Taming Latency In Data Center Applications

Ph.D. Defense of Dissertation

Mohan Kumar Advisor: Taesoo Kim

Motivation: Importance of Latency

Speed Matters for Google Web Search

Jake Brutlag
Google, Inc.
June 22, 2009

The Cost of Latency

retail giant found that every 100ms of latency cost them 1
same time, a study by Tabb Group revealed that a broker
$4 million in revenues per millisecond if its electronic tradi
5ms behind the competition.

Although low-latency connectivity has become more com
Amazon, Tabb and others began this conversation about l
ago, low latency is still a critical element of business succe
practically a foregone conclusion that all companies these days will prioritize

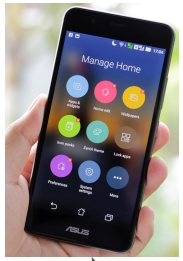
DOI:10.1145/3015146

Microsecond-scale I/O means tension

Latency Critical In Data Center Applications

Attack of the Killer Microseconds

Data Center Applications



Key-value Stores

redis

Web Servers

APACHE HTTP SERVER

NGINX

Distributed Services

Apache ZooKeeper™

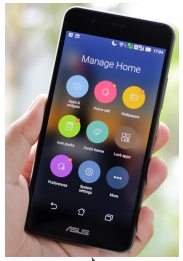
Cloud Spanner

Contemporary Data Center Characteristics

- Optimized network – microsecond round-trip time
- Moving from 10/25 Gbps to 100/200 Gbps network
- Software running in servers induce high latency:
  - ➢ 66% of the inter-rack latency [1]
  - ➢ 81% of the intra-rack latency [1]

[1] Network requirements for resource disaggregation, OSDI'16

Data Center Applications - Server Latency



Protocol stack - 80% overhead

TLB shootdown - 30% overhead

Key-value Stores

redis

Web Servers

APACHE HTTP SERVER

NGINX

Consensus - 82% overhead

Distributed Services

Apache ZooKeeper™

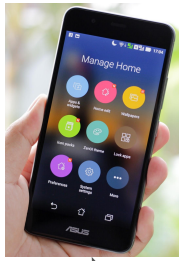
Cloud Spanner

bing

Thesis Statement

System abstractions and optimizations are needed at different levels of the software stack, from the software services running in the user space and the kernel to the software running on SmartNICs, to reduce the latency and improve the throughput of current data-center applications.

Data Center Applications - Server Latency



Protocol stack - Xps

TLB shootdown - LATR

Key-value Stores

redis

Web Servers

APACHE HTTP SERVER

NGINX

Consensus - Dyad

Distributed Services

Apache ZooKeeper™

Taming Application Latency- Thesis

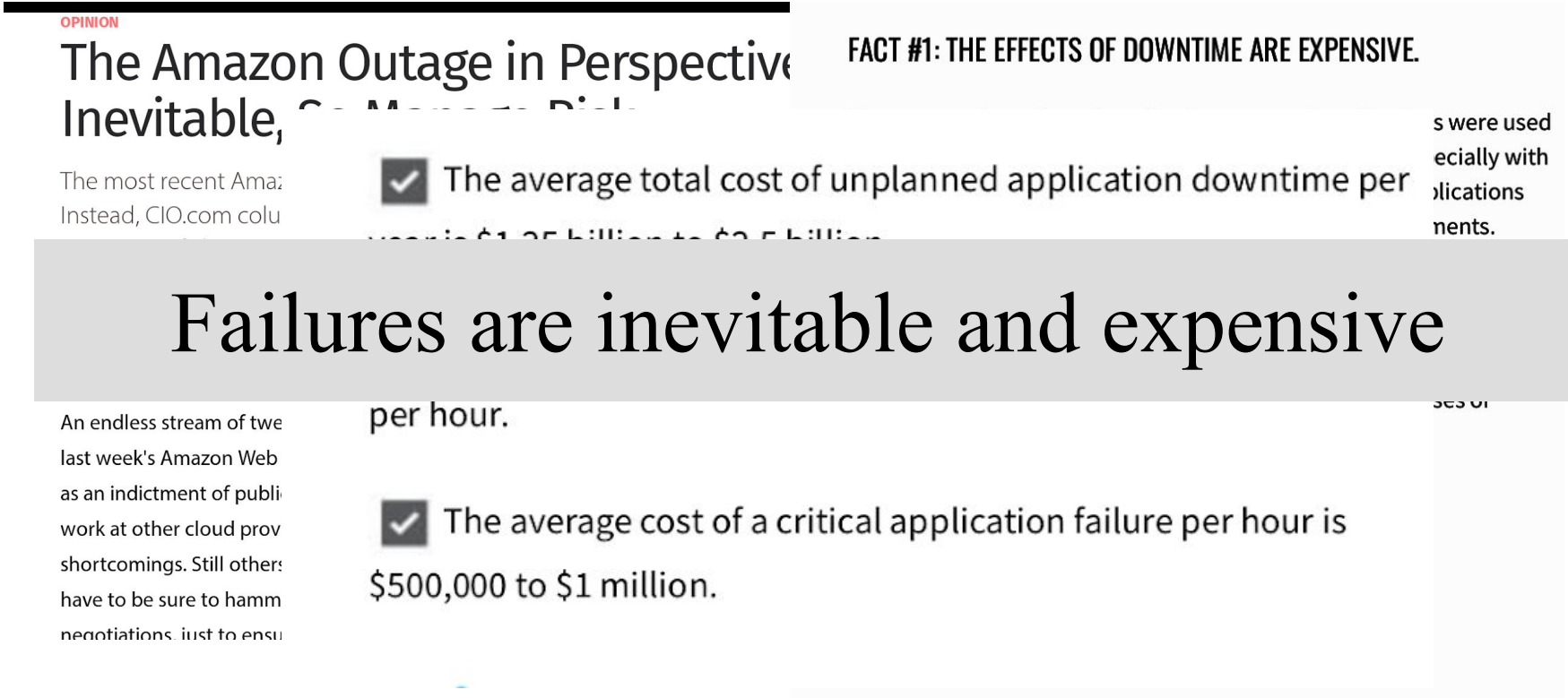
- Xps - Extensible Protocol Stack:
  - ➢ Abstraction in kernel and user-space protocol stacks, and SmartNICs
  - ➢ Reduces Redis latency by up to 73.3%
- LATR - Lazy Translation Coherence:
  - ➢ Kernel mechanism for free operations, page migration and swapping
  - ➢ Reduces Apache latency by up to 26.1%

Taming Application Latency - Thesis

- Dyad - Untangling Logically-Coupled Consensus:
  - ➢ Abstraction in SmartNIC for consensus
  - ➢ Reduces timestamp server latency by up to 79%

Dyad: Untangling Logically-Coupled Consensus

Motivation - Consensus Algorithms

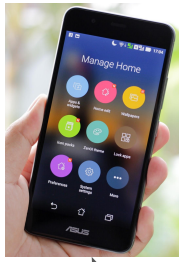


Failures are inevitable and expensive

Consensus Algorithms

- Consensus Algorithms:
  - Provides high availability by state machine replication
  - Keeps data consistent - linearizable
  - Consensus algorithms:
    - Multi-Paxos/Viewstamp Replication (VR)
    - Raft and Zookeeper Atomic Broadcast (ZAB)

Consensus Algorithms - Applications



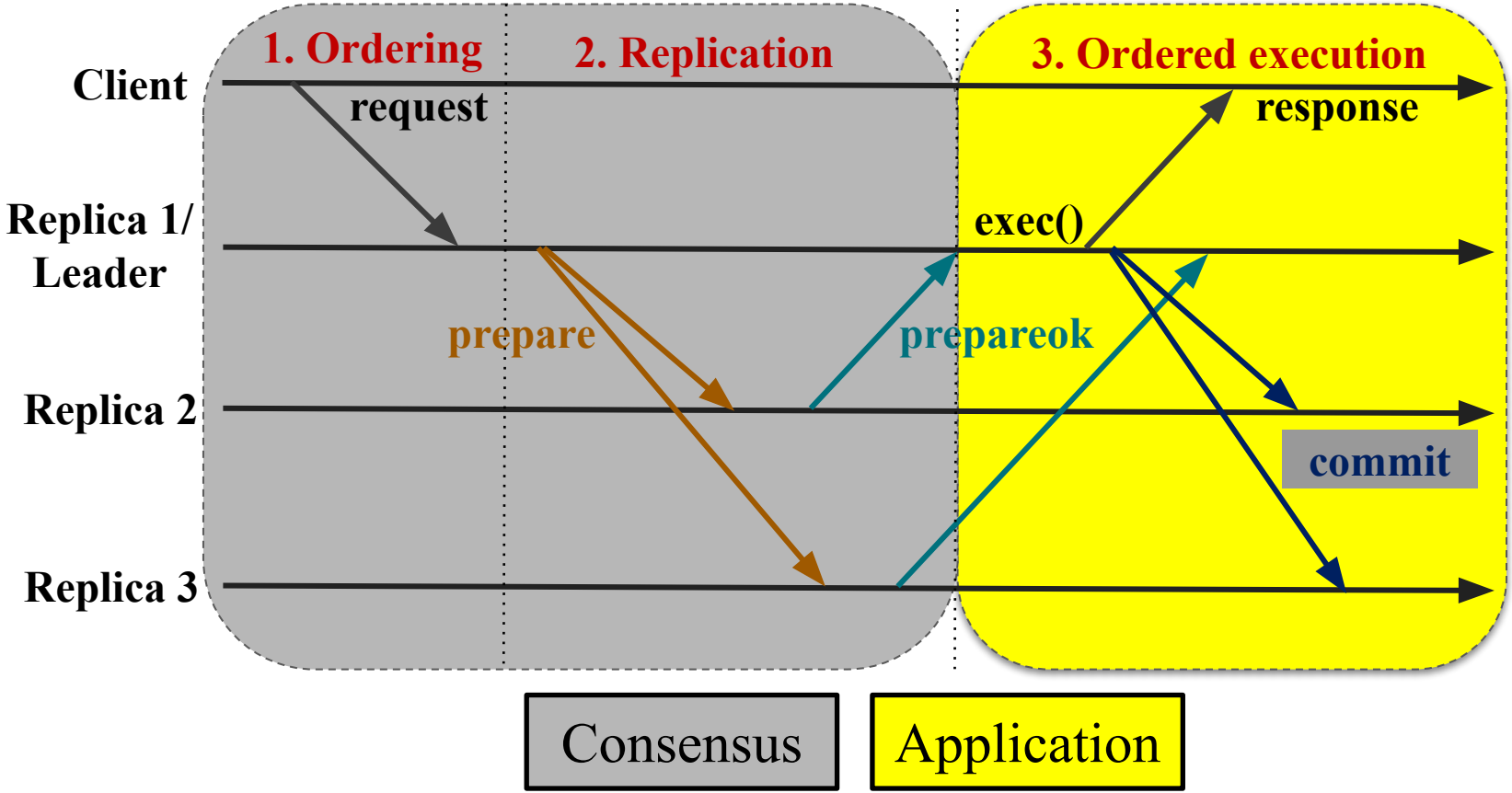
Distributed Services

- Timestamp Servers
- Key-value stores
- Database
- Lock managers

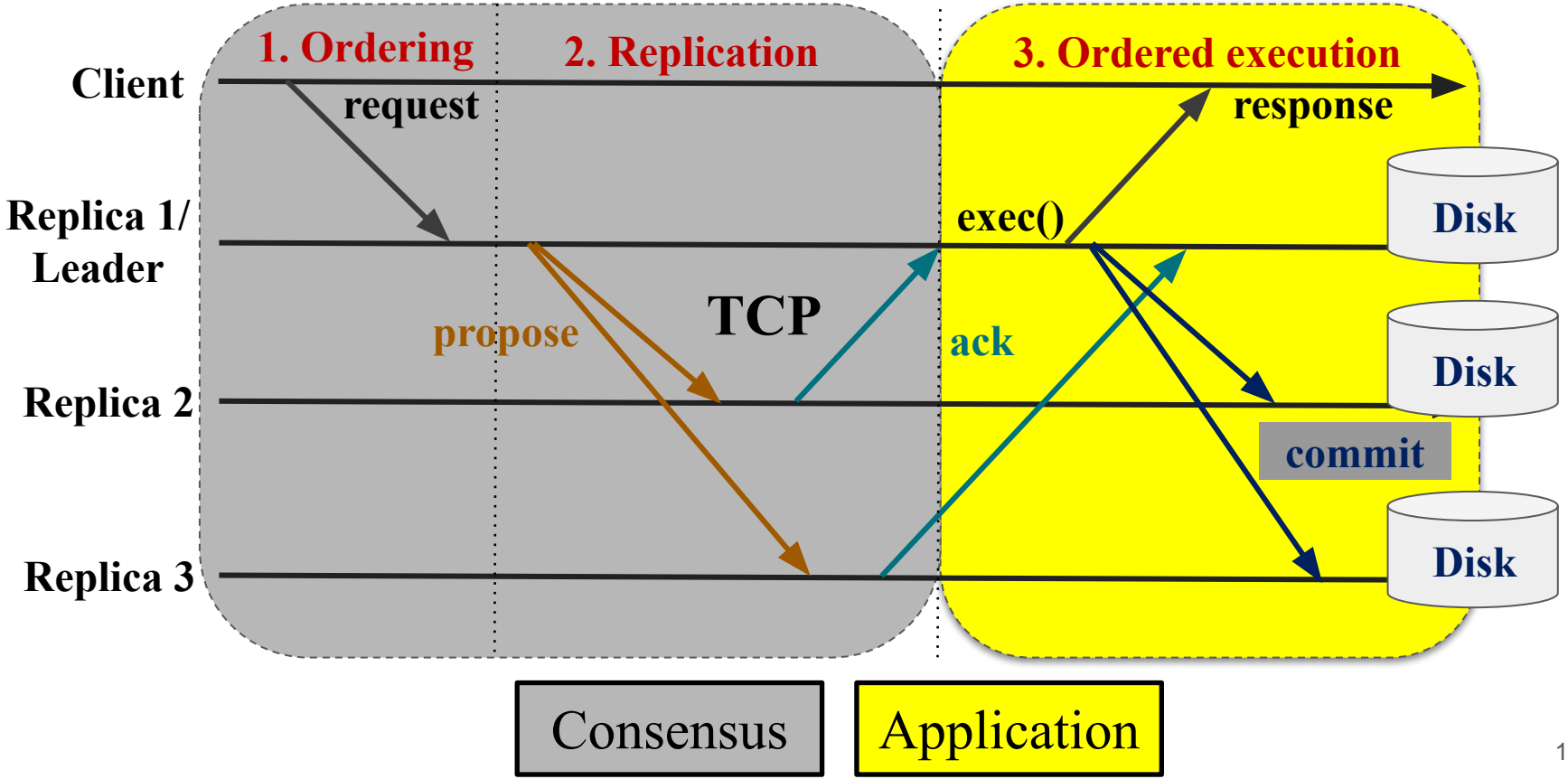
Dyad: Untangling Logically-Coupled Consensus

- **Background**
- Overview
- Design and Evaluation
- Conclusion

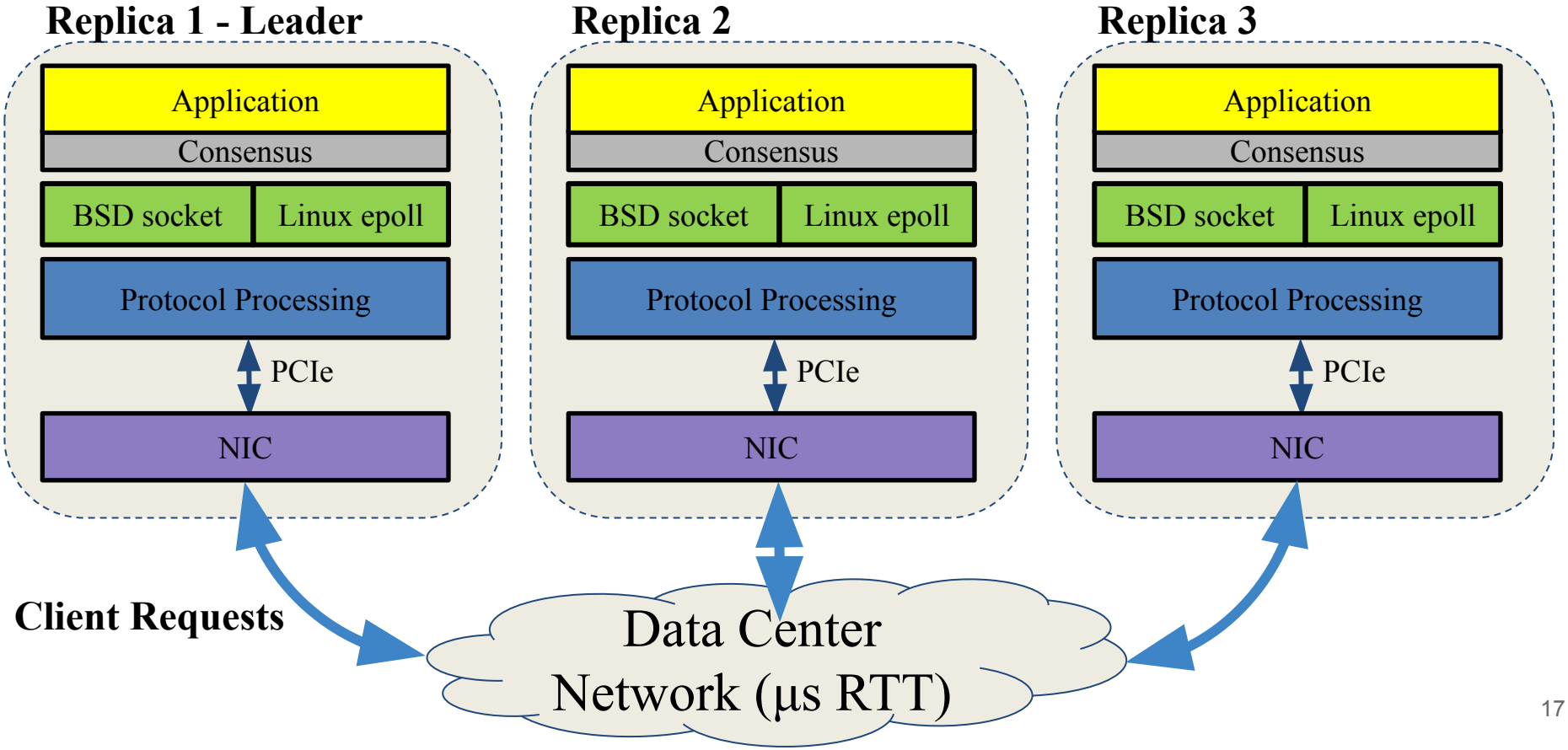
Consensus – VR Data Operation



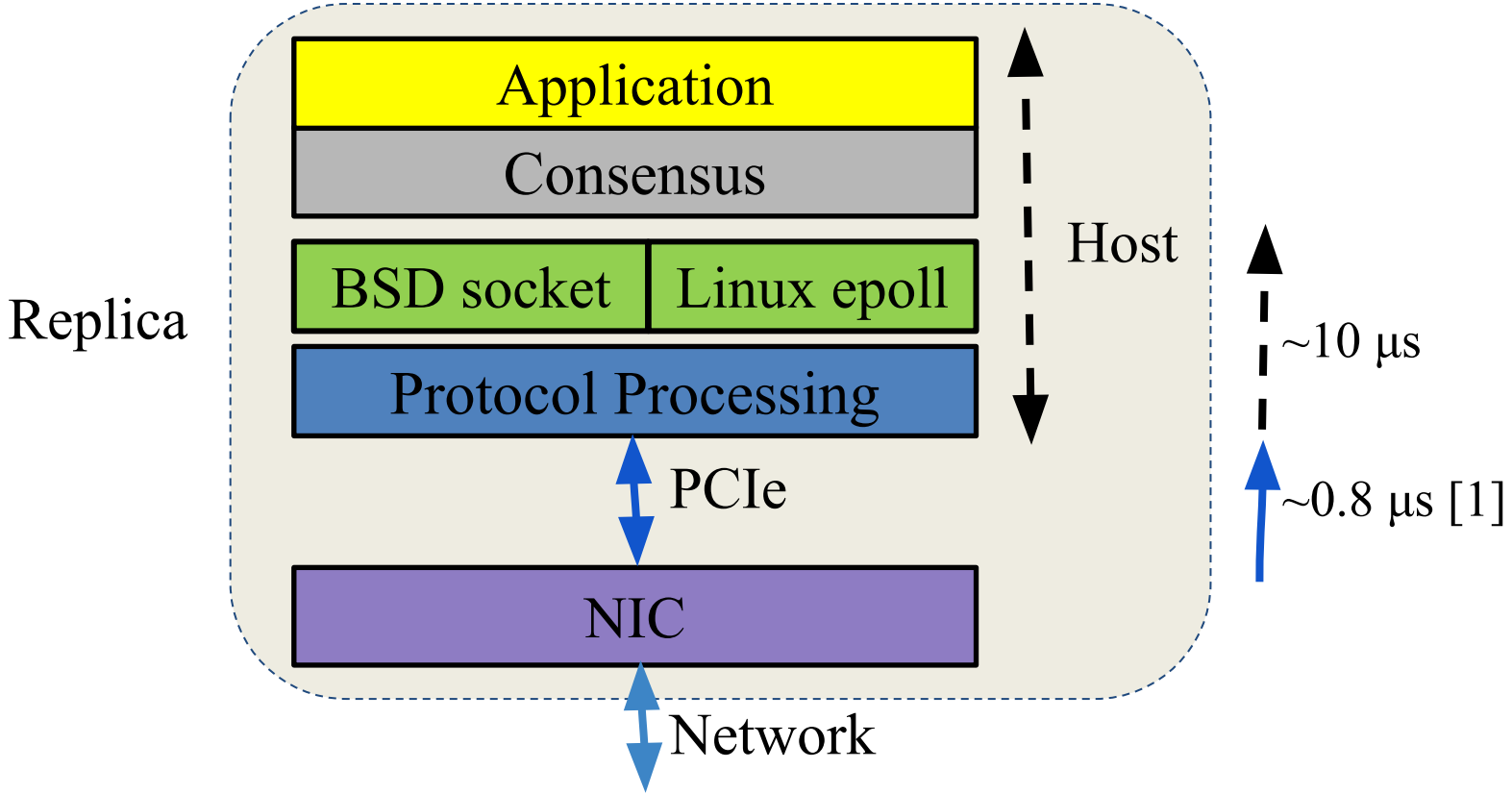
Consensus – ZAB or Raft Data Operation



Replicas in a Data Center

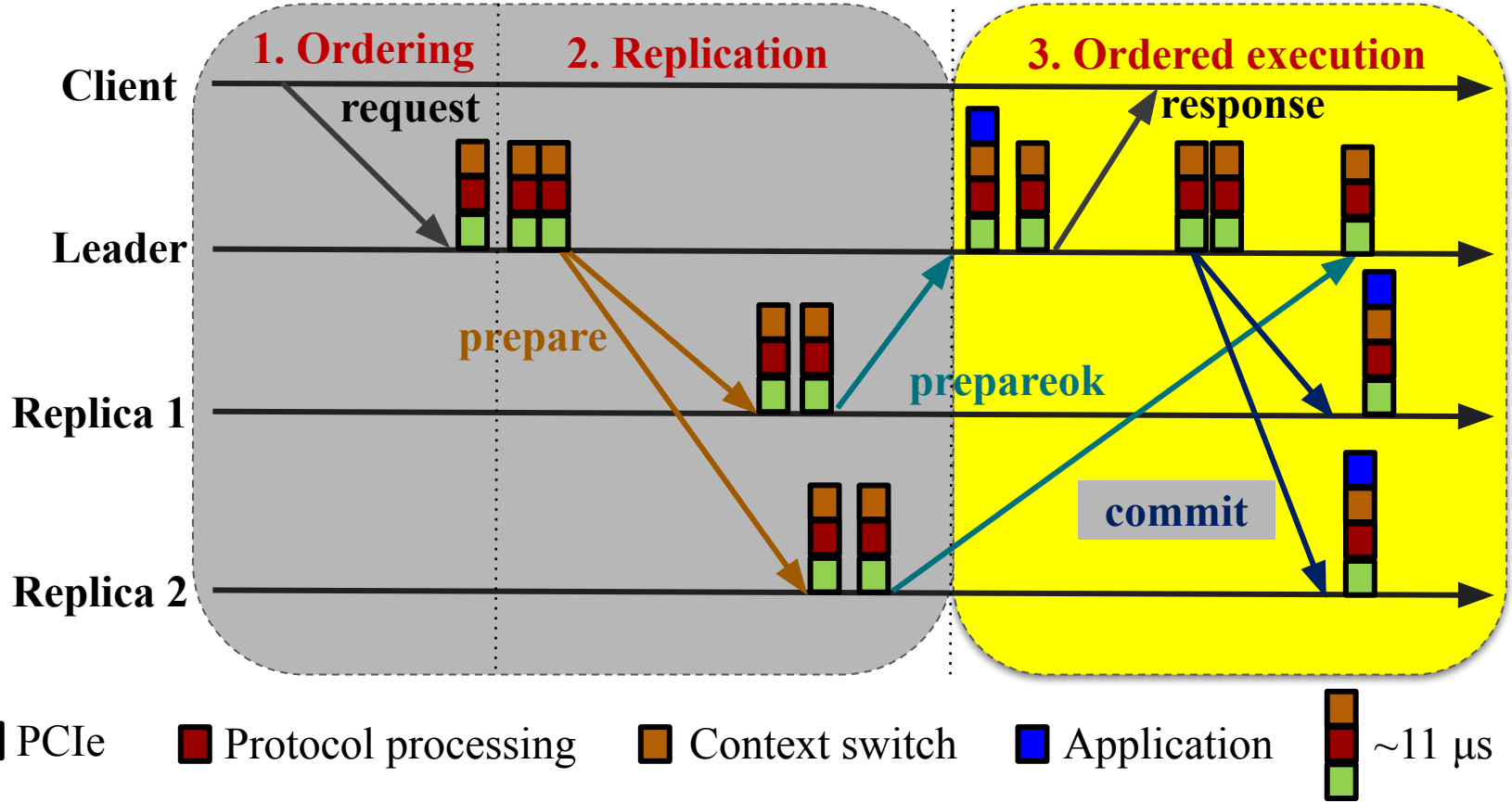


Logically-Coupled Consensus

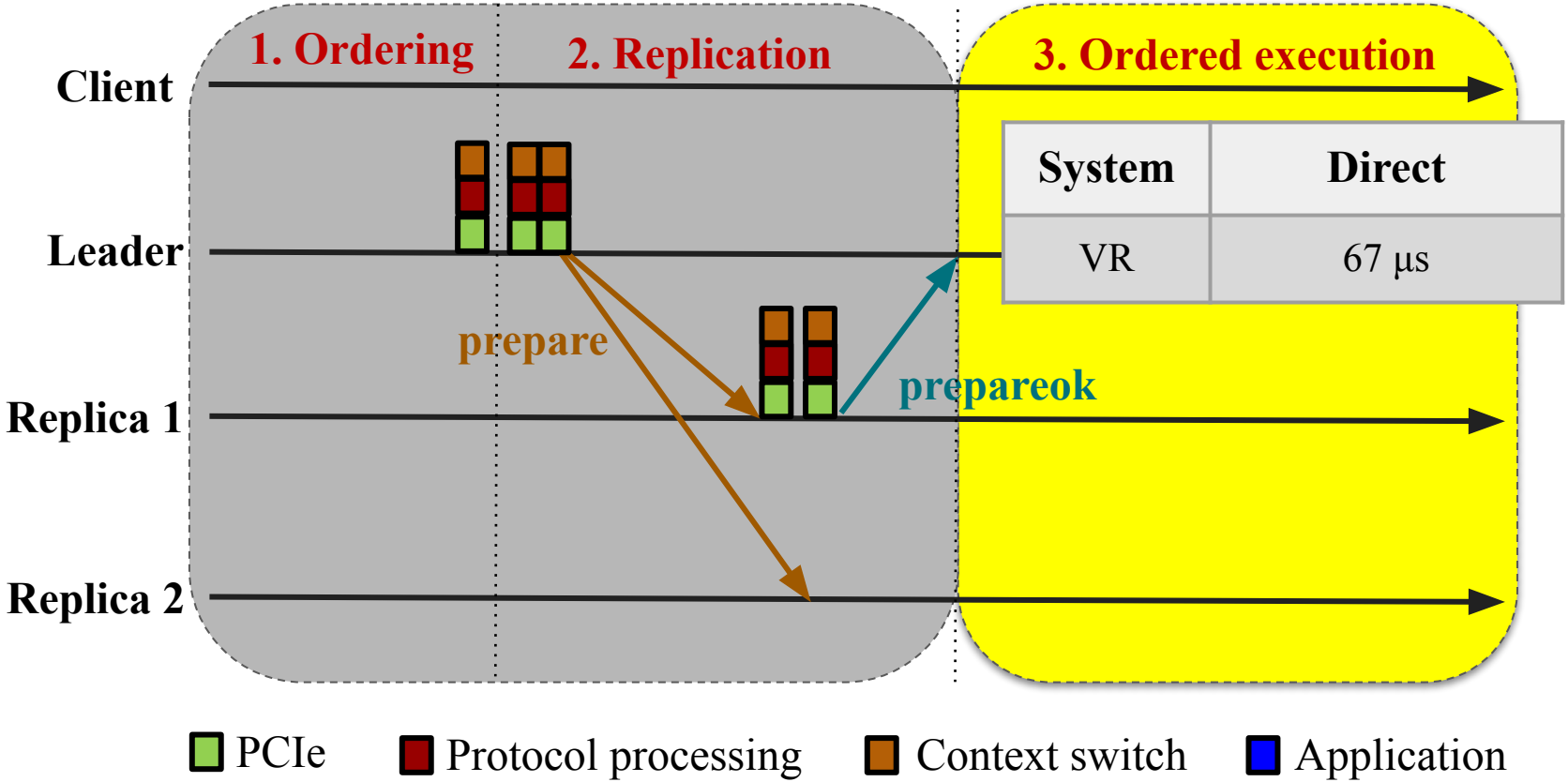


[1] Understanding PCIe performance for end host, SIGCOMM'18

Consensus – VR Data Operation

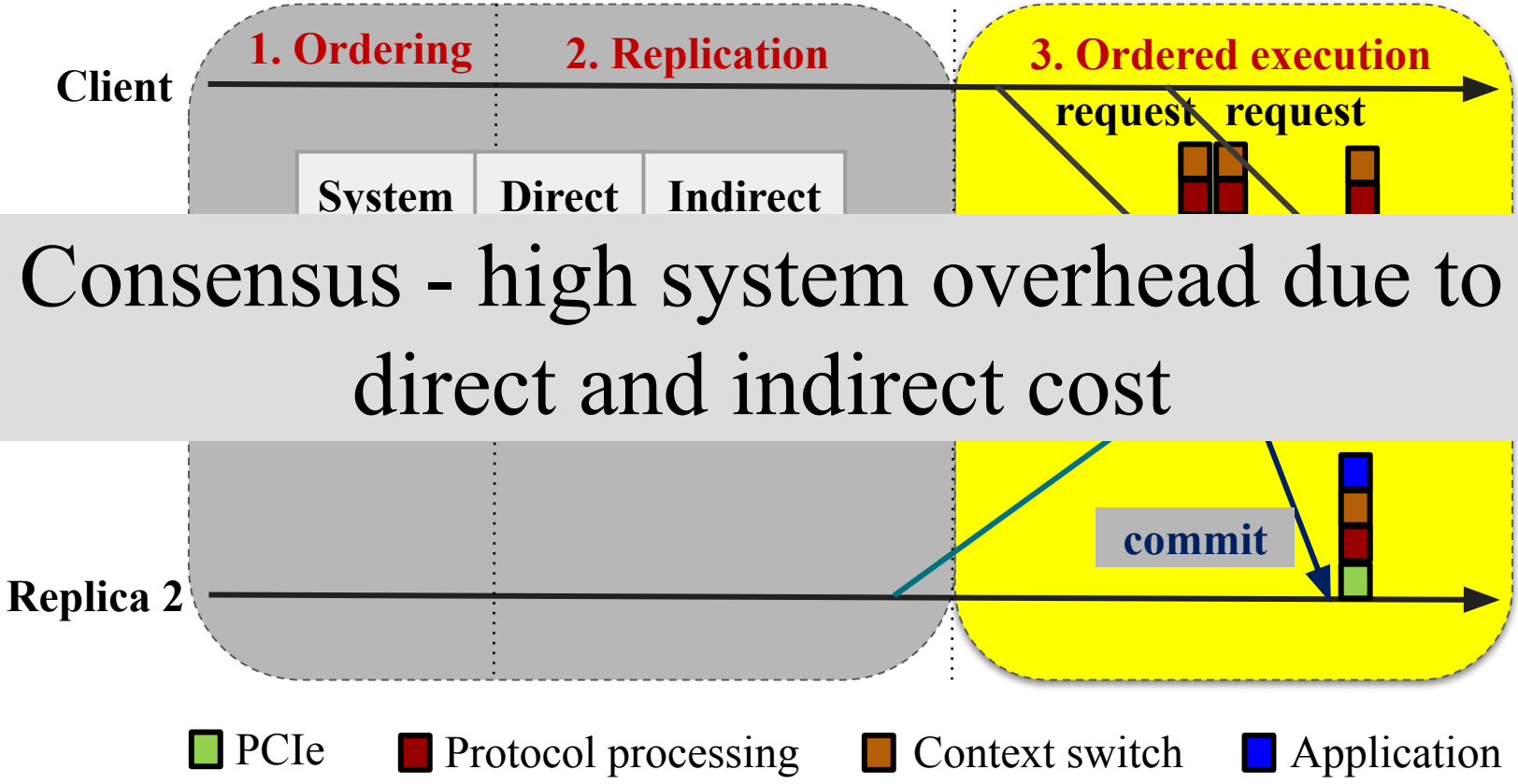


Consensus – Direct Cost of Latency

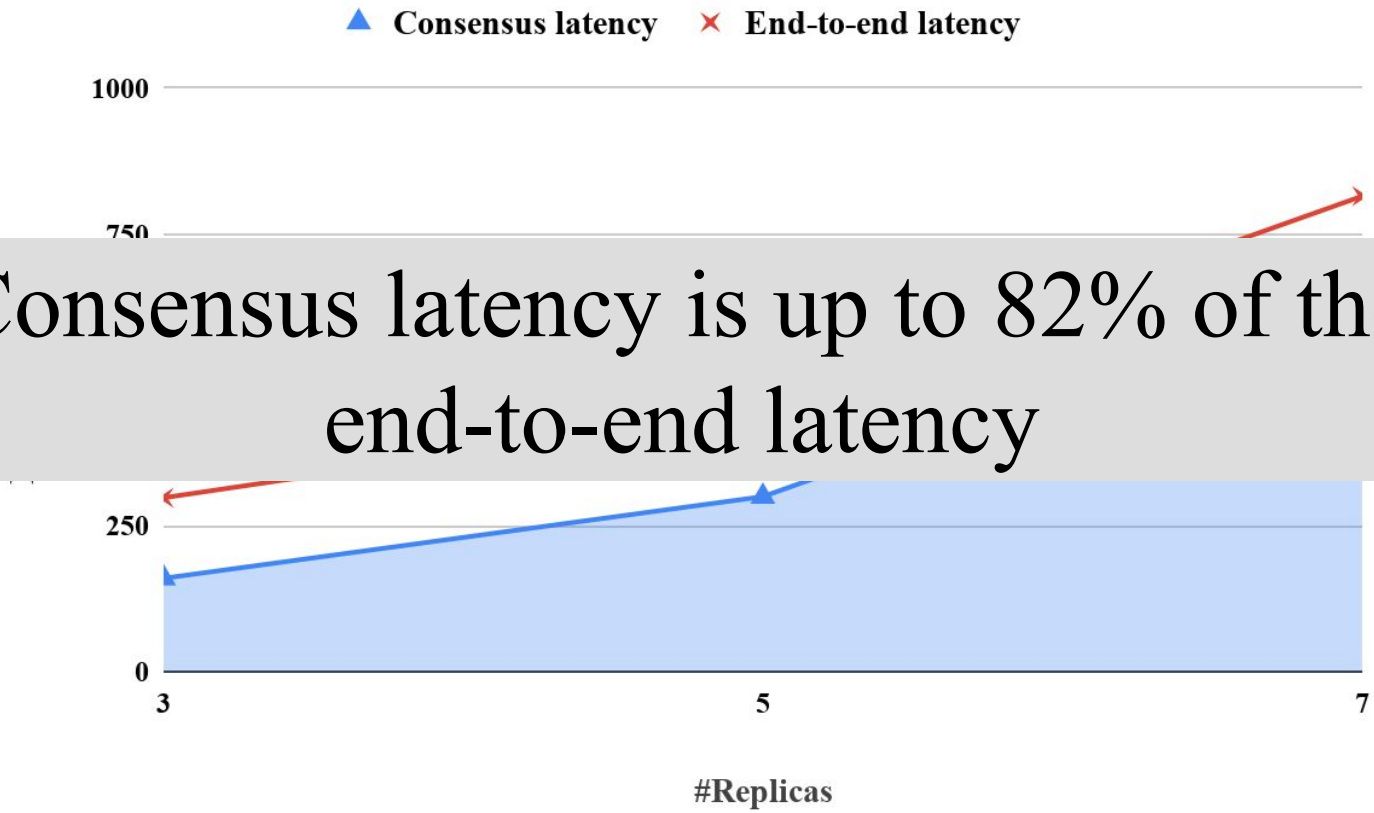


System	Direct
VR	67 µs

Consensus – Indirect Cost of Latency



Consensus Latency - Increasing Replicas



Consensus latency is up to 82% of the end-to-end latency

Consensus Operations

- Data Operation:
  - ➢ Critical path for handling a client request
- Control Operations:
  - ➢ Recovery – application recovery after failure
  - ➢ View Change – new replicas joining/leaving the group, new leader
  - ➢ Heartbeats – health status messages exchanged across replicas

Cost of Consensus - Summary

- Every client request has high consensus overhead
- Consensus algorithms share resources with application
- Consensus overhead increases with increasing replicas

Consensus - Existing Research

- Network approaches:
  - ...sts
  - ...es
- Hardware approach:
  -
  -

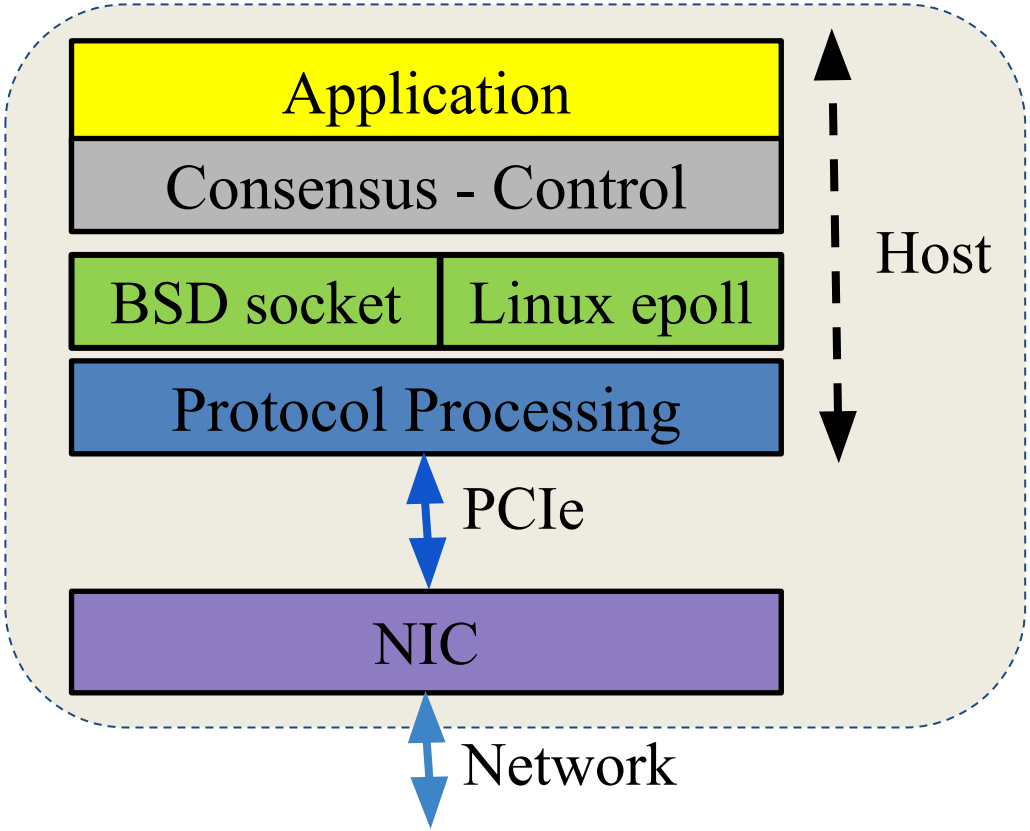
Rely on Network Guarantees

Logically Coupled Consensus

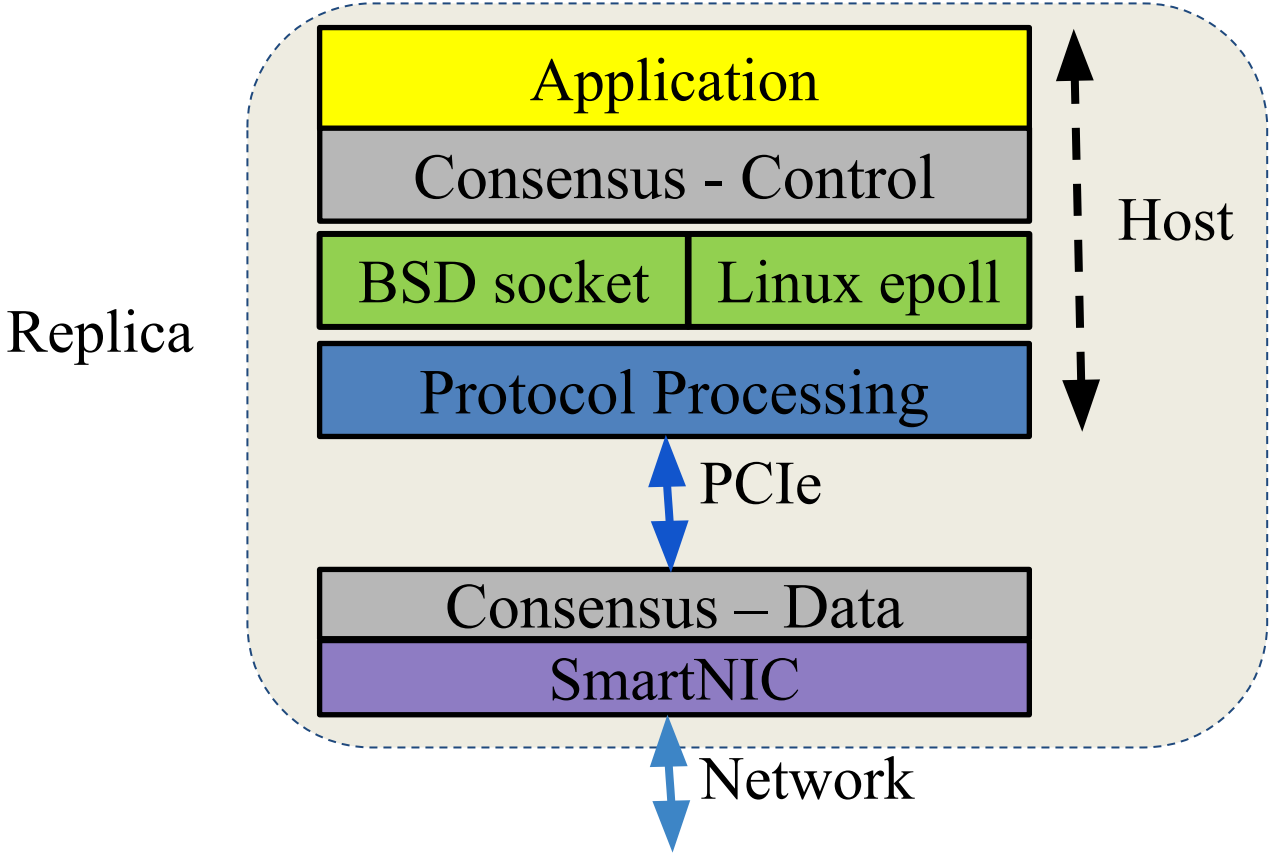
Dyad: Untangling Logically-Coupled Consensus

- Background
- **Overview**
- Design and Evaluation
- Conclusion

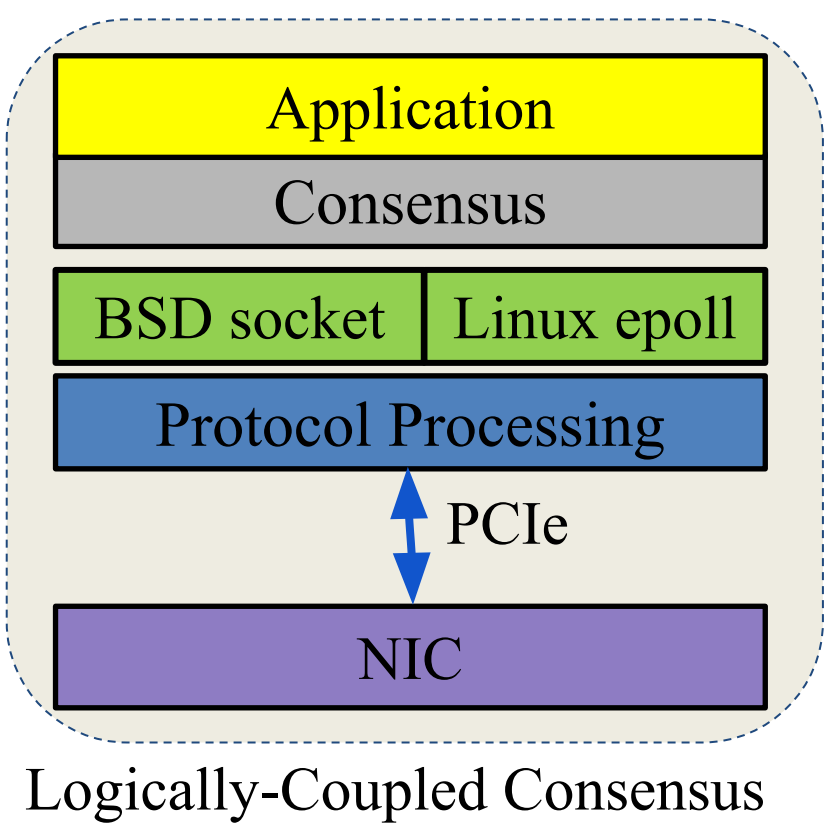
Logically-Coupled Consensus



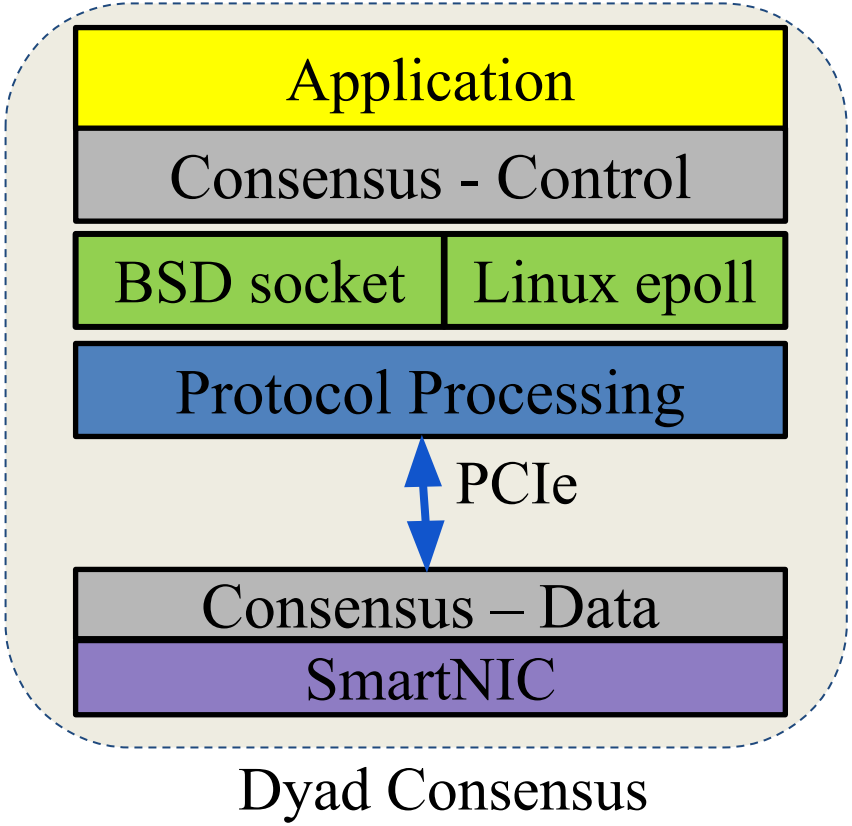
Dyad: Untangling Logically-Coupled Consensus



Dyad: Untangling Logically-Coupled Consensus



Logically-Coupled Consensus



Dyad Consensus

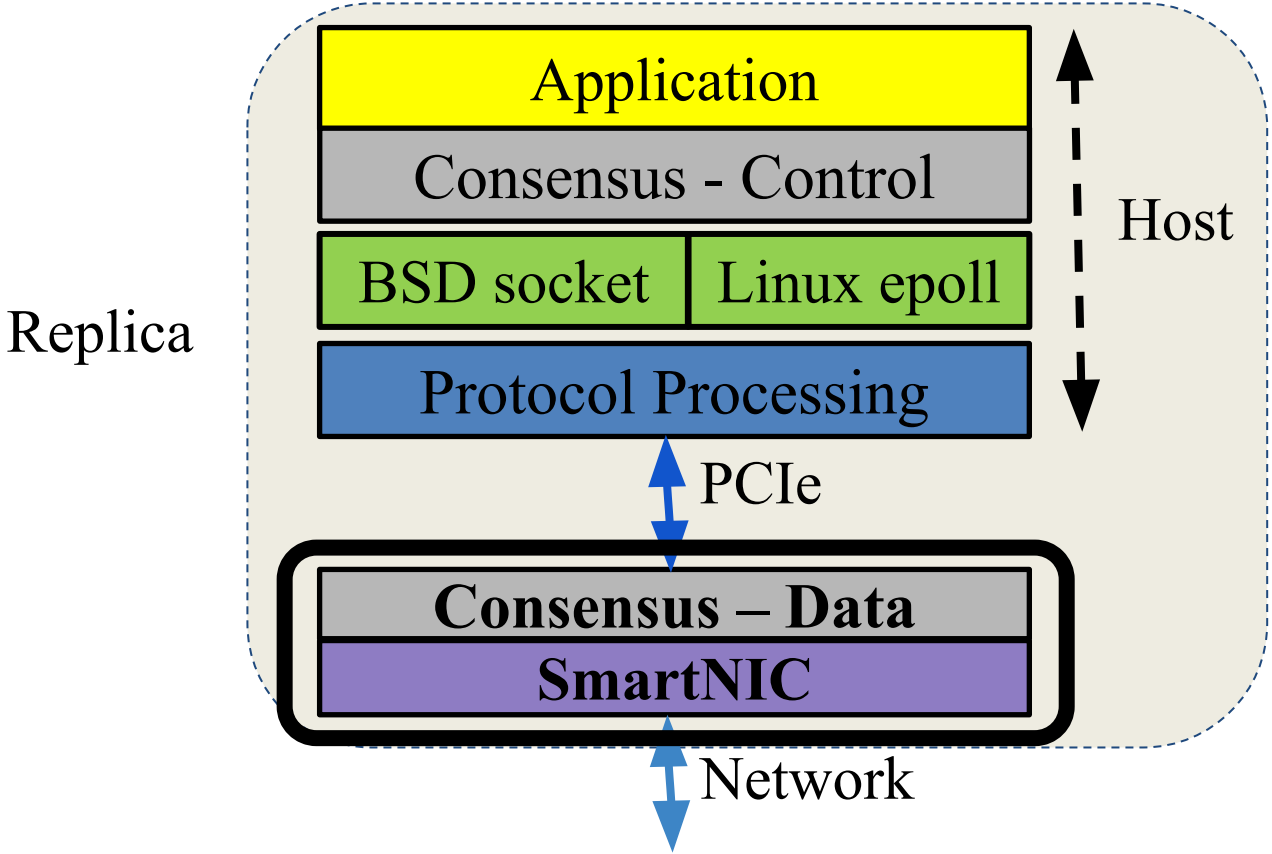
Dyad: Classifying Consensus Operations

- Data Operation - SmartNIC:
  - ➢ Critical path for handling a client request
- Control Operations - Host:
  - ➢ Recovery – application recovery after failure
  - ➢ View Change – new replicas joining/leaving the group, new leader
  - ➢ Heartbeats – health status messages exchanged across replicas

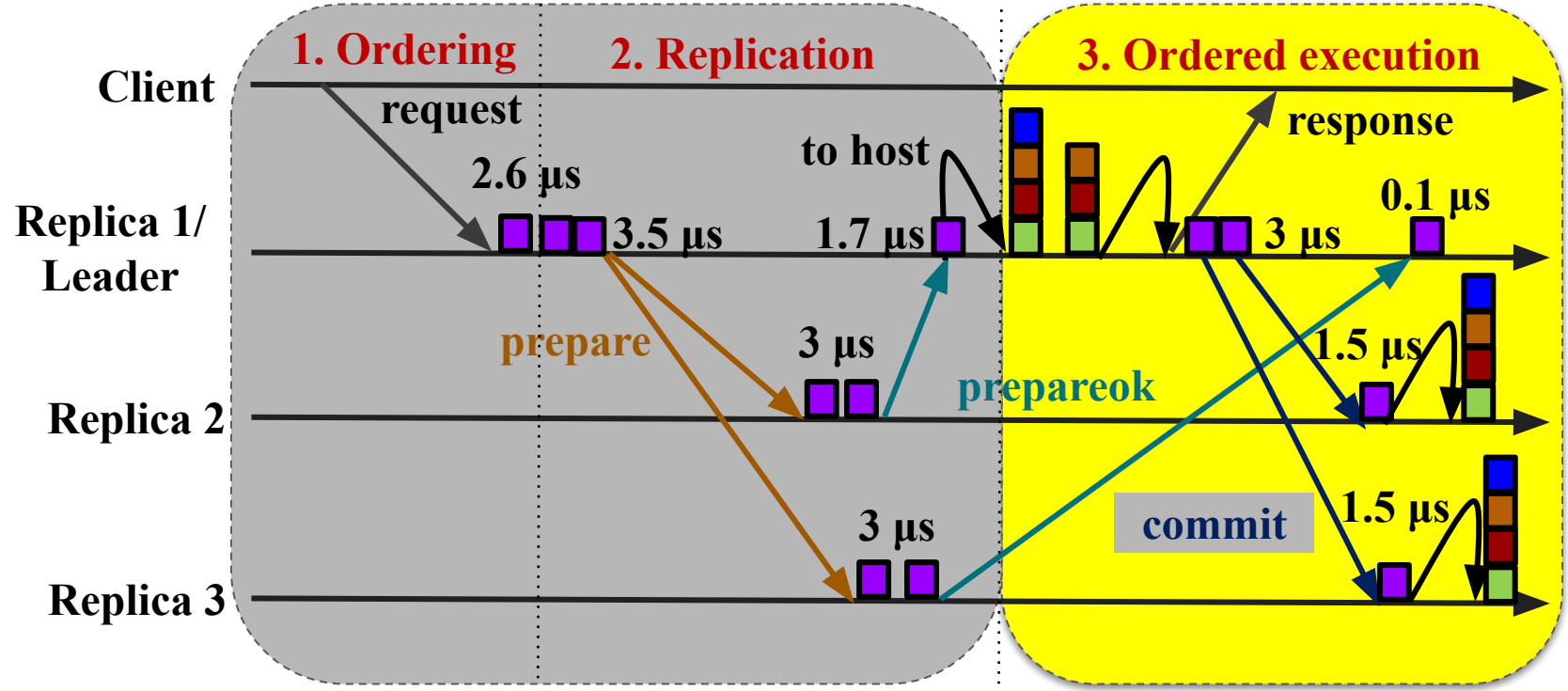
Dyad: Untangling Logically-Coupled Consensus

- Background
- Overview
- **Design and Evaluation**
- Conclusion

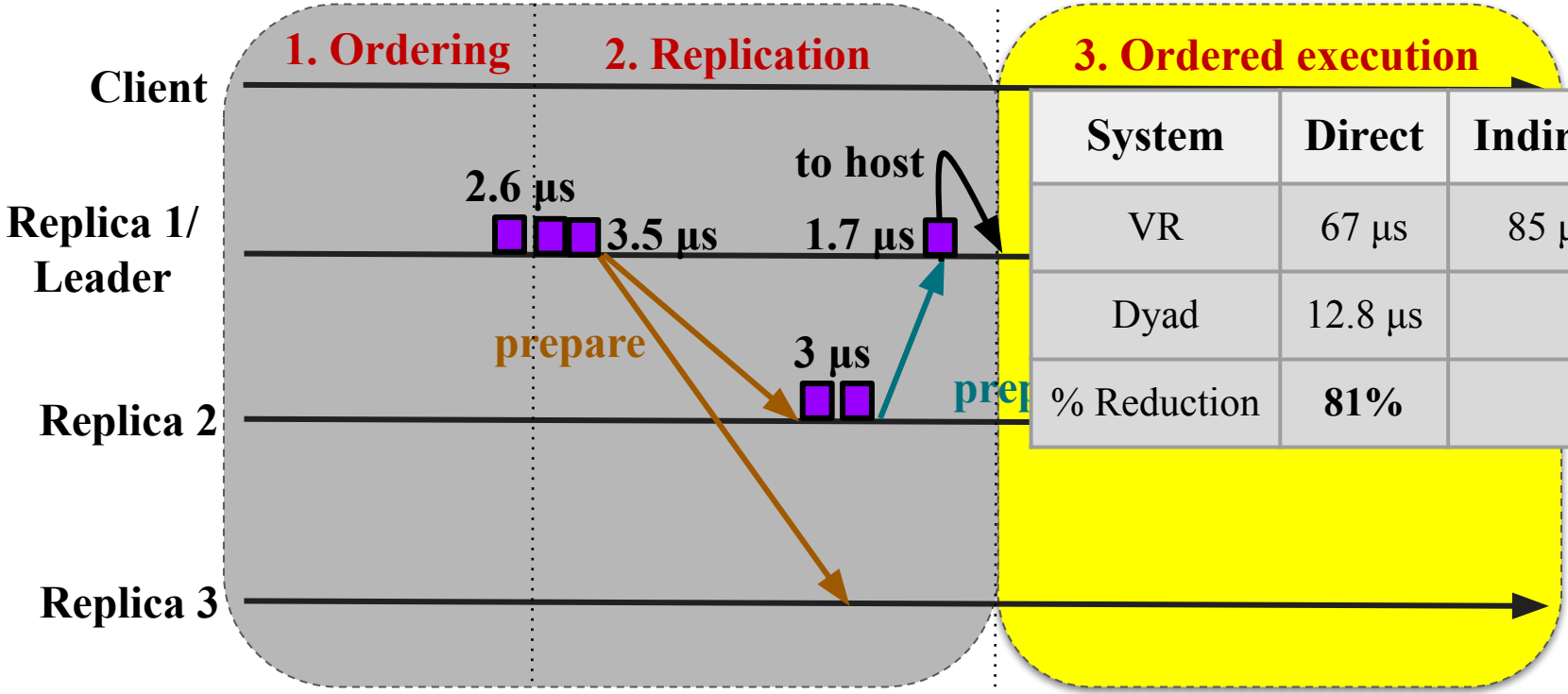
Dyad: Data Operations



Dyad – Viewstamp Replication (VR)



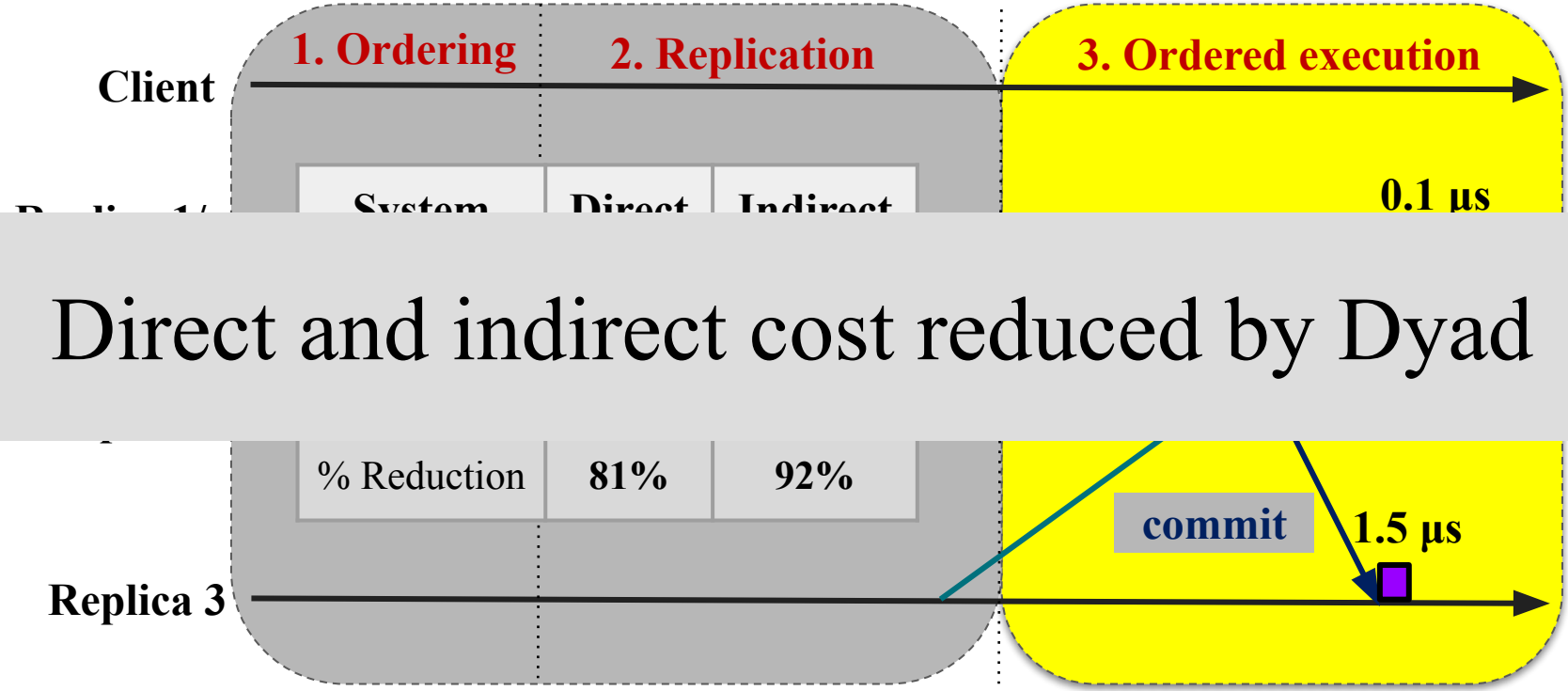
Dyad – Direct Cost



System	Direct	Indirect
VR	67 µs	85 µs
Dyad	12.8 µs	
% Reduction	**81%**	

SmartNIC

Dyad – Indirect Cost



Direct and indirect cost reduced by Dyad

PCIe | Protocol processing | Context switch | Application | SmartNIC

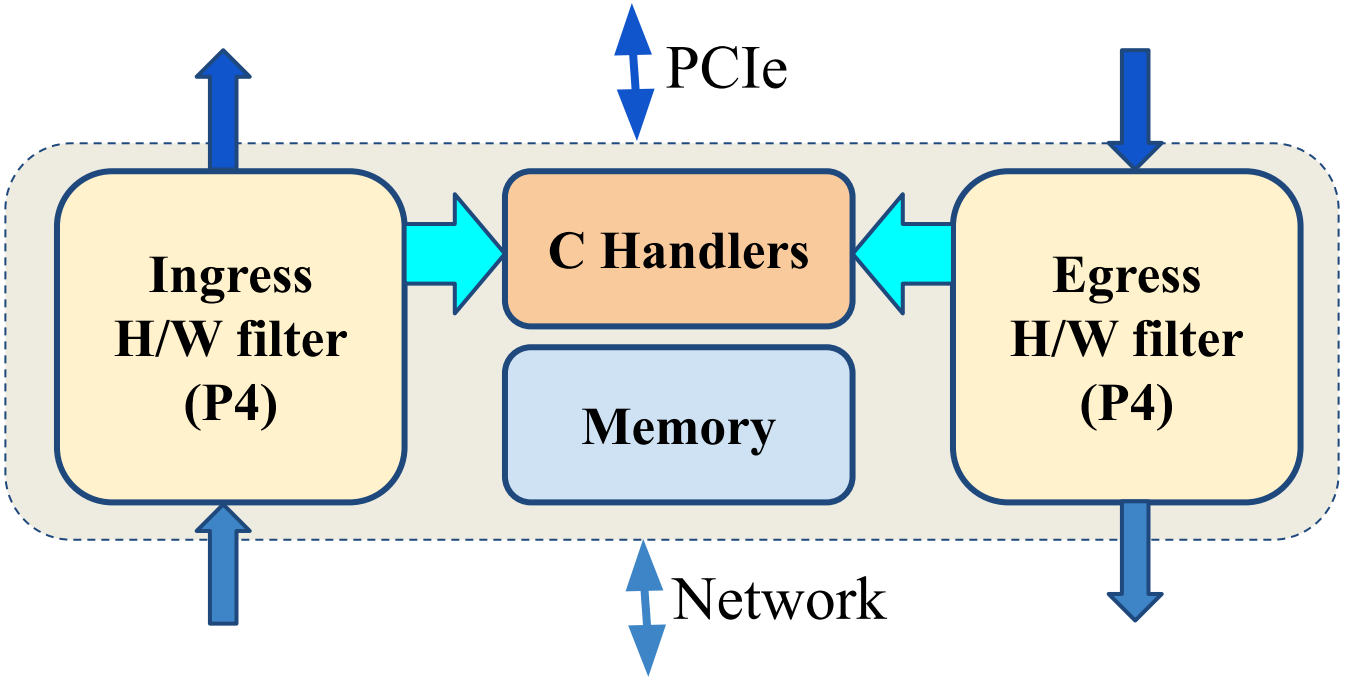
Dyad: SmartNIC Primitives

- Hardware Filtering:
  - ➢ Specify packet format in domain-specific language (P4)
  - ➢ Filter messages based on the header and payload
  - ➢ Filters are applied to messages coming from the network and the host

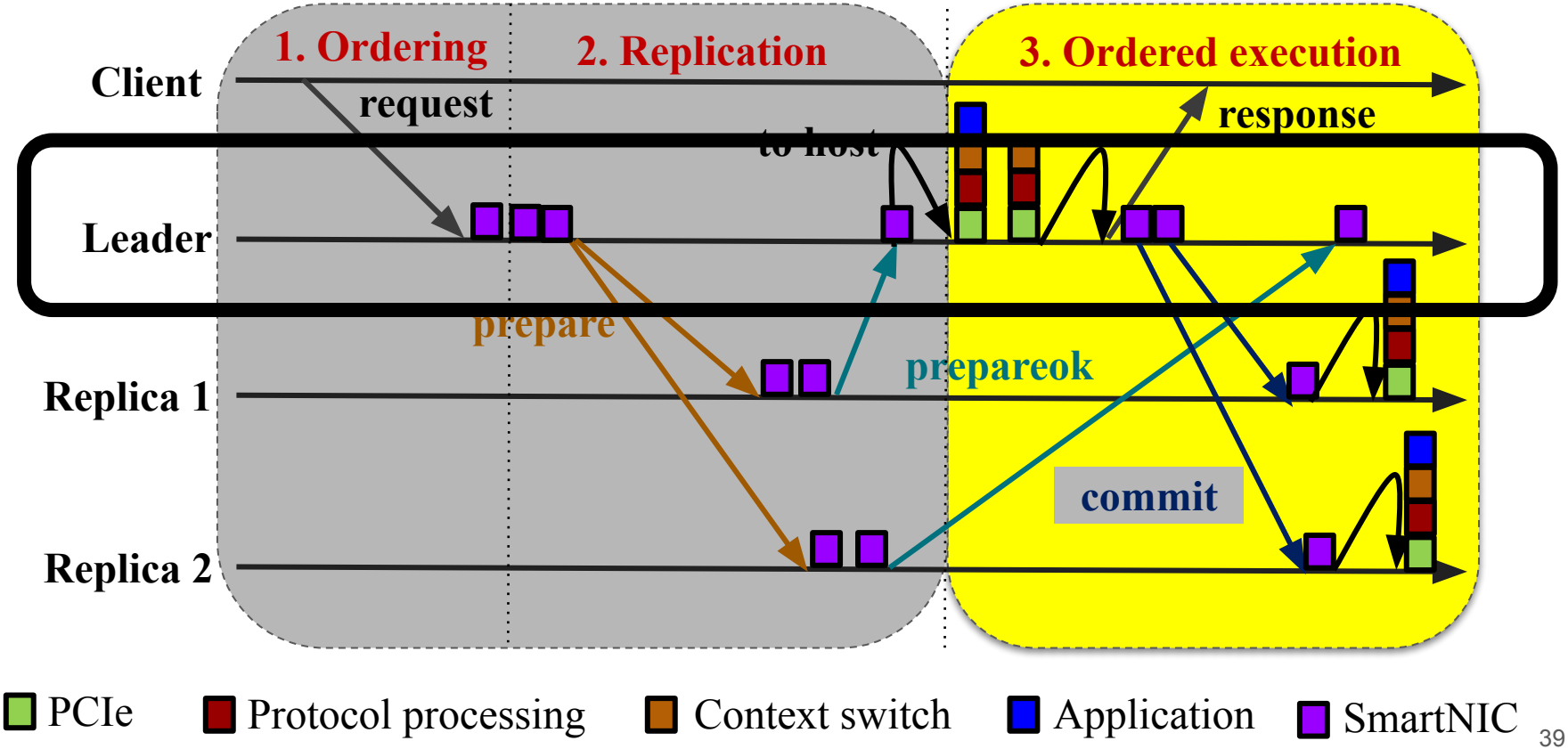
Dyad: SmartNIC Primitives

- Packet Processing:
  - ➢ Filtered messages invoke request/consensus/response handlers
  - ➢ Handlers drop/forward/modify a packet
  - ➢ Generate new packets

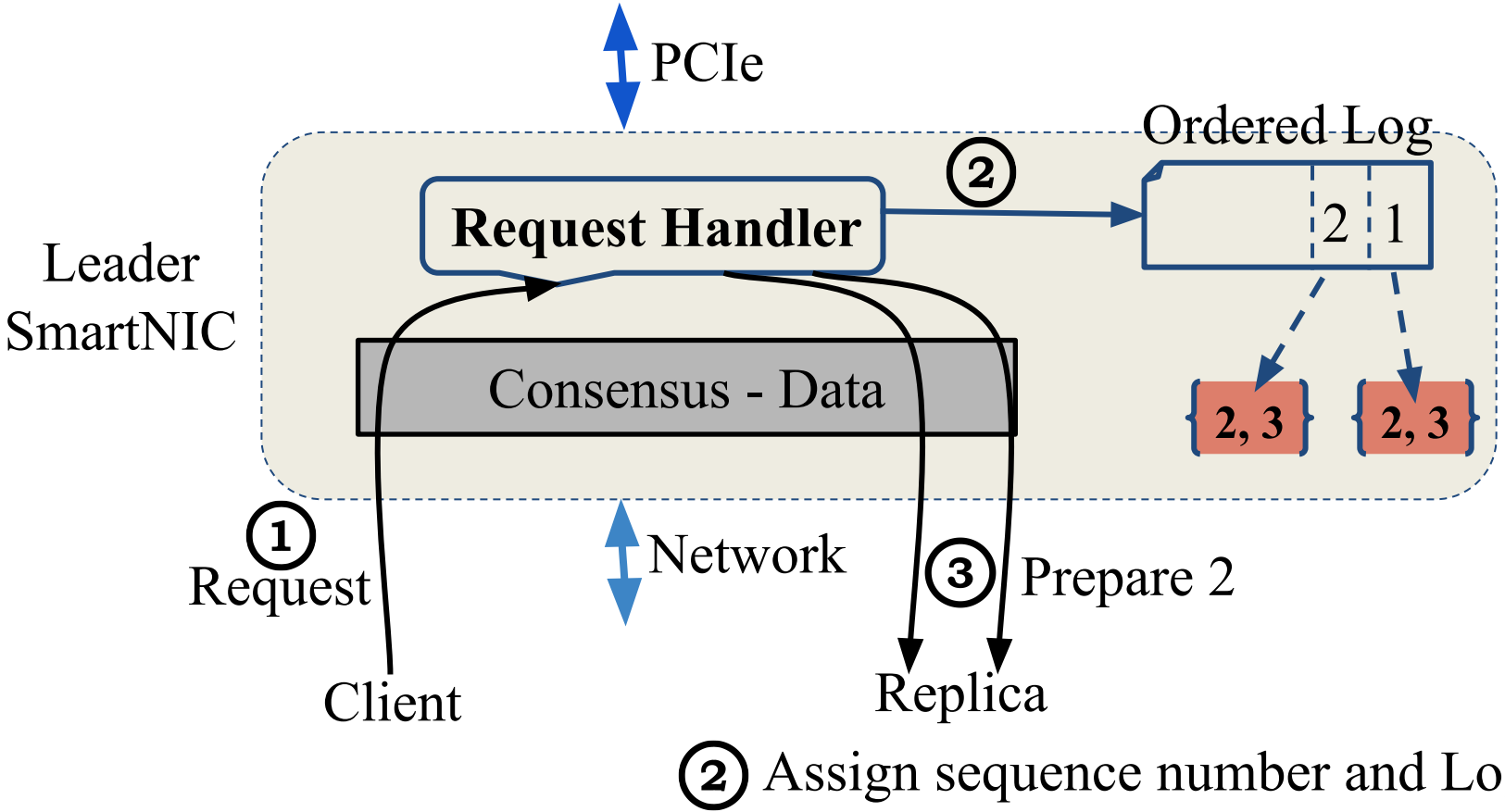
Dyad: SmartNIC Primitives



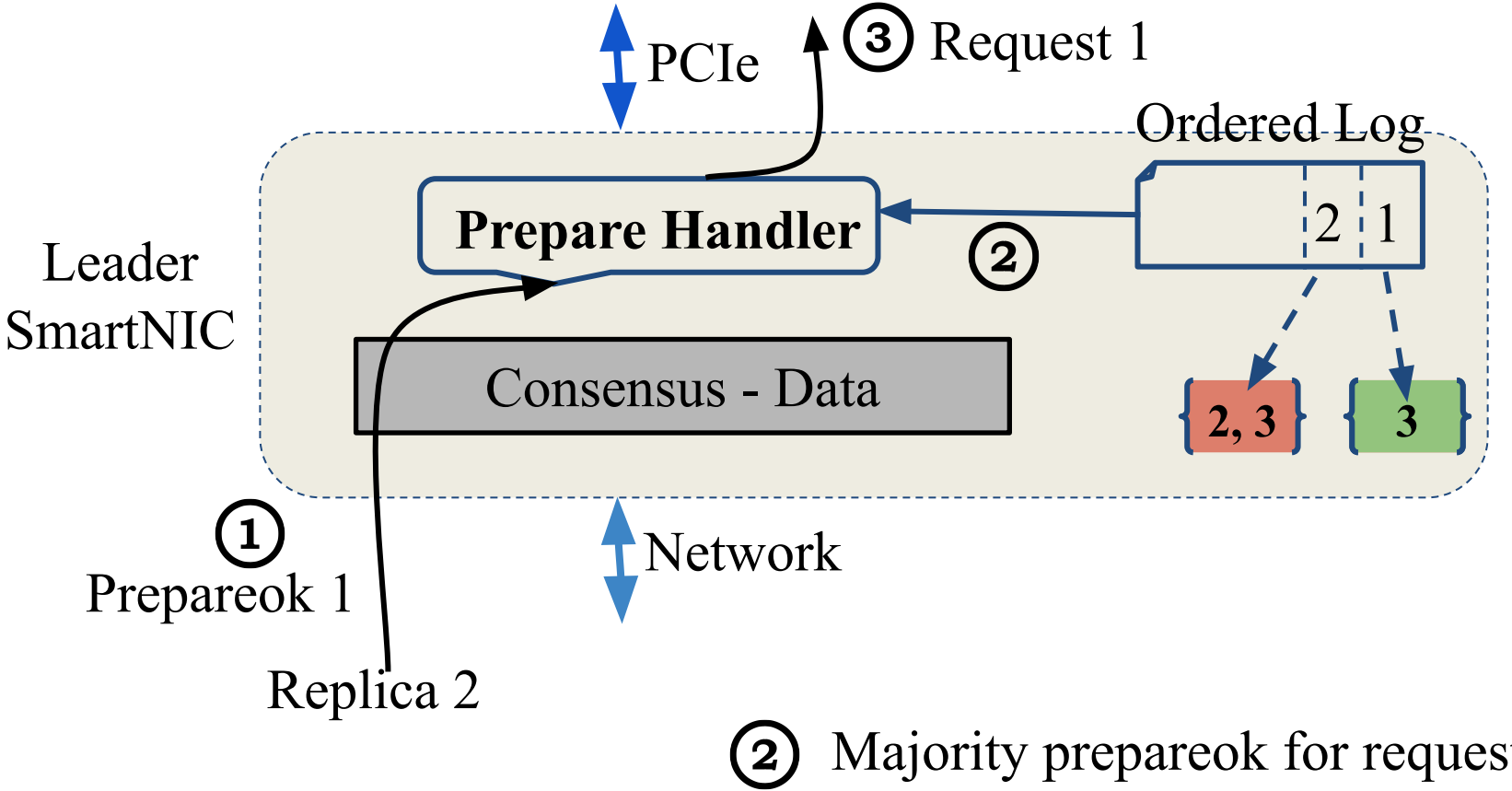
Dyad - Leader Data Operations



Dyad: Ordering on Leader SmartNIC

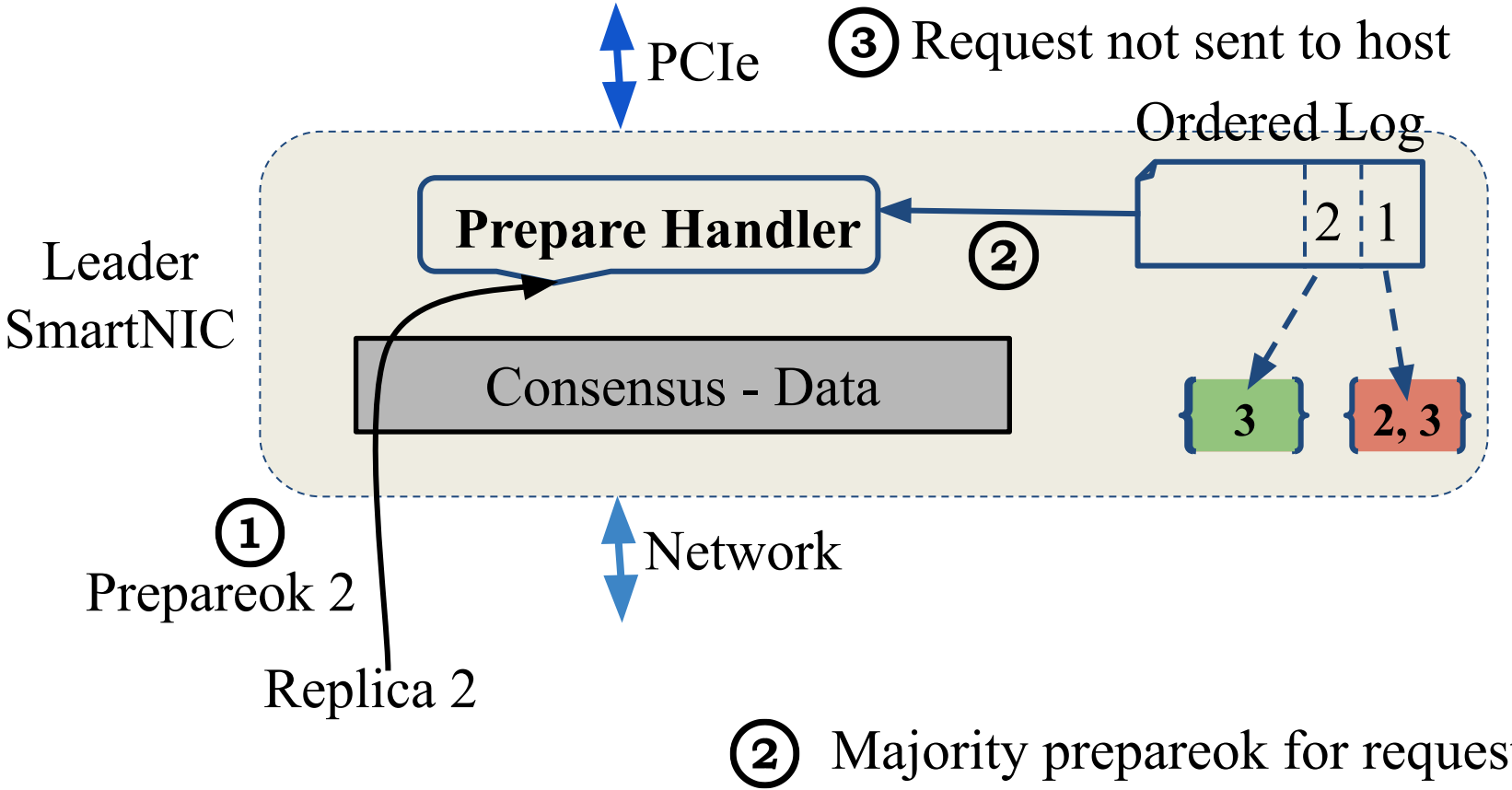


Dyad: Replication on Leader SmartNIC

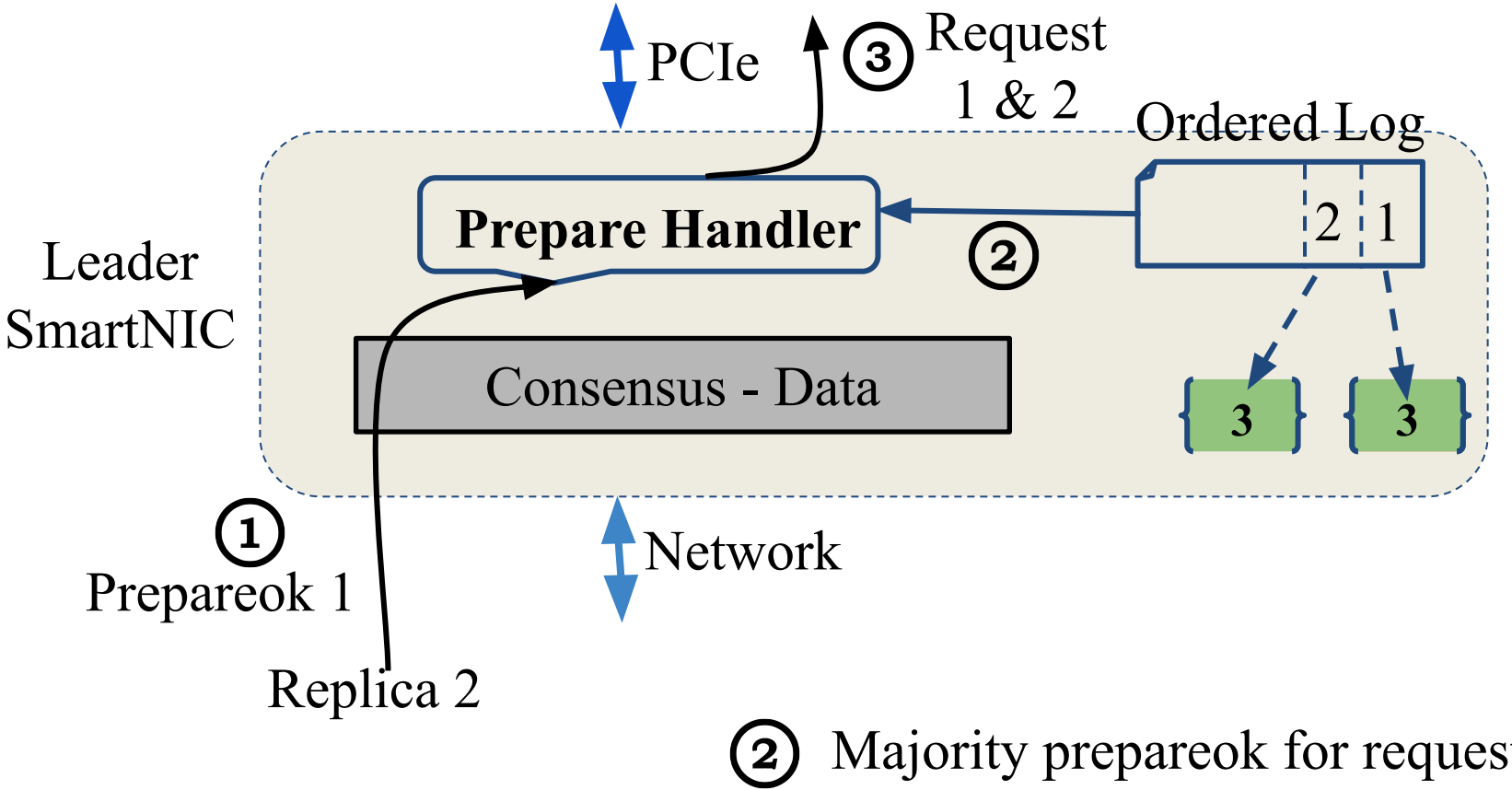


2 Majority prepareok for request 1

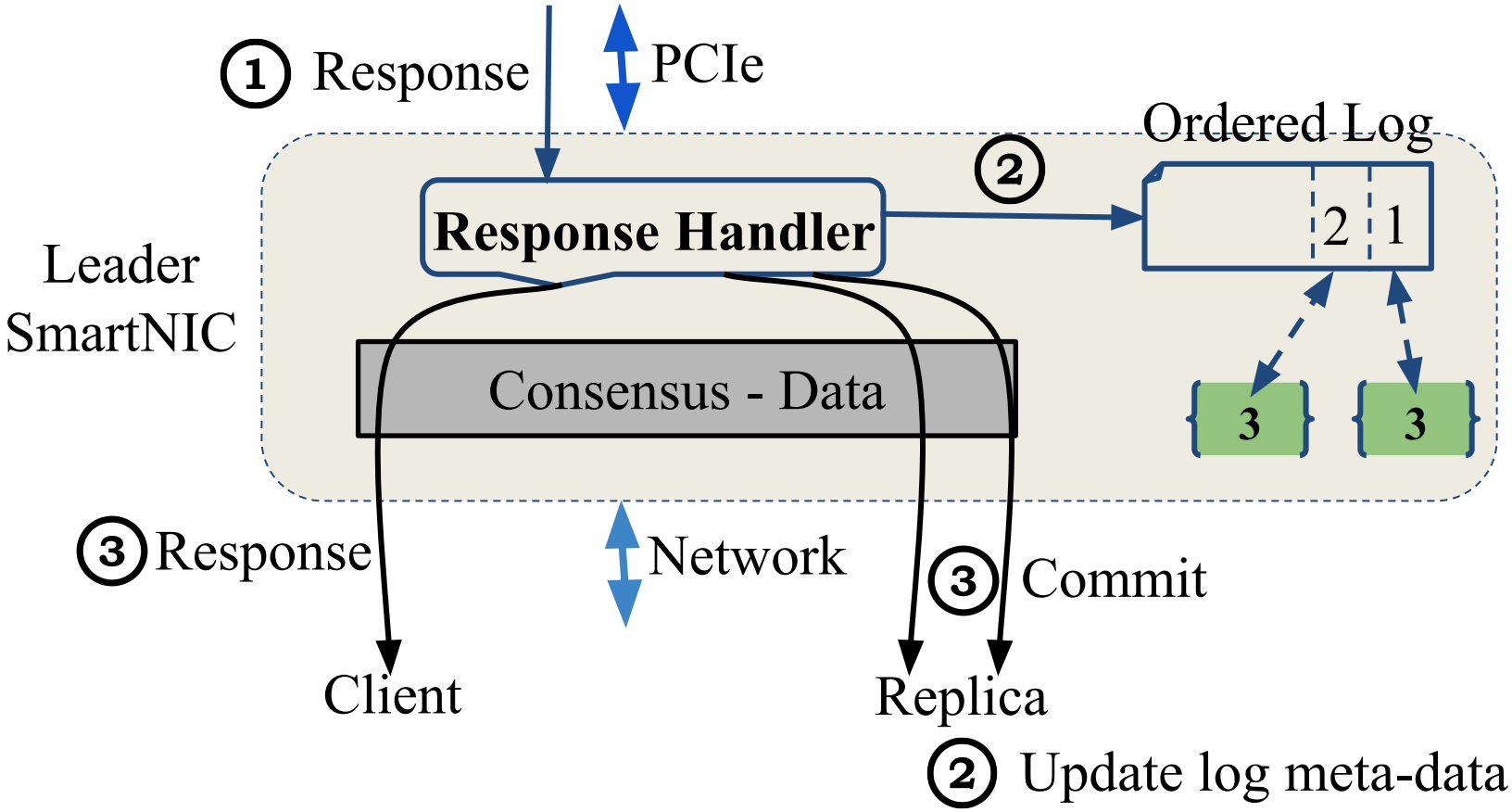
Dyad: Reordered Consensus Message



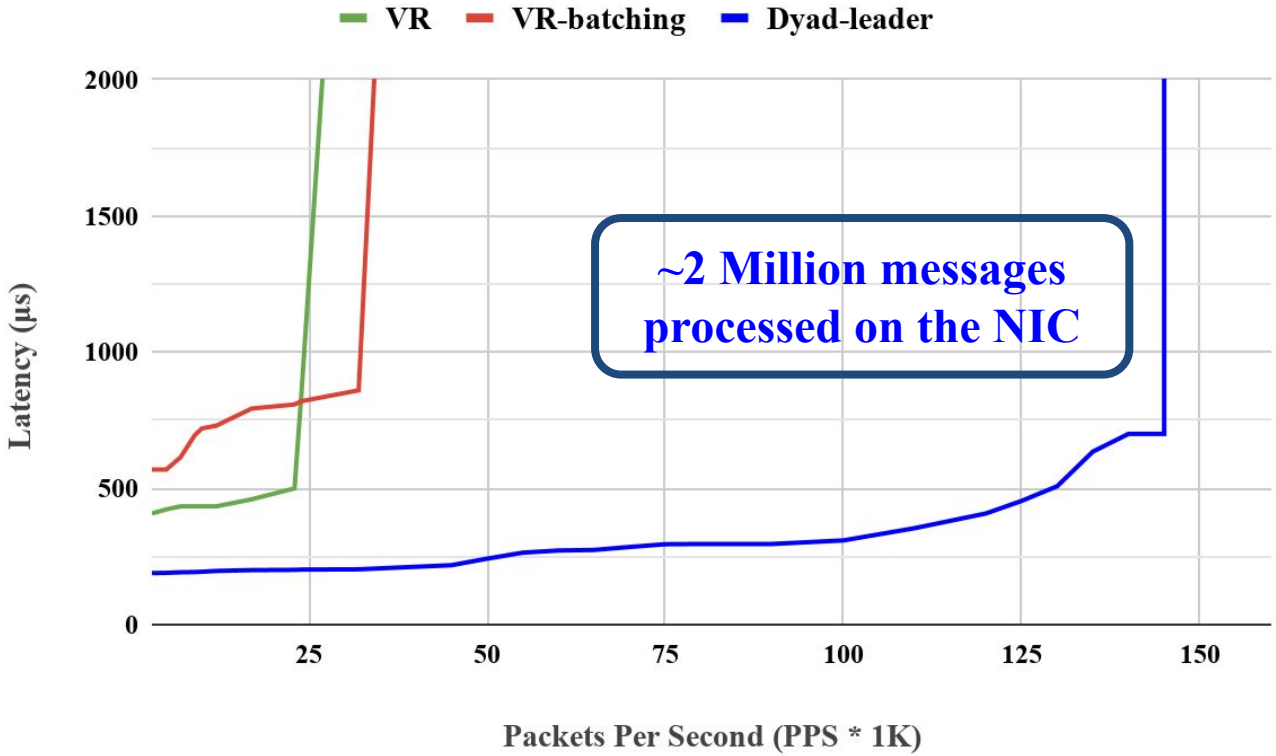
Dyad: Reordered Consensus Message



Dyad: Response and Commit

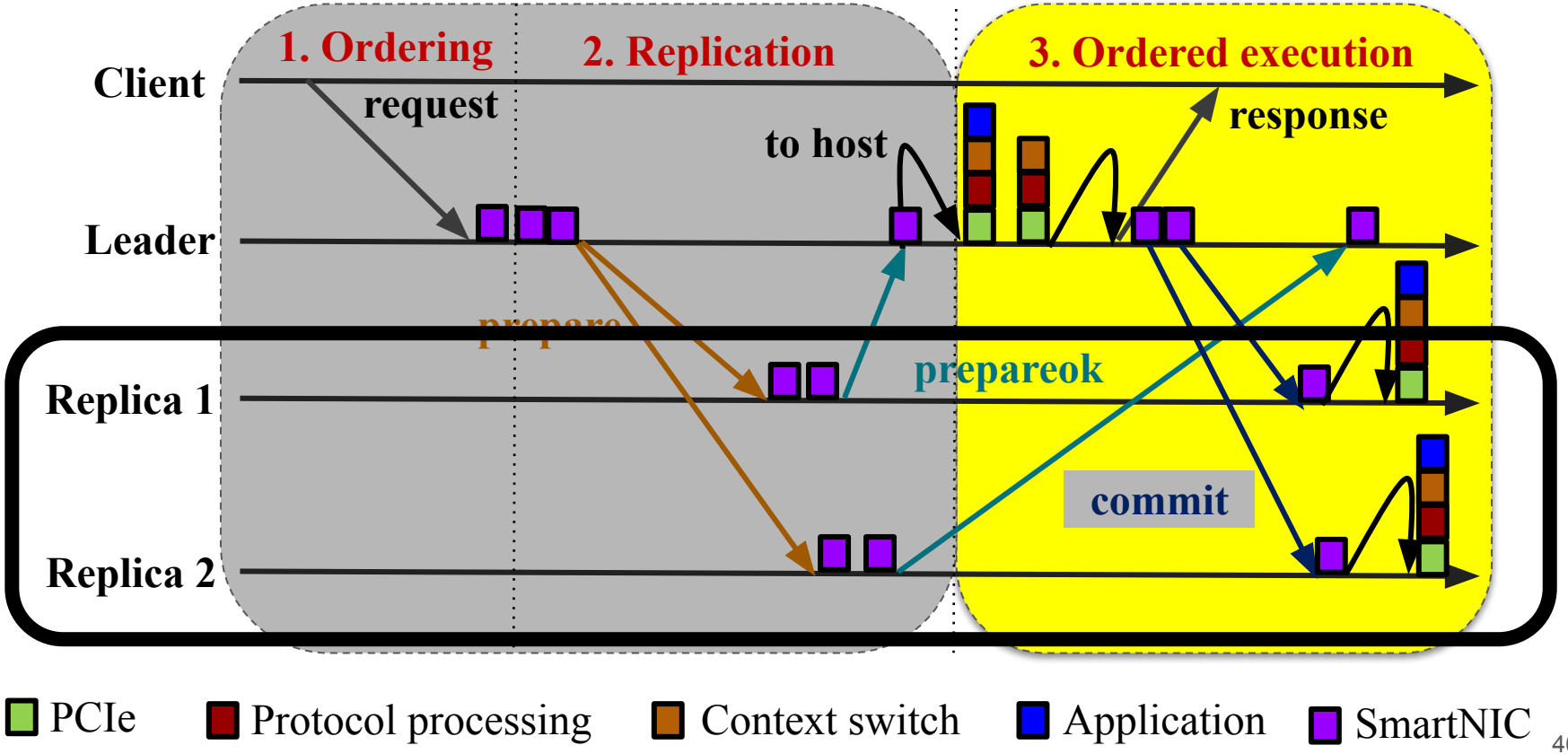


Dyad: Timestamp Server with 5 replicas



➢ Reduce latency by up to **76%**, Improves throughput by **5.8x**

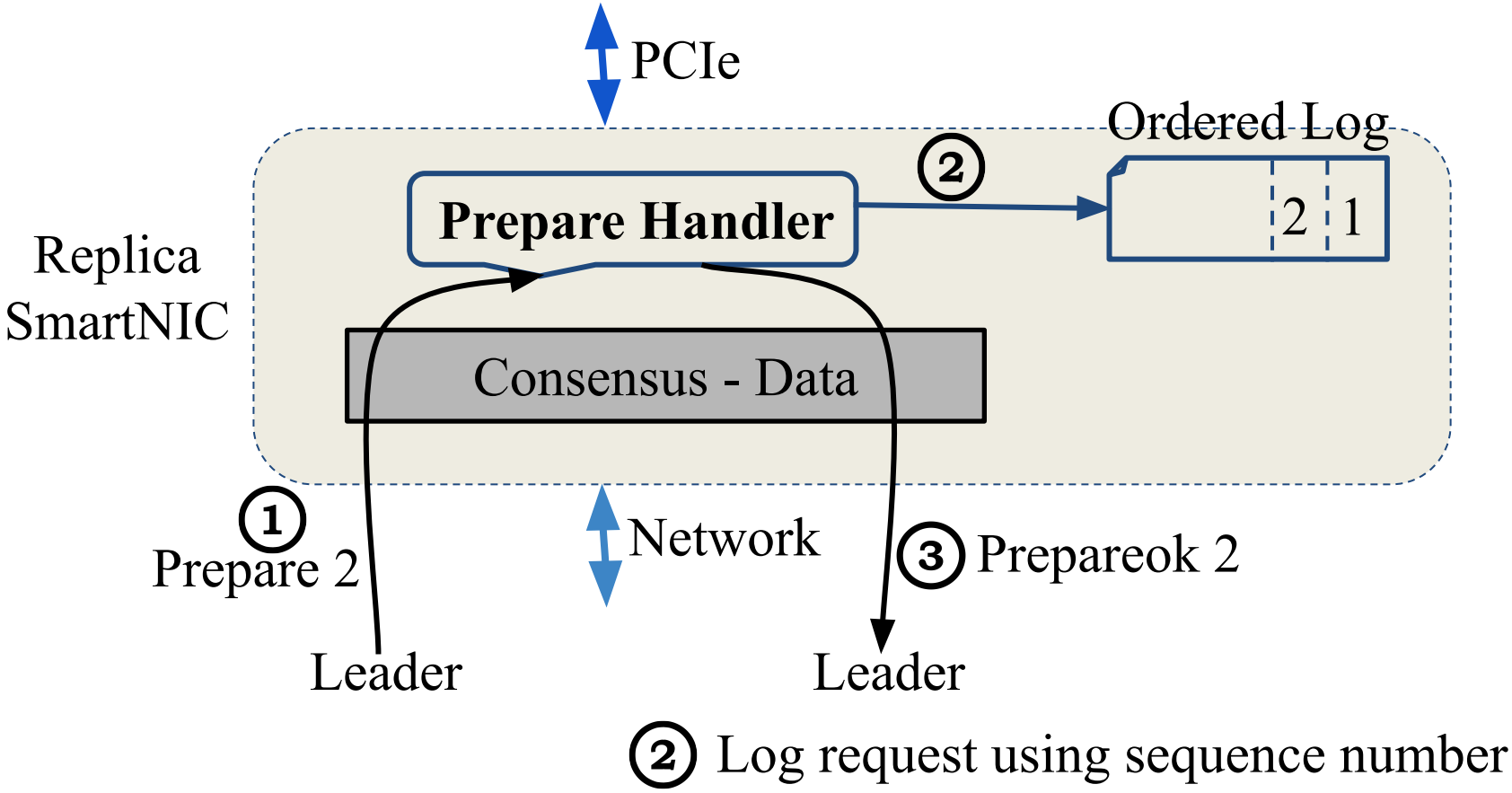
Dyad – Replica Data Operations



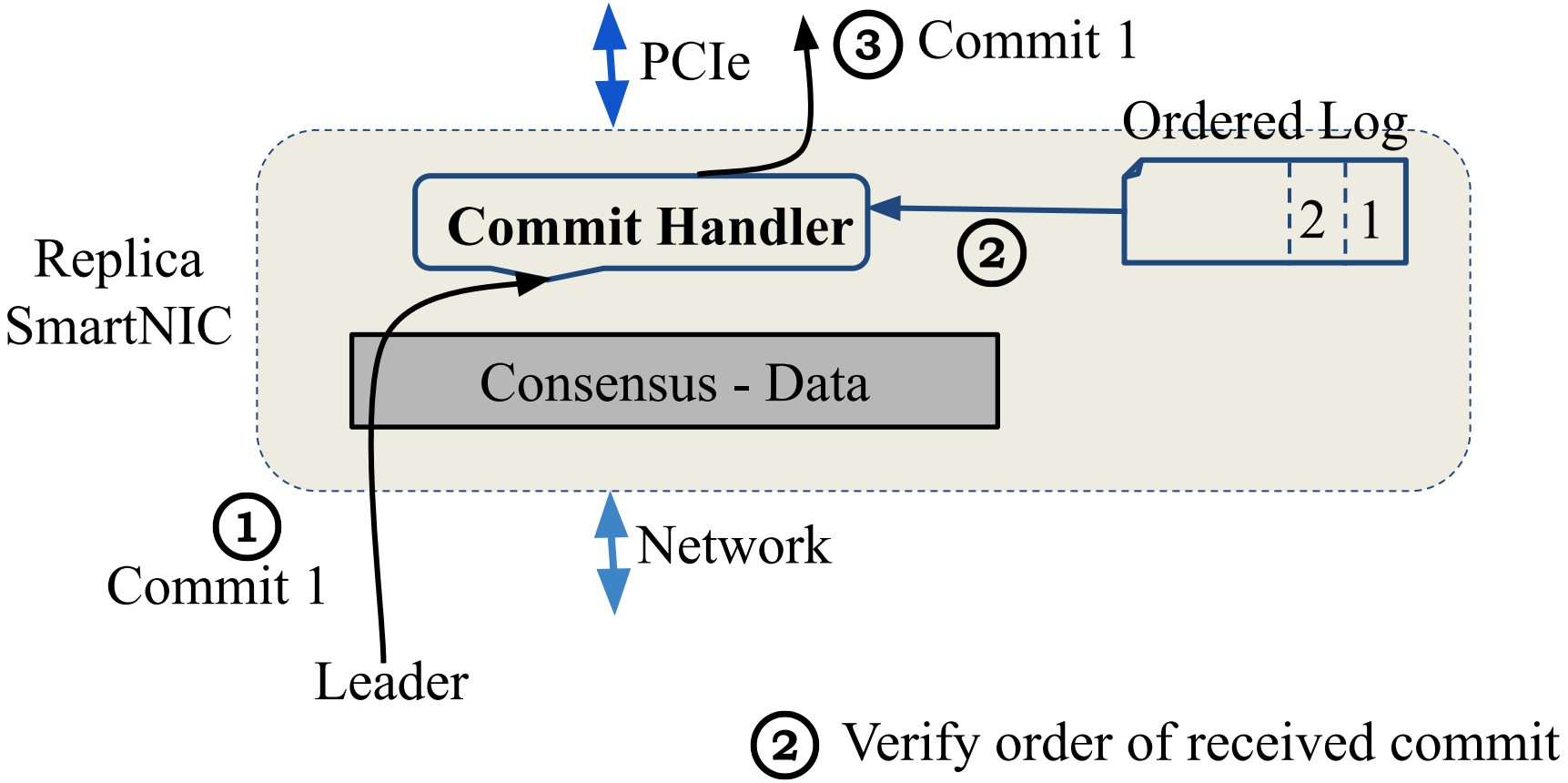
Dyad: Ordering on Replica SmartNIC

- Ordering and Logging:
  - ➢ Logs ordered by the sequence number in prepare message
  - ➢ Prepare message are processed and dropped on the SmartNIC
- Ordered Execution:
  - ➢ Commit messages forwarded to the host processor
  - ➢ The request is appended to the commit message by SmartNIC

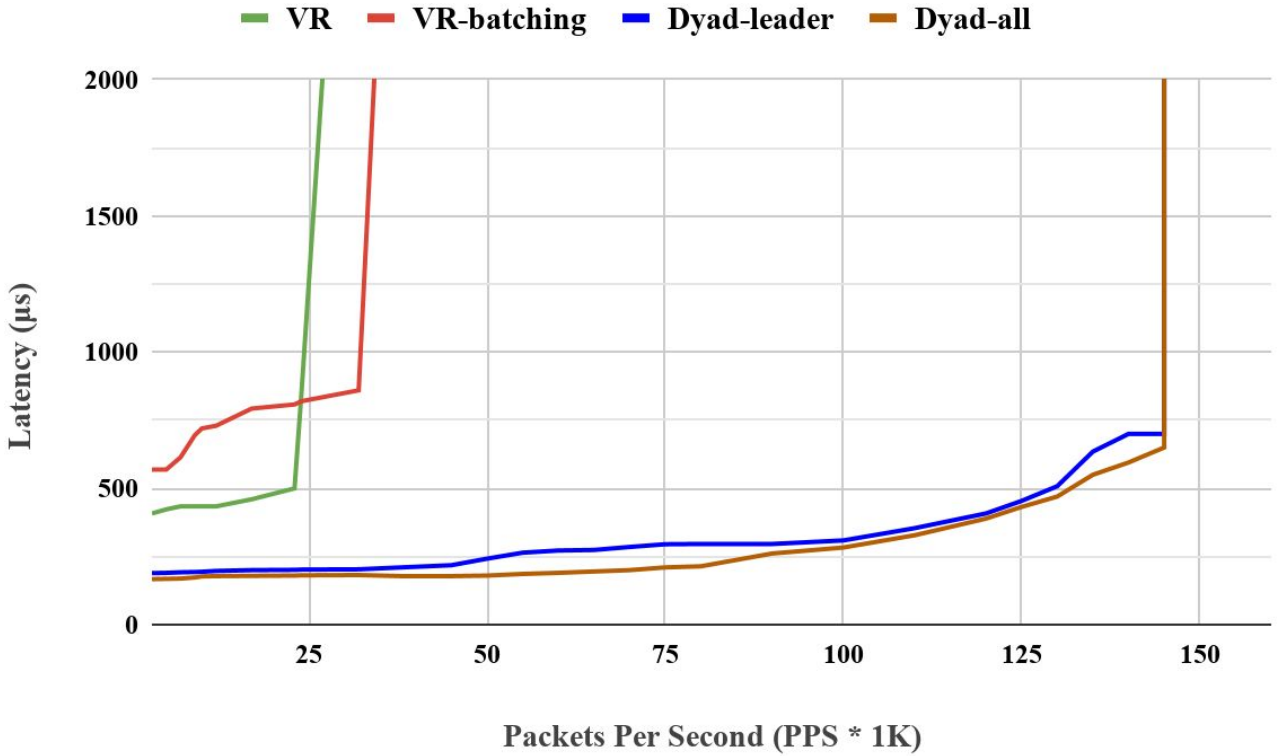
Dyad: Logging on Replica SmartNIC



Dyad: Ordered Execution on the Replica



Dyad: Timestamp Server with 5 replicas



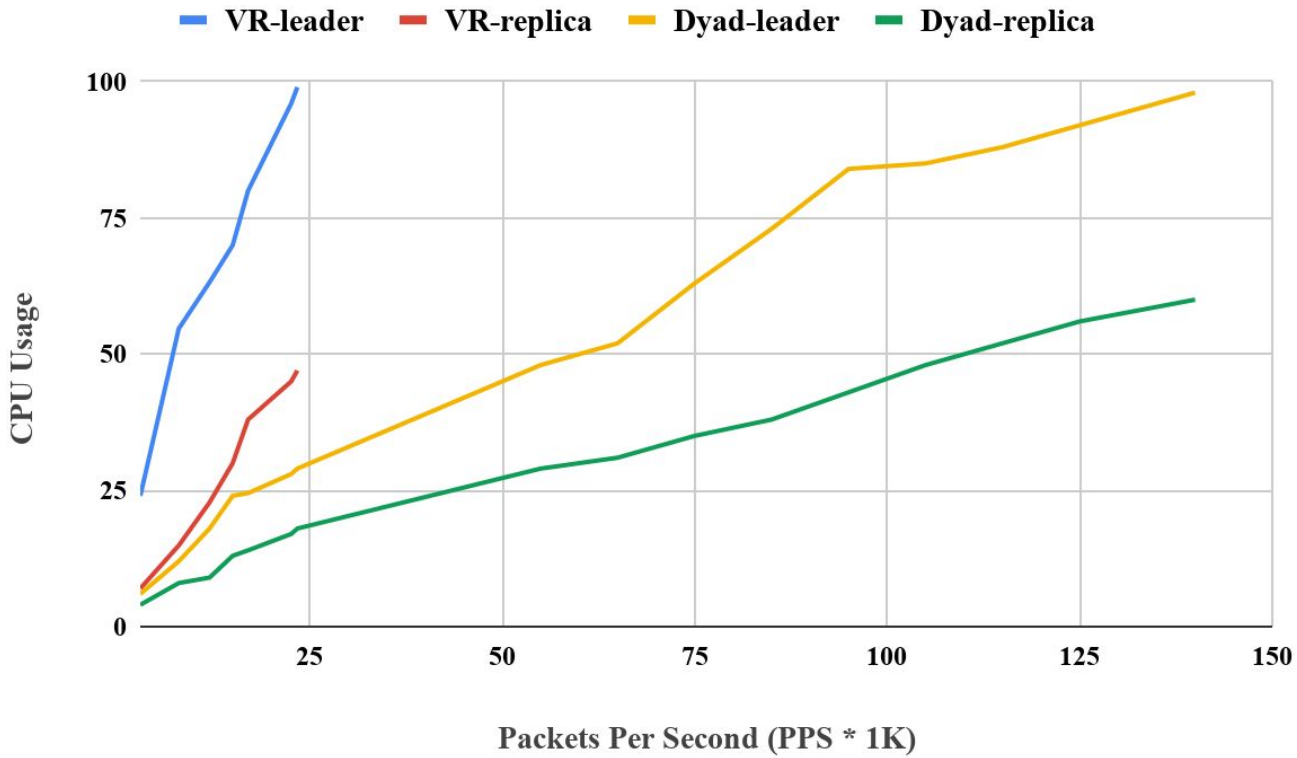
➢ Reduce latency by **30 µs**

Dyad: Consensus Latency

System	Consensus latency (µs)	% reduction
VR	350	N/A
VR-batching	409	N/A
Dyad-Leader	48	**86%**
Dyad-All	17	**95%**

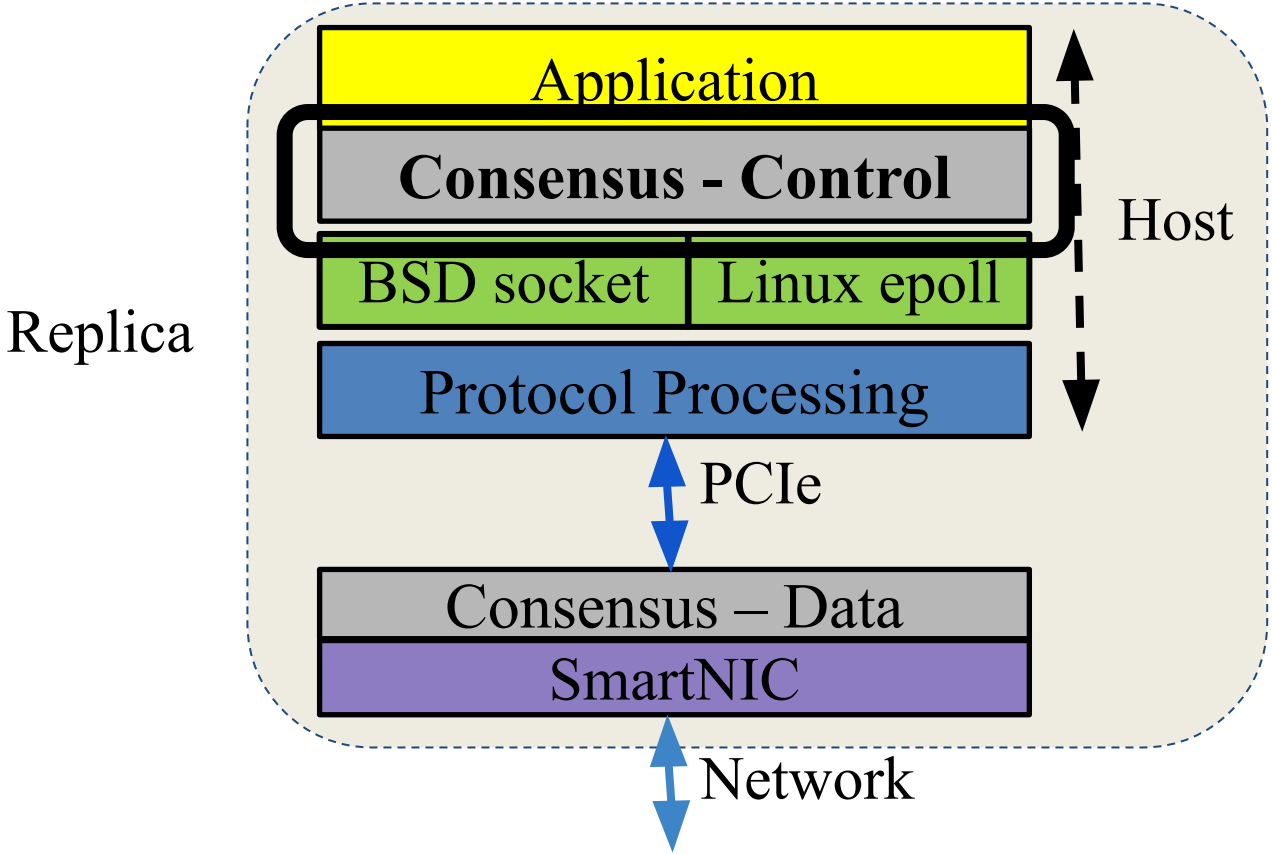
Timestamp server - 5 replicas

Dyad: CPU Usage Timestamp Server



➢ Reduce CPU usage by up to **70%** on the leader

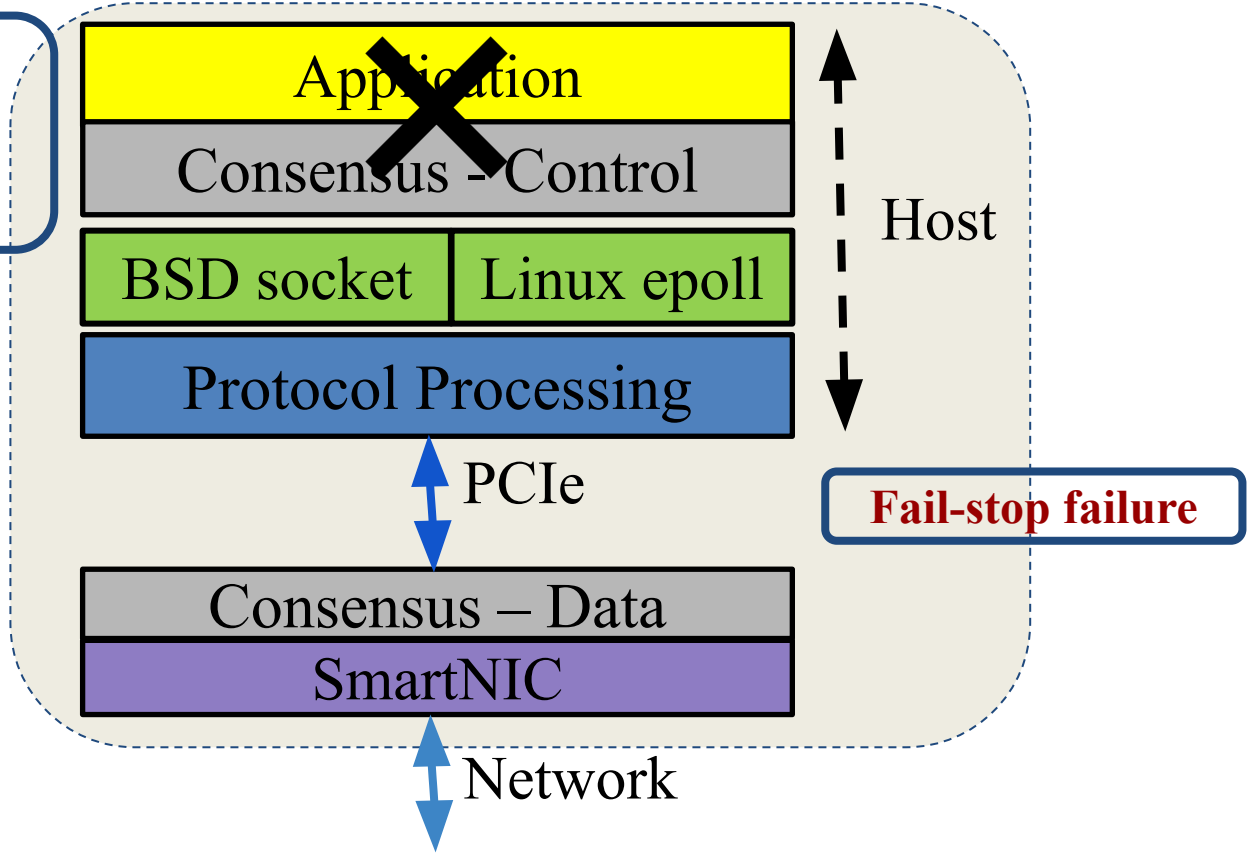
Dyad: Control Operations



Dyad: Application Failures

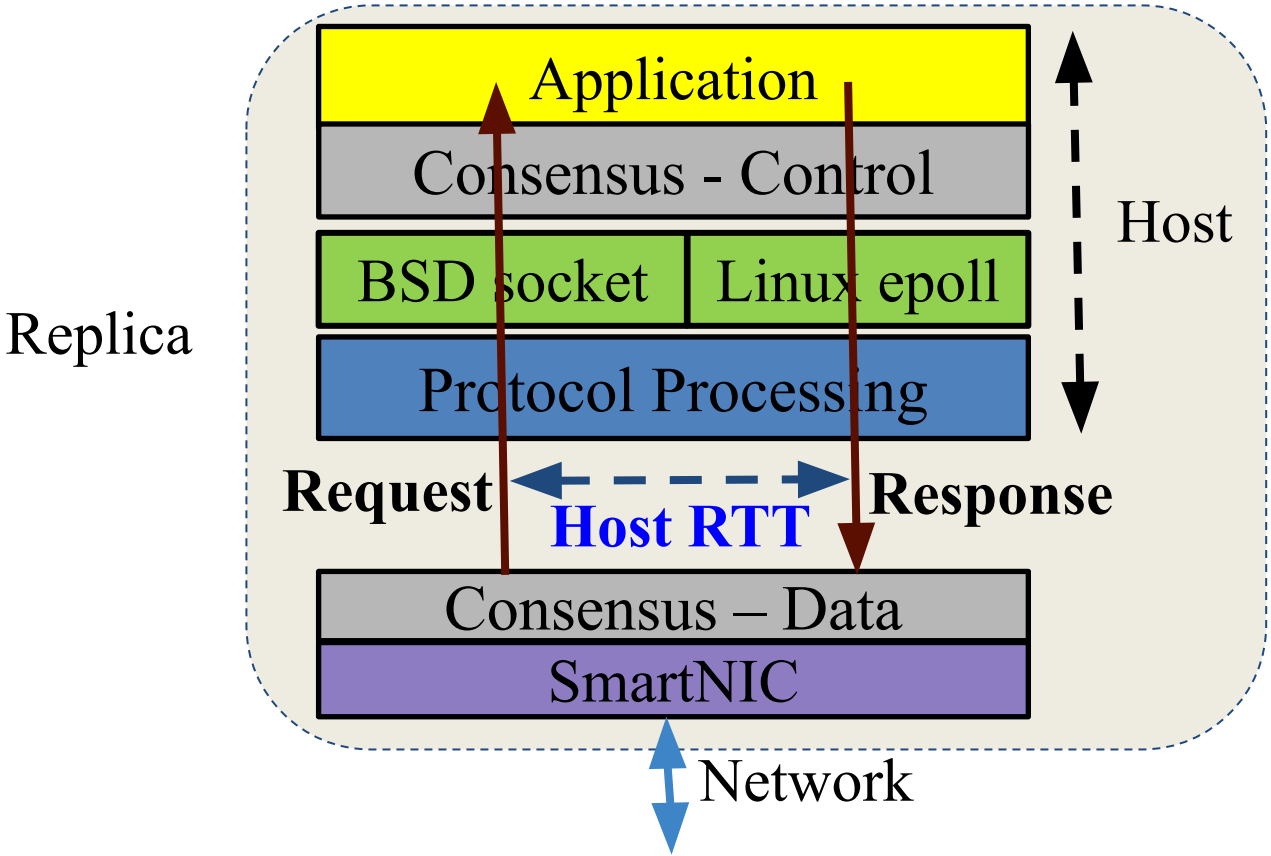
92% catastrophic failure - due to software [1]

Replica



[1] Simple Testing Can Prevent Most Critical Failures, OSDI'14

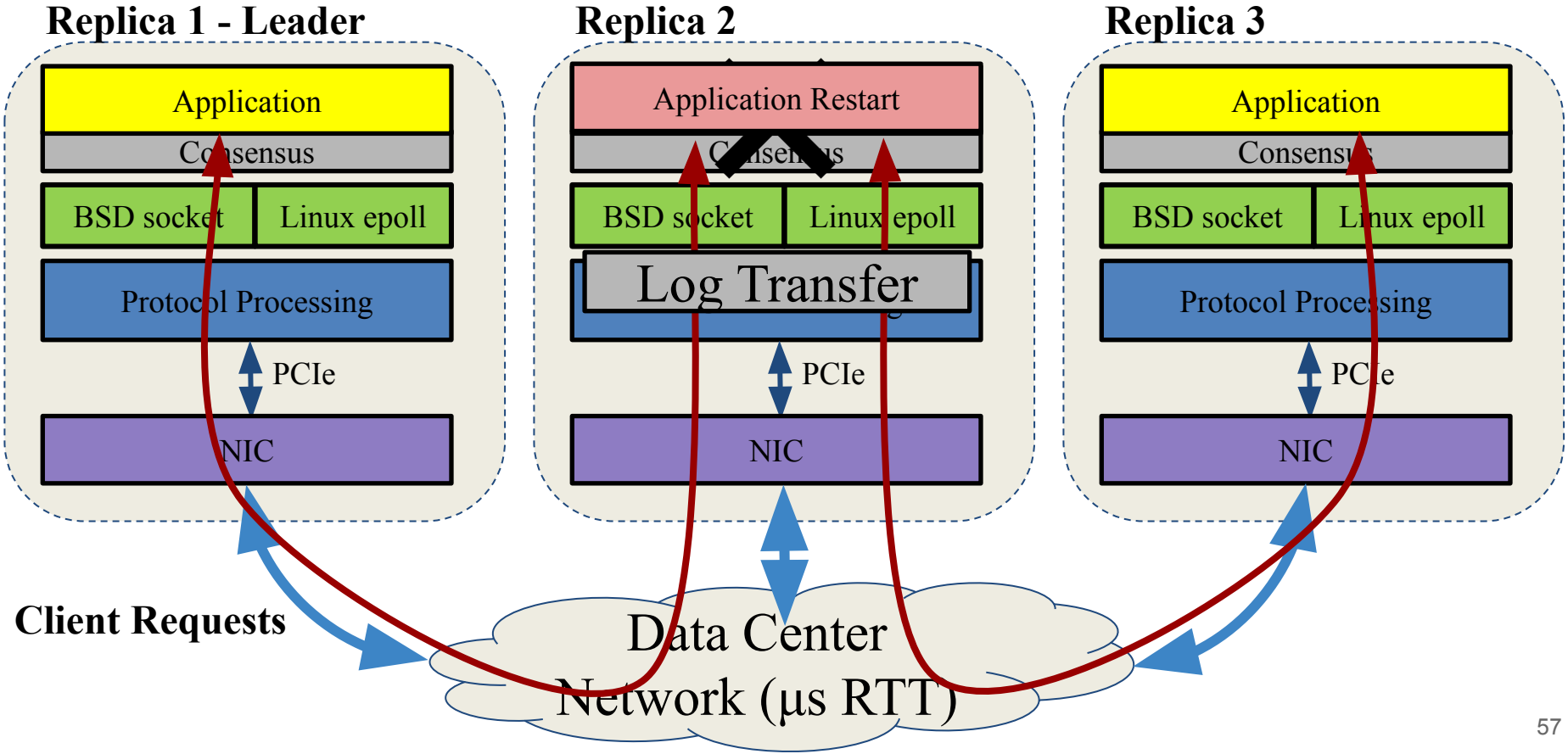
Dyad: Detecting Application Failures



Dyad: Detecting Application Failures

- Measure host RTT for each request
- Computed weighted average of host RTTs
- Detect failure - response not within host RTT threshold

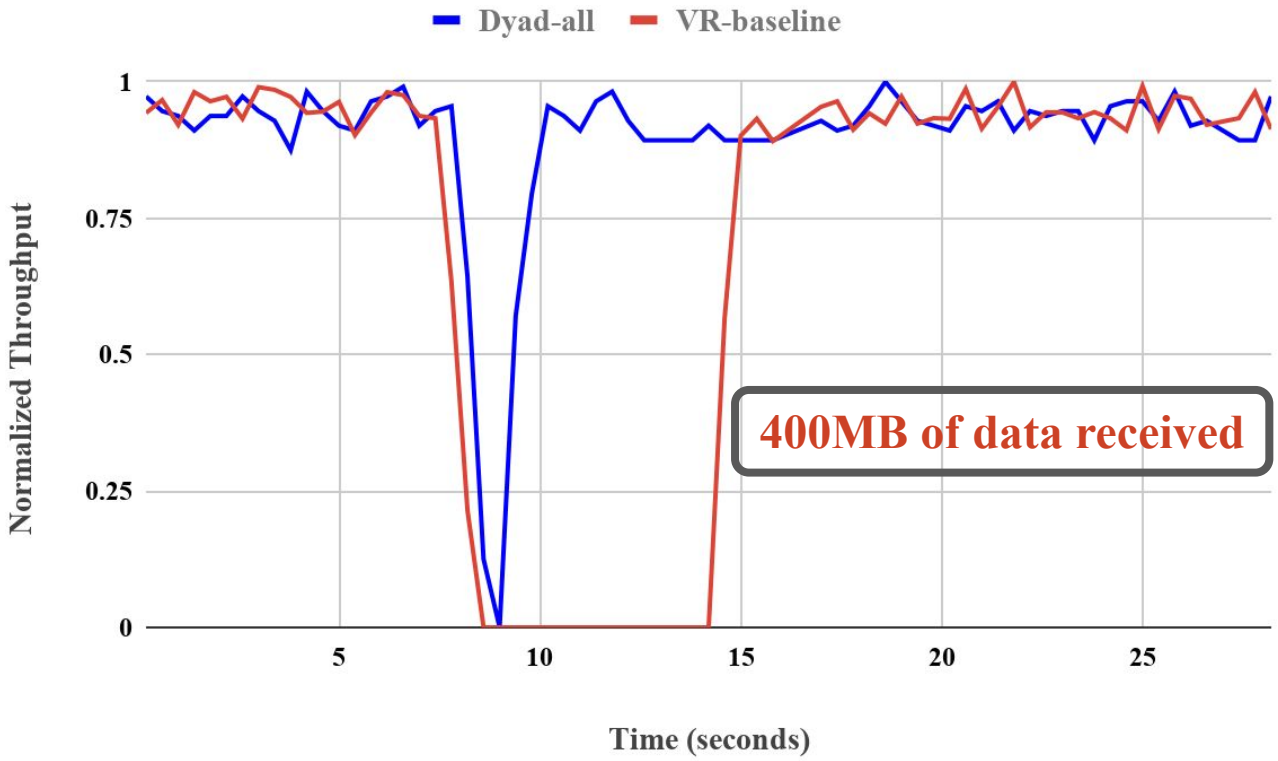
Application Recovery - VR



Dyad: Application Recovery

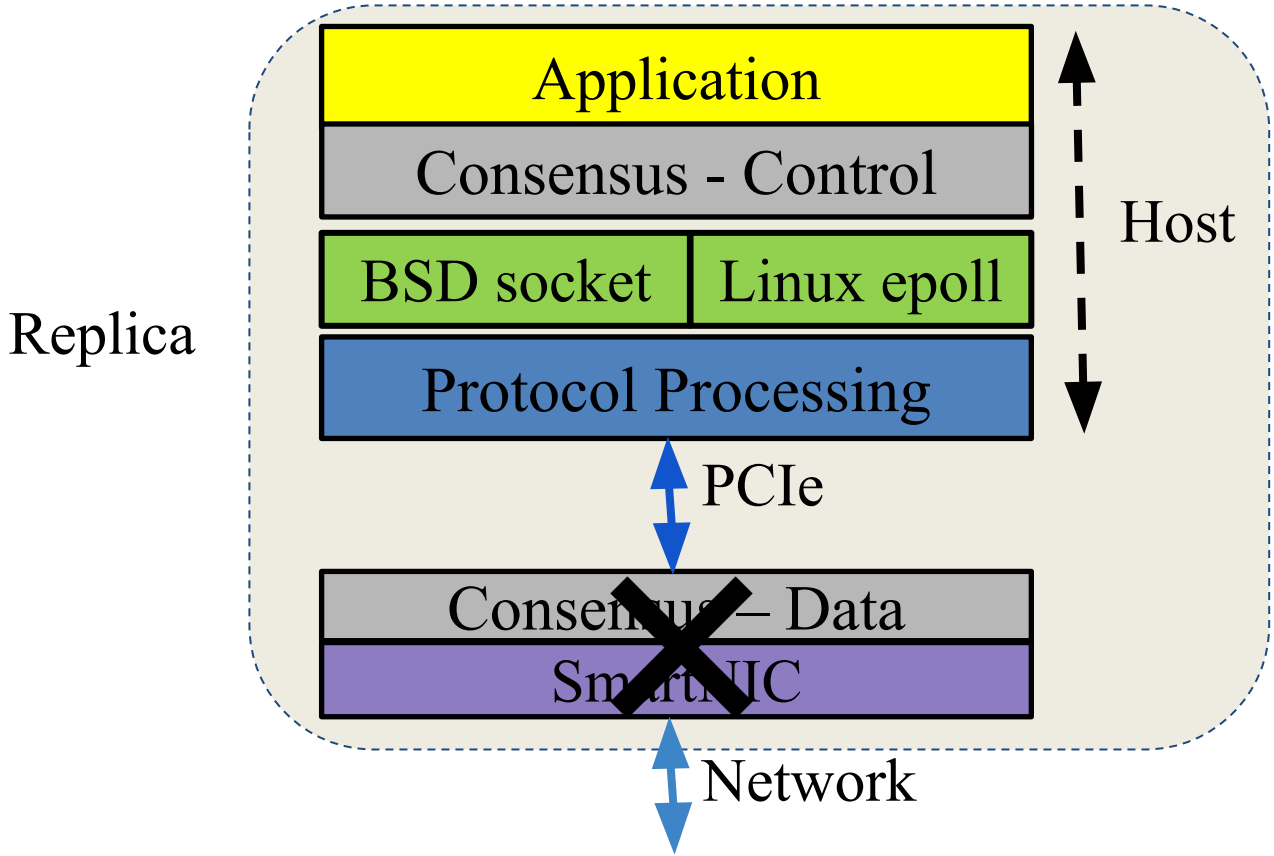
- Recovery using logs on SmartNIC
- Two stage recovery:
  - ➢ Recover logs from the SmartNIC
  - ➢ Recover remaining logs from other replicas

Dyad: Application Recovery



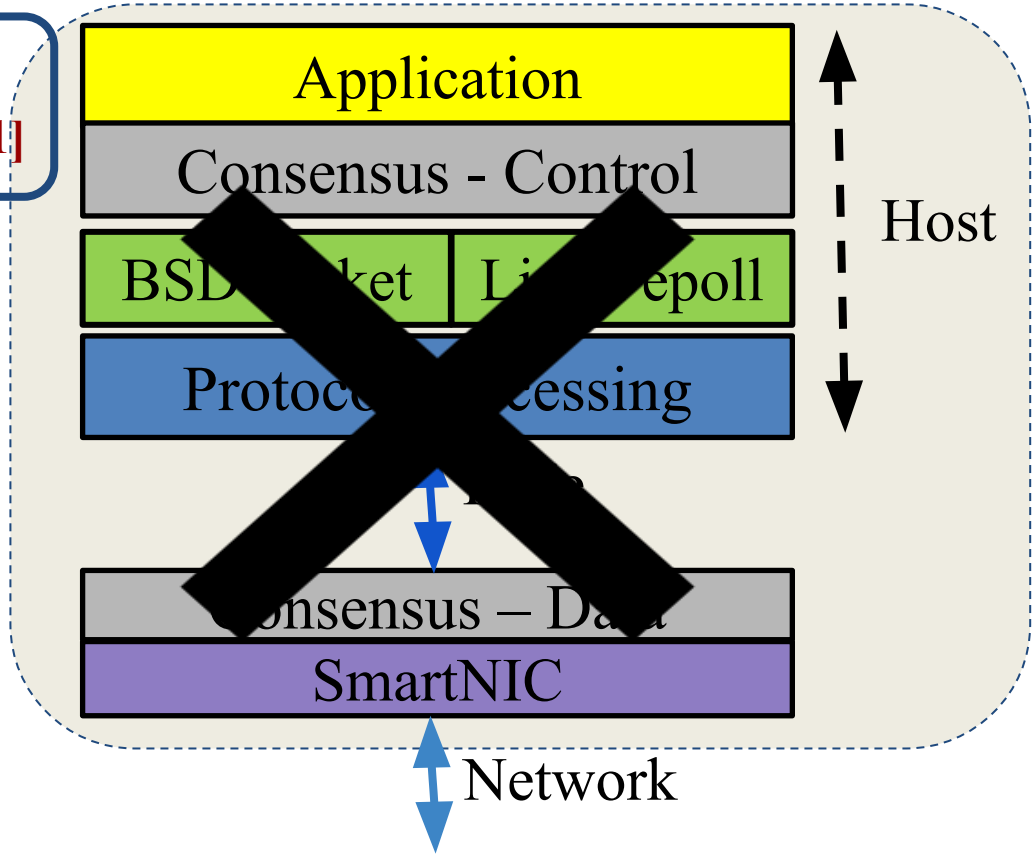
➢ Dyad reduces recovery time by up to **67%**

Dyad: SmartNIC Failure



Dyad: System Failure

8% - hardware faults, misconfigs [1]



[1] Simple Testing Can Prevent Most Critical Failures, OSDI'14

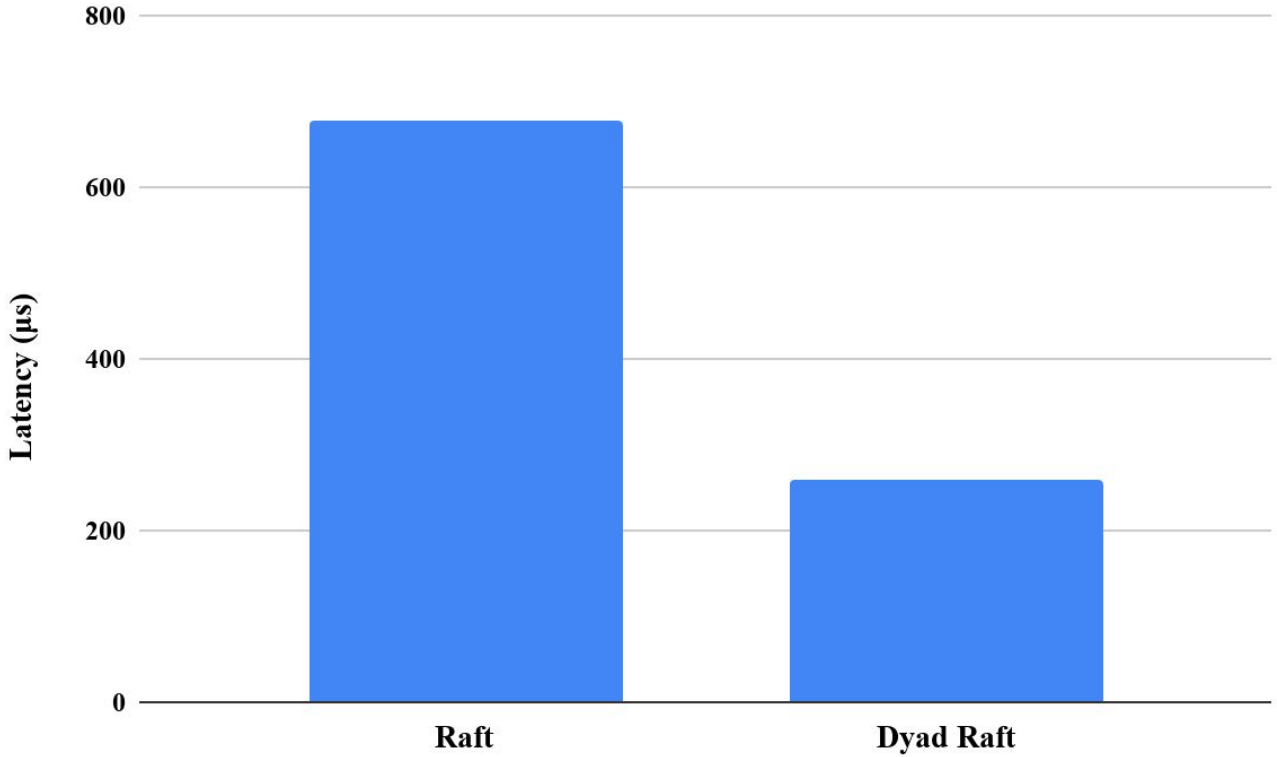
Dyad: System Recovery

- SmartNIC Failure:
  - ➢ Detected on the host using heartbeat/client messages
  - ➢ Existing VR recovery: fetch remaining logs from other replicas
- System Failure:
  - ➢ Existing VR recovery: fetch logs from other replicas
  - ➢ Dyad supports logging to disk from host (Raft)

Dyad: Reliable Connection

- Dyad Supports Raft:
  - ➢ Using TCP connection to replicas
  - ➢ TCP stack specifically decode Raft headers and payload
  - ➢ Host application logs client commands to disk for persistence

Dyad: Raft Latency

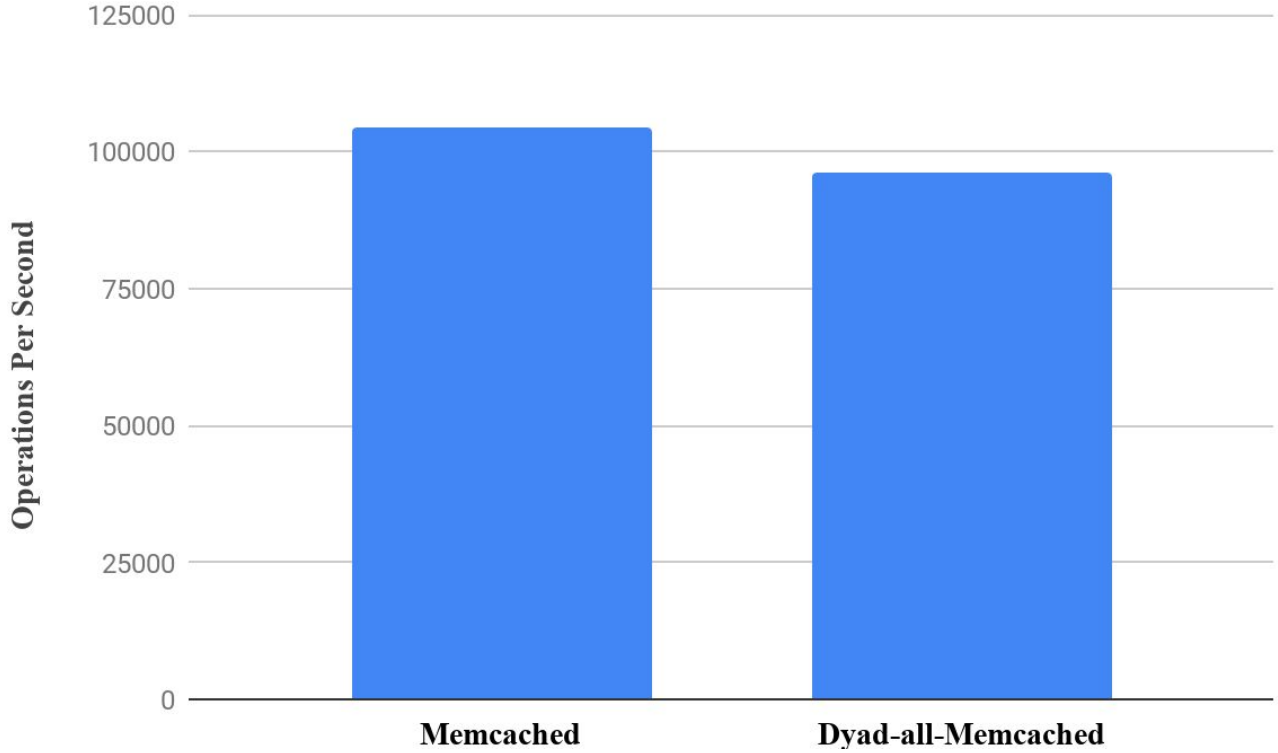


➢ Improves latency by up to **62%**

Dyad: Ease of Use

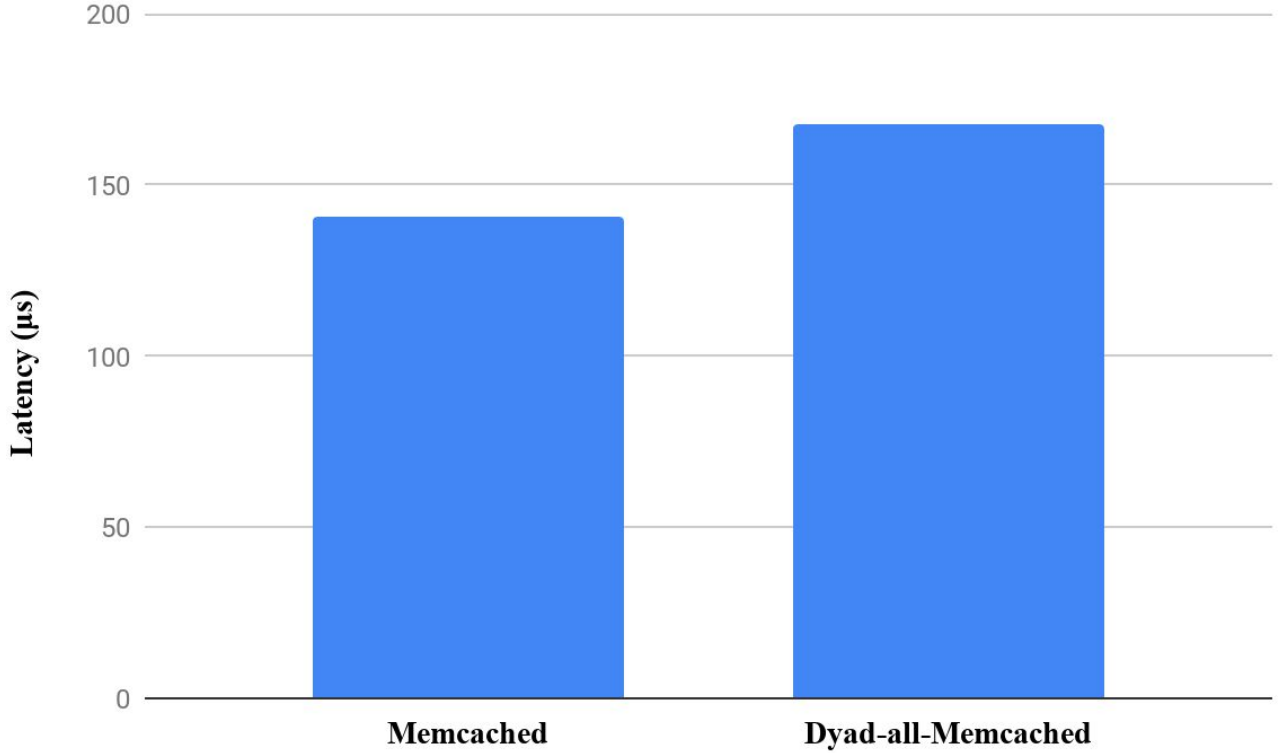
- Memcached:
  - ➢ Enable consensus for Memcached
    - ~100 lines of code for data operations on replica
  - ➢ Evaluate impact on latency and throughput

Dyad: Memcached Throughput



➢ Provides consensus with ~7% **reduction** in throughput

Dyad: Memcached Latency



➢ Provides consensus with **~16% increase** in latency

Dyad: Untangling Logically-Coupled Consensus

- Motivation
- Background
- Overview
- Design and Evaluation
- **Conclusion**

Dyad: Conclusion

- SmartNIC abstraction for consensus
- Data operations performed on the SmartNIC
- Control operations performed on the Host
- Enables consensus as a service on SmartNICs

Thesis: Conclusion

- Xps - Extensible Protocol Stack:
  - ➢ Abstraction in kernel, user space, and SmartNIC
- Latr - lazy TLB shootdown:
  - ➢ Kernel mechanism for TLB shootdown

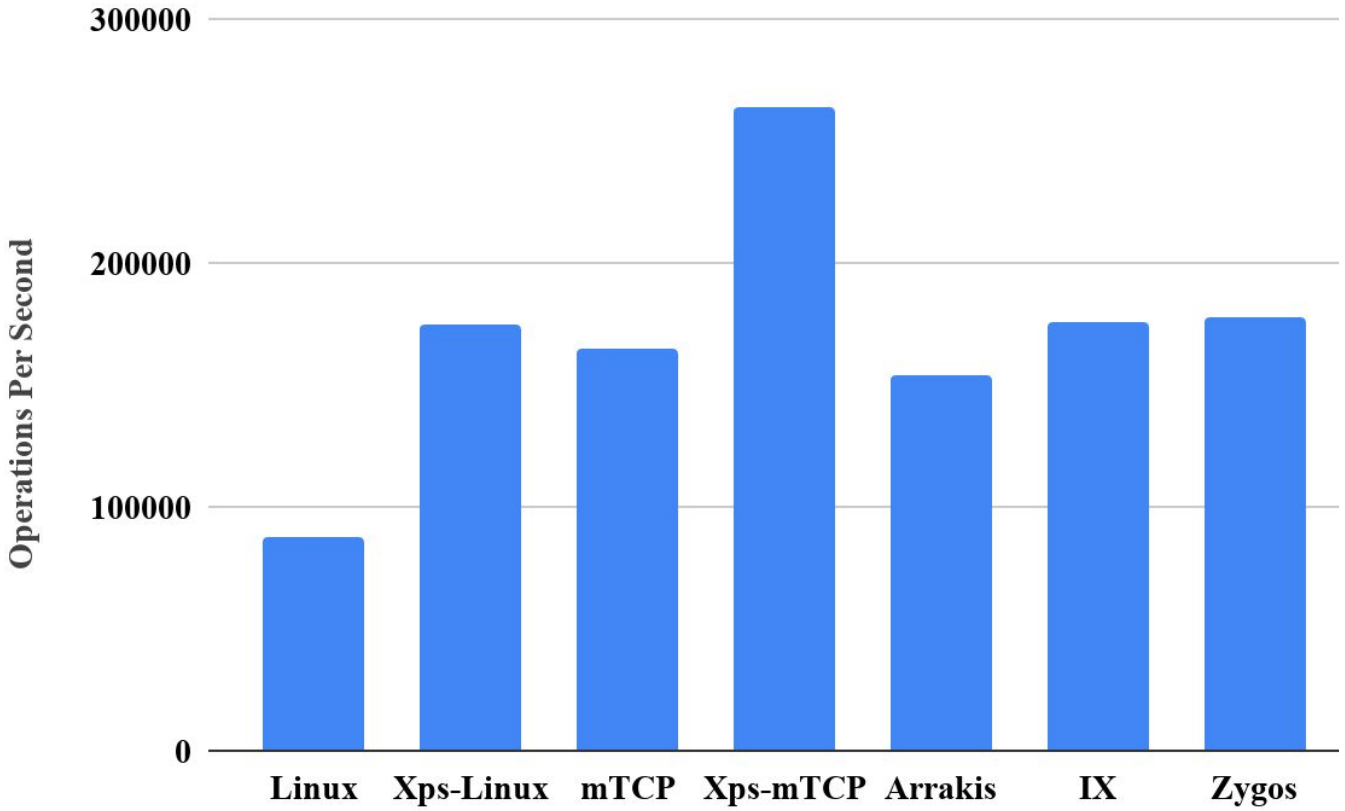
System abstractions and optimizations are needed at different levels of the software stack to reduce the latency and improve the throughput of current data-center applications.

Thank you!

Backup Slides

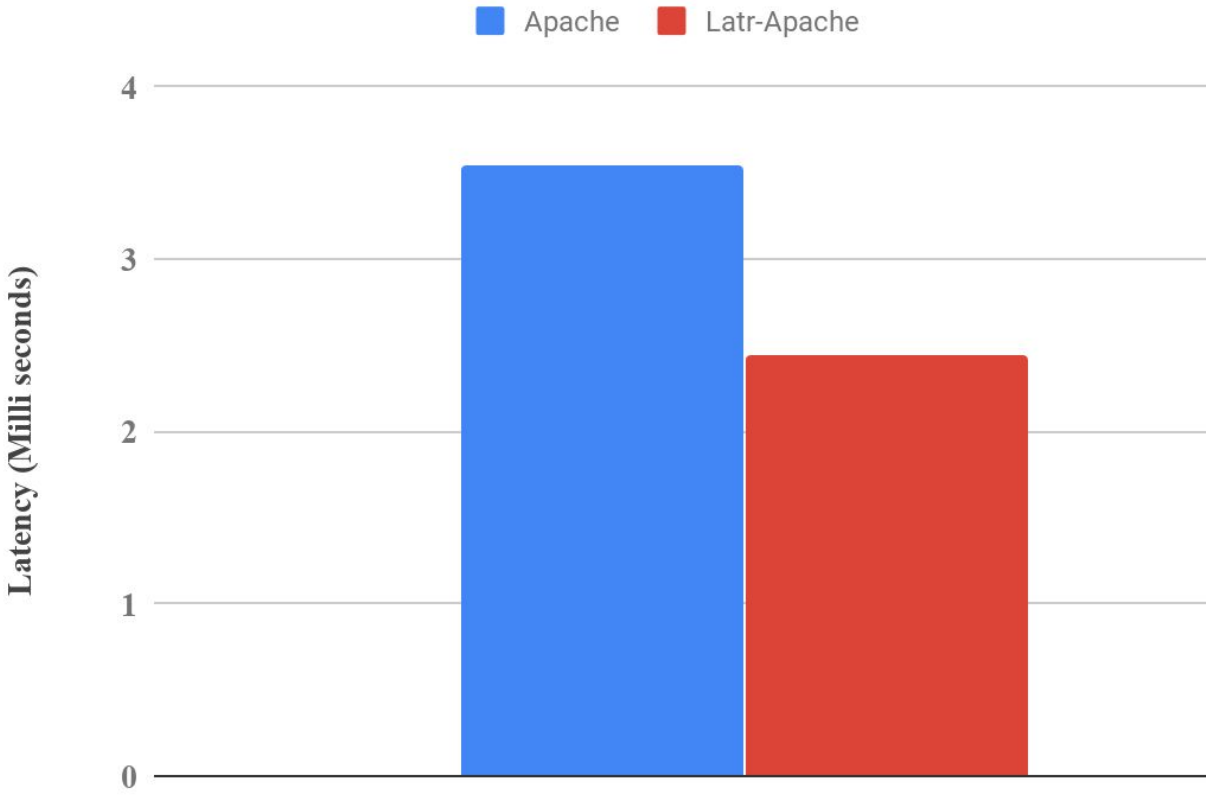
Arrakis

Redis comparison with Arrakis



Latr - Apache

Latr - Apache latency

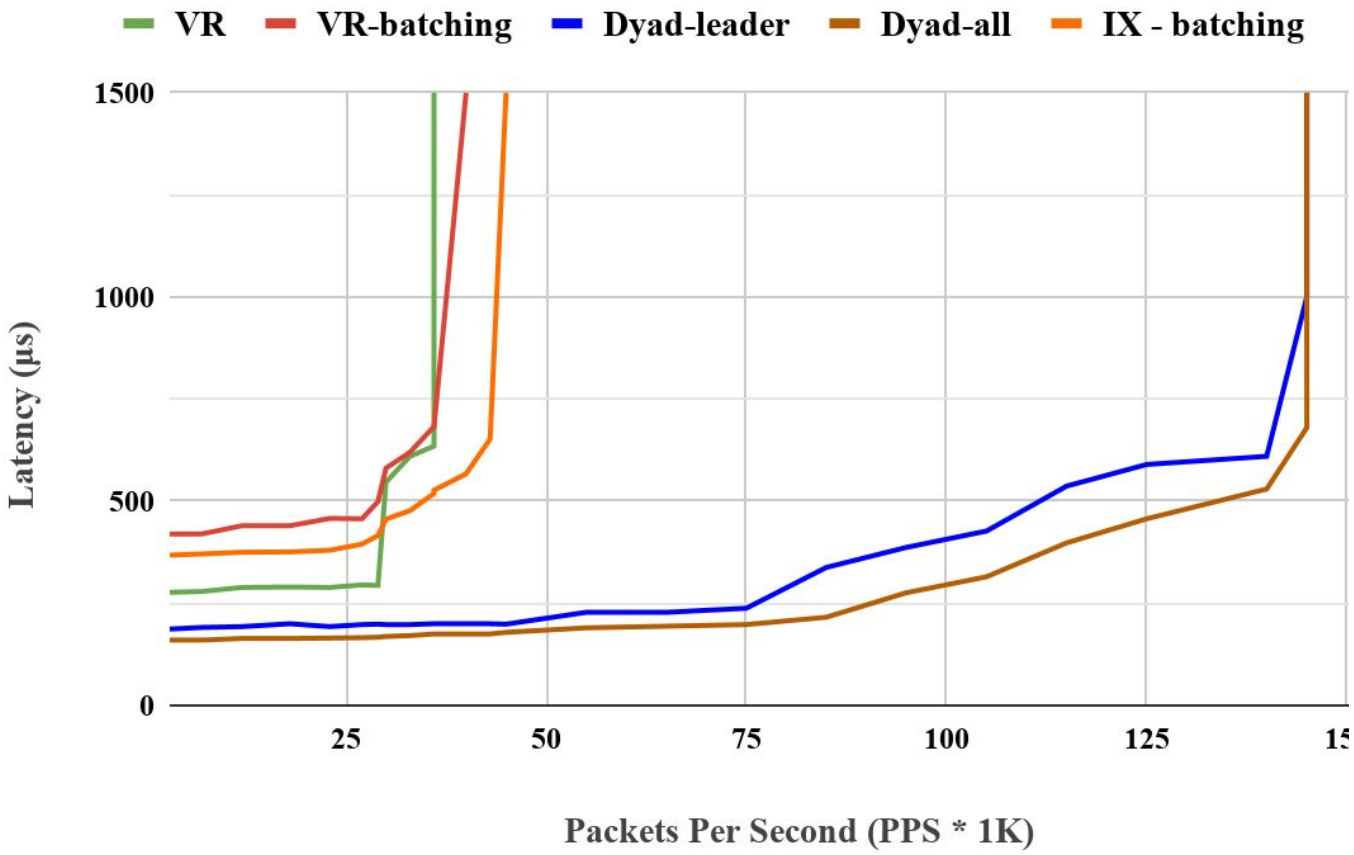


User-Space Stacks

User Space: Protocol processing

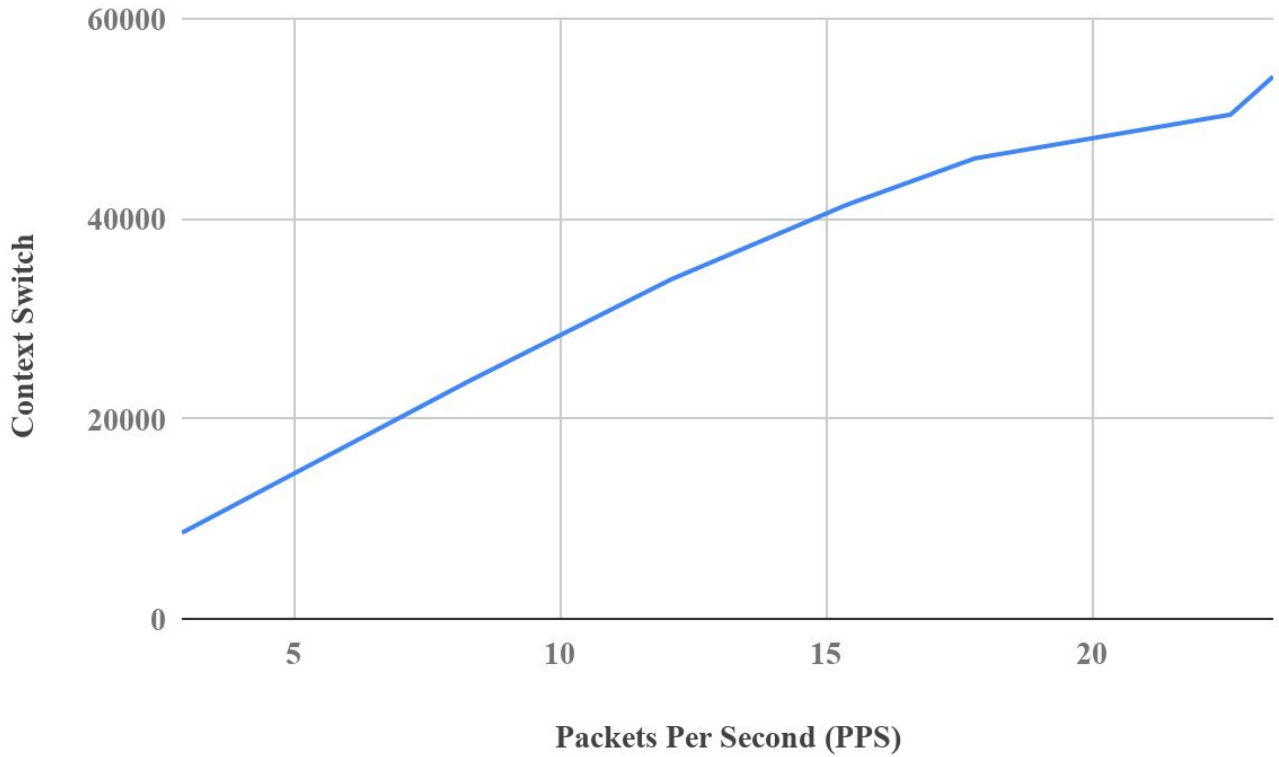
Systems	Latency (µs)	Mitigation
mTCP	~ 23	Batching
IX	~12	Batching
Arrakis	~2.6 - 6.3	None

VR: IX batching with 3 Replicas



Context Switch

VR - Leader Context Switch

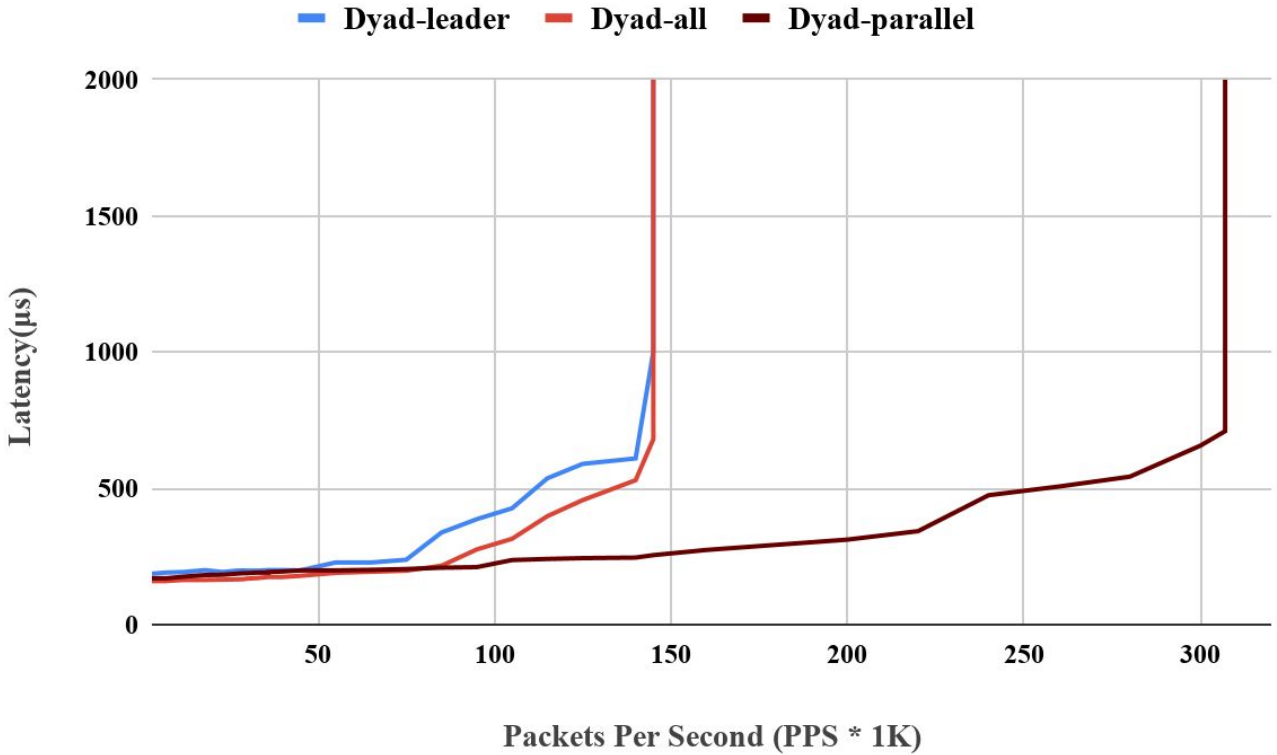


Dyad - Parallelism

Dyad: Application Parallelism

- Without SmartNIC:
  - ➢ Sequence numbers are available in prepareok message
  - ➢ Multi-thread execution by using the sequence number
- Dyad:
  - ➢ Request are ordered without containing the sequence number
  - ➢ SmartNIC appends the sequence number to the client request

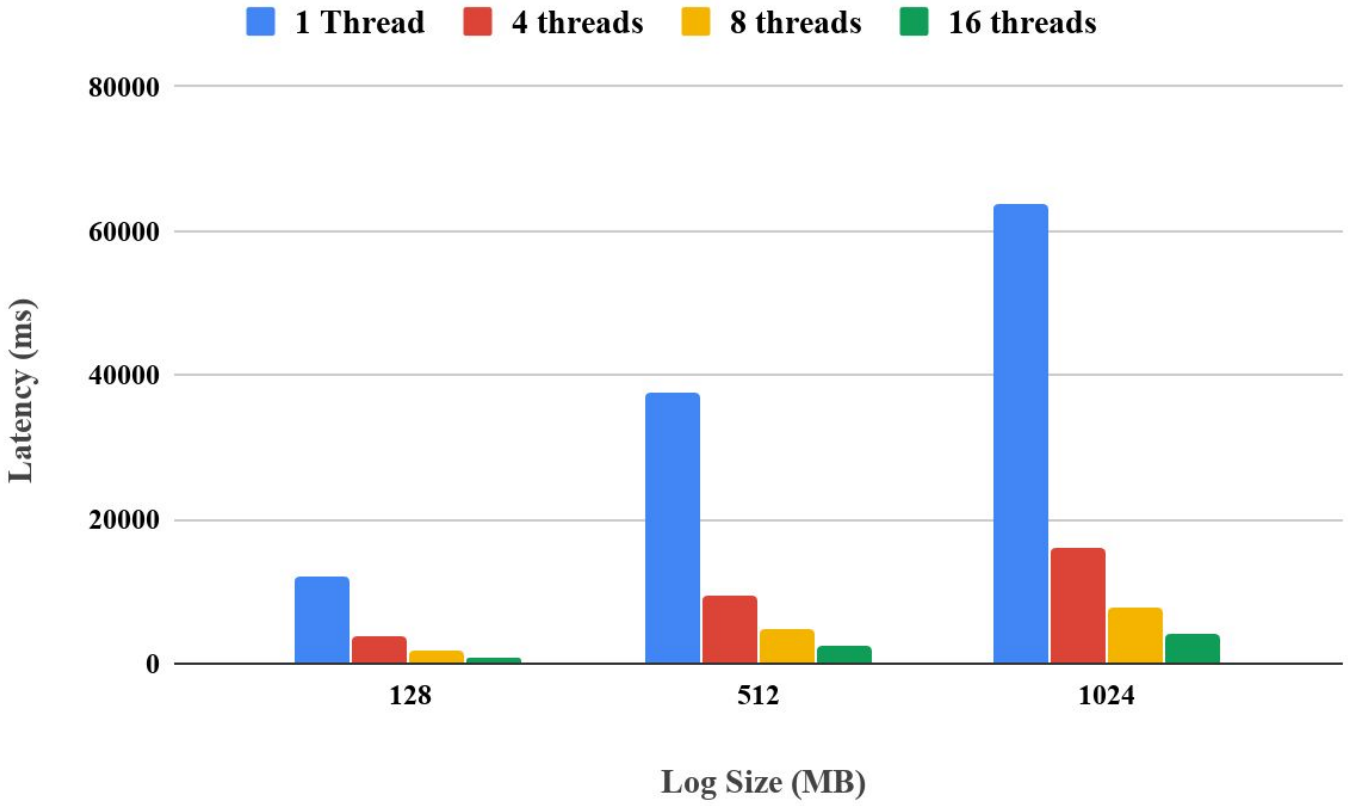
Dyad: Parallelism Timestamp Server



➢ Improves throughput by up to 2.1x

Reading Logs

Dyad: Log Read Throughput



- Log read throughput ~256 MB with 16 threads

Direct Cost Formula

Cost of Consensus - Direct and Indirect

$$Latency_{direct} = \frac{n-1}{2} * RTT + \overbrace{(n-1) * TX + \frac{n-1}{2} * RX}^{\text{Leader prepare}}$$

Replicas prepare

Consensus overhead increases with increasing replicas

$$Latency_{indirect} = \overbrace{\frac{n-1}{2} * RX}^{} + \overbrace{(n-1) * TX}^{}$$

Replicas commit

$$+ \overbrace{(n-1) * RX + t_{processing}}^{}$$

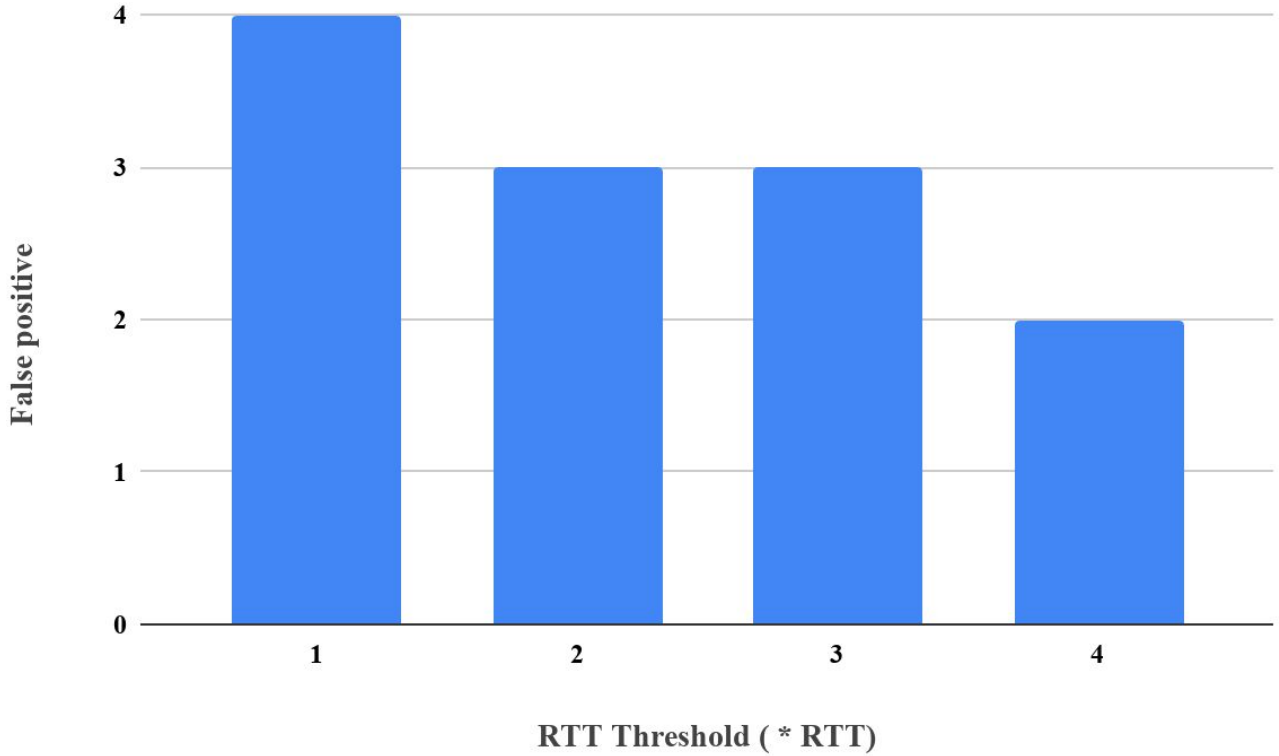
VR Recovery Data Transfer

Application Recovery - VR data transfer

Replicas	Log Size (MB)	Data transferred (MB)
3	100	200
5	100	400
7	100	600

False Positives RTT

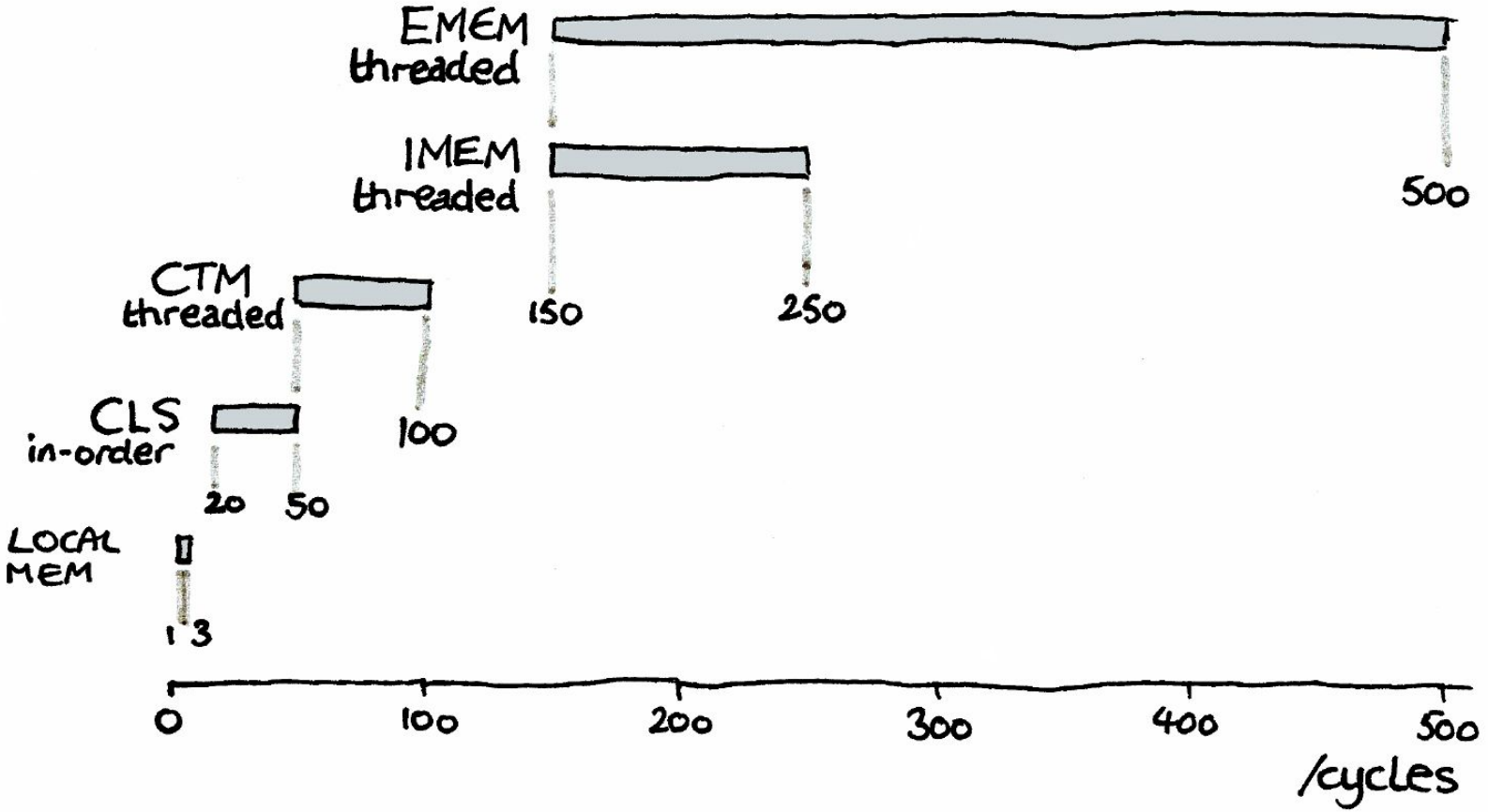
Dyad: False Positives with Timestamp Server



➢ RTT = ~96 µs

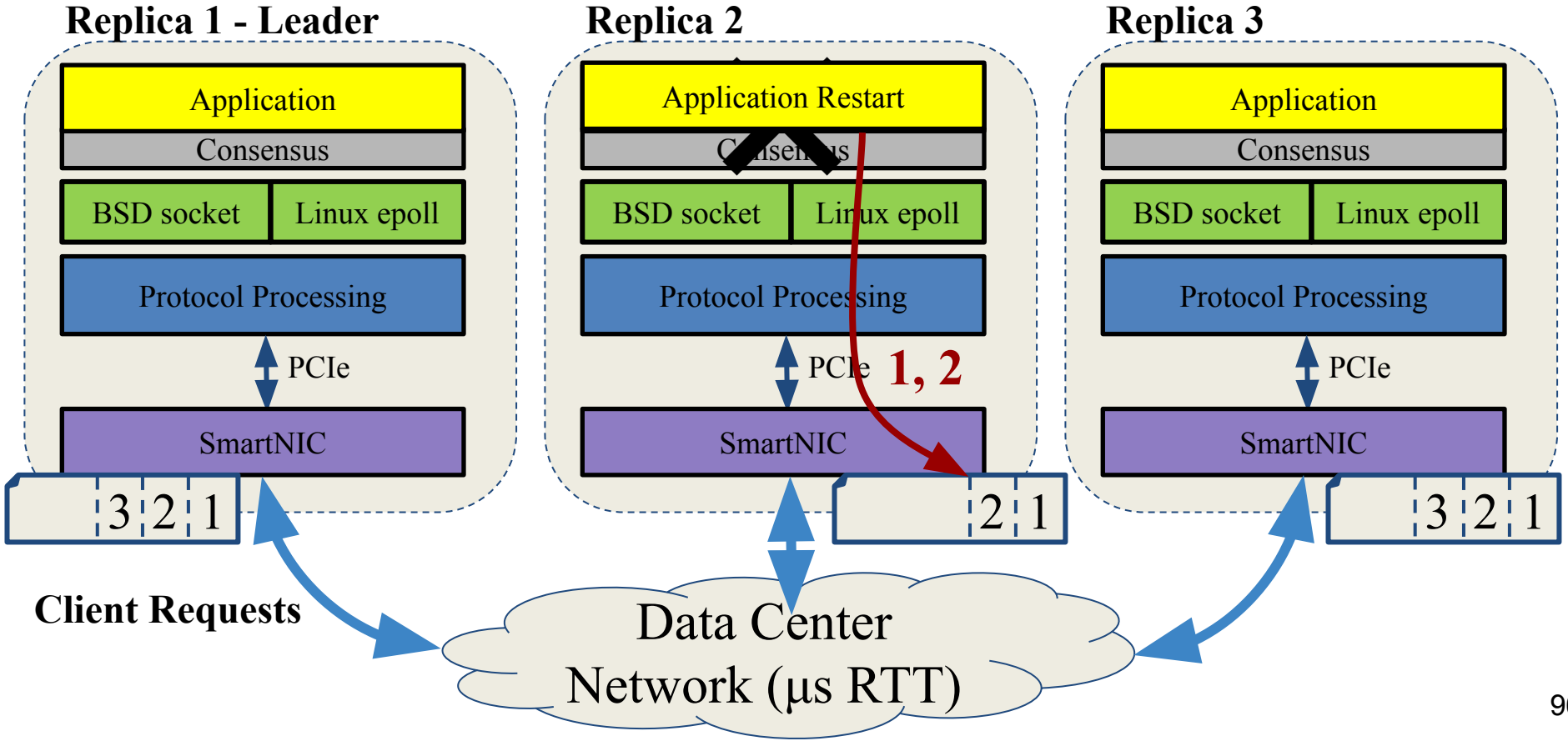
SmartNIC - Netronome

SmartNIC: Memory Hierarchy and Latency

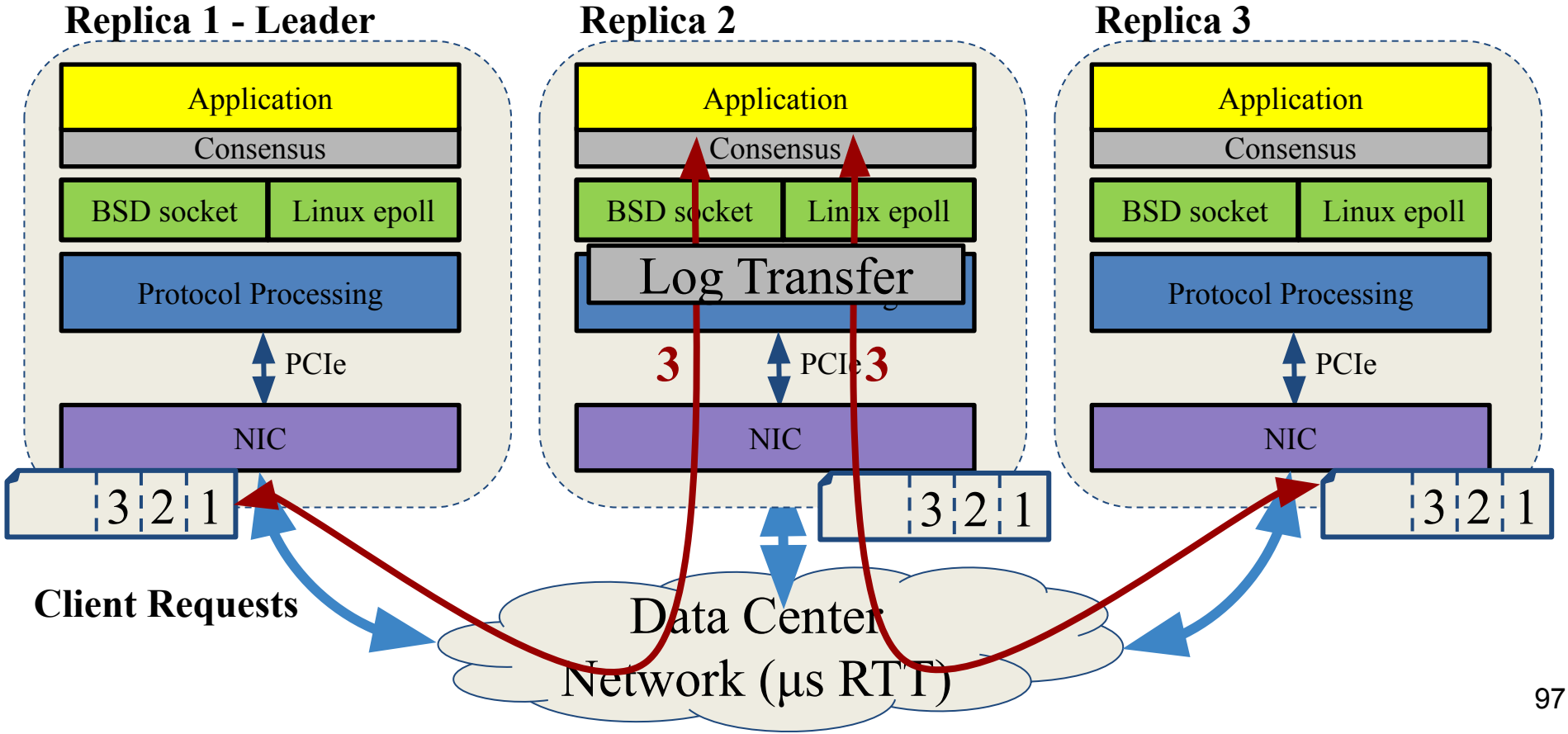


Recovery Example

Dyad - Recovery Phase1

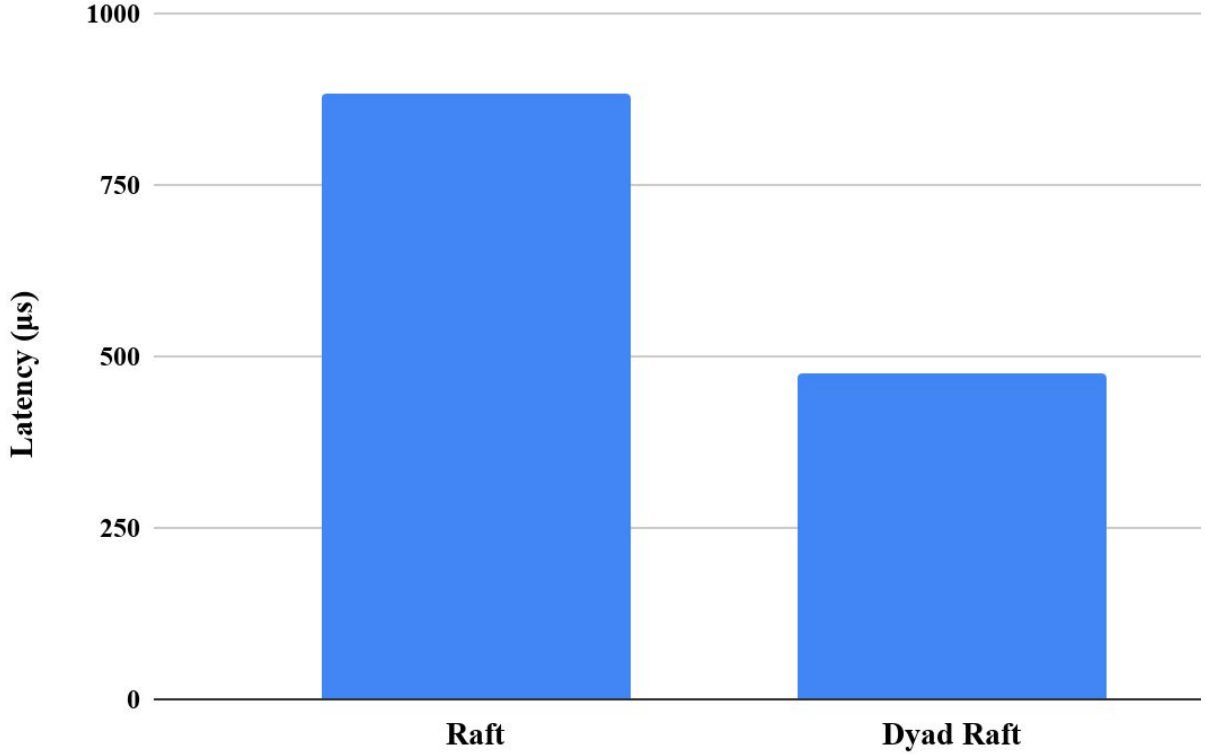


Dyad - Recovery Phase2



Raft - Logging to Disk

Dyad: Raft Latency with disk logging



➢ Improves latency by up to 46%

Dyad - Future Work

Dyad: Future Work

- Logging to disk from SmartNIC:
  - ➢ Possible with NVMe over fabric
  - ➢ Possible over PCIe? - ARM, FPGA, or NPU
- Optimize request handling:
  - ➢ Sending parsed requests to host