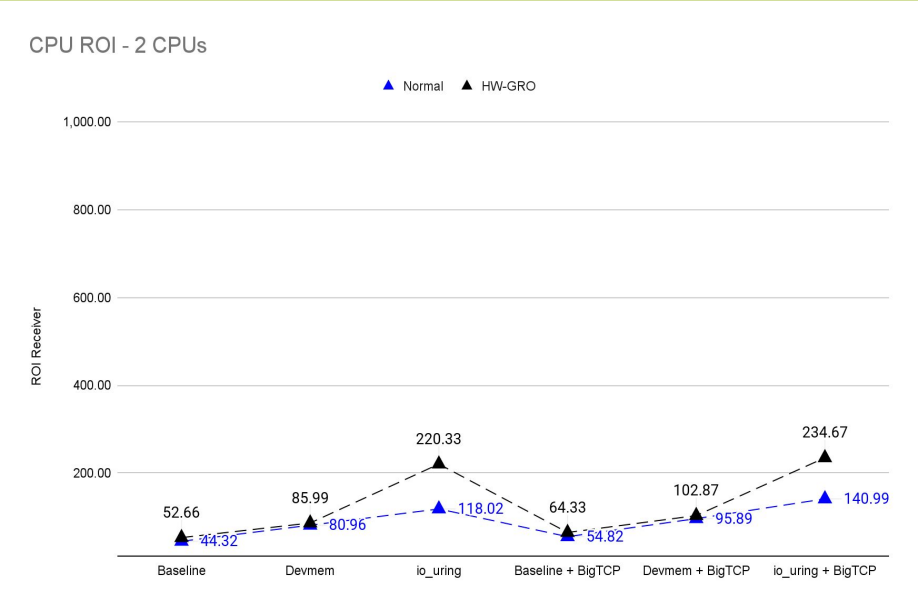


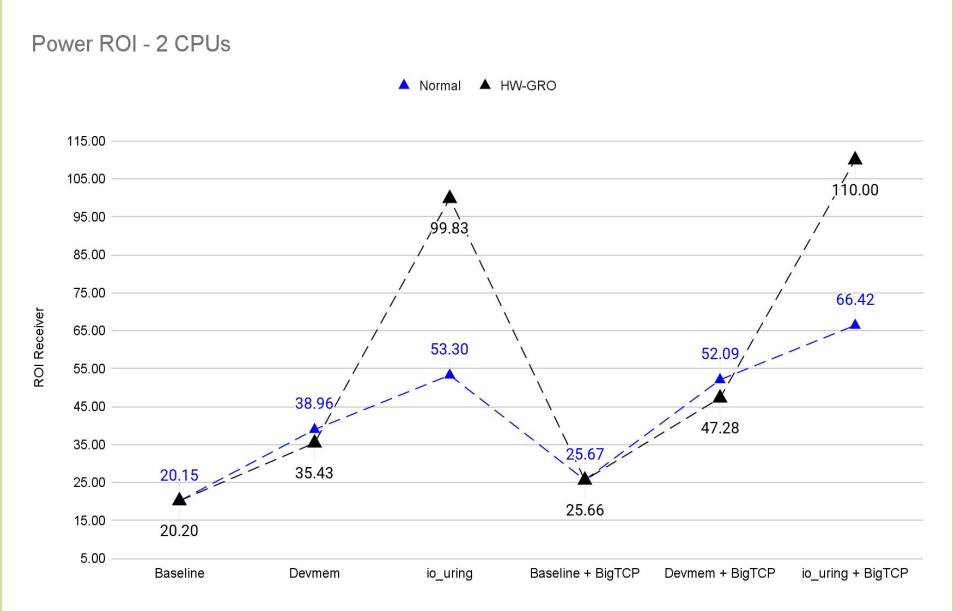
Power - 2 CPUs


Receiver Power - 4K MTU - 2 CPUs
Normal
HW-GRO
Receiver Power (Watts)
400.00
300.00
200.00
100.00
0.00
348.91
349.51
360.22
353.71
364.89
361.84
354.99
349.37
360.27
353.28
373.06
372.91
Baseline
Devmem
io_uring
Baseline + BigTCP
Devmem + BigTCP
io_uring + BigTCP

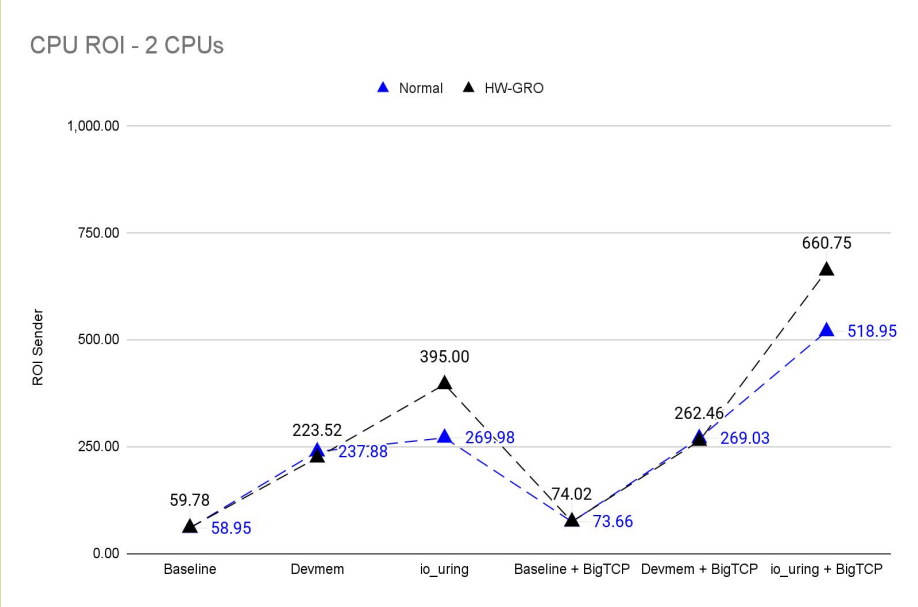

Sender Power - 4K MTU - 2 CPUs
Normal
HW-GRO
Sender Power (Watts)
400.00
300.00
200.00
100.00
0.00
340.85
340.78
337.39
336.81
340.38
345.06
335.82
335.53
331.38
331.94
333.09
339.09
Baseline
Devmem
io_uring
Baseline + BigTCP
Devmem + BigTCP
io_uring + BigTCP

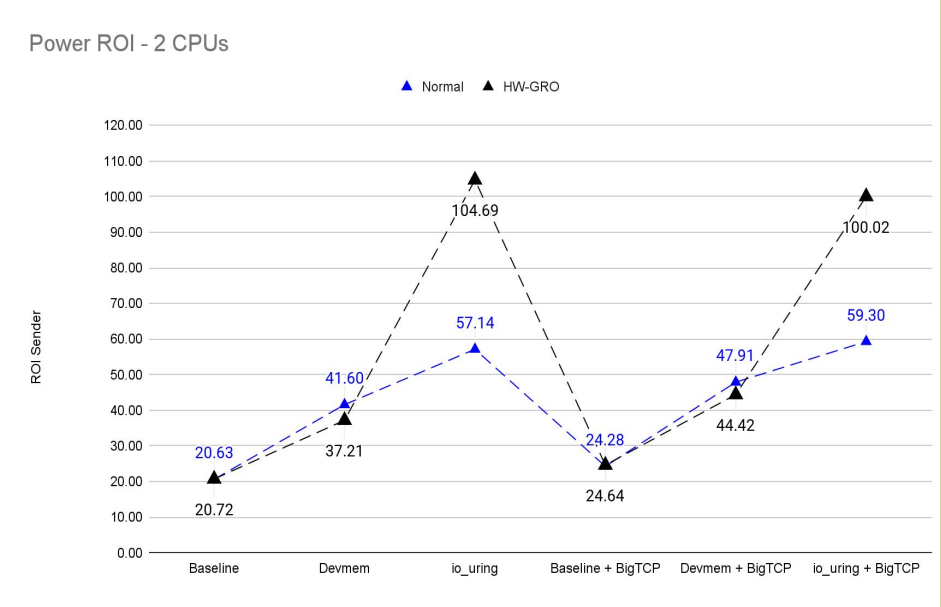
ROI Receiver - 2 CPUs





ROI Sender - 2 CPUs



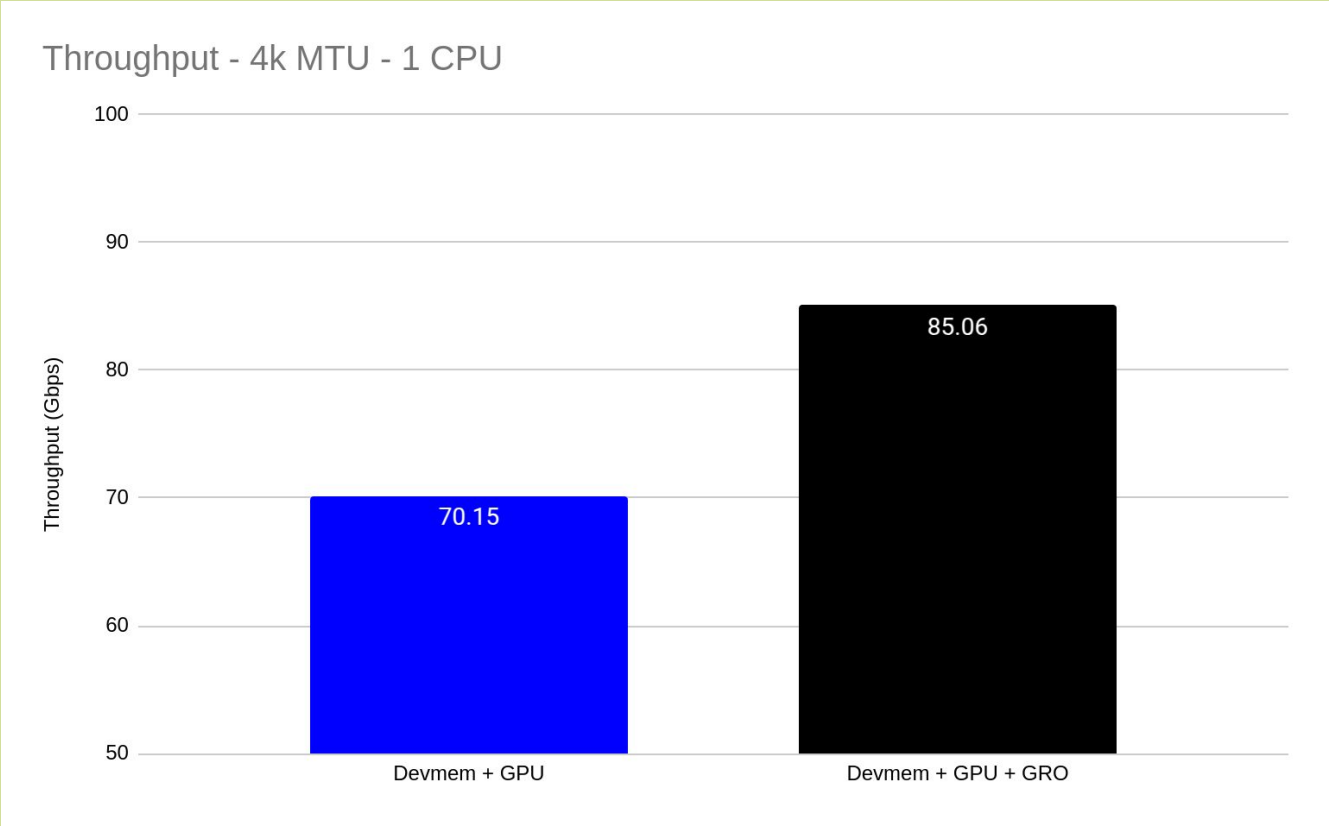


Sample Accelerator

Cheap GPU: NVidia T1000

- Sitting on a PCIE3 x16 riser card (max expected ~100Gbps)
- 4G RAM
- Run some dummy CUDA code to keep the GPU awake

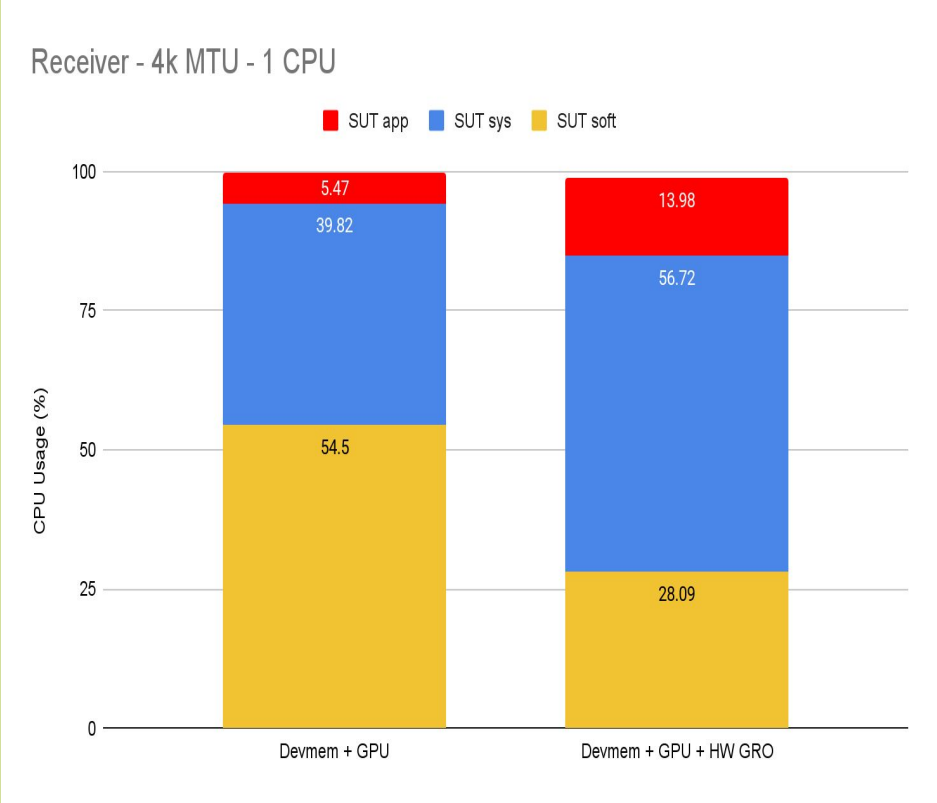
Throughput - 1 CPU - GPU Mem

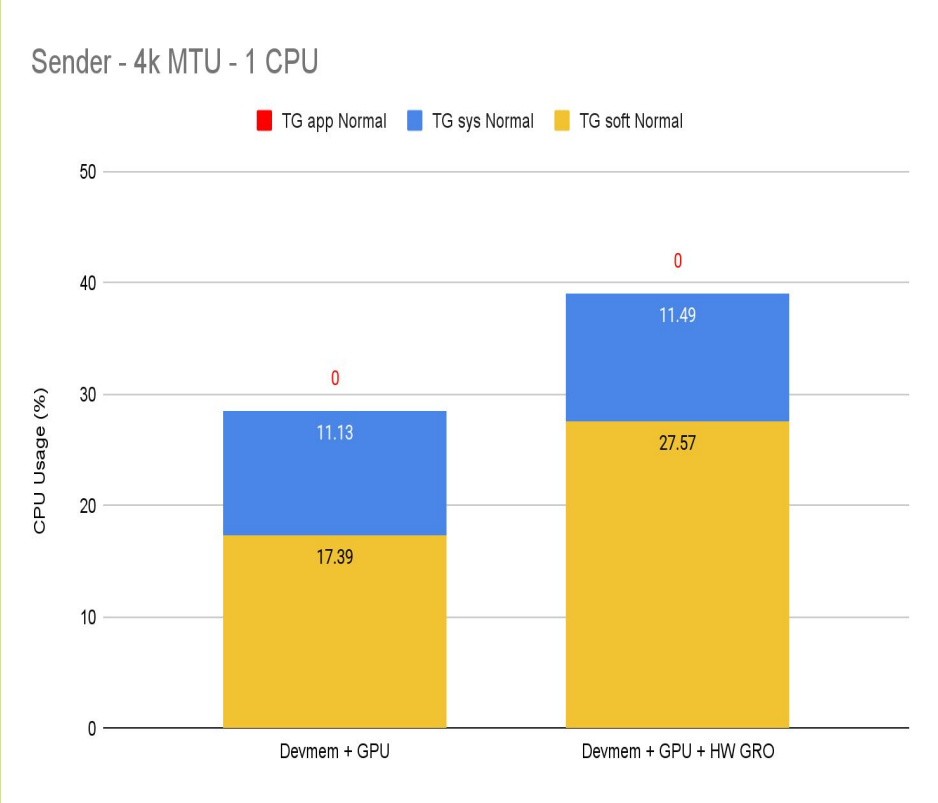


Nvtop - Receiver

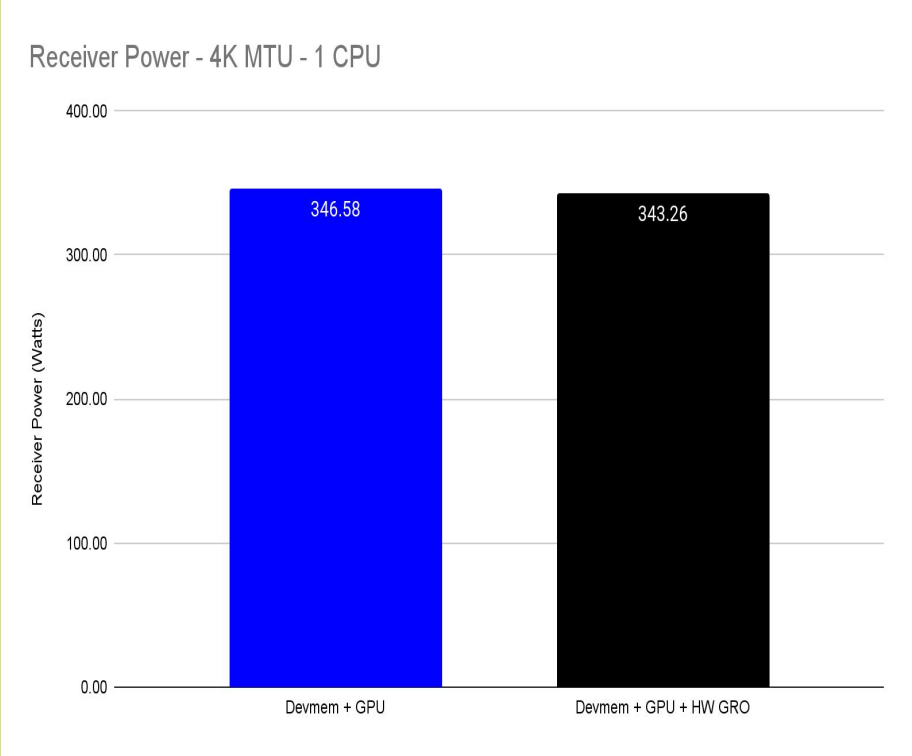
```
Device 0 [NVIDIA T1000] PCIe GEN 3@16x RX: 10.81 GiB/s TX: 2.930 MiB/s
GPU 1065MHz MEM 5000MHz TEMP  64°C FAN  47% POW N/A /  50 W
GPU[|                          3%] MEM[|||          0.469Gi/4.000Gi]

100 GPU0 %
 75 GPU0 mem%

 50
 25
  0
   55s                    41s                    27s                    13s                    0s
```

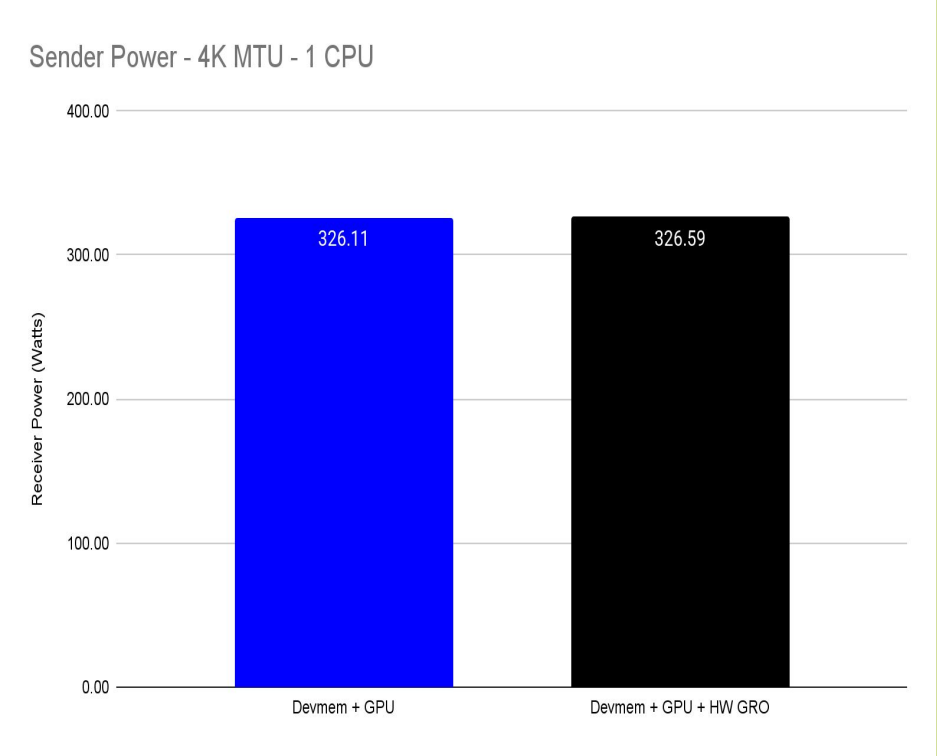
CPU Usage - 1 CPU - GPU Mem



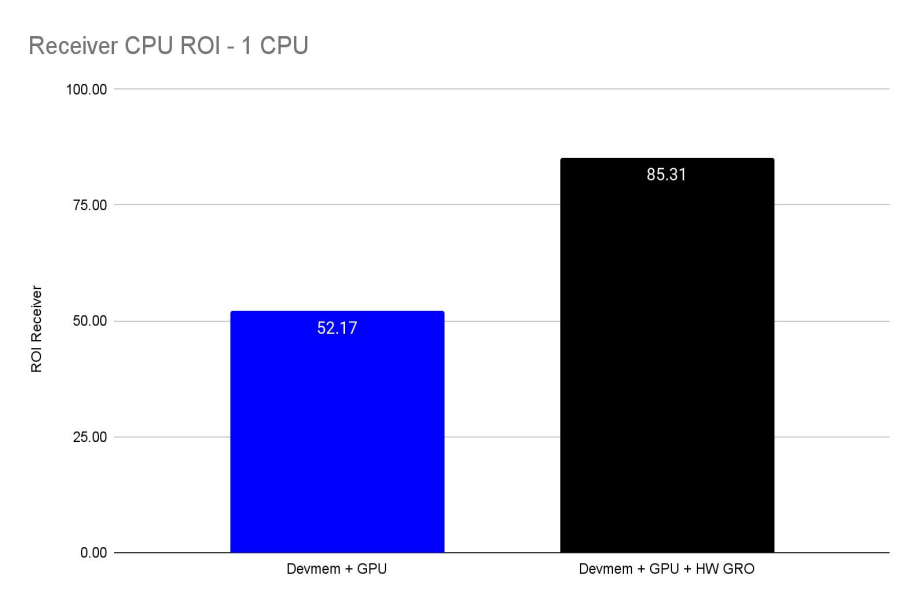


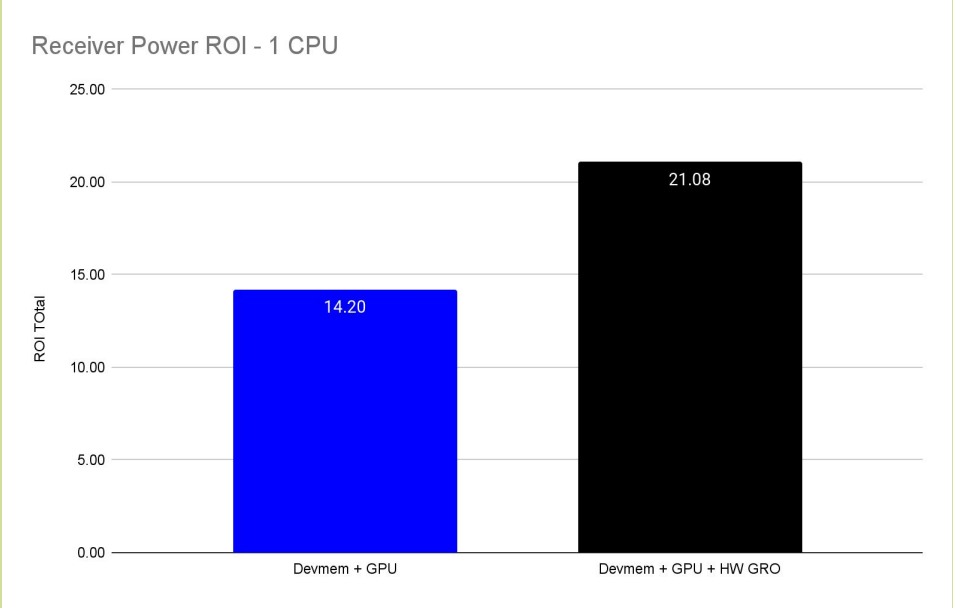
Power - 1 CPU - GPU Mem



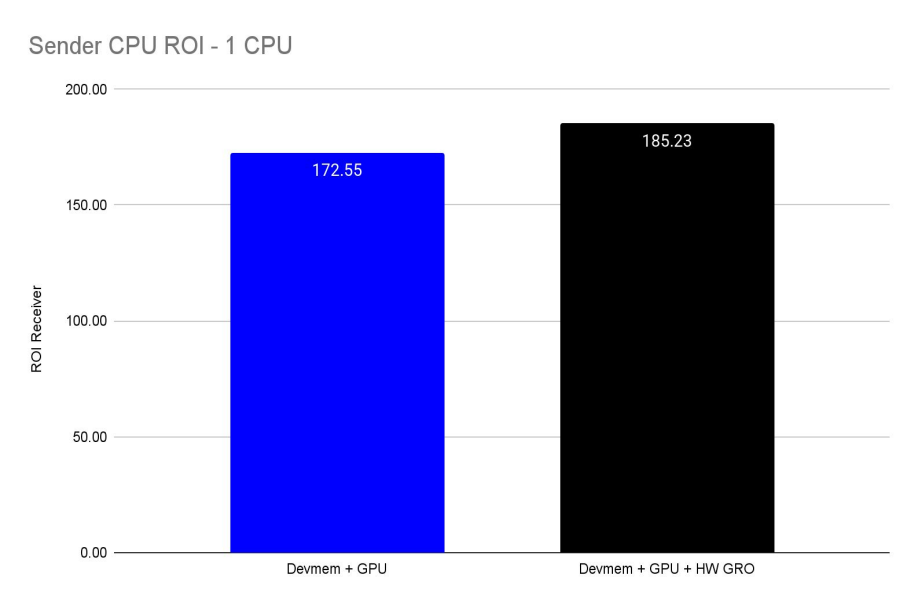


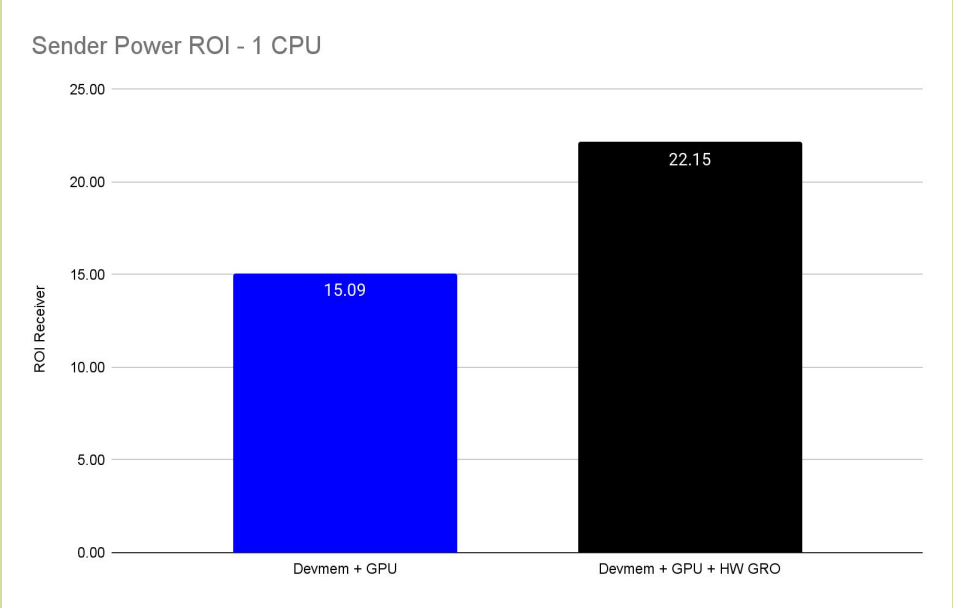
ROI Receiver - 1 CPU - GPU Mem



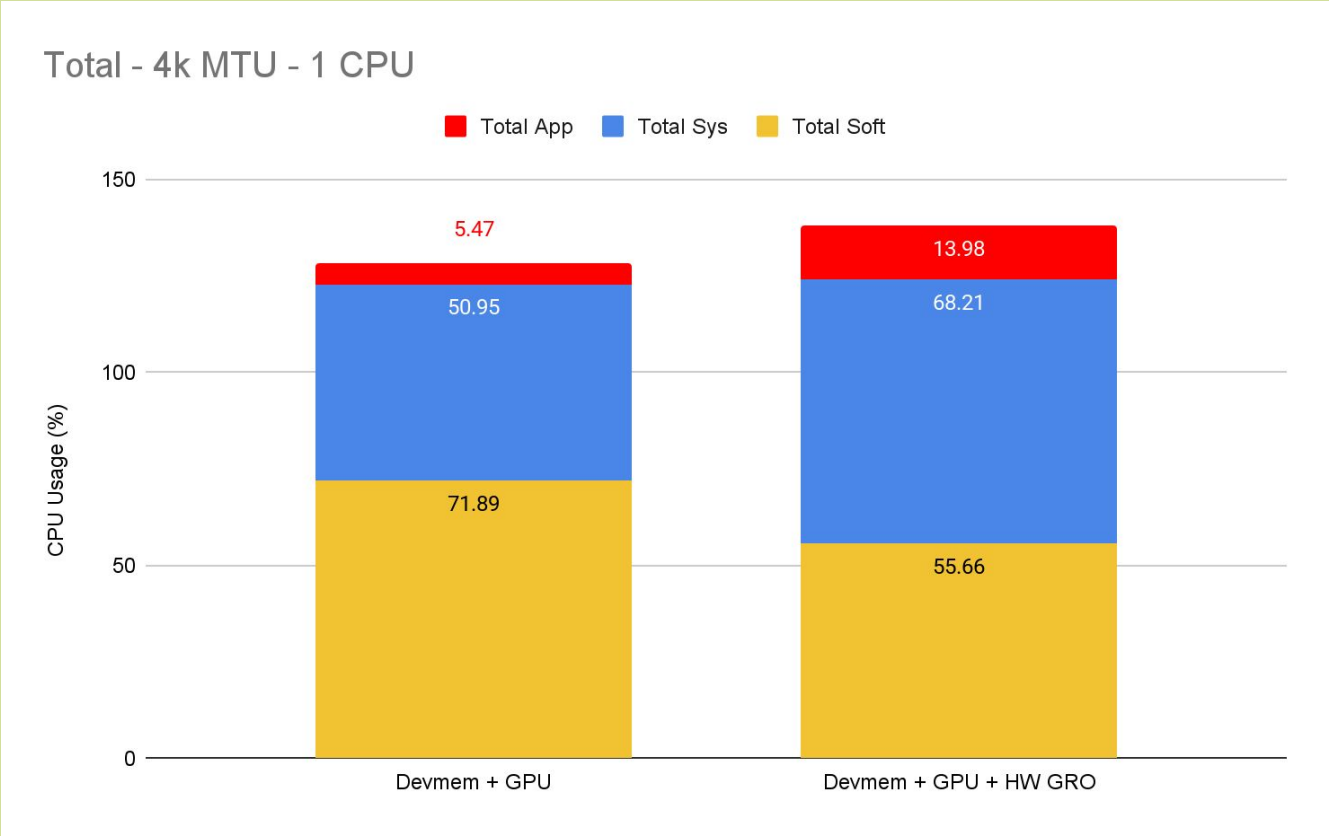


ROI Sender - 1 CPU - GPU Mem

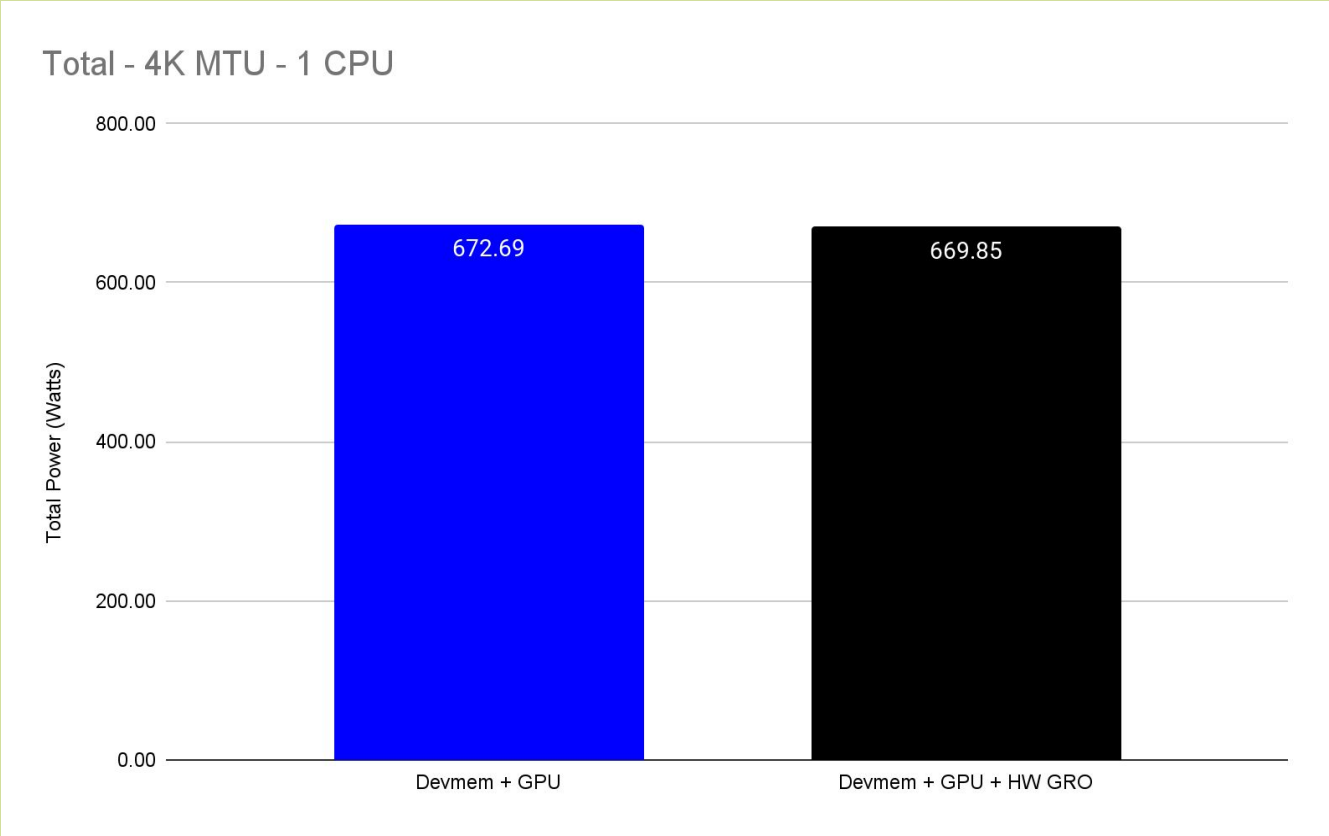




CPU Usage Total - 1 CPU - GPU Mem



Power Total - 1 CPU - GPU Mem



ROI Total - 1 CPU - GPU Mem