Timely Mobile Routing: An Experimental Study

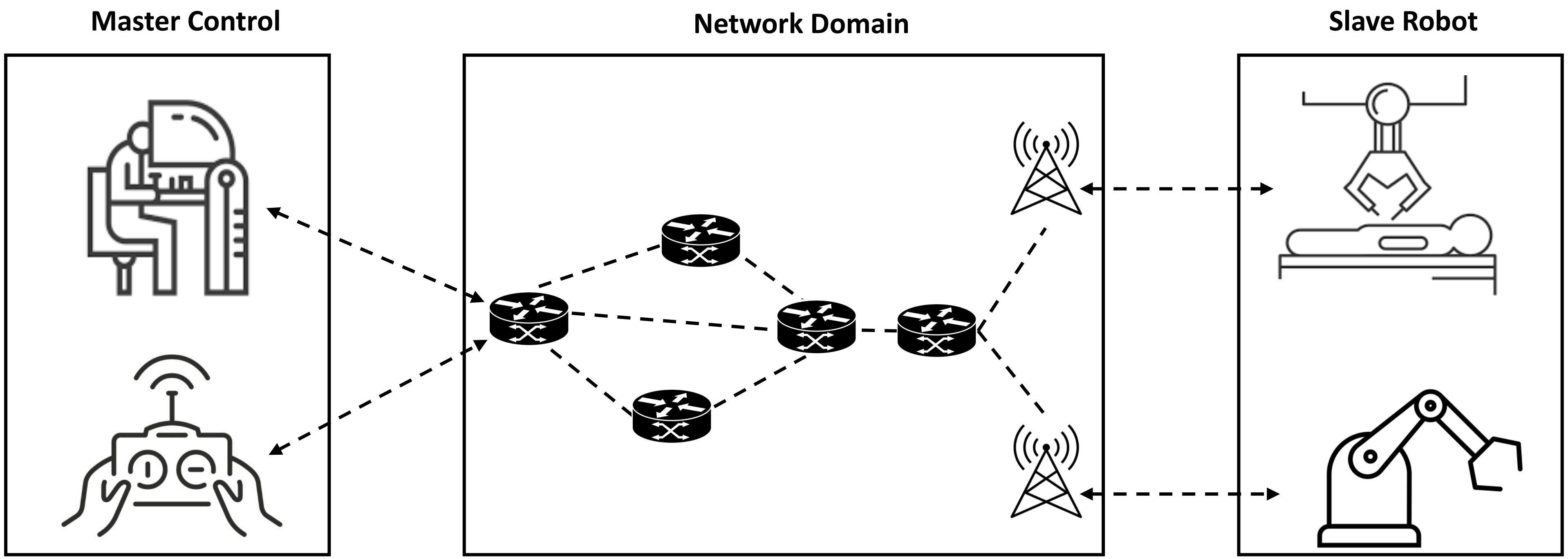
Vishakha Ramani, Jiachen Chen, Roy D. Yates
{vishakha, jiachen, ryates}@winlab.rutgers.edu
WINLAB, ECE Dept.

AoI INFOCOM Workshop
May 20, 2023

RUTGERS

WINLAB

Understanding the behavior of a single network node



End devices want *fresh* information

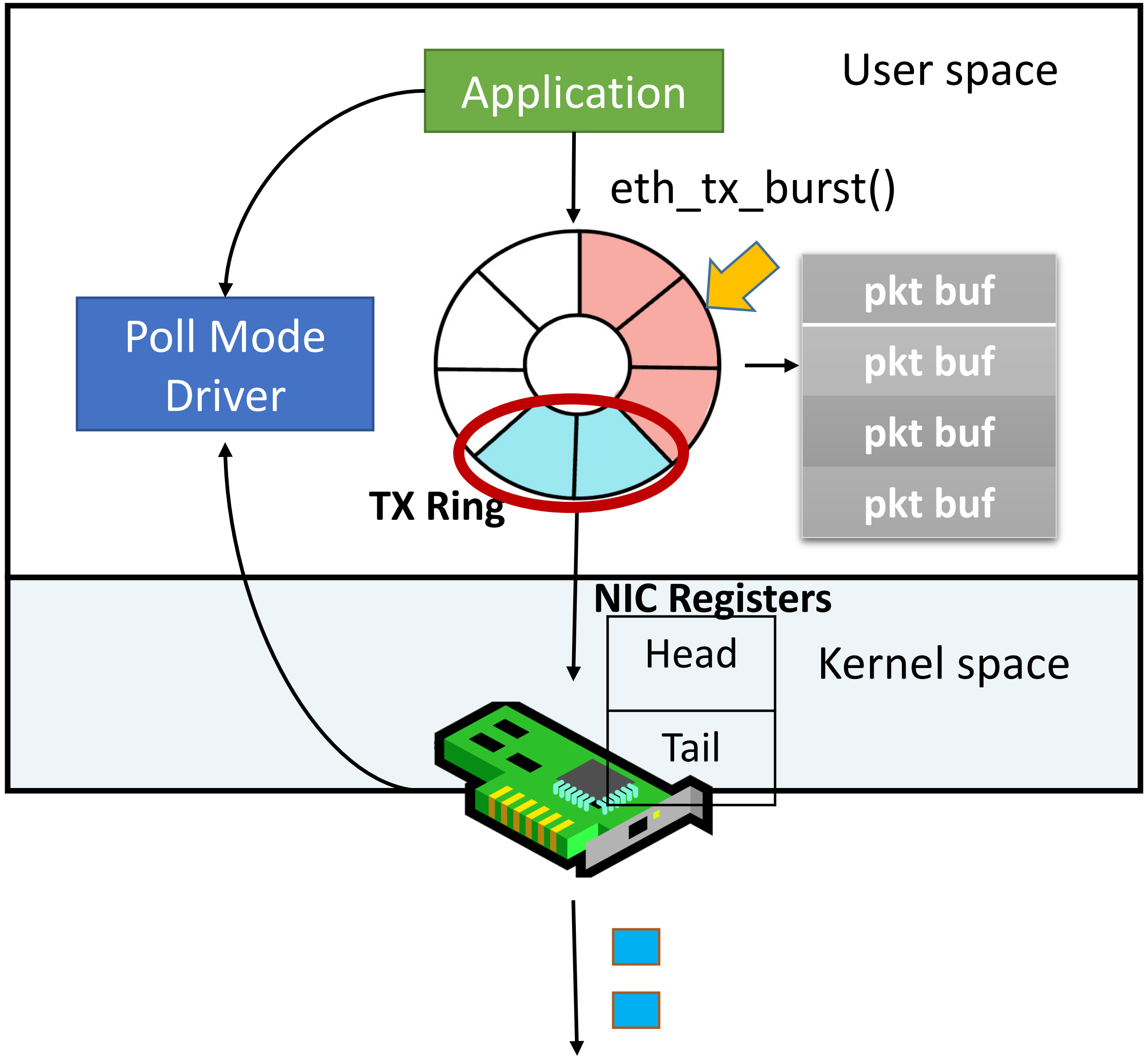
$1\ ms$ end-to-end latency (unidirectional)
$2\ ms$ cycle time

Application Processing
$250\ \mu s$

Robotic Device Processing
$250\ \mu s$

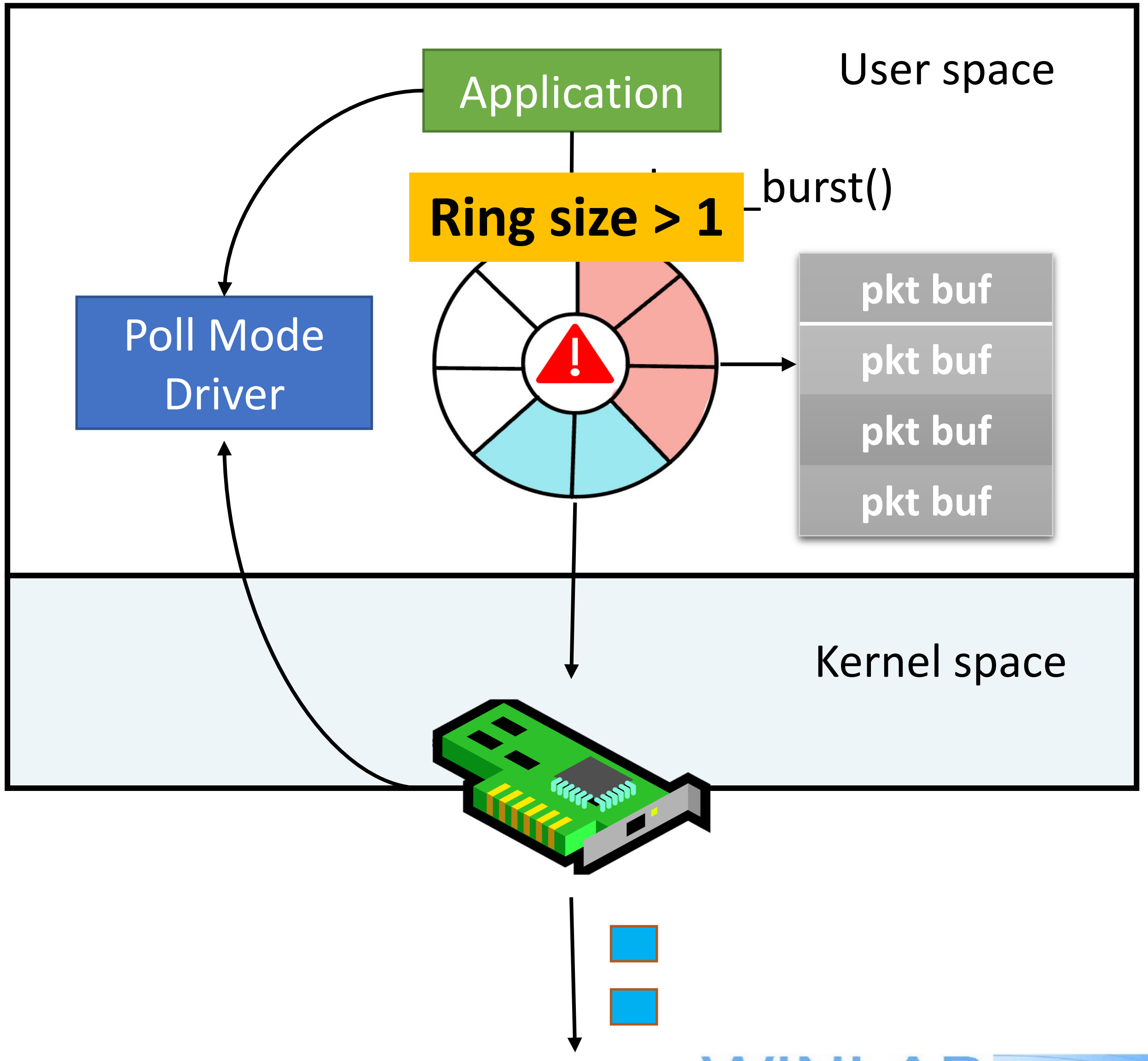
Network Processing
$500\ \mu s$

TX Path of Application Packet

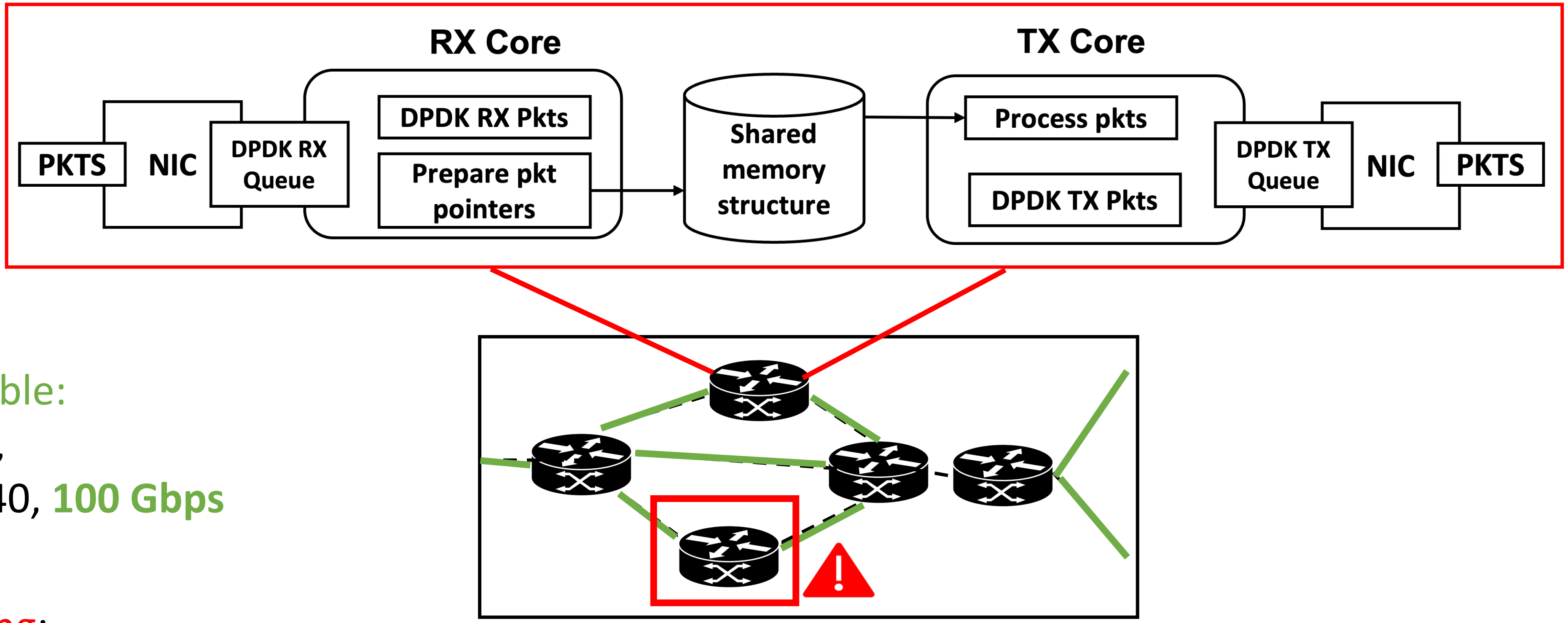


Impact of Input Queueing

- Large ring sizes beneficial in burst traffic.
- AOI favorable to “bufferless” systems.
  - Ring size = 1.
- Update packets become stale while queued in the ring.



High Performance Packet Processing

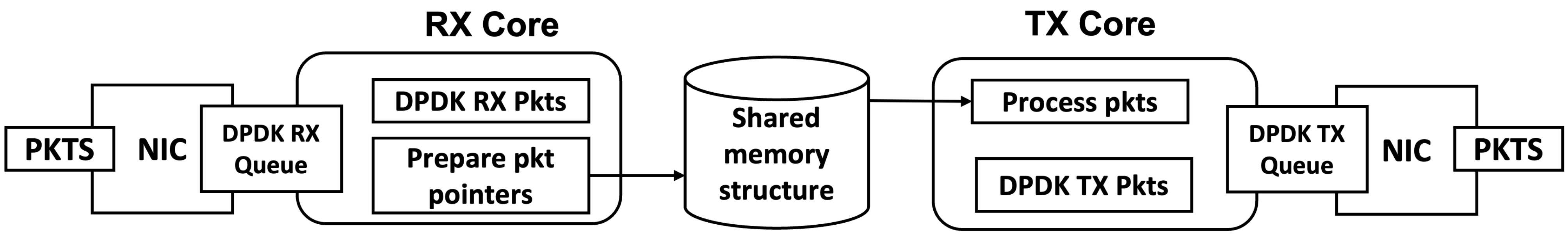


- Propagation delay is negligible:
  - Widespread – 10 Gbps,
  - Gaining traction – 25, 40, **100 Gbps**

- Bottleneck in packet handling:
  - I/O processing limits networks performance

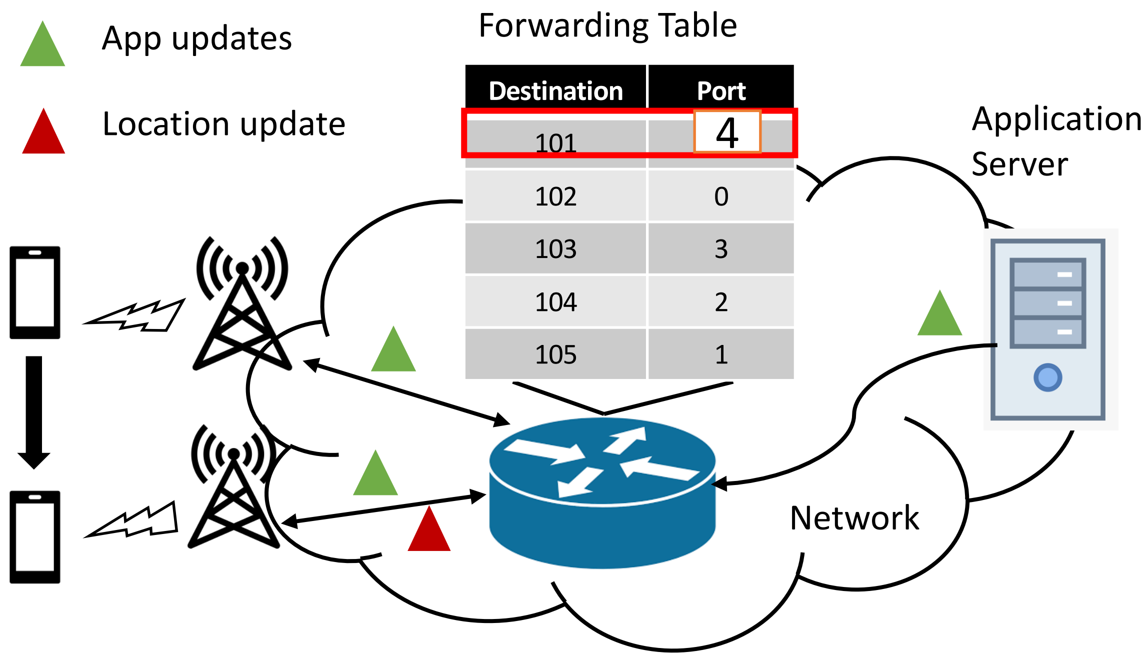
Key bottleneck: Synchronization between multiple processors accessing shared memory

Information Freshness In Shared Memory



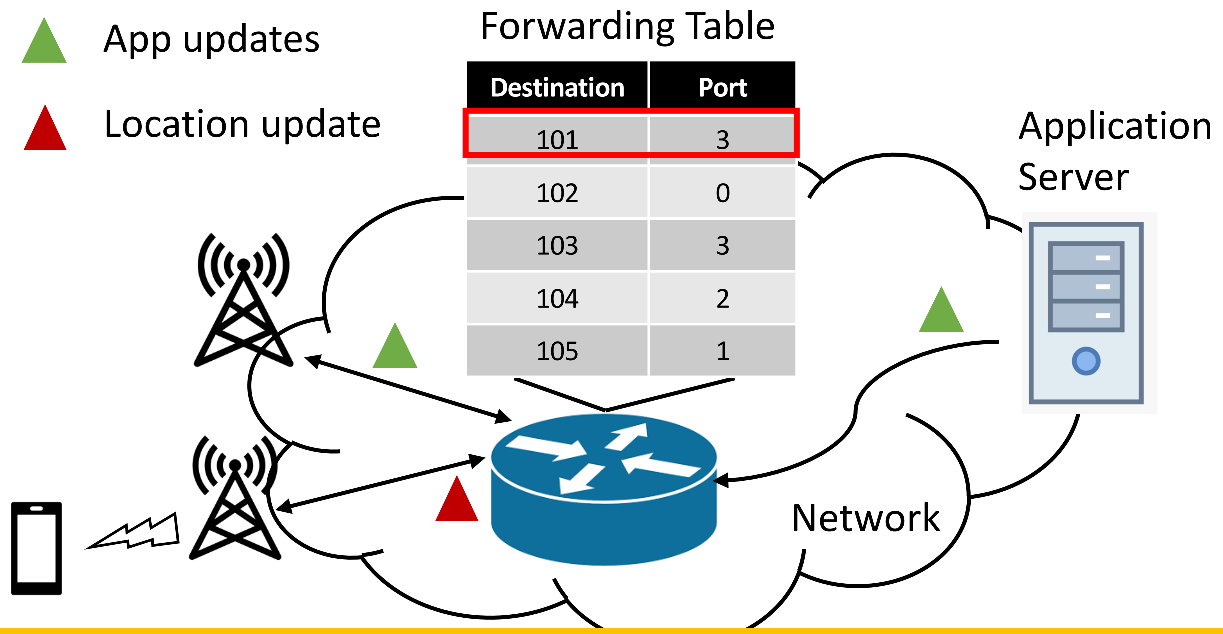
- Synchronization primitives on shared data structures:
  - Avoid race conditions
  - Ensure data correctness
- Impact of synchronization primitives
  - Delay in packet processing?
  - Timeliness of information?

Timely Update Forwarding



- Update forwarding with coupled updating processes:
  - *Location Updates* from mobile terminals stored/written in FIB.
  - *App Server Updates* addressed (by reading from FIB) and forwarded to mobile terminal.

Timely Update Forwarding



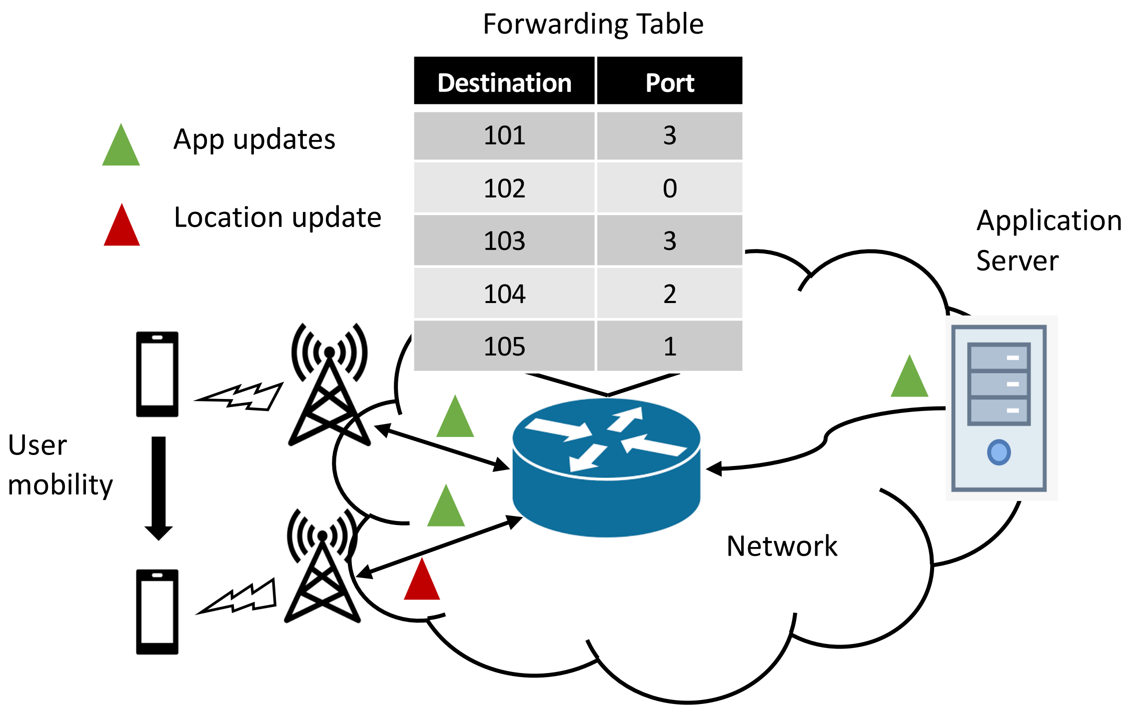
Destination	Port
101	3
102	0
103	3
104	2
105	1

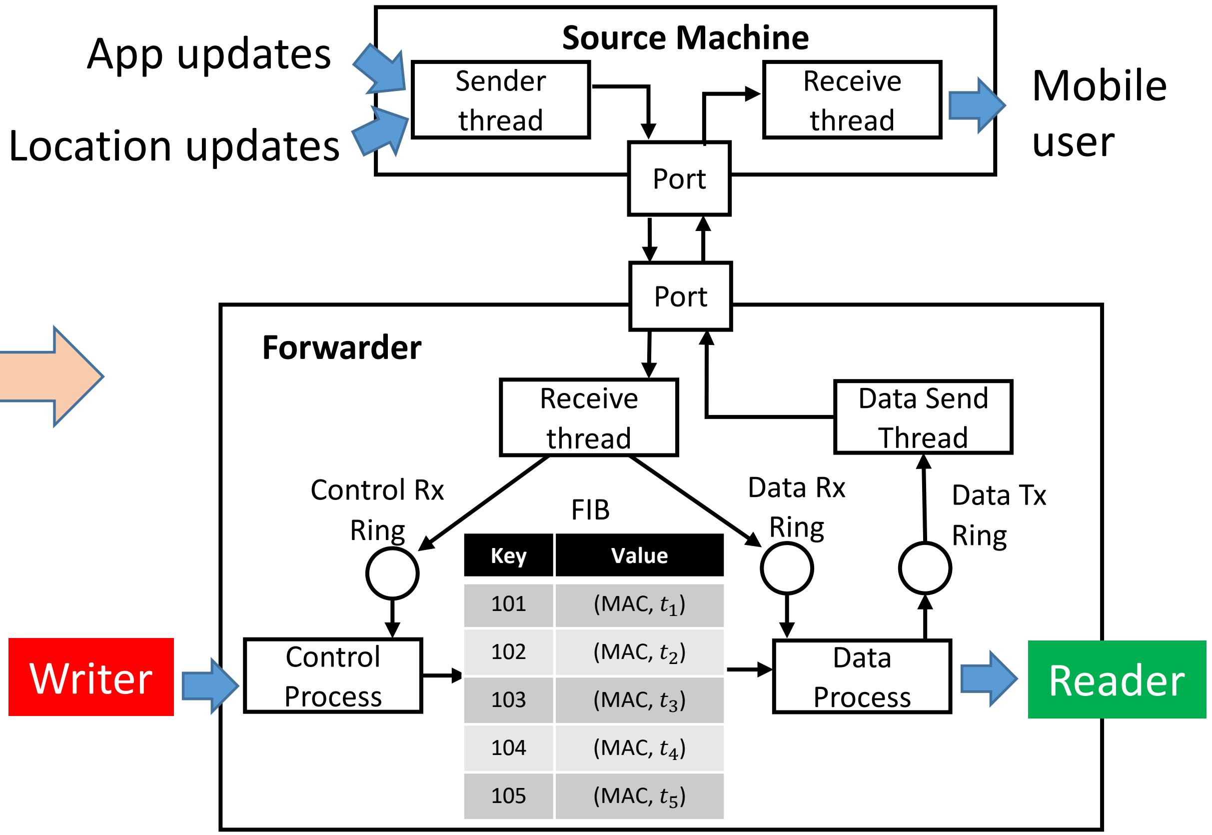
Performance Metric: Age of app update at the mobile terminal

App Update AoI bottlenecks:

- Updates queued at the forwarding node
- Misaddressed app updates due to synchronization primitives
  - lost in transit -> increased age at the mobile

Experiment Design and Testbed



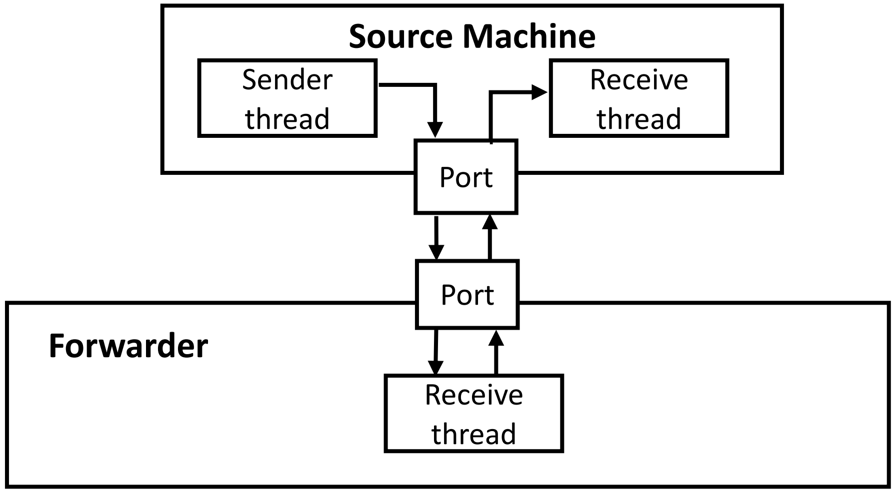


Packet forwarding testbed

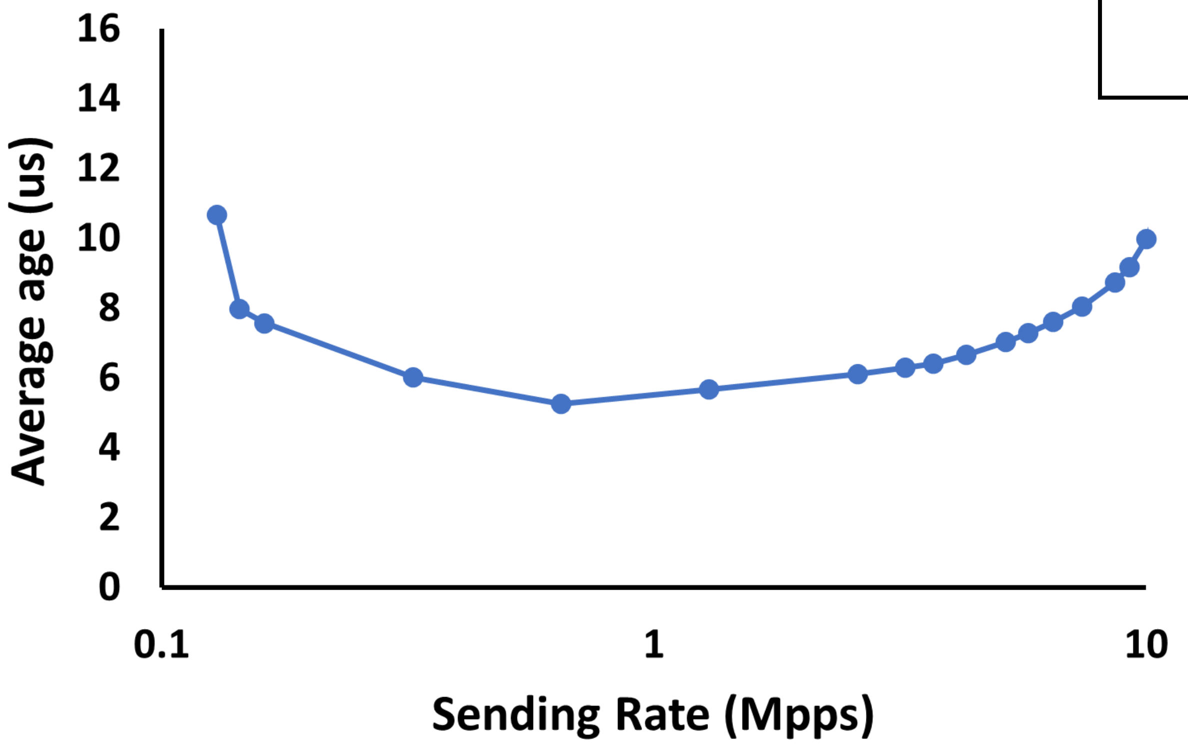
- **DPDK Experimental Eval:**
  - evaluate impact of *input queueing* caused by DPDK batch admission procedure.
  - quantitatively analyze the *effects of concurrency constructs*
  - using high-speed packet processing framework Data Plane Development Kit (DPDK).
  - Source and Forwarder machines on COSMOS testbed.

Experimental Evaluation of Input Queueing

- DPDK optimized for maximum packet throughput.
- DPDK allows user to access the port using a TX API
  - Sends a burst of output packets on a transmit queue of an Ethernet device.
- Throughput focused frameworks support timely delivery of information?

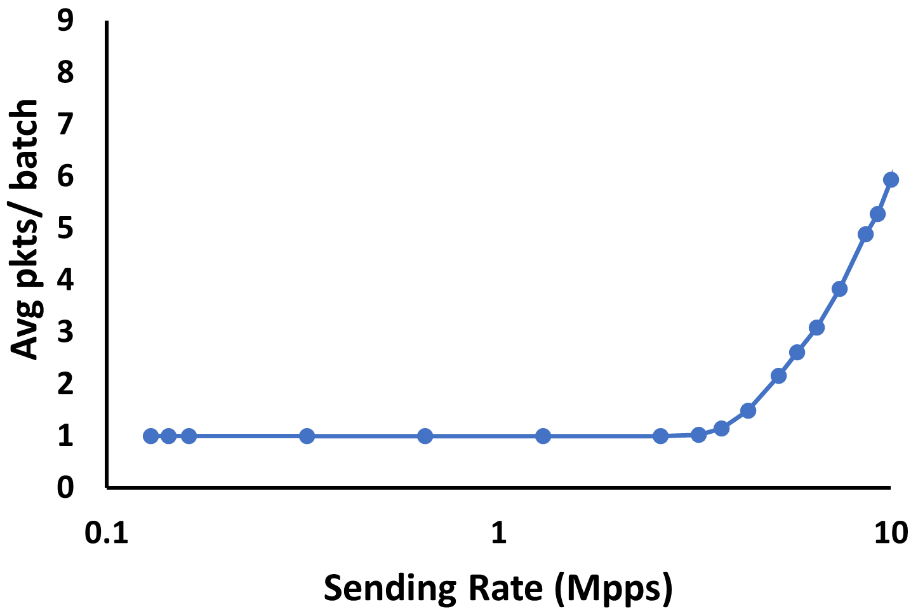


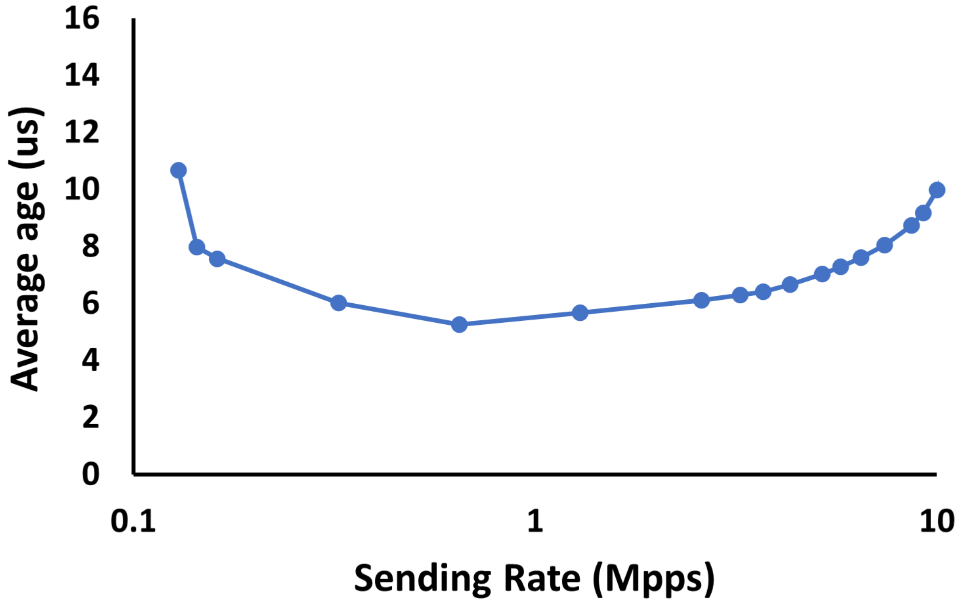
Baseline experiment setup



Experimental Evaluation of Input Queueing

- Use token bucket rate control mechanism to control the sending rate.





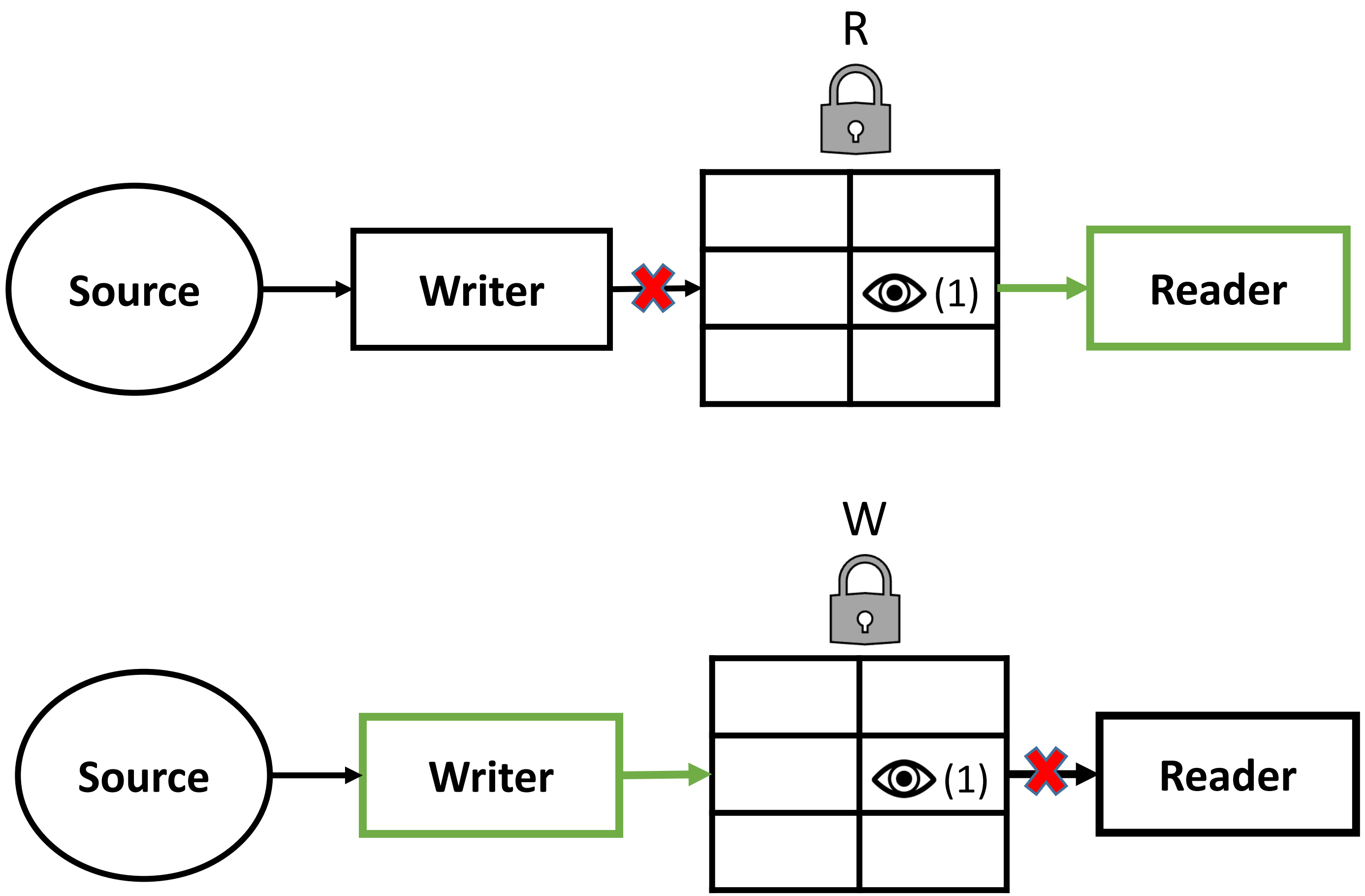
Throughput focused frameworks not necessarily favorable to timeliness

Experimental Evaluation of Synchronization Primitives

Compare Readers-Writer lock (RWL) and Read-Copy-Update (RCU)

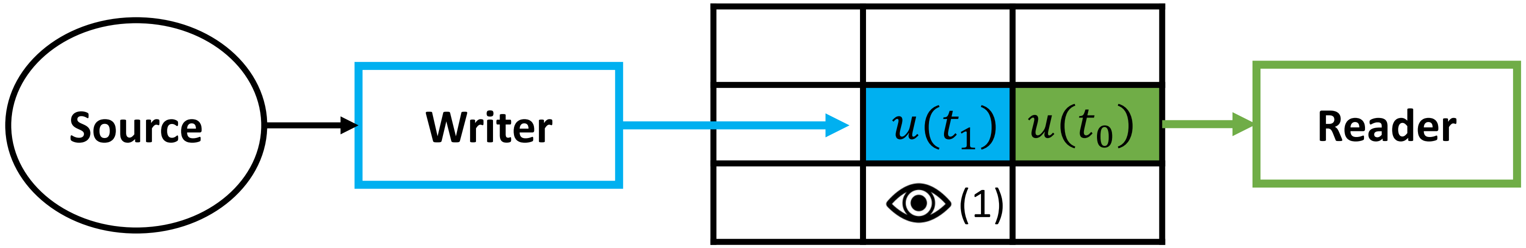
- as shared-memory synchronization primitives for FIB access

Synchronization Primitive: Readers-Writer Lock (RWL)



Enforces mutual exclusion between readers and writer.

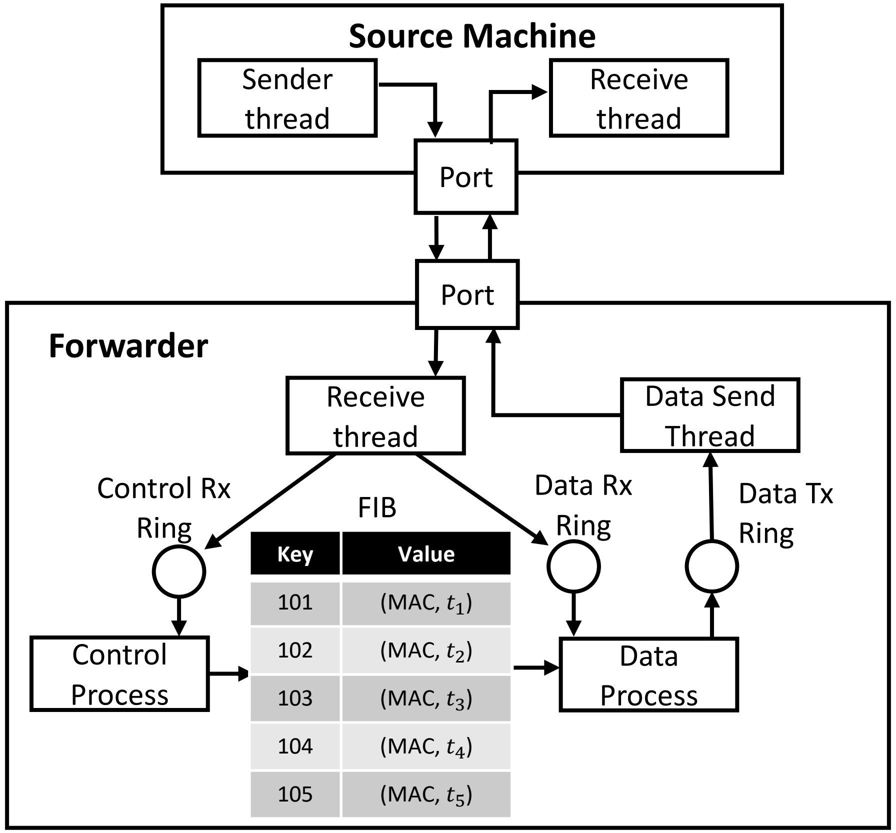
Synchronization Primitive: Read-Copy-Update (RCU)

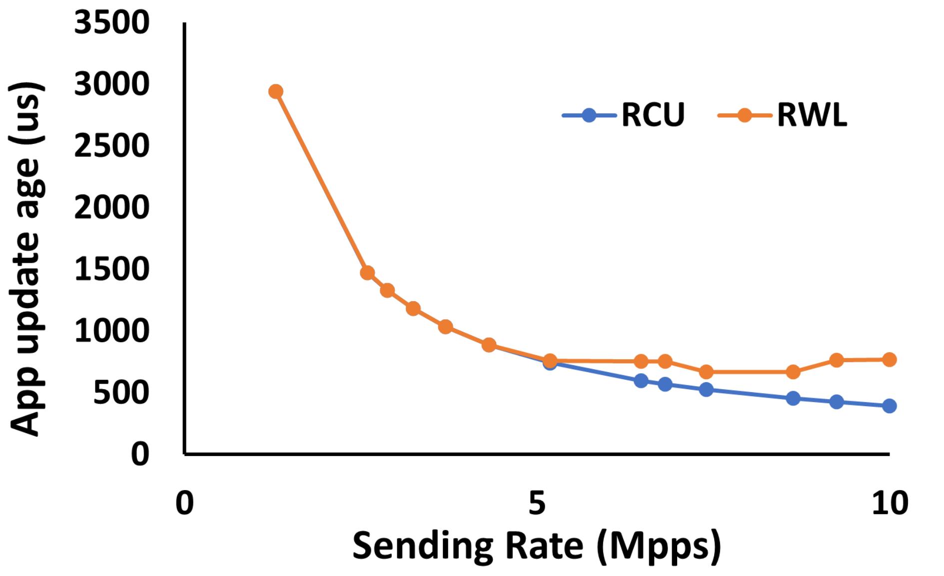


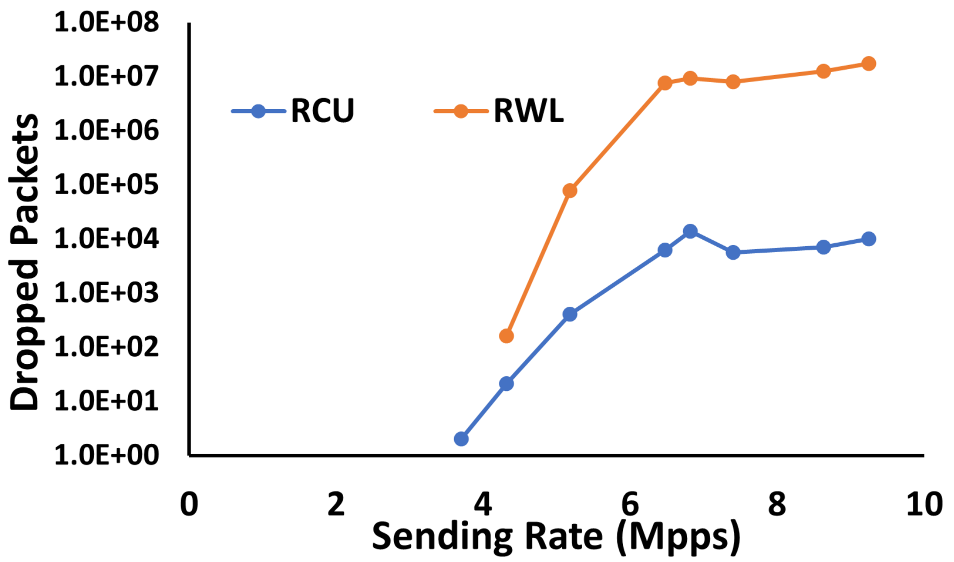
- Lock-less concurrency construct:
  - Concurrent forward progress of both readers and writer.

Experimental Evaluation of Synchronization Primitives

Low Contention



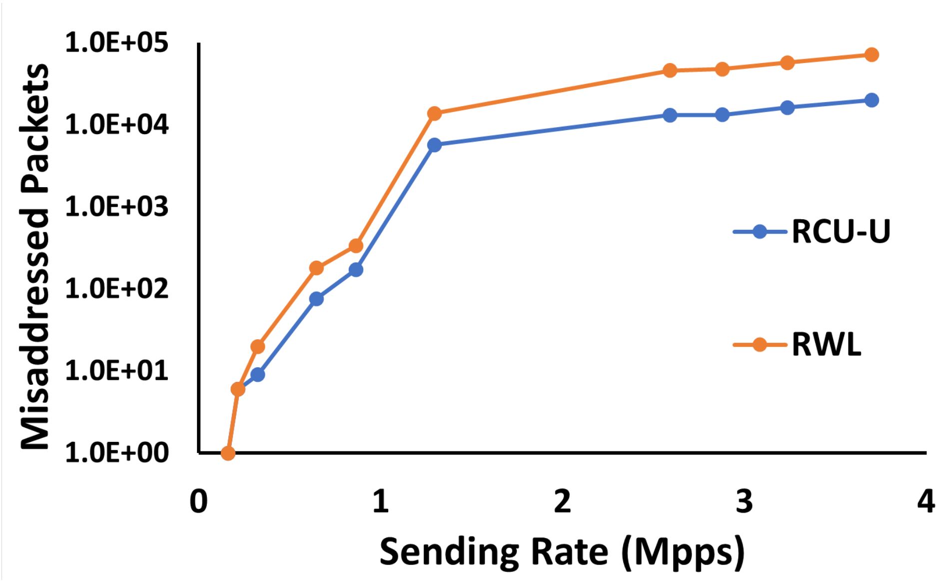


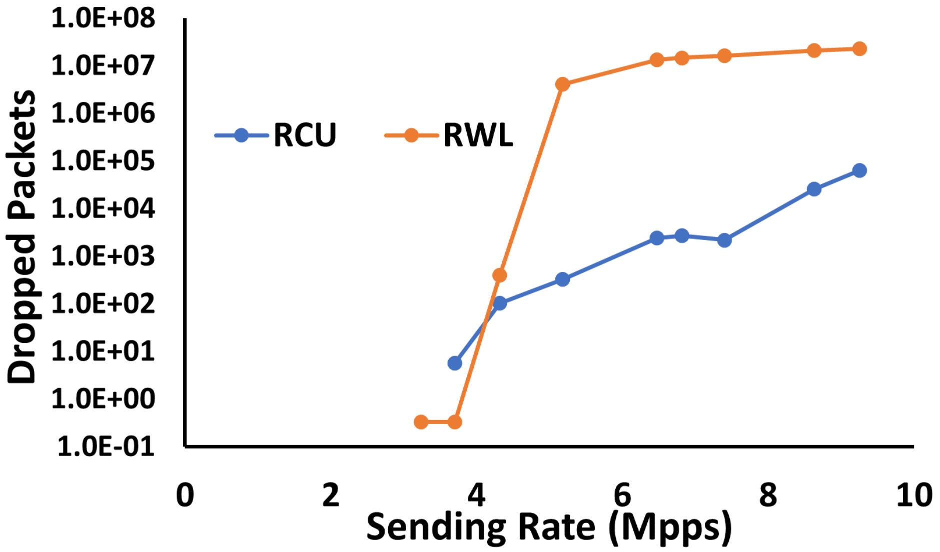


- Mutual exclusion in RWL
  - slows processing of app updates -> updates dropped at data process ring

Experimental Evaluation of Synchronization Primitives

High Contention





- RCU writes are heavy and RWL enforces mutual exclusion
  - Slow processing of location updates -> updates dropped at control process ring.
  - FIB updates with stale location updates.

- RCU reads are faster that RWL
  - Slow processing of app updates -> updates dropped at data process ring

Conclusion

- Designed and implemented a DPDK-based packet forwarding experiment
- Evaluated AoI performance of Readers-Writer lock (RWL) and Read-Copy-Update (RCU)
- Complex interactions between FIB synchronization mechanisms and packet queueing
- Lesson: Optimize packet processing frameworks for timely status updating