SNIA DEVELOPER CONFERENCE

SDC 24

BY Developers FOR Developers

September 16-18, 2024
Santa Clara, CA

Optimizing NVMe Over TCP for Disaggregated Storage on DPUs

Satananda Burla, Pradeep Kumar Nalla

Forward-looking statements

Except for statements of historical fact, this presentation contains forward-looking statements (within the meaning of the federal securities laws) including statements related to future revenue, future earnings, and the success of our product releases that involve risks and uncertainties. Words such as “anticipates,” “expects,” “intends,” “plans,” “projects,” “believes,” “seeks,” “estimates,” “can,” “may,” “will,” “would” and similar expressions identify such forward-looking statements. These statements are not guarantees of results and should not be considered as an indication of future activity or future performance. Actual events or results may differ materially from those described in this presentation due to a number of risks and uncertainties.

For other factors that could cause Marvell’s results to vary from expectations, please see the risk factors identified in Marvell’s Quarterly Report on Form 10-Q for the fiscal quarter ended July 29, 2024 as filed with the SEC on August 25, 2024 and Marvell’s Annual Report on Form 10-K for the fiscal year ended January 28, 202 as filed with the SEC on March 9, 2024, and other factors detailed from time to time in Marvell’s filings with the SEC. Marvell undertakes no obligation to revise or update publicly any forward-looking statements.

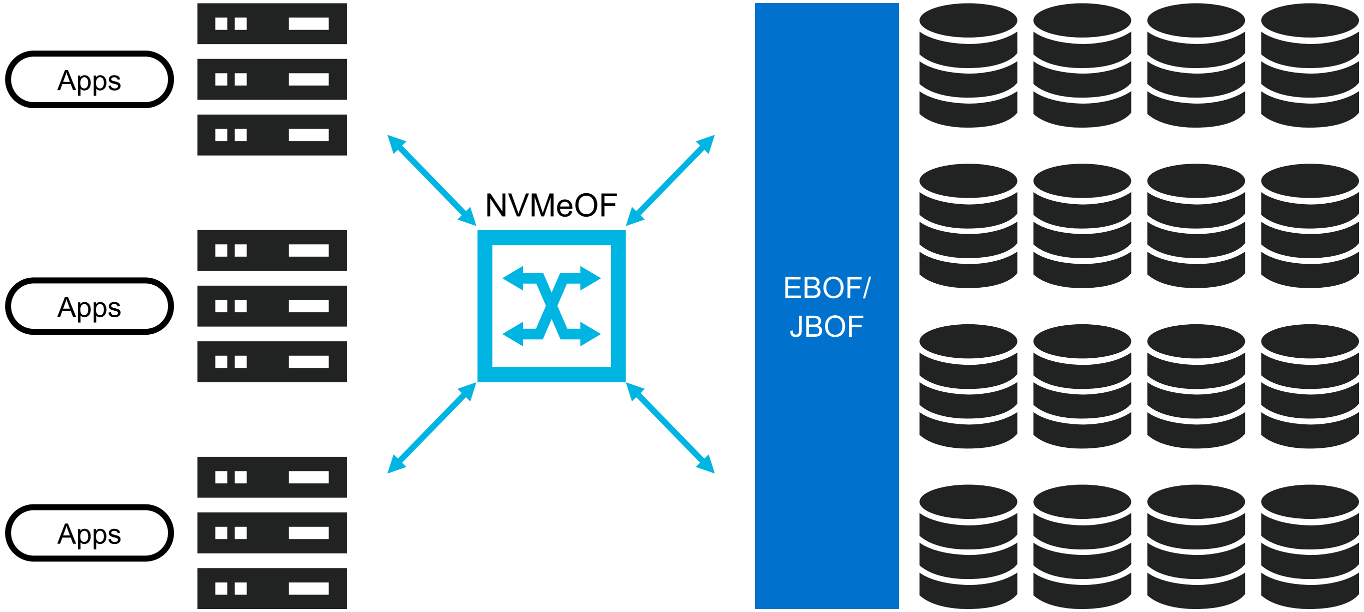
Agenda

- ✓ Disaggregated storage
- ✓ NVMeOTCP on DPU – problems and solutions
- ✓ Next steps, future work

Disaggregated Storage

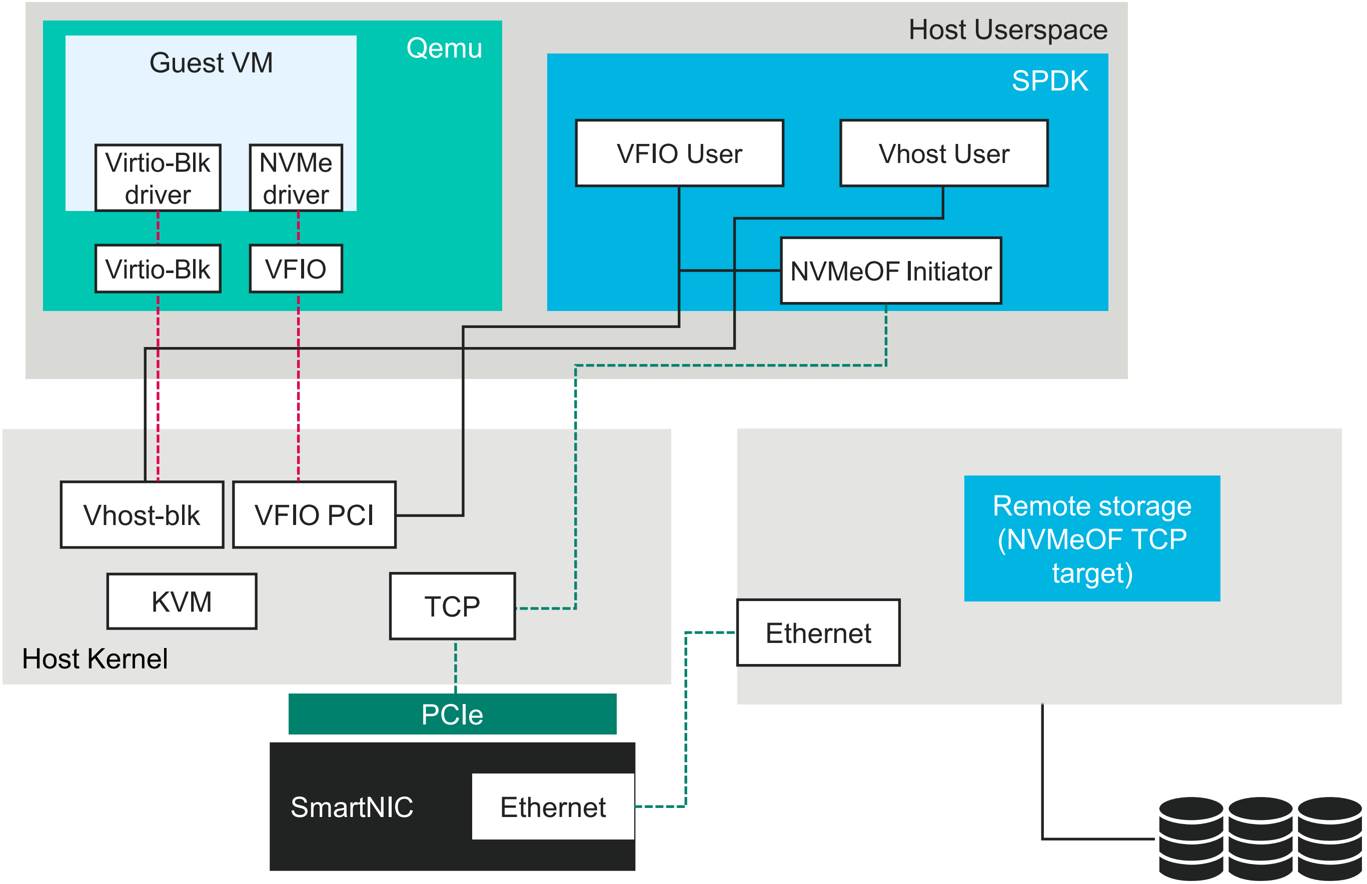
Disaggregated storage

- Compute and Storage resources are separated
- Storage can be collocated in the DC
- Multiple advantages
  - Allows for scale out and greater efficiency
  - Allows for flexibility during upgrade, replace cycles
  - Does not need dedicated hardware like SANs, works using Ethernet
- NVMe provides simplistic and fast control/data interface
- NVMeOTCP enables storage over the most ubiquitous network protocols (TCP/IP + Ethernet)
- SmartNICs and DPUs have supported NVMeOTCP for cloud/VM use cases
- DPUs provide additional control plane capability offloading NVMeOF initiation and termination from hypervisor



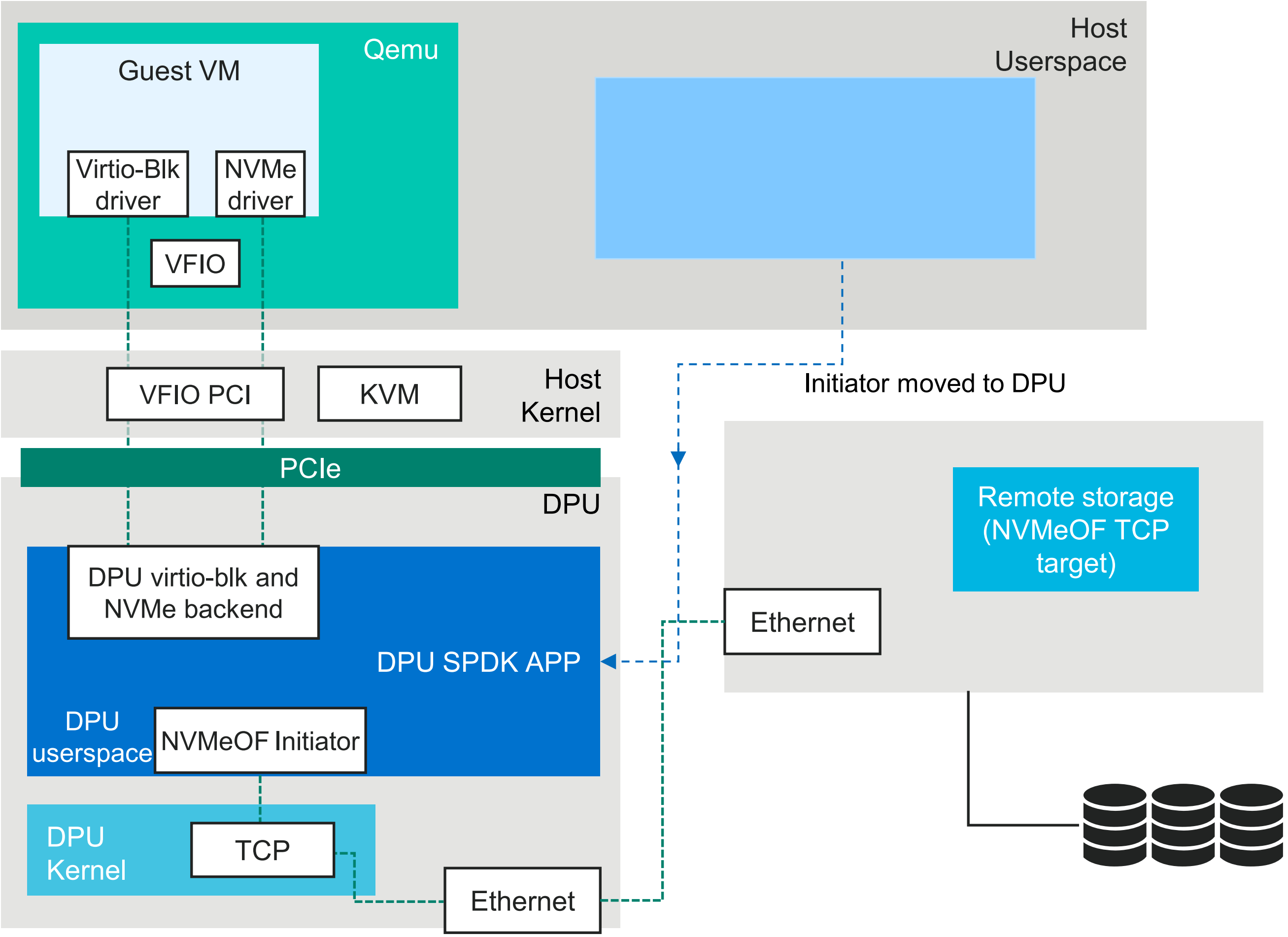
Disaggregated storage for VMs with SmartNIC

- SmartNIC provides Ethernet connectivity with protocol offloads(csum/TSO/LRO)
- NVMeOTCP initiator runs on the host either kernel/SPDK
- Virtio-blk/NVMe backend emulation has to be run on host
- Requires CPU cycles on Host which cannot be used for Tenant workloads
- QoS for storage becomes complicated with Network bandwidth sharing



Disaggregated storage with DPU

- DPU provides Local NVMe/Virtio-Blk-PCI PF/VF devices to host
- DPU handles the Virtio-blk/NVMe backend and runs NVMeOTCP initiator
- Guest VMs use NVMe/Virtio-Blk devices directly with passthrough
- NVMeOF initiator configuration needs to be done on DPU
- Saves Host NVMe/Virtio emulation and NVMeOTCP CPUcycles
- Provides better QoS separation for VMs by throttling local VFs

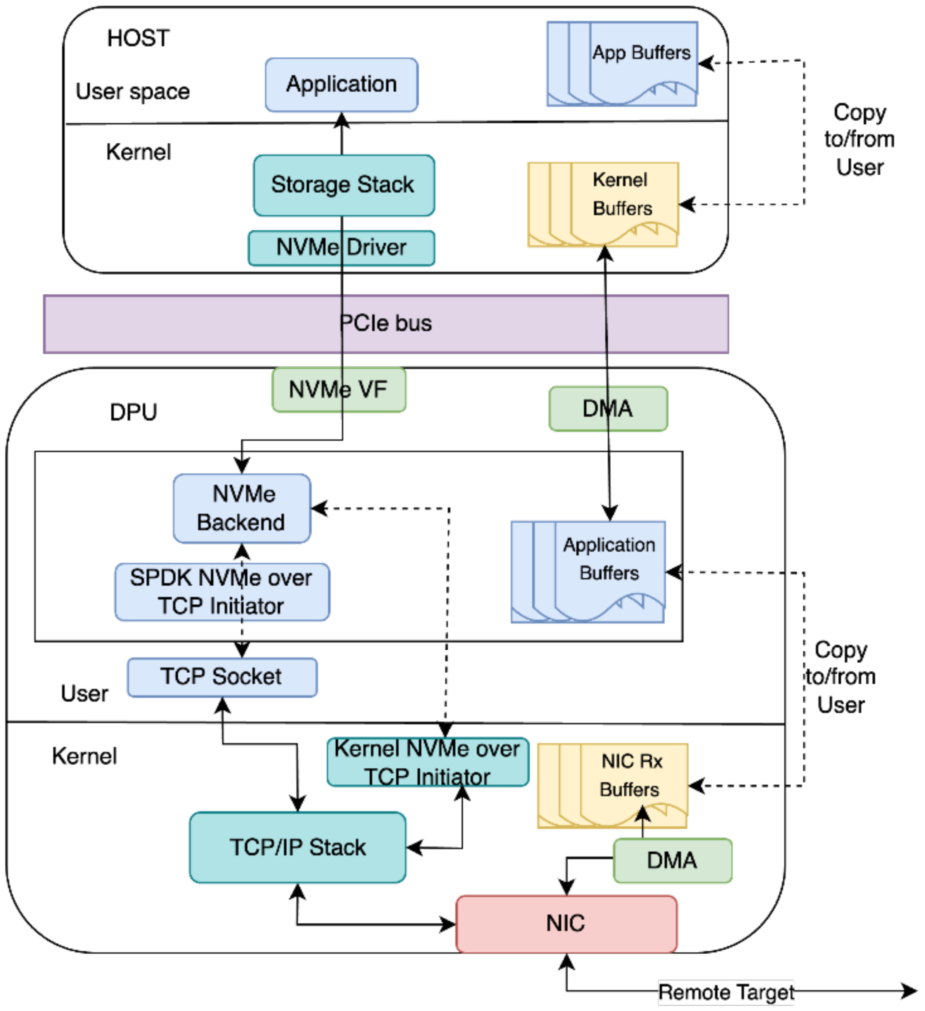


NVMeOTCP on DPU

Problems and Solutions

Problems in Offloading NVMeOTCP on DPU

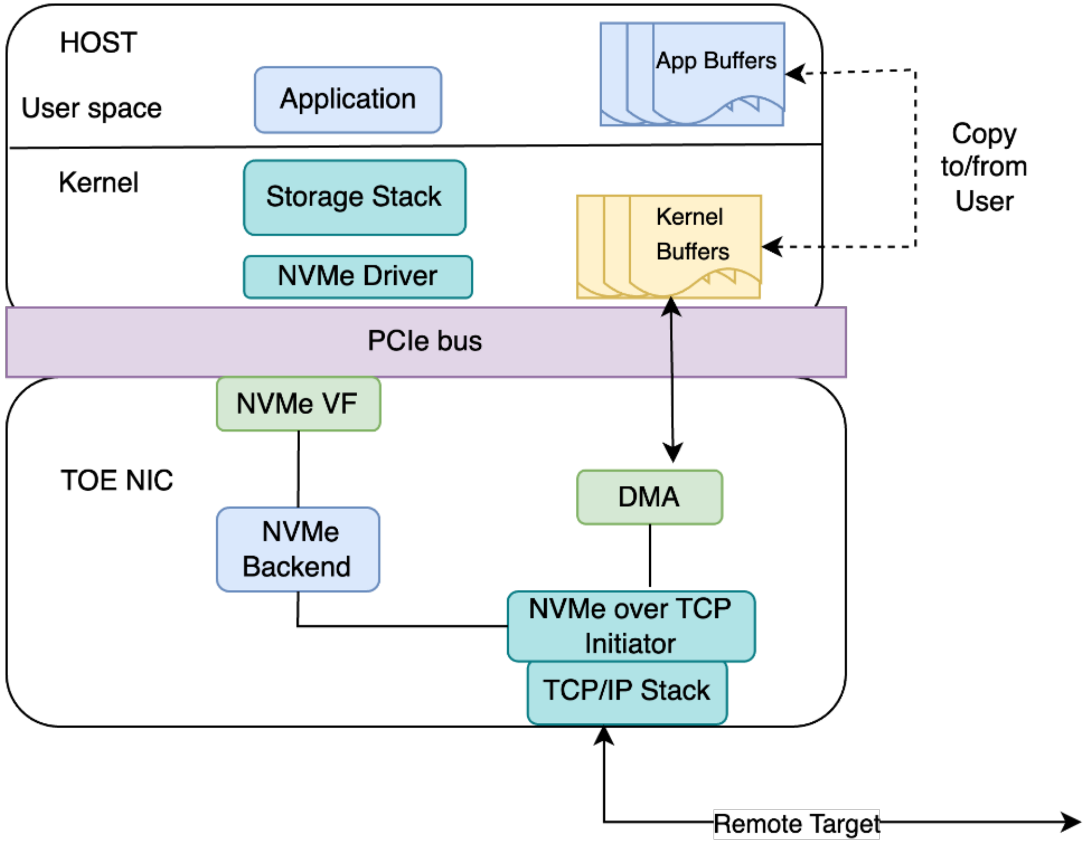
Bounce buffering	▪ One DMA from DPU NIC to DPU RX buffers ▪ A second DMA from DPU Application Buffers to Host Kernel Buffers ▪ Increases DDR utilization on the DPU though payload is not destined for DPU ▪ Noisy neighbor problems for DPU workloads
User copy	▪ There is an extra User copy from DPU kernel from/to DPU user ▪ If not eliminated increases DPU DDR utilization
PDU CRC	▪ NVMe Over TCP PDU has digest ▪ Digest calculation consumes CPU cycles
Out of order PDU	▪ The remote side is allowed to reorder the transmitted PDUs ▪ TCP is in order but PDUs can be out of order ▪ Direct buffer copy is not possible in such situations
Security	▪ TLS security (data in transit) ▪ OPAL TCG SED (data at rest) ▪ Offloading TLS is complex
Latency	▪ Latency increases due to additional hop in DPU



Custom TCP stack (TCP offload engine)

Custom TOE (TCP offload engine)

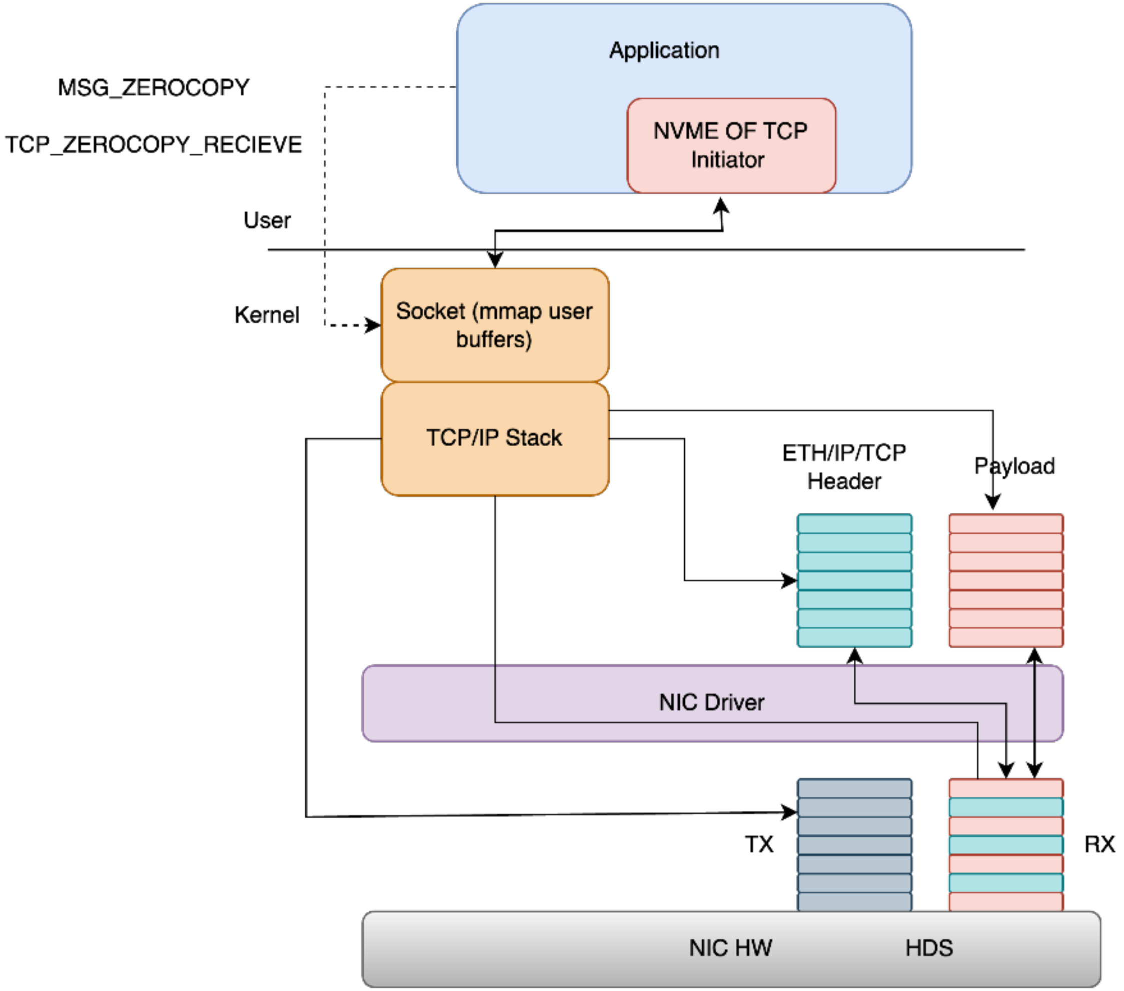
- TOE based implementations provide solution by running Custom TCP stack in Hardware
- Security Concerns, Maintenance concerns, slower update cycles, delay in new adding new features have hampered adoption
- Eliminates both usercopy and Bounce buffering
- Security can be handled by NIC itself



Zerocopy sockets

Zerocopy sockets

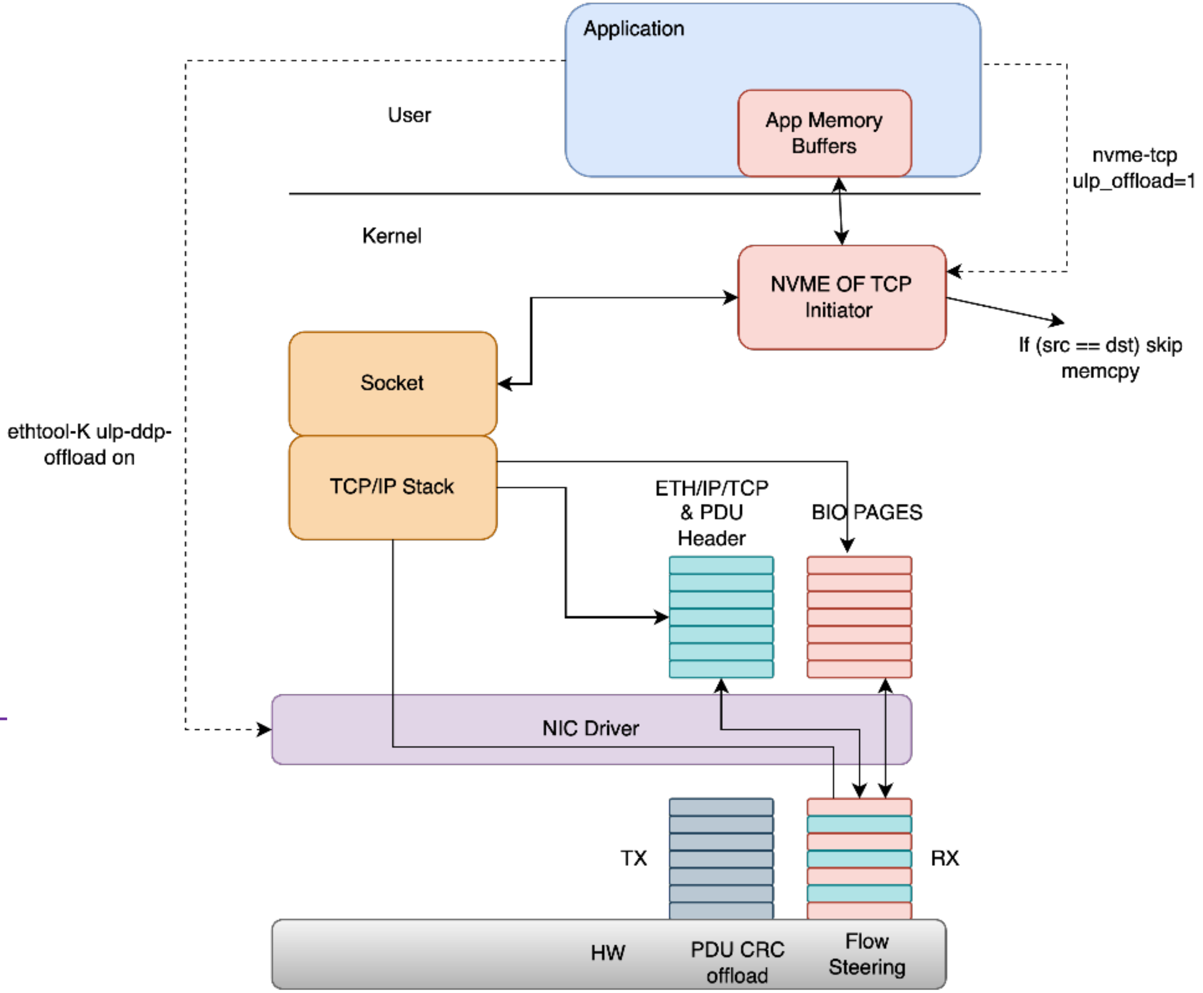
- MSG_ZEROCOPY socket option for send
- TCP_ZEROCOPY_RECIEVE socket option is used to map pages into userspace
- Zerocopy works only for page aligned payload
- PDU CRC is not handled, Needs Userspace Initiator
- Needs scatter receive from NIC if not HDS
- Need to map/unmap for every recieve
- Eliminates Copy
- Bounce Buffering is not avoided
- Security has to be handled by kernel



ULP DDP offload

Kernel ULP DDP offload

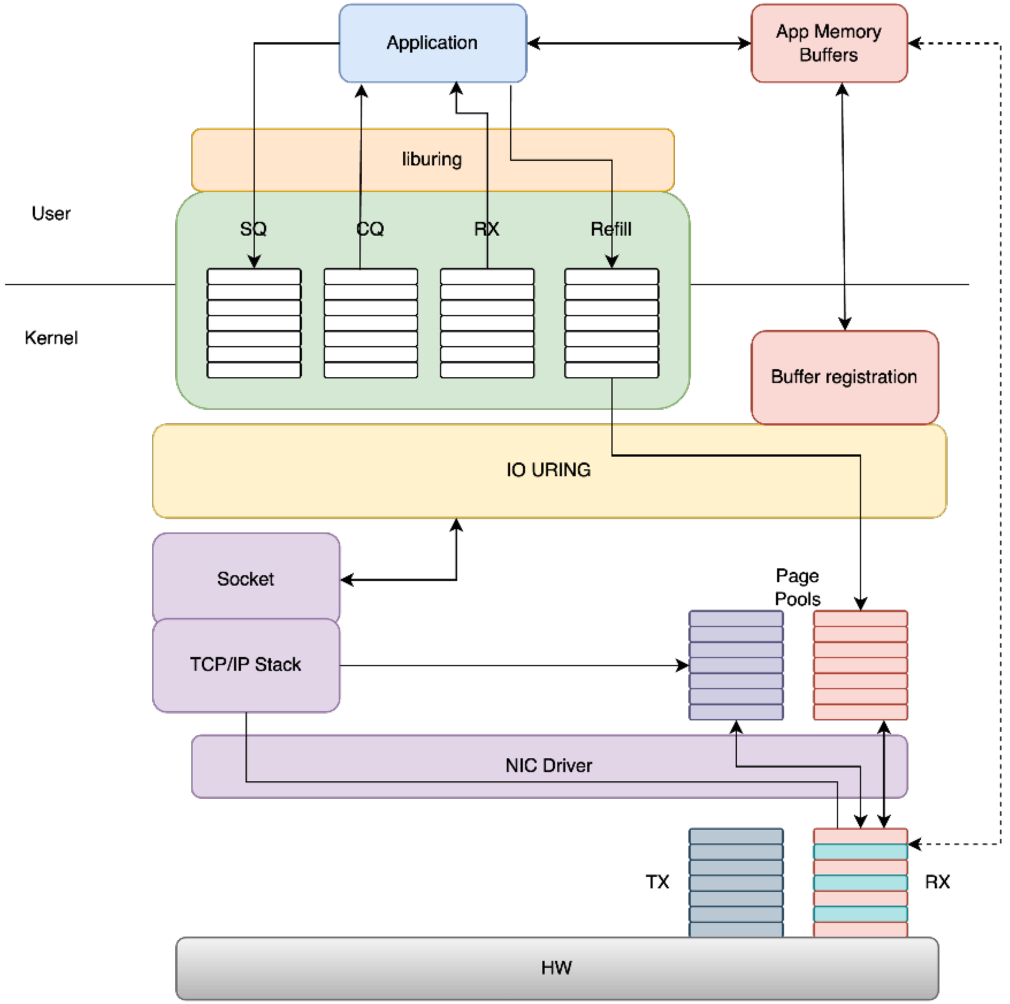
- Kernel NVMe over TCP offload
- Other solutions do not address NVMe OF PDU CRC validation
- PDUs can get reordered
- HDS (with TCP and PDU header from receive ring, Payload from userspace buffers)
- Storage protocol skips copy
- Eliminates Usercopy
- PDU CRC processing is offloaded
- Out of order PDU is a concern
- Bounce buffering is not avoided
- Security can be offloaded



IO URING Zerocopy

Liburing

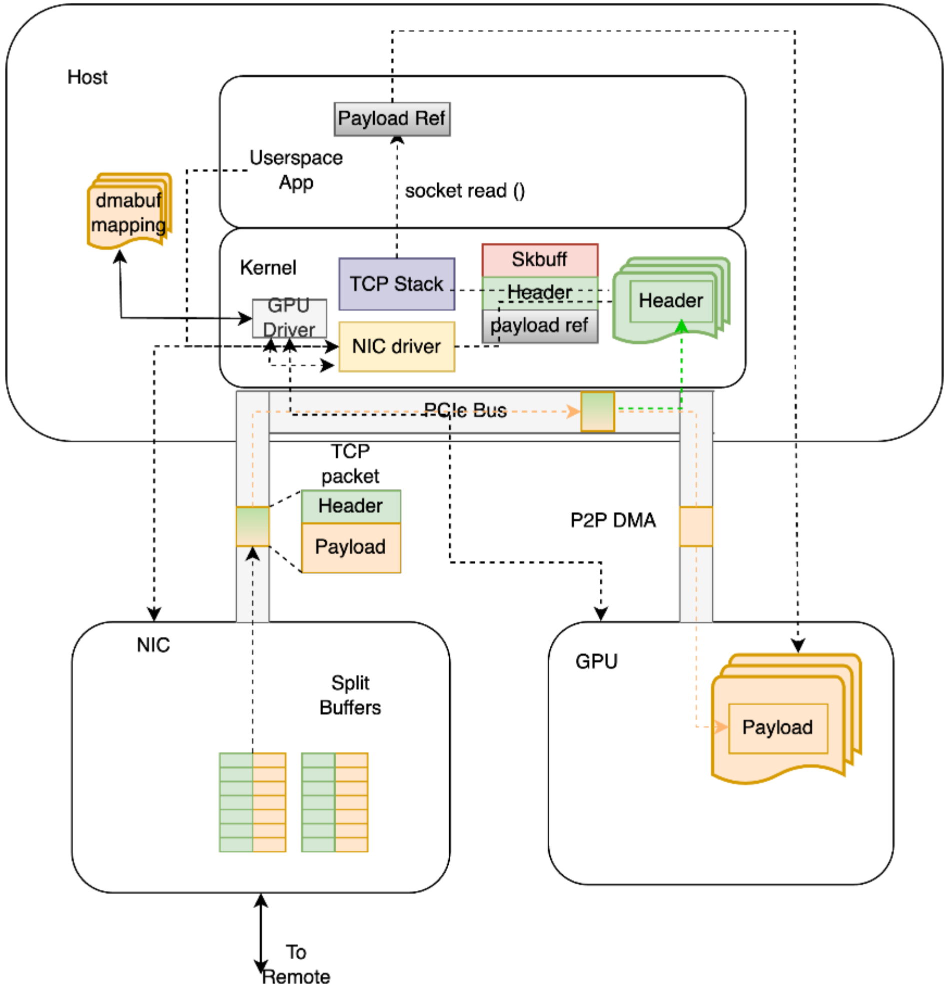
- Rings shared between Userspace and Kernel
- Userspace submits through SQ
- Kernel completions through CQ
- Register userspace memory with IO URING
- HDS(Header data Split) + RSS flow steering
- Fill HW RX payload ring with userspace buffers
- Needs change in userspace applications to switch from system calls to IO URING
- Needs userspace initiator like SPDK
- Eliminates usercopy
- Bounce buffering is not avoided
- Security has to be handled in kernel



TCP DEVICEMEM

TCP device memory

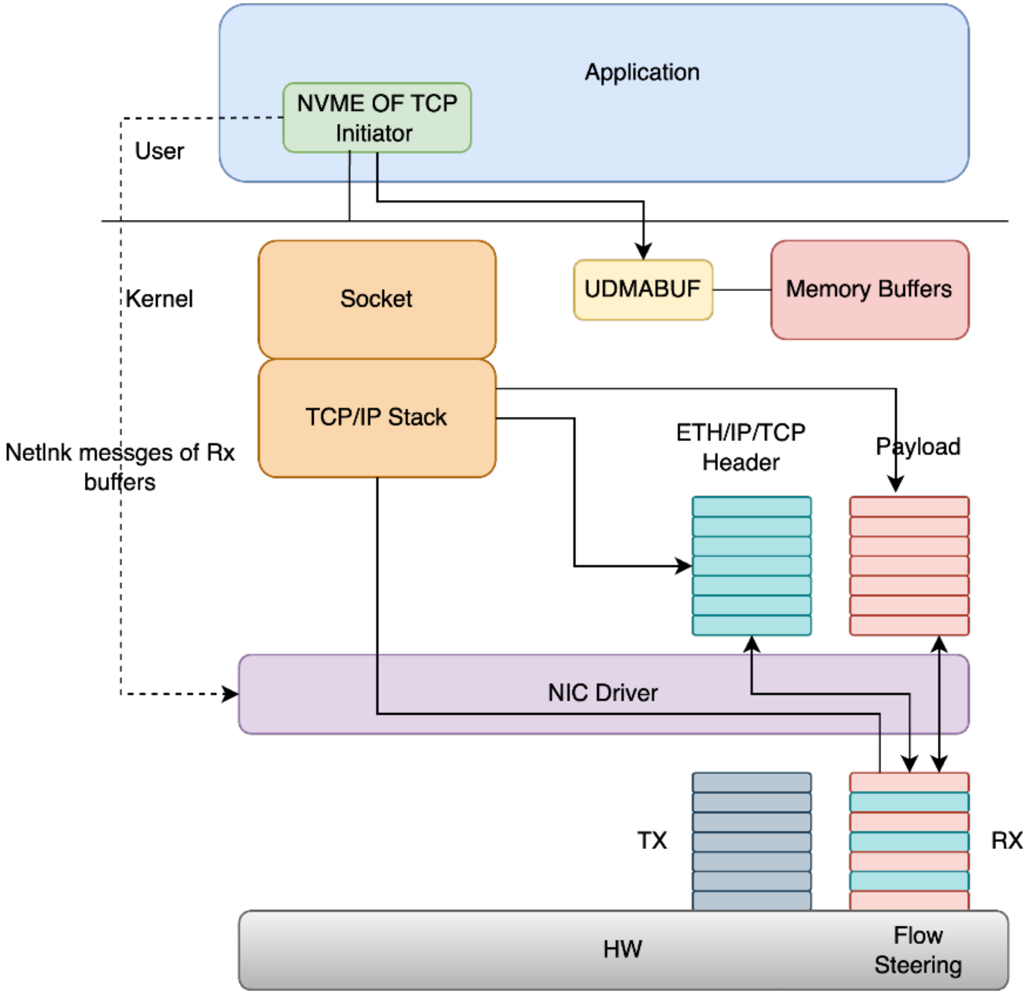
- Designed for direct Data placement of TCP Payload using peer to peer DMA into GPU memory by NIC
- Linux DMABUF infrastructure allows buffer sharing across multiple device drivers. (NIC and GPU)
- Netlink messages to fill buffers into netdev's rx queue, from the pool created out of dma buffers
- Avoids mapping and un-mapping receive buffer to userspace
- NIC HDS feature needed to split header and payload
- NIC flow steering desired flow to this particular rx queue
- TX is not yet supported
- Not yet upstreamed



Adapting TCP DEVICEMEM for NVMe over TCP

Storage with TCP Devmem

- Udmabuf is a kernel mechanism to allocate contiguous memory blocks in the kernel space as DMA buffers and makes them available from the user space
- Userspace Initiator can send the buffers through netlink to NIC supporting TCP Devicemem
- NIC needs to support flow steering and HDS split
- Usercopy is eliminated
- DMA bounce buffering is not avoided
- Security cannot be offloaded



Summary of Different TCP Optimization methods

Issue	Custom TOE	Zerocopy Socket	ULP DDP offload	IO URING Zerocopy	TCP Devicemem
Bounce Buffering/Latency	Yes	No	No	No	No
Usercopy	Yes	Yes	Yes	Yes	Yes
PDU CRC calculation/validation	Yes	No	Yes	No	No
Out of order PDU	Yes	No	No	No	No
NVMeOTCP TLS offload	Yes	No	Yes	No	No
TCG OPAL SED security offload	Yes	Yes	Yes	Yes	Yes

Next Steps and Future Work

Next steps and future work

1 Explore solutions to avoid DMA bounce buffering without using custom TCP stack solutions

2 Explore TLS offload while avoiding DMA bounce buffering

3 Extend SPDK initiator to use udmabuf with TCP Devicemem

4 Extend in kernel NVMe over TCP initiator to use TCP Devicemem

Q&A

Thank You