HA/TCP: A Reliable and Scalable Framework for TCP Network Functions

Haoyu Gu, Ali José Mashtizadeh, Bernard Wong

RCS Lab @ University of Waterloo

NSDI '25 – April 29, 2025

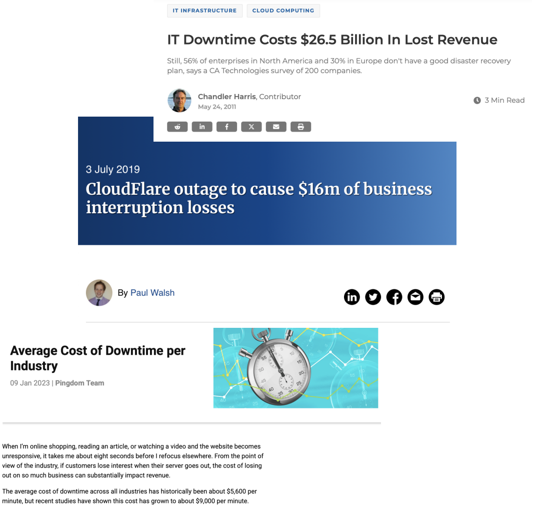
UNIVERSITY OF WATERLOO

Outline

1. Introduction
2. Design
3. How It Works
4. Optimizations
5. Evaluation
6. Conclusion

Network Functions (NFs)

- Packets often traverse multiple network functions (NFs)
- Building blocks for networks
- NF failures lead to large scale disruptions
- Network outage is expensive



IT INFRASTRUCTURE | CLOUD COMPUTING

IT Downtime Costs $26.5 Billion In Lost Revenue

Still, 56% of enterprises in North America and 30% in Europe don't have a good disaster recovery plan, says a CA Technologies survey of 200 companies.

Chandler Harris, Contributor
May 24, 2011
3 Min Read

3 July 2019

CloudFlare outage to cause $16m of business interruption losses

By Paul Walsh

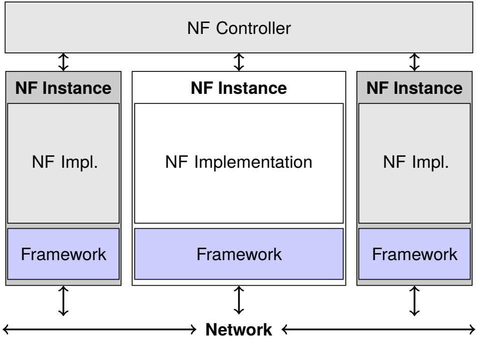
Average Cost of Downtime per Industry

09 Jan 2023 | Pingdom Team

When I'm online shopping, reading an article, or watching a video and the website becomes unresponsive, it takes me about eight seconds before I refocus elsewhere. From the point of view of the industry, if customers lose interest when their server goes out, the cost of losing out on so much business can substantially impact revenue.

The average cost of downtime across all industries has historically been about $5,600 per minute, but recent studies have shown this cost has grown to about $9,000 per minute.

Framework for NFs



- **Goal:** Reliability and scalability
- **Requires:** Failover, migration, and load-balancing

Two Categories of NFs

	L2–3	L4–7
Operation	Packet oriented	Stream oriented
Examples	Firewall, IDS, NAT, ...	WAN accel., Proxy, TLS term., ...
States		
States size		
Update frequency		

Two Categories of NFs

	L2–3	L4–7
Operation	Packet oriented	Stream oriented
Examples	Firewall, IDS, NAT, ...	WAN accel., Proxy, TLS term., ...
States		
States size		
Update frequency		

- Existing frameworks make use of small infrequent updates
- State replication with batching to reduce replication costs

Two Categories of NFs

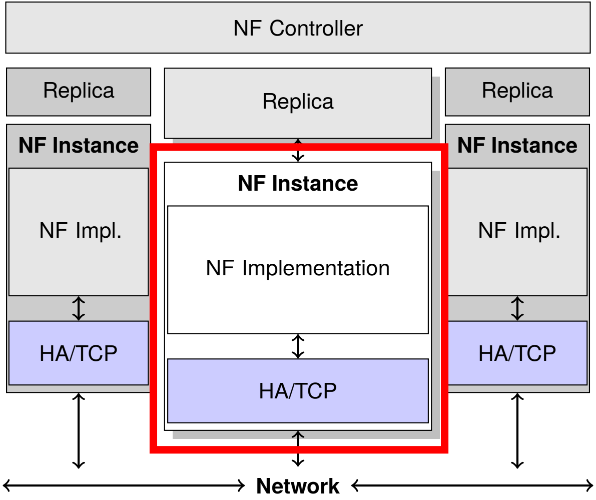
	L2–3	L4–7
Operation	Packet oriented	Stream oriented
Examples	Firewall, IDS, NAT, ...	WAN accel., Proxy, TLS term., ...
States	NF	NF & TCP (incl. buffers)
States size	10s Bytes	KBs
Update frequency	Per-flow or per-packet	Multiple times per-packet

- Existing frameworks make use of small infrequent updates
- State replication with batching to reduce replication costs
- Hard to apply for L4–7:
  - Combination of increased size, frequency, and complexity

Outline

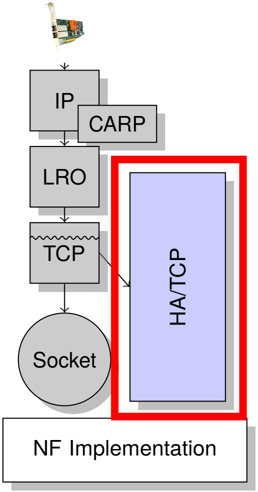
Our Approach: HA/TCP

- Framework for L4–7 NFs
  - Build NF on top of HA/TCP



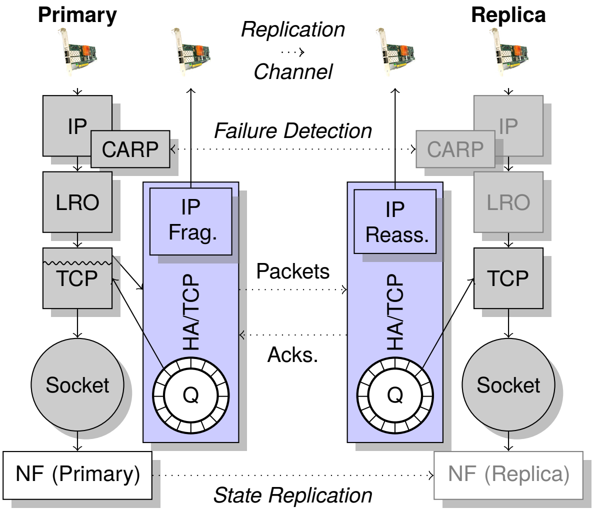
Our Approach: HA/TCP

- Framework for L4–7 NFs
  - Build NF on top of HA/TCP
  - Extend existing TCP stack



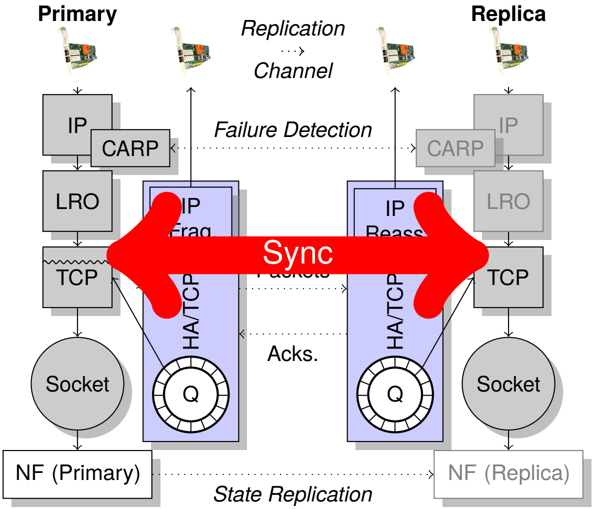
Our Approach: HA/TCP

- Framework for L4–7 NFs
  - Build NF on top of HA/TCP
  - Extend existing TCP stack
- Active–active replication
  - Reconstruct states locally



Our Approach: HA/TCP

- Framework for L4–7 NFs
  - Build NF on top of HA/TCP
  - Extend existing TCP stack
- Active–active replication
  - Reconstruct states locally

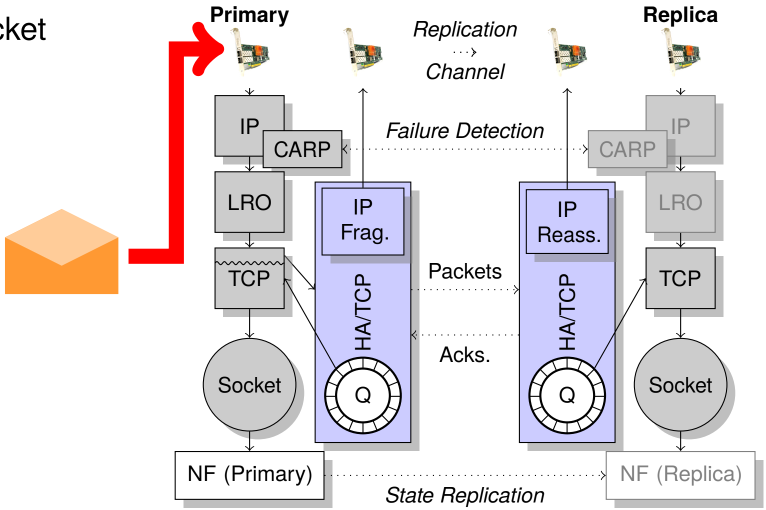


Outline

1. Introduction
2. Design
3. How It Works
4. Optimizations
5. Evaluation
6. Conclusion

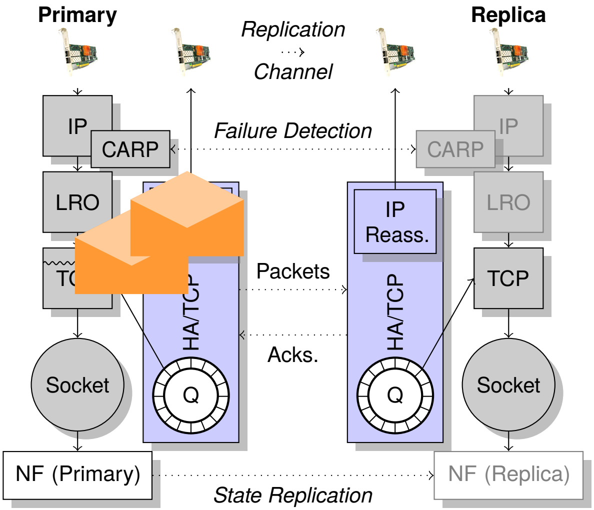
Steady State Processing

- Primary receives the packet



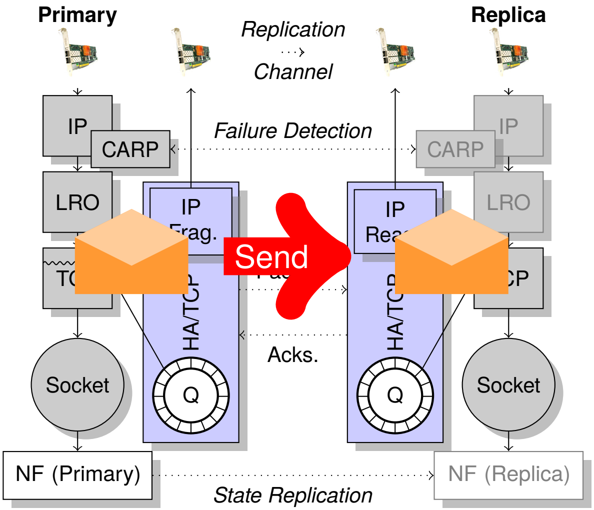
Steady State Processing

- Primary receives the packet
  - Duplicate the packet
  - Queue original packet



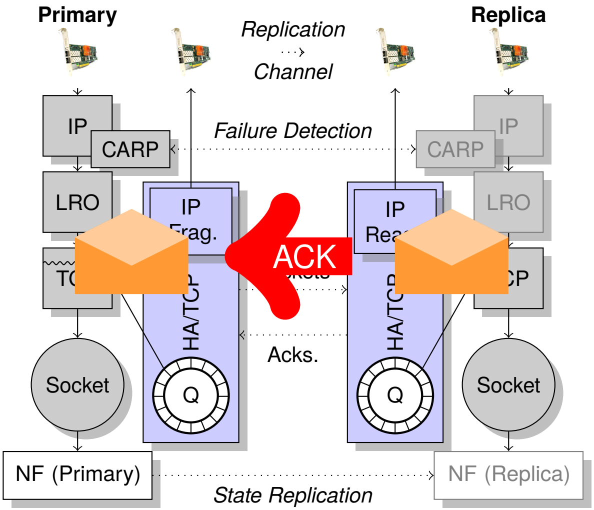
Steady State Processing

- Primary receives the packet
  - Duplicate the packet
  - Queue original packet
  - Sends to replica



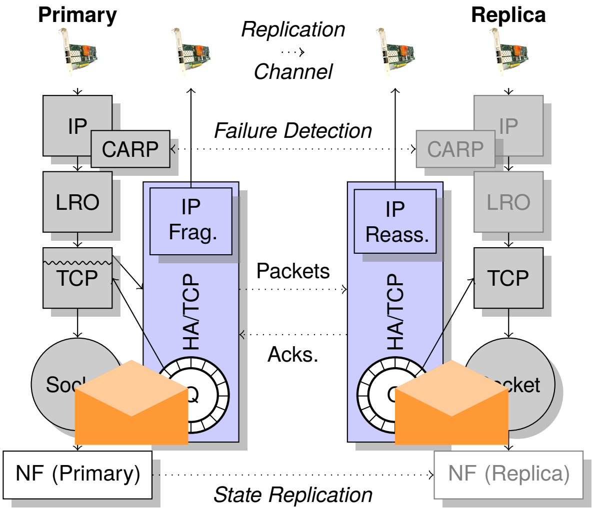
Steady State Processing

- Primary receives the packet
  - Duplicate the packet
  - Queue original packet
  - Sends to replica
- Replica acknowledges
  - Place in replica's packet queue



Steady State Processing

- Primary receives the packet
  - Duplicate the packet
  - Queue original packet
  - Sends to replica
- Replica acknowledges
  - Place in replica's packet queue
- Both sides process packet
  - Primary: deliver to stack
  - Replica: dequeue and deliver if meets criteria

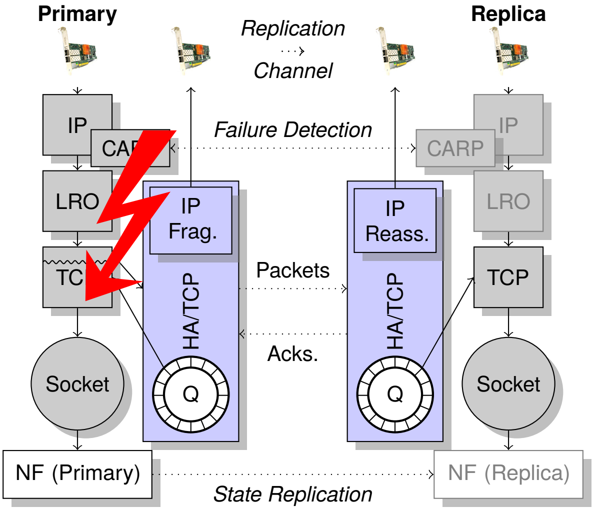


Dequeueing on Replica

- Replica packet queue to hide the differences
- Tolerate replica application lag behind primary
  - Improve throughput and tail latency
- Ensure delivery only when TCP will accept the packet
  - We don't want to make another copy of the packet (expensive!)
  - Valid TCP state (seq# and ack#), don't overrun window, ...

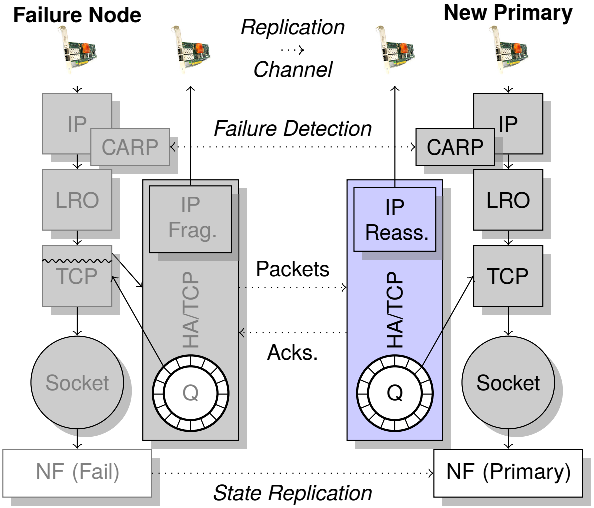
Failure Recovery

- Fast failover/migration
  - Active–active replication



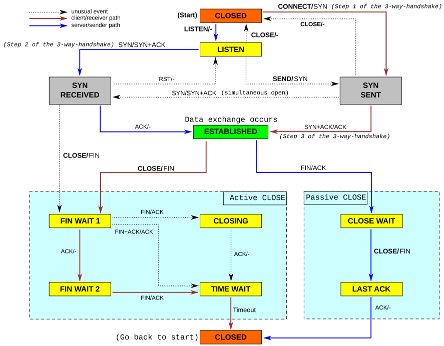
Failure Recovery

- Fast failover/migration
  - Active–active replication
- Process:
  - Drain the packet queue
  - Take over IP address
  - Continue service…



Application Integration

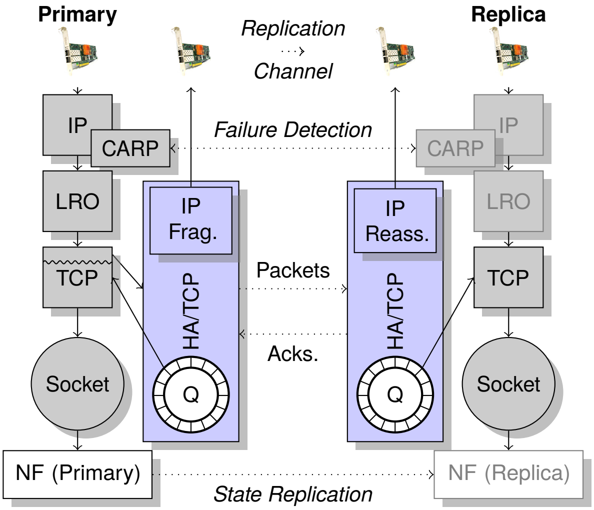
- 3 lines of code to enable HA/TCP
- HA/TCP only replicates stack state
  - HA/TCP guarantees that the network inputs are identical
- Use output determinism programming model



Outline

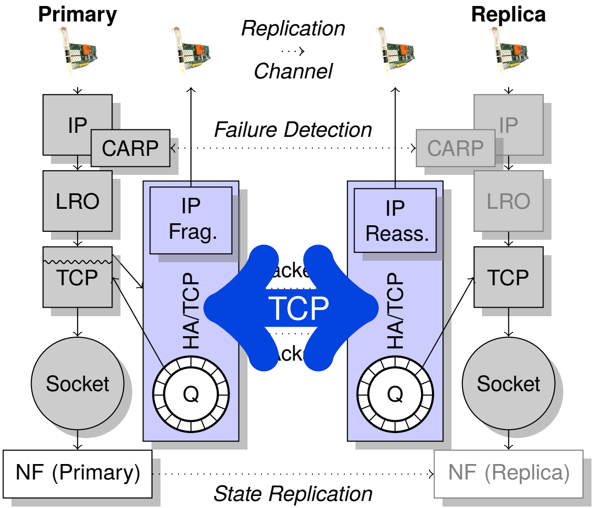
High Performance Replication Channel

- **Performance goal:** support 100 Gbps link
- First design uses TCP for the replication channel
  - Simple to build
  - No need to think about packet loss/retransmits
- But only achieves 54 Gbps



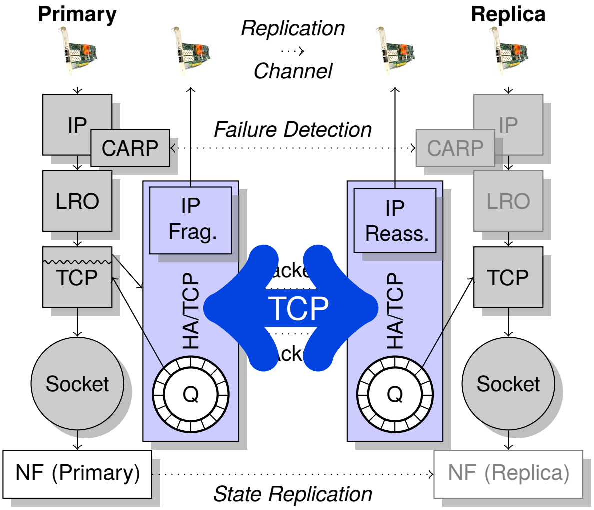
High Performance Replication Channel

- **Performance goal:** support 100 Gbps link
- First design uses TCP for the replication channel
  - Simple to build
  - No need to think about packet loss/retransmits
- But only achieves 54 Gbps



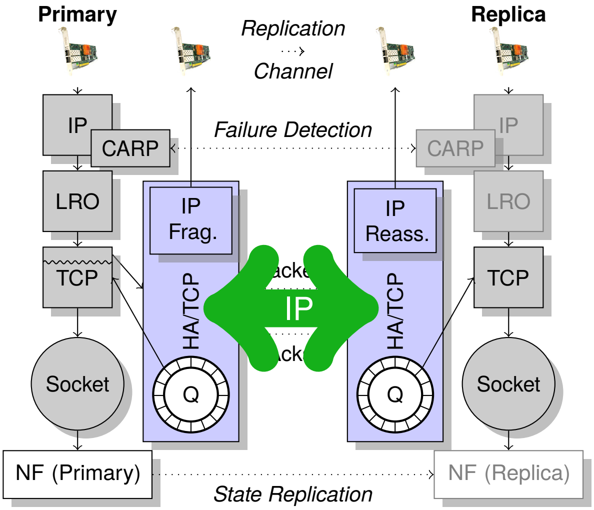
Challenges with TCP for Replication Channel

- High overheads
  - High CPU on TCP encapsulation and processing
- TCP over TCP (TCP meltdown)
  - CC disagreement
  - Unstable performance



Challenges with TCP for Replication Channel

- High overheads
  - High CPU on TCP encapsulation and processing
- TCP over TCP (TCP meltdown)
  - CC disagreement
  - Unstable performance
- **Solution:** Switch to IP for replication channel



Challenges of Using IP Protocol

- Loss of TCP Reliability
- Loss of TCP Optimization/Offload

Loss of TCP Reliability

- No packet loss detection/retransmission
- Observation
  - Active-active replication designed for LAN and Metro area networks
  - Lower packet loss and better latency than WAN

Loss of TCP Reliability

- No packet loss detection/retransmission
- Observation
  - Active-active replication designed for LAN and Metro area networks
  - Lower packet loss and better latency than WAN
- Rely on TCP between client and primary to retransmit
  - Packet loss in channel prevents TCP acks
  - Client resend after retransmit timeout *(rtt + 4*rttvar)*
- Benefits
  - Simplifies design
  - TCP congestion control adapts to overall link quality

Loss of TCP Optimization/Offload

- TCP Segment Offload/Large Receive Offload (TSO/LRO)
  - Main insight: stack performance is proportional to PPS (not Gbps)
  - Significantly improve performance
  - Processing 64 KiB packets (Up to $\times 6.2$ improvement)

Loss of TCP Optimization/Offload

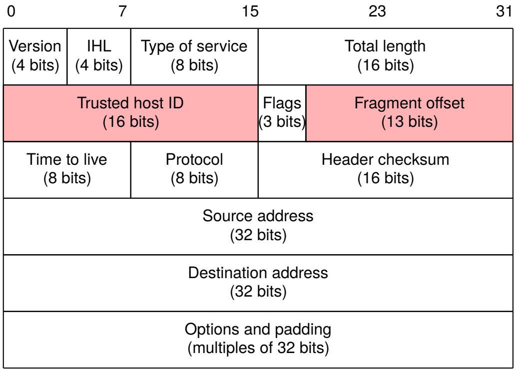
- TCP Segment Offload/Large Receive Offload (TSO/LRO)
  - Main insight: stack performance is proportional to PPS (not Gbps)
  - Significantly improve performance
  - Processing 64 KiB packets (Up to ×6.2 improvement)
- IP fragmentation/reassembly can serve the same purpose
  - Used by UDP for large packets

Loss of TCP Optimization/Offload

- TCP Segment Offload/Large Receive Offload (TSO/LRO)
  - Main insight: stack performance is proportional to PPS (not Gbps)
  - Significantly improve performance
  - Processing 64 KiB packets (Up to ×6.2 improvement)
- IP fragmentation/reassembly can serve the same purpose
  - Used by UDP for large packets
- Challenge: IP reassembly suffers from collisions

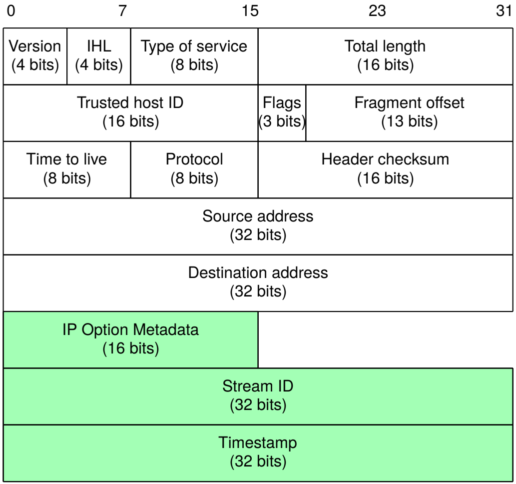
IP Fragmentation/Reassembly Collisions

- IP reassembly collision (RFC 4963)
  - QUIC and SCTP
    do not support IP fragmentation
- 200 collisions per second at
  25 Gbps with multiple connections
  - Results in packet loss or data corruption
  - Packet loss $\rightarrow$ TCP retransmits
  - CC bandwidth reduction



Eliminating Collisions with New IP Option

- Stream ID IP option
- Identifies the replication channel
- Useful for other protocols



Outline

Evaluation

- FreeBSD 13.1 kernel
- Micro-benchmark
  - Replication overhead
  - Latency overheads
  - Migration and failover
- Application benchmarks

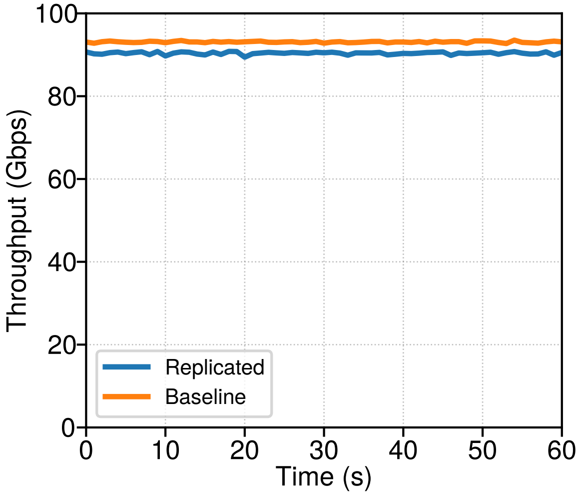
	Component	SLOC
System	HA/TCP TCP extension	10K
	HA/TCP IP clustering	1.4K
Apps	SOCKS proxy	3.3K
	WAN sccelerator	8.7K
	Distributed load-balancer	1.2K

Setup

- Dual Intel Xeon 6342 processors
- 100 Gbps Mellanox ConnectX-6 NICs
- Mellanox SN2100 100 Gbps switch
  - Worst case: client $\leftrightarrow$ primary latency = primary $\leftrightarrow$ replica latency

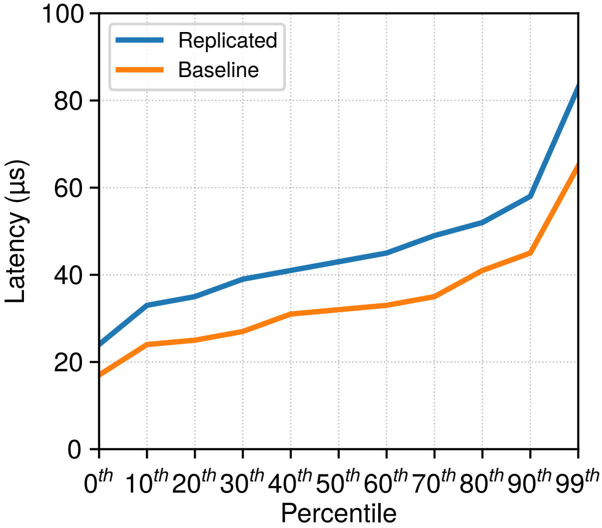
Replication Overhead

- Worst case for HA/TCP
  - Receive-bound traffic
- Baseline: 93.60 Gbps
- HA/TCP: 90.38 Gbps
- 3.4% decrease in throughput



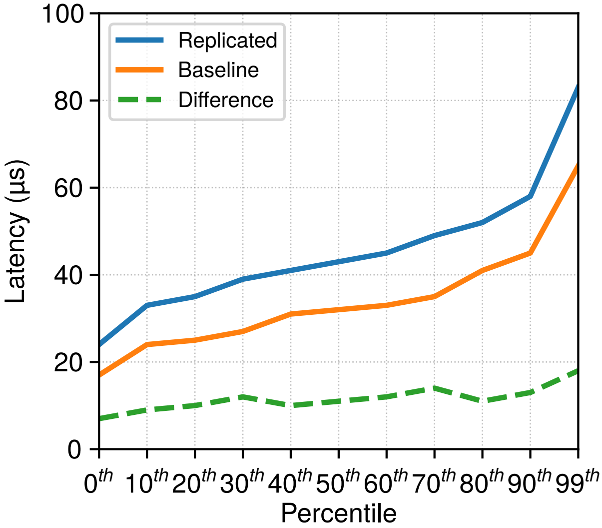
Latency

- Worst case for HA/TCP
  - All nodes on the same network



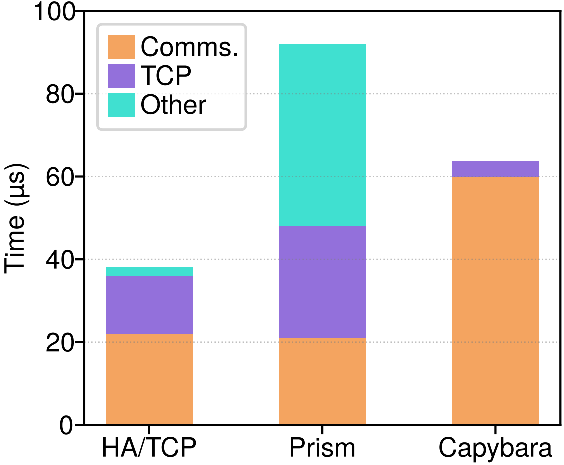
Latency

- Worst case for HA/TCP
  - All nodes on the same network
- Low tail latency



Migration

- Compares TCP migration
- HA/TCP migrates in 38 µs



Failover

- On average 300 ms disruptions
- CARP failure detection dominates
  - Configured to 300 ms average detection time
  - Experimentally set to minimize false positives

Outline

Conclusion

- Challenges of building on real systems
  - FreeBSD network stack is 150K SLOC
  - HA/TCP is 10K SLOC
- See the paper for:
  - More challenges, optimizations, benchmarks
  - Our distributed load balancer
- Our code is available at `https://github.com/rcslab/hatcp`

Thank You

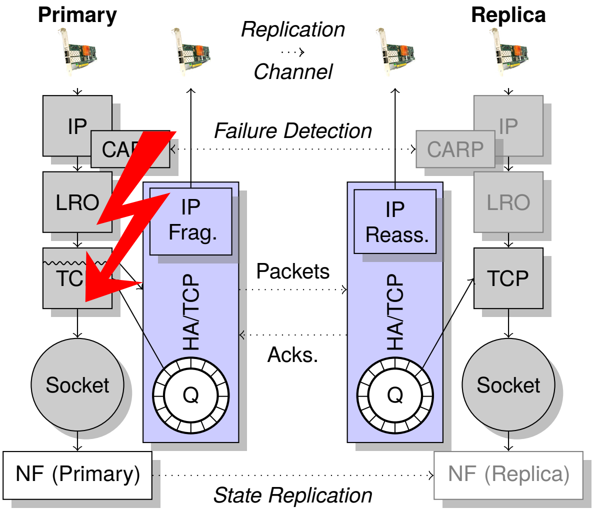
- Questions?

Appendix 1: IP-Based Replication Channel

- IP provides nice helper functions
- IP is still routable
- Eliminates the TCP stack overhead

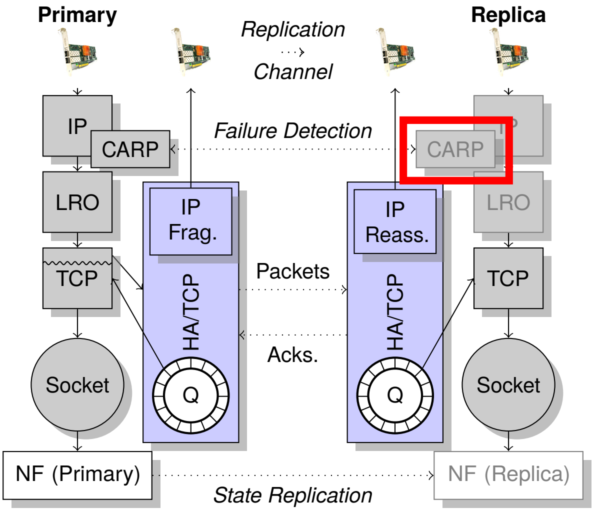
Appendix 2: Steady State Processing

- Primary receives the client packet
  - duplicates the packet
  - sends to replica
- Replica acknowledges the reception
- Both sides process packet
- When failure happens…



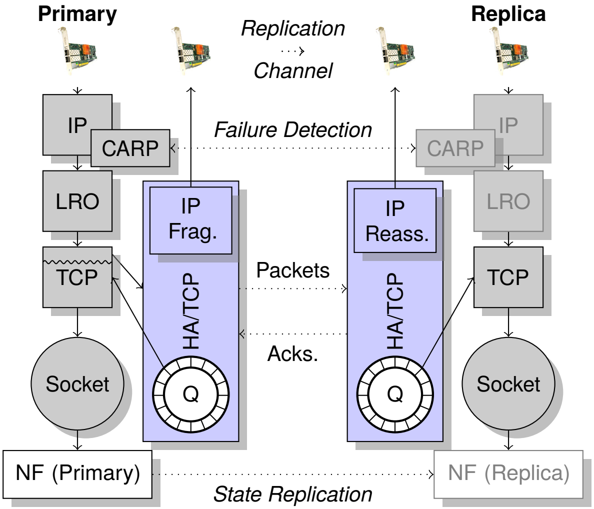
Appendix 2: Failure Handling

- CARP timeout



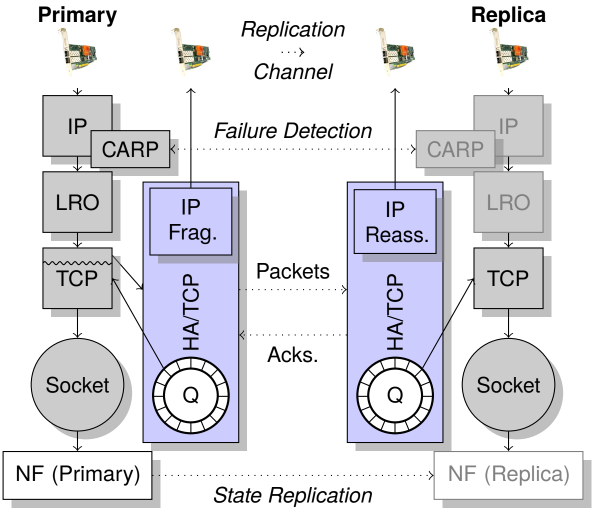
Appendix 2: Failure Handling

- CARP timeout
  - Primary: promote replica
  - Replica: remove



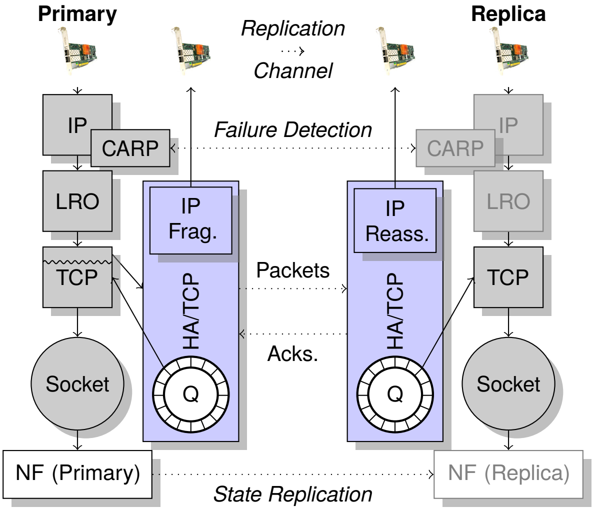
Appendix 2: Failure Handling

- CARP timeout
  - Primary: promote replica
  - Replica: remove
- Update ARP



Appendix 2: Failure Handling

- CARP timeout
  - Primary: promote replica
  - Replica: remove
- Update ARP
- Continue service



Appendix 3: Application Benchmarks