PASTE: A Network Programming Interface for Non-Volatile Main Memory

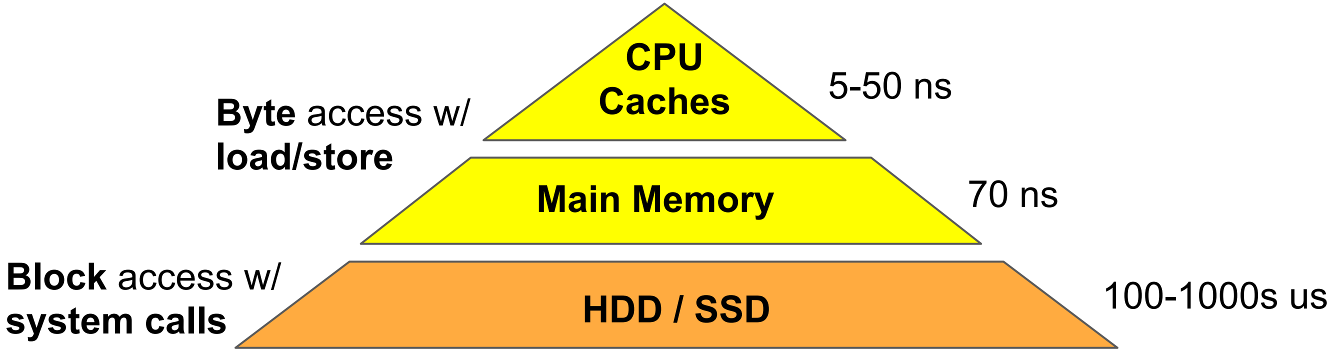
Michio Honda (NEC Laboratories Europe)
Giuseppe Lettieri (Università di Pisa)
Lars Eggert and Douglas Santry (NetApp)

USENIX NSDI 2018

SSICLOPS

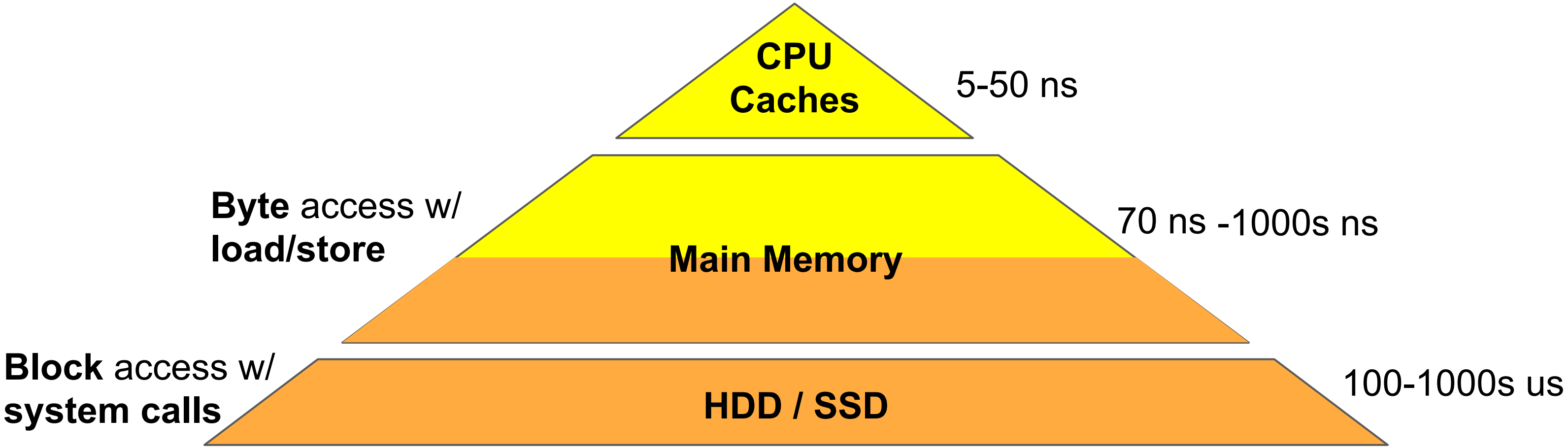
Review: Memory Hierarchy

Slow, block-oriented persistence



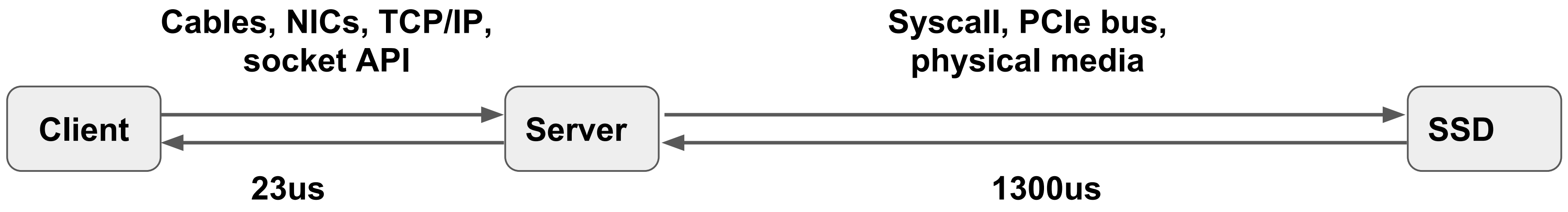
Review: Memory Hierarchy

Fast, byte-addressable persistence



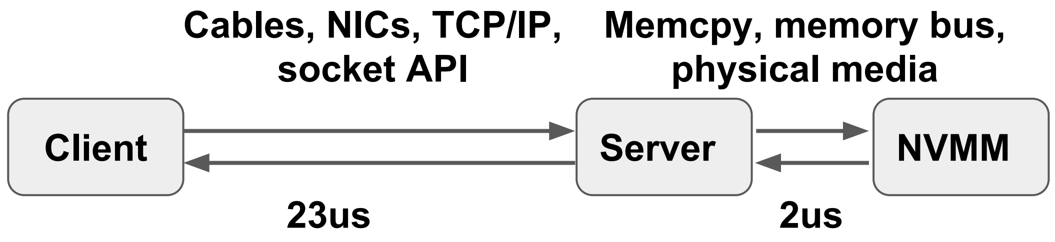
Networking is faster than disks/SSDs

1.2KB durable write over TCP/HTTP



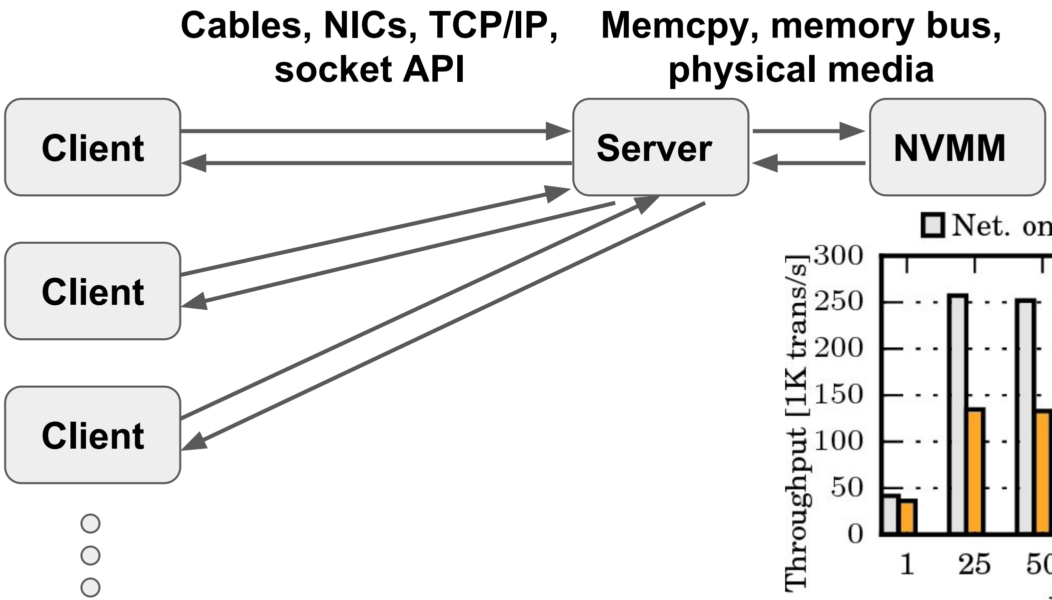
Networking is slower than NVMM

1.2KB durable write over TCP/HTTP

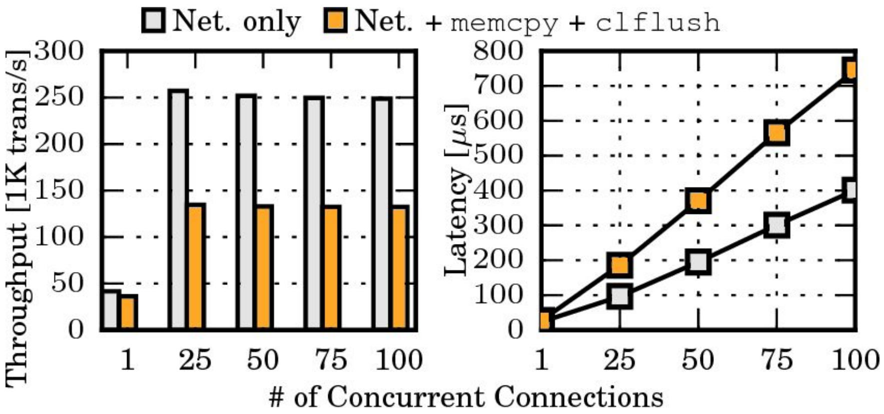


Networking is slower than NVMM

1.2KB durable write over TCP/HTTP




```
nevts = epoll_wait(fds)
for (i =0; i < nevts; i++) {
    read(fds[i], buf);
    ...
    memcpy(nvmm, buf);
    ...
    write(fds[i], reply)
}
```



Innovations at both stacks

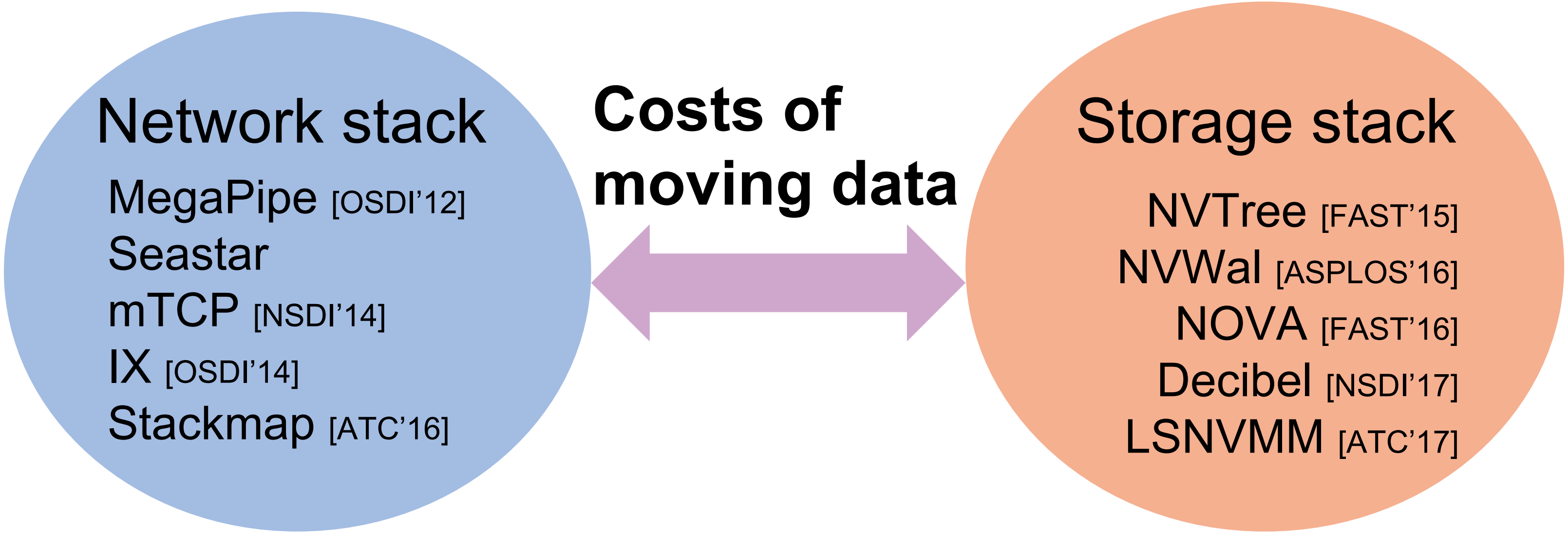
Network stack

MegaPipe [OSDI'12]
Seastar
mTCP [NSDI'14]
IX [OSDI'14]
Stackmap [ATC'16]

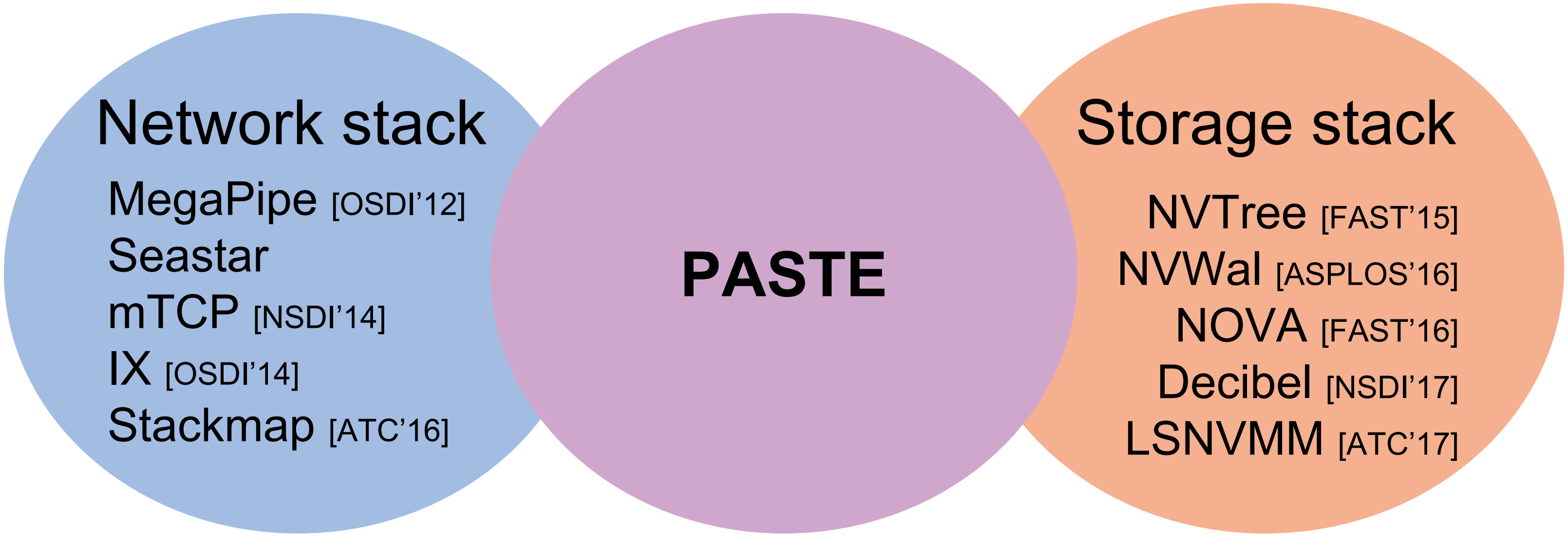
Storage stack

NVTree [FAST'15]
NVWal [ASPLOS'16]
NOVA [FAST'16]
Decibel [NSDI'17]
LSNVMM [ATC'17]

Stacks are isolated



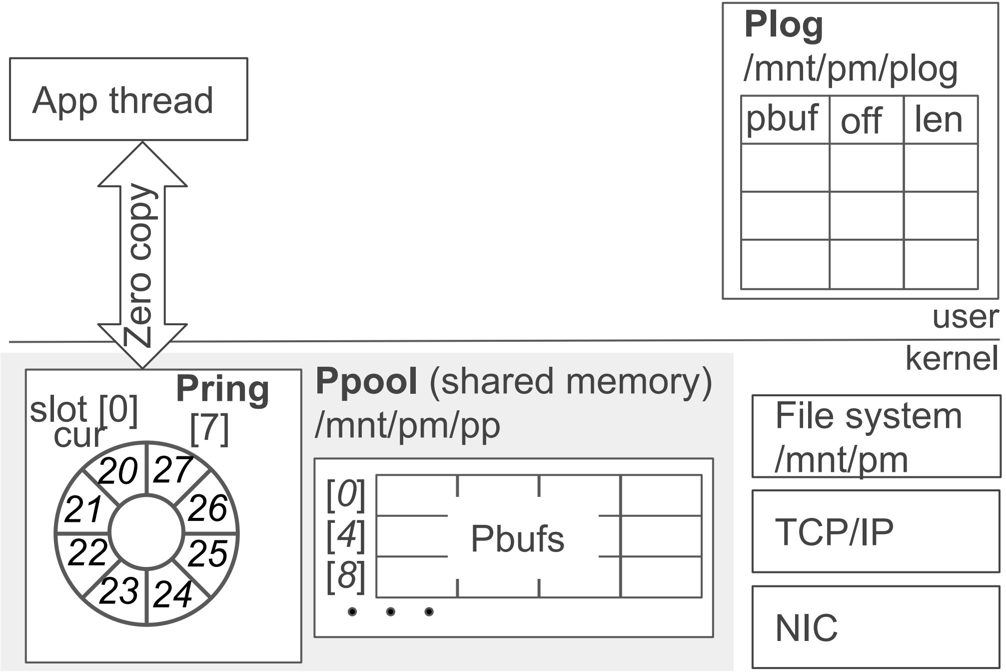
Bridging the gap



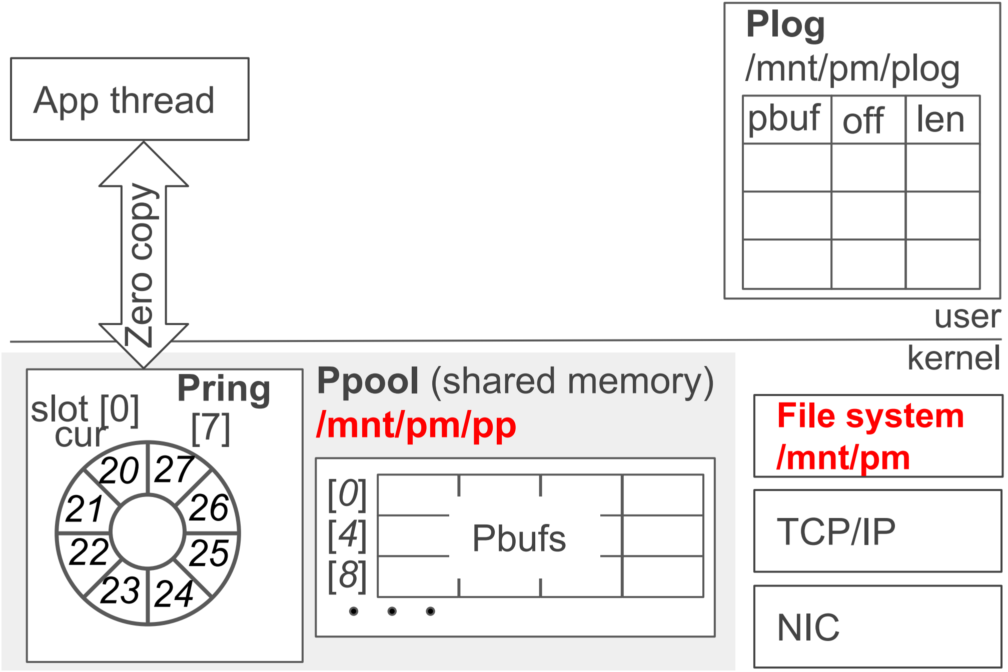
PASTE Design Goals

- Durable zero copy
  - **DMA to NVMM**
- Selective persistence
  - **Exploit modern NIC's DMA to L3 cache**
- Persistent data structures
  - **Indexed, named packet buffers backed fy a file**
- Generality and safety
  - **TCP/IP** in the kernel **and netmap API**
- Best practices from modern network stacks
  - Run-to-completion, blocking, busy-polling, batching etc

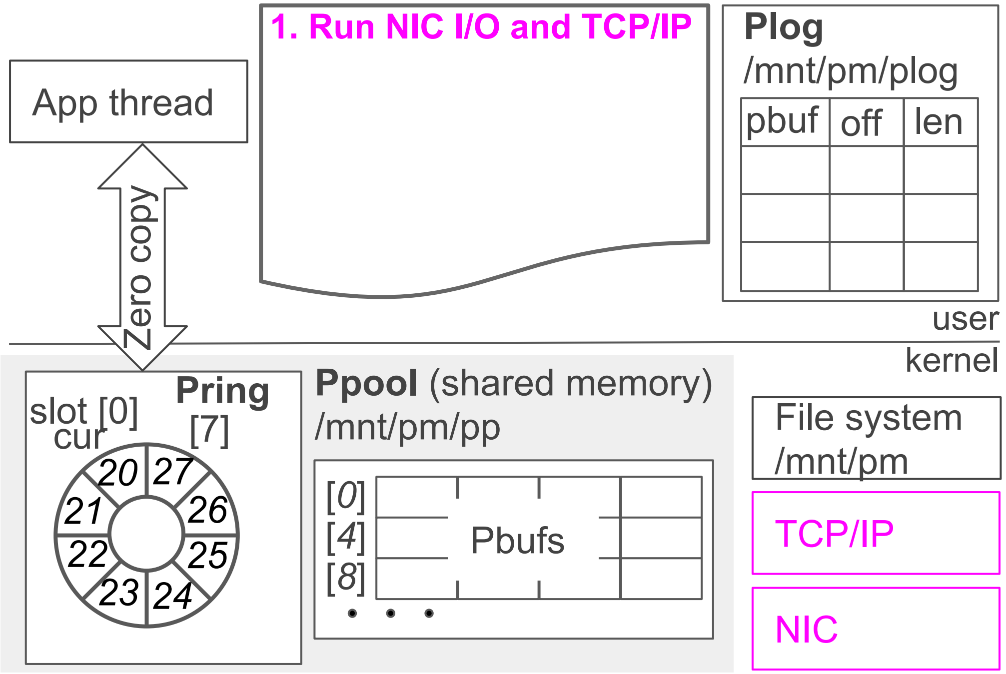
PASTE in Action



PASTE in Action

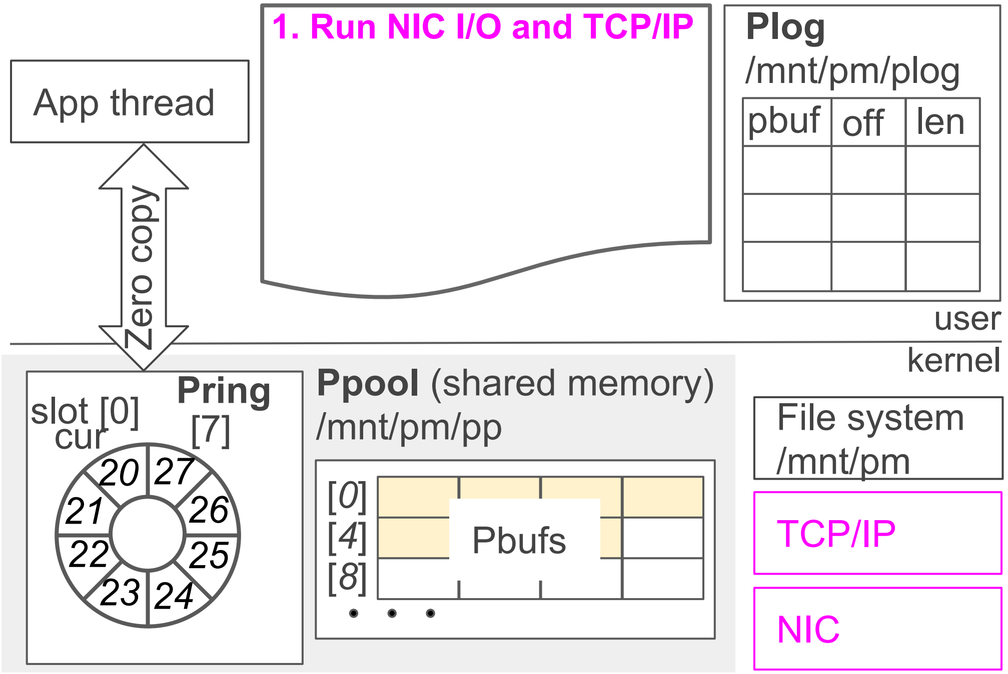


PASTE in Action



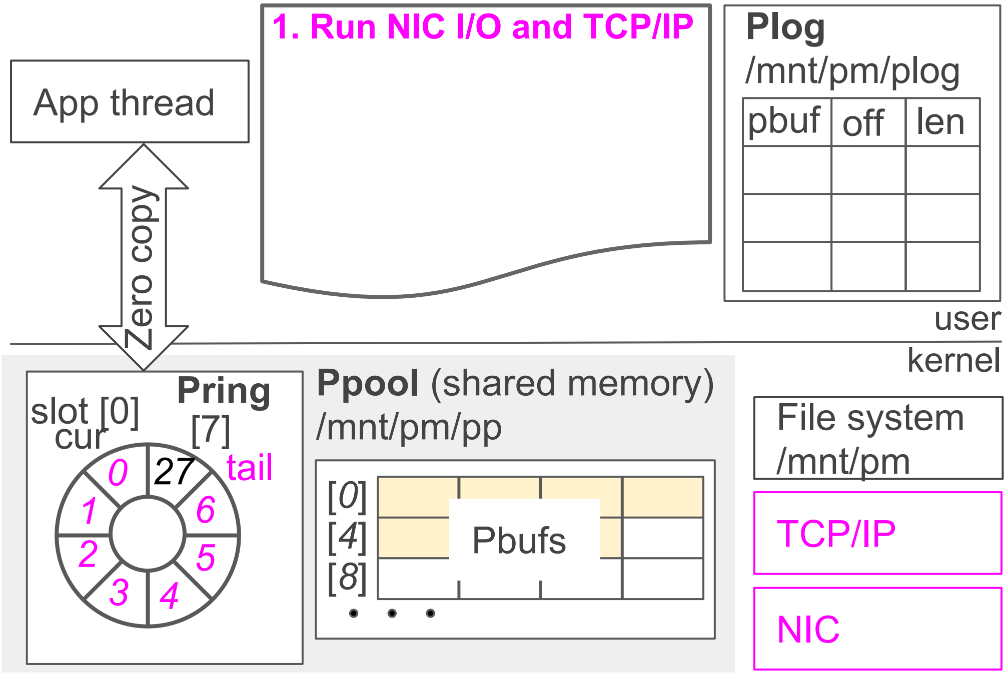
- poll() system call

PASTE in Action



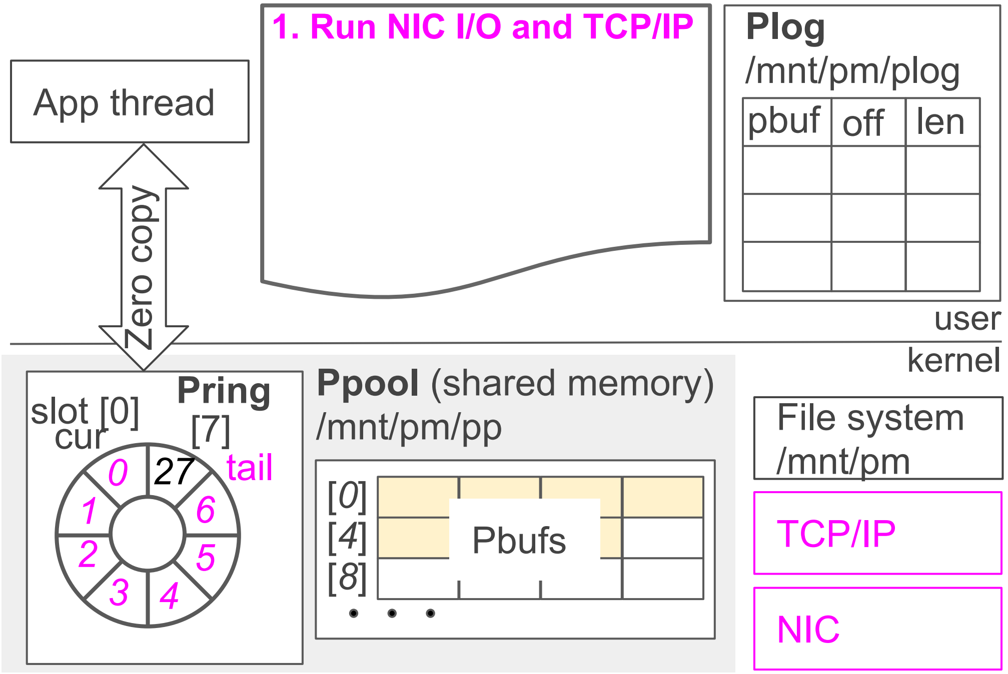
- poll() system call
  - Got 6 in-order TCP segments

PASTE in Action



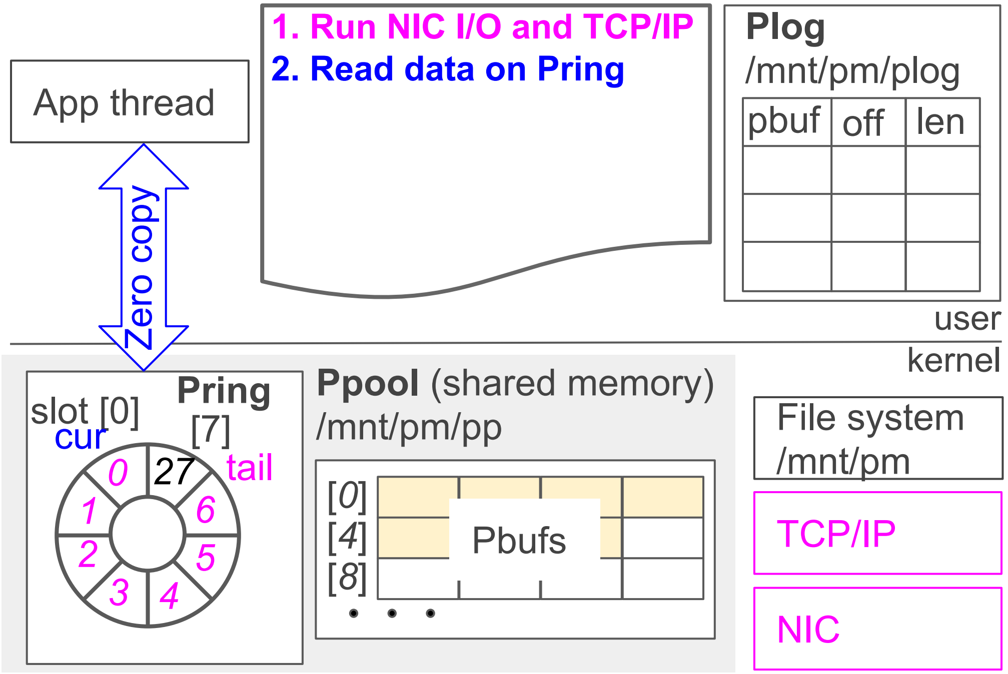
- poll() system call
  - They are set to Pring slots

PASTE in Action

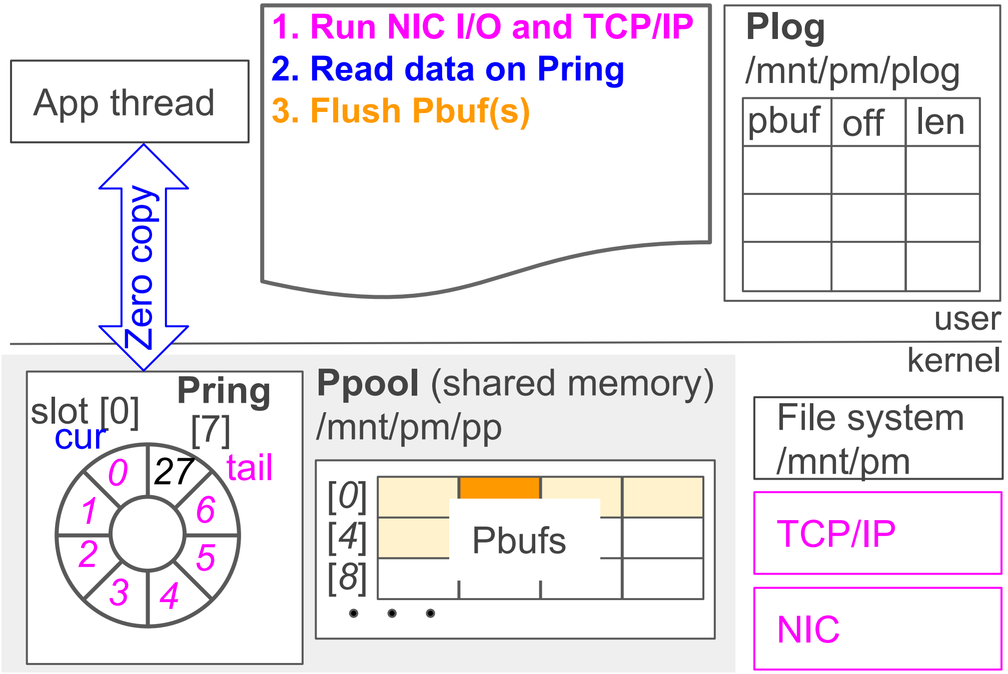


- Return from poll()

PASTE in Action

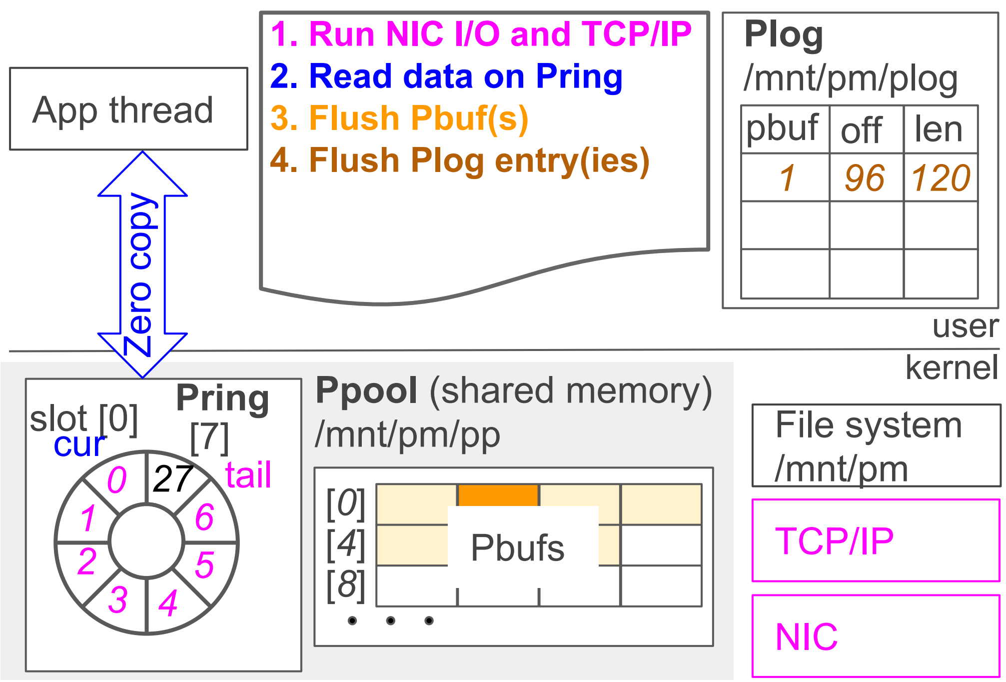


PASTE in Action



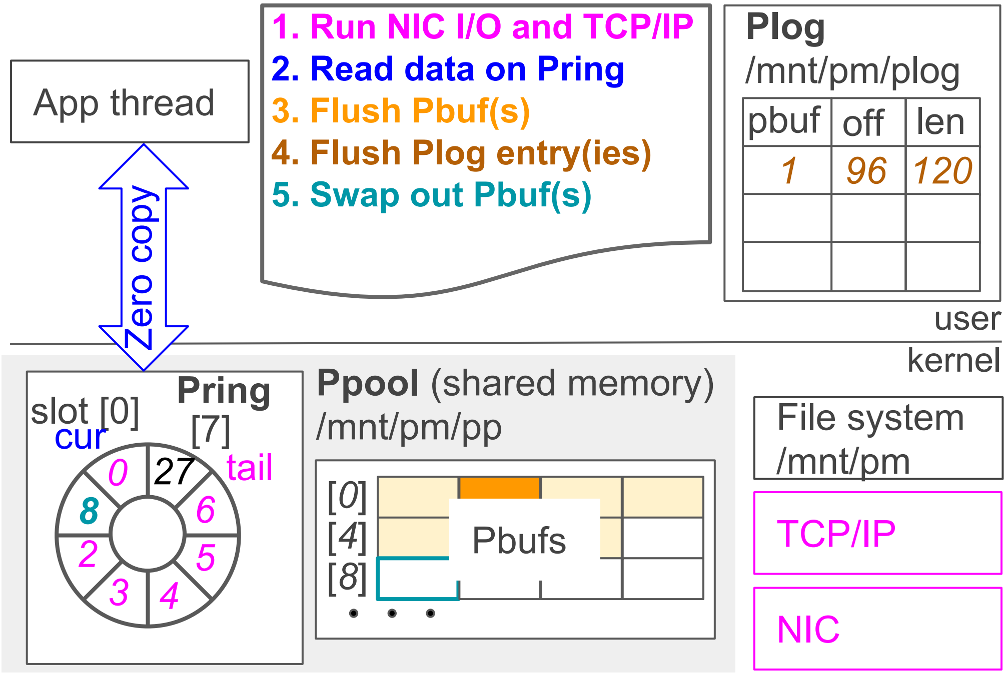
- **flush Pbuf data from CPU cache to DIMM**
    - clflush(opt) instruction

PASTE in Action



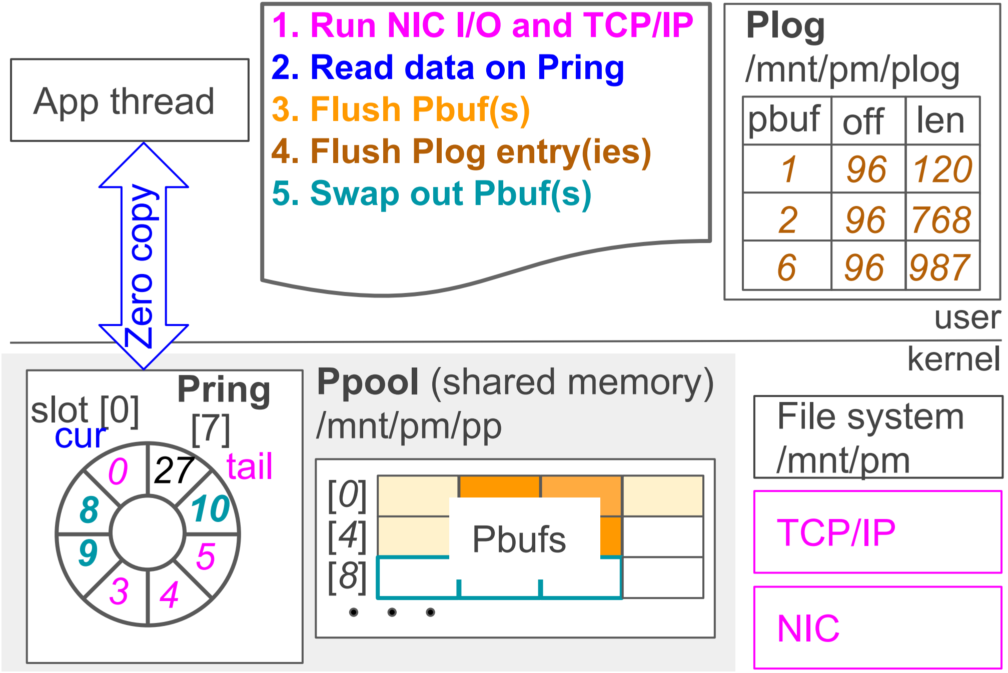
- **Pbuf is persistent data representation**
  - Base address is static i.e., file (/mnt/pm/pp)
  - Buffers can be recovered after reboot

PASTE in Action



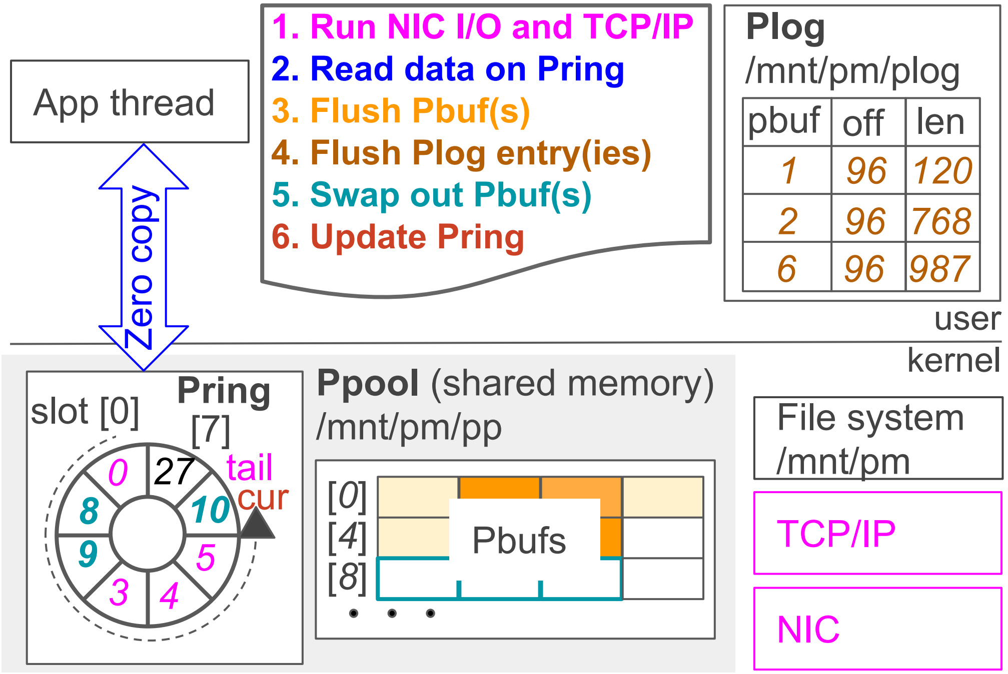
- **Prevent the kernel from recycling the buffer**

PASTE in Action



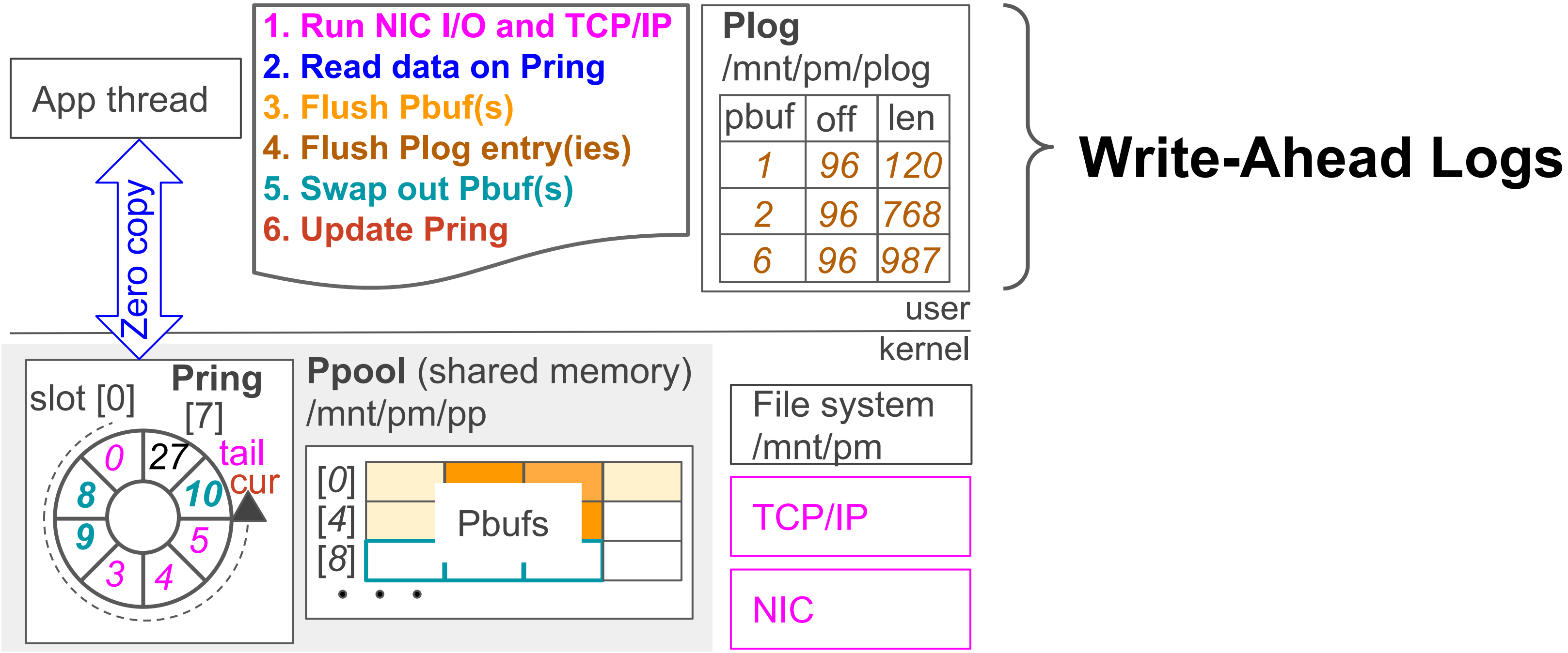
- Same for Pbuf 2 and 6

PASTE in Action

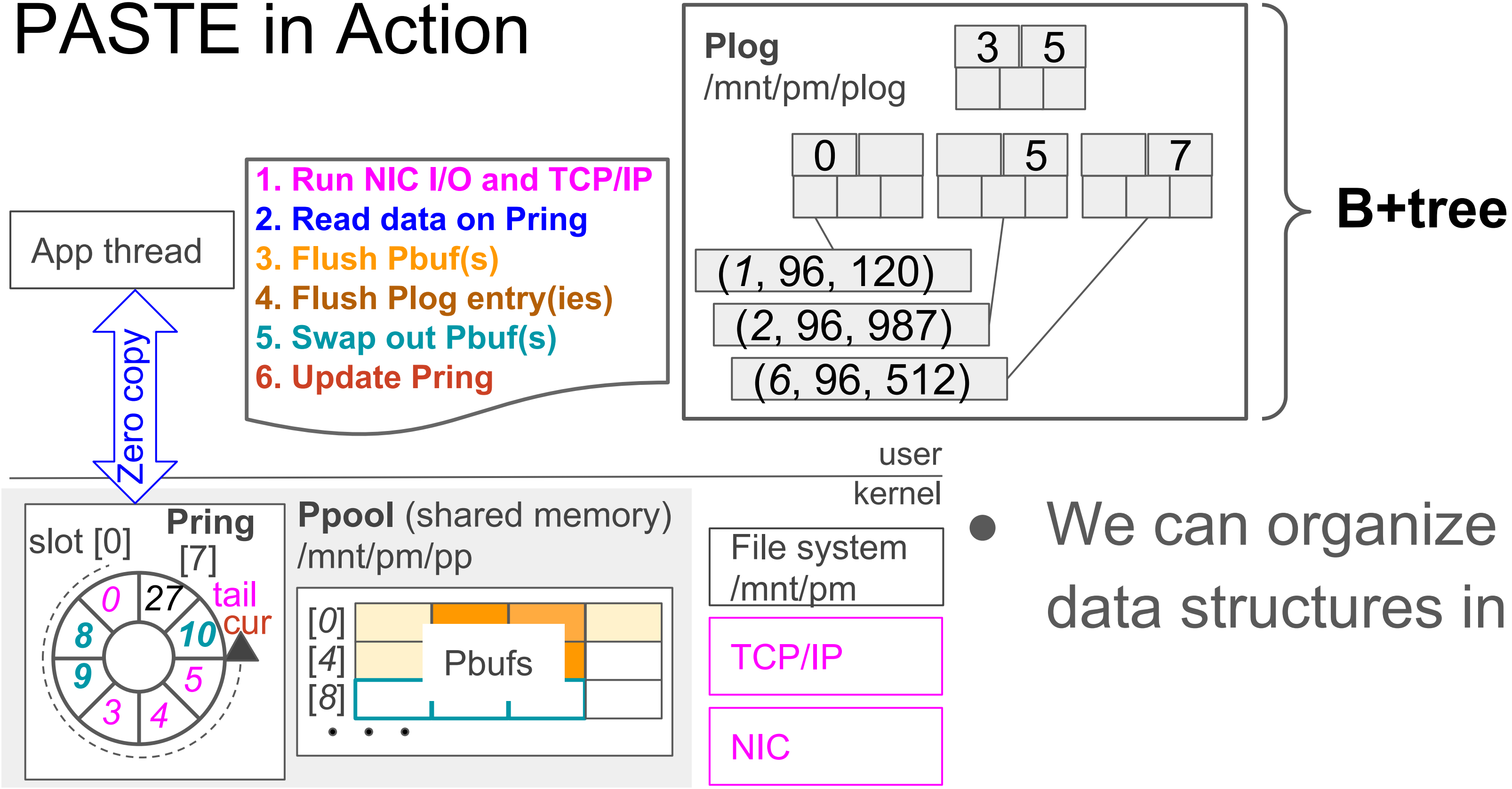


- Advance cur
  - Return buffers in slot 0-6 to the kernel at next poll()

PASTE in Action



PASTE in Action



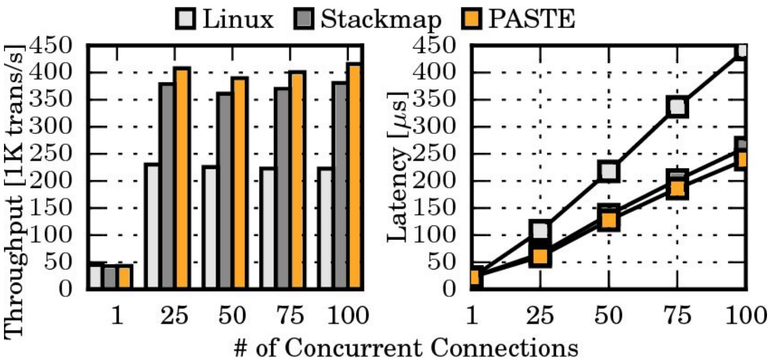
- We can organize various data structures in **Plog**

Evaluation

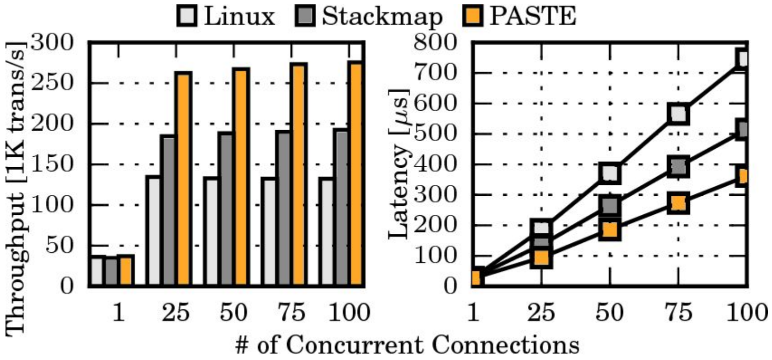
1. How does PASTE outperform existing systems?
2. Is PASTE applicable to existing applications?
3. Is PASTE useful for systems other than file/DB storage?

How does PASTE outperform existing systems?

64B



1280B

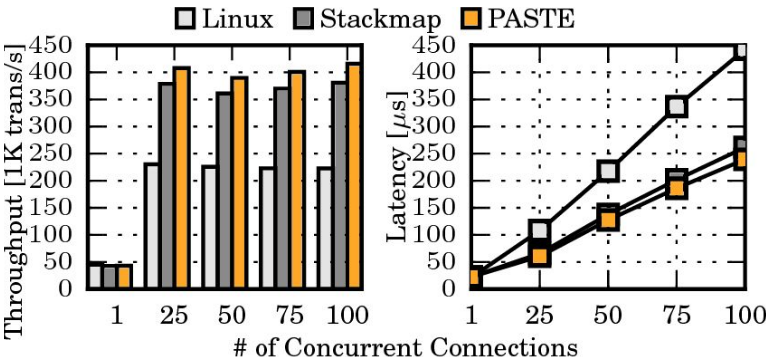


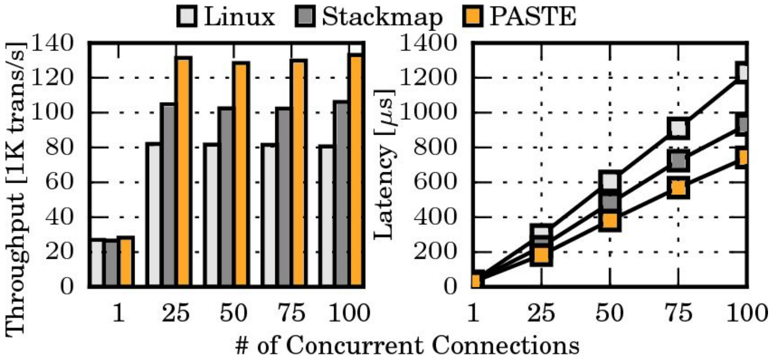
WAL

What if we use more complex data structures?

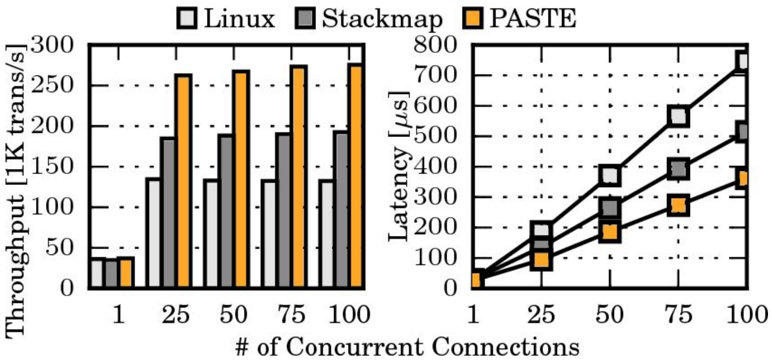
How does PASTE outperform existing systems?

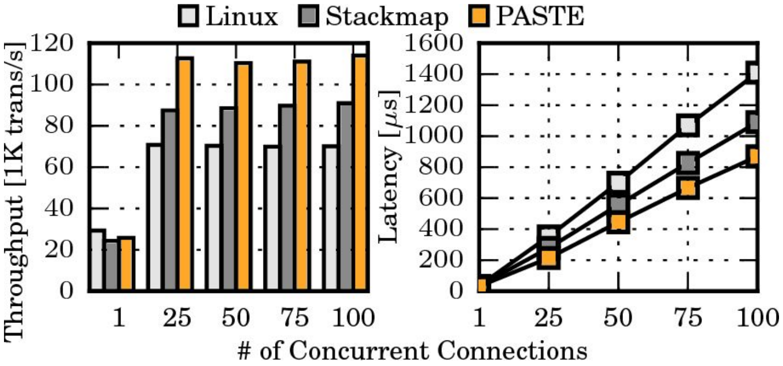
64B





1280B



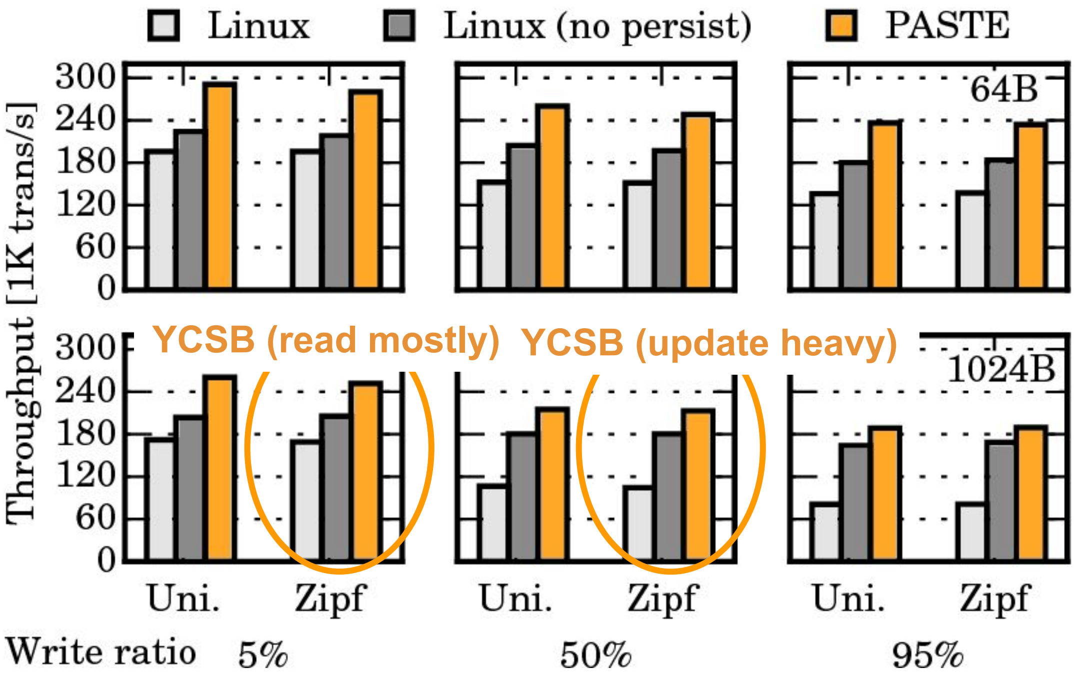


WAL

B+tree (all writes)

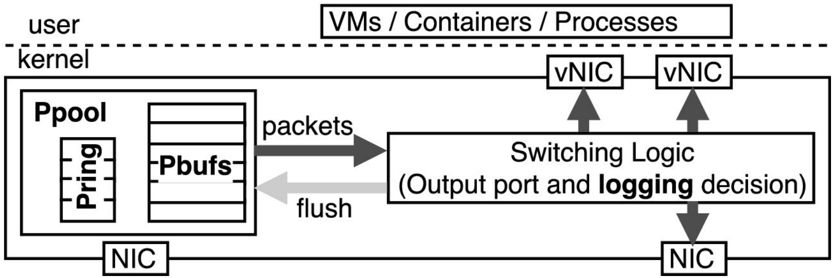
Is PASTE applicable to existing applications?

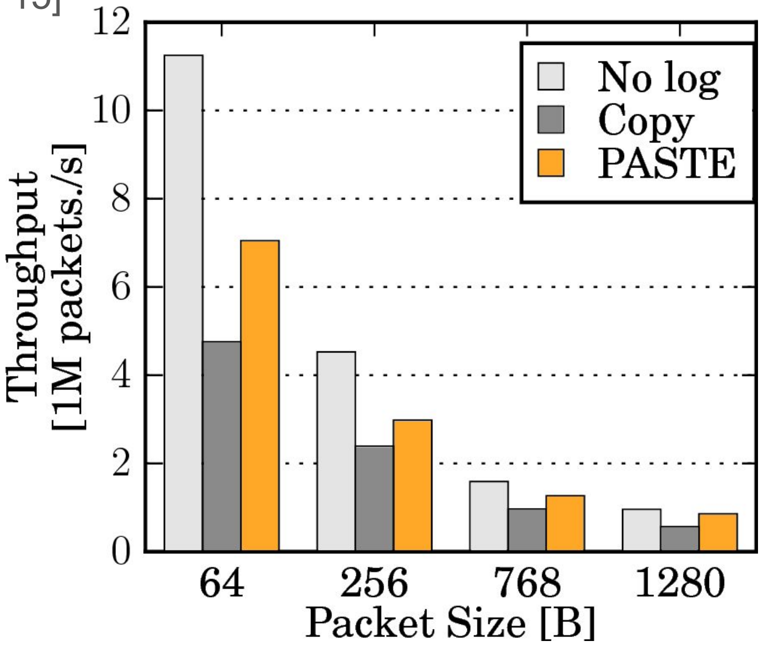
- Redis



Is PASTE useful for systems other than DB/file storage?

- Packet logging *prior to forwarding*
  - Fault-tolerant middlebox [Sigcomm'15]
  - Traffic recording
- Extend mSwitch [SOSR'15]
  - Scalable NFV backend switch





Conclusion

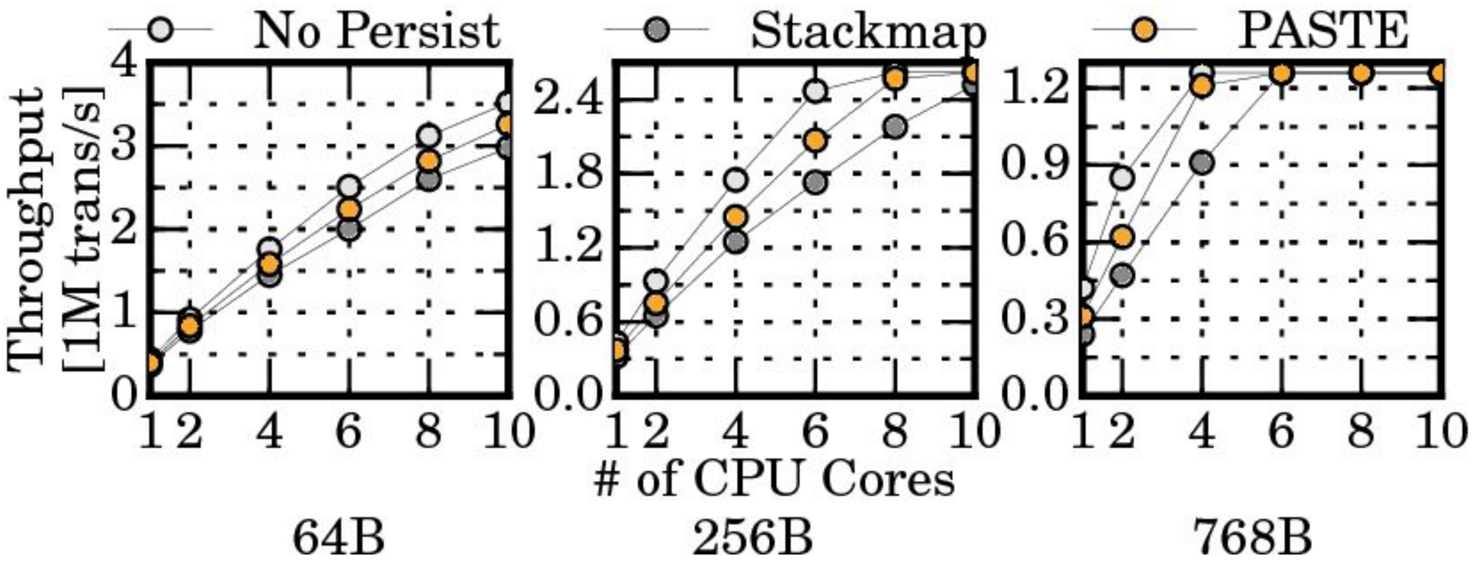
- PASTE is a network programming interface that:
  - Enables durable zero copy to NVMM
  - Helps apps organize persistent data structures on NVMM
  - Lets apps use TCP/IP and be protected
  - Offers high-performance network stack even w/o NVMM

https://github.com/luigirizzo/netmap/tree/paste

micchie@sfc.wide.ad.jp or @michioh

Multicore Scalability

- WAL throughput



Further Opportunity with Co-designed Stacks

- What if we use higher access latency NVMM?
  - e.g., 3D-Xpoint
- Overlap flushes and processing with clflushopt and mfence before system call (triggers packet I/O)
  - See the paper for results

Experiment Setup

- Intel Xeon E5-2640v4 (2.4 Ghz)
- HPE 8GB NVDIMM (NVDIMM-N)
- Intel X540 10 GbE NIC
- Comparison
  - Linux and Stackmap [ATC'15] (current state-of-the art)
  - Fair to use the same kernel TCP/IP implementation