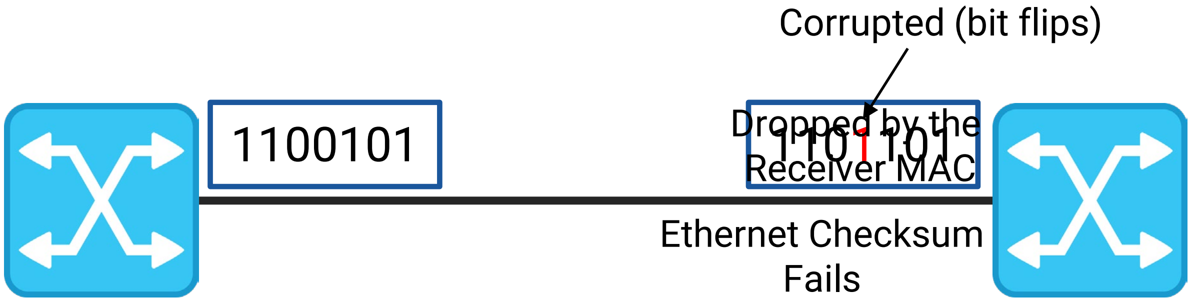
LinkGuardian
Mitigating the impact of packet corruption loss with link-local retransmission

Raj Joshi, Qi Guo, Nishant Budhdev, Ayush Mishra, Mun Choon Chan, Ben Leong

NUS National University of Singapore

Image Credit: Google Datacenters

What is corruption packet loss?

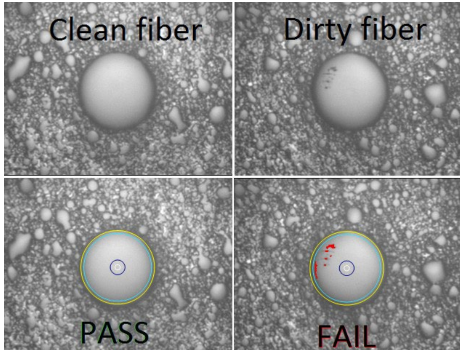


What causes packet corruption?

Switch-to-switch Links - Optical

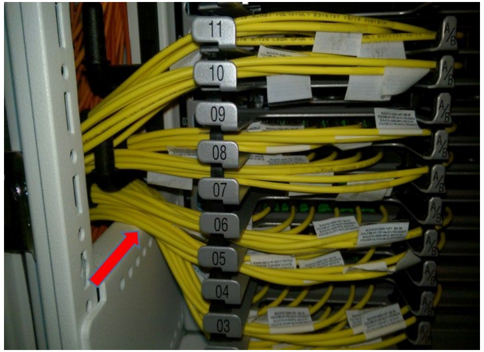
Support high link speeds (up to 400 Gbps)
over long distances (~100m)

Optical links – susceptible to corruption



Fiber Contamination

Airborne dust particles



Fiber Bending

Decaying Transmitters

Photo Credits: Zhou et al., SIGCOMM 2017

Why do we care?

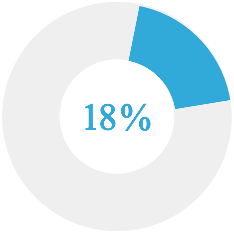
Corruption Packet Loss – Significant

Comparable to Congestion Loss

Large-scale study

(350K links, 15 datacenters)

[Zhuo et al., SIGCOMM'17]



Packet drops affected customers

due to corruption

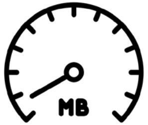
[Zhou et al., SIGCOMM'20]

Why do we care?

Affects Application Performance
(like any packet loss)

Increase in FCTs

For latency-sensitive flows



Drop in throughput

For throughput-sensitive flows

How can we fix packet corruption?

Physical Repair

Several hours to days

Until then →

Mitigate

Effects of packet corruption

Existing Solutions to Mitigate Packet Corruption

	Localize Impact No network-wide disruption	**Capacity↓ Proportional** to corruption loss rate	**End-host Agnostic** No end-host changes
Avoid corrupting links [e.g. NetPilot, CorrOpt]	✗		
End-to-end redundancy [e.g. RAIL, CloudBurst]	✓	✗	✗
Link-local redundancy [e.g. Wharf]	✓	✗	✓

Disabling the links

Network-wide disruption: re-routing of 1000's of flows

Incur higher reduction in capacity even for small corruption loss rate (e.g. 10^{-3})

LinkGuardian

Key Idea

Link-local Retransmission

Within the network to recover corruption pkt loss

Wireless Networks

Not explored in the context of datacenter networks

Link-local ReTx in DC networks is non-trivial

A complete link-local ReTx scheme

- Detect the packets lost
- Hold on the transmission until lost packet is retransmitted
- Put packets back in order to continue transmission

Challenging:

- High link speeds
- Dataplane h/w contraints

In this paper – first step

Investigating whether a simple out-of-order retransmission scheme could work in datacenter networks

In this talk

Small measurement study

Potential "**recovery delay**":
out-of-order reTx scheme

Insights

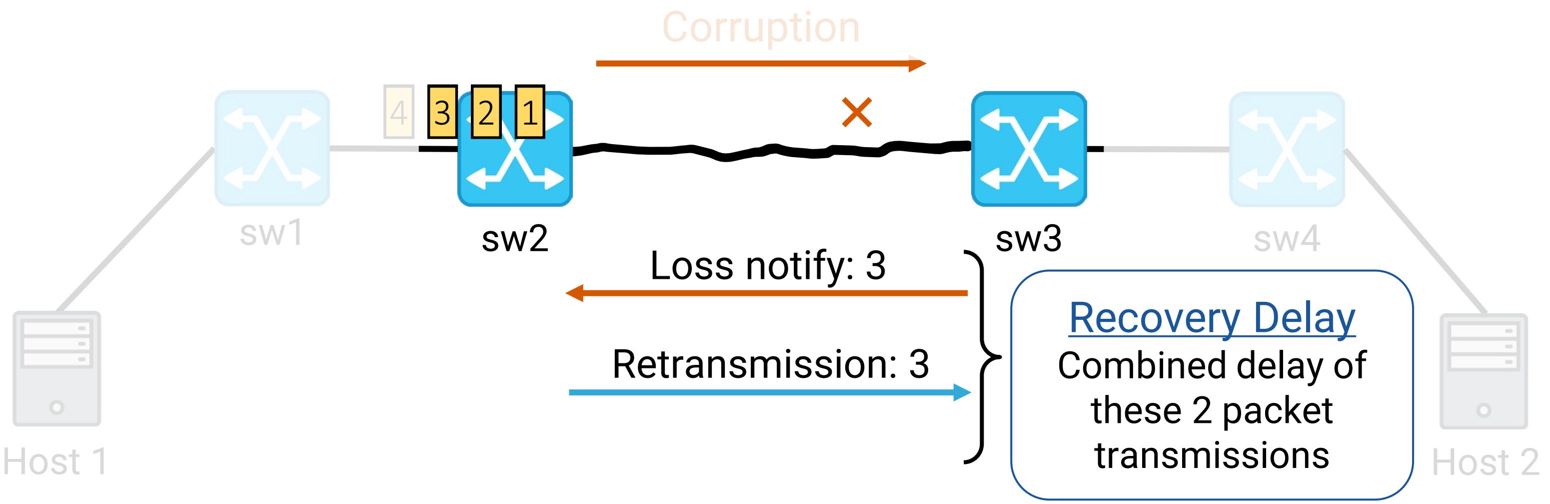
LinkGuardian

Design & Implementation

Evaluation Results

Recovery Delay for out-of-order ReTx

Simple out-of-order link-local ReTx scheme



Measurement Study

– 10 Gbps network with h/w timestamping on switches

* more details in the paper

Measurement Study – potential recovery delay

Measured Recovery Delays in µs

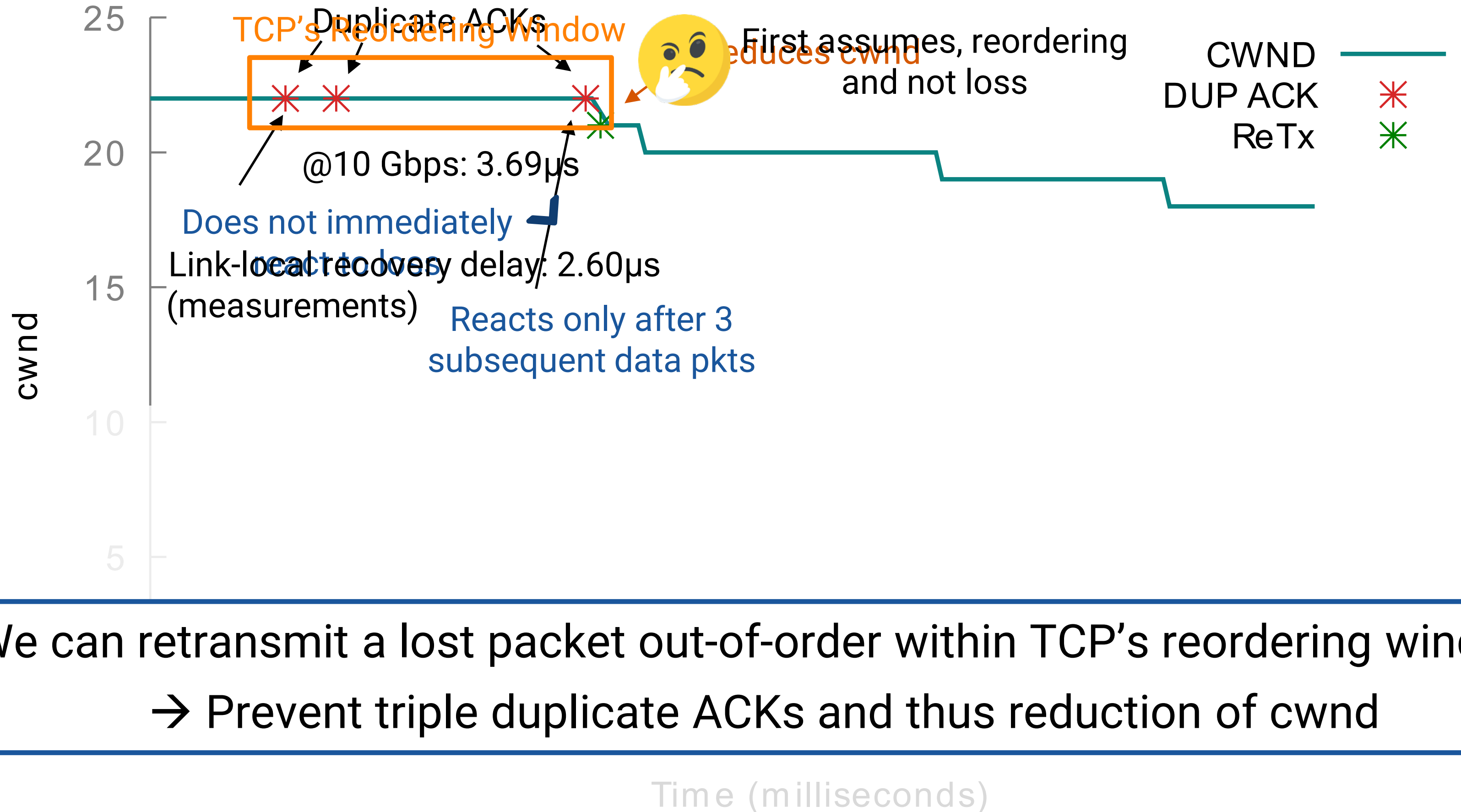
	Link-local ReTx	End-to-end (kernel)
Mean	2.59	32.73
50%	2.59	32.50
99%	2.60	44.00

10x lower

Implication for latency-sensitive flows

– Smaller recovery delay → significantly reduce the increase in FCT due to packet loss

Implication for Throughput-sensitive TCP flows



We can retransmit a lost packet out-of-order within TCP's reordering window:

→ Prevent triple duplicate ACKs and thus reduction of cwnd

Key Insight

Fast out-of-order Recovery

in the network dataplane

Loss → reorder

for end-host TCP

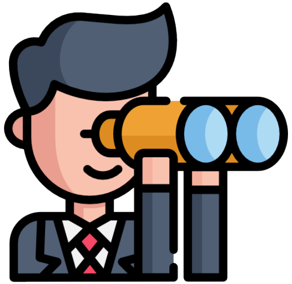
Prevent performance degradation

By avoiding end-host based loss recovery & reduction of cwnd

LinkGuardian

Network w/ LinkGuardian

Runs normally when no link is corrupting



Monitor the links for packet corruption

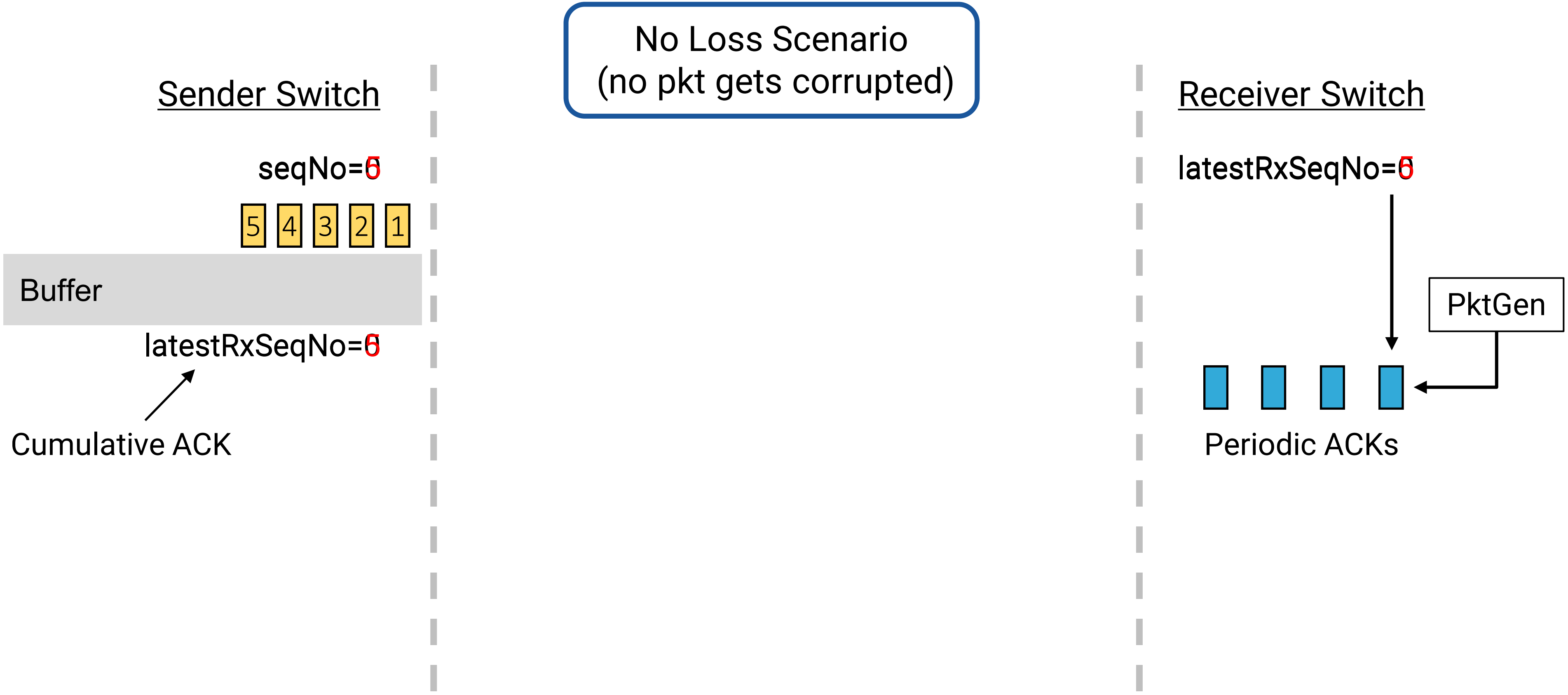
Using existing techniques

Link corrupting

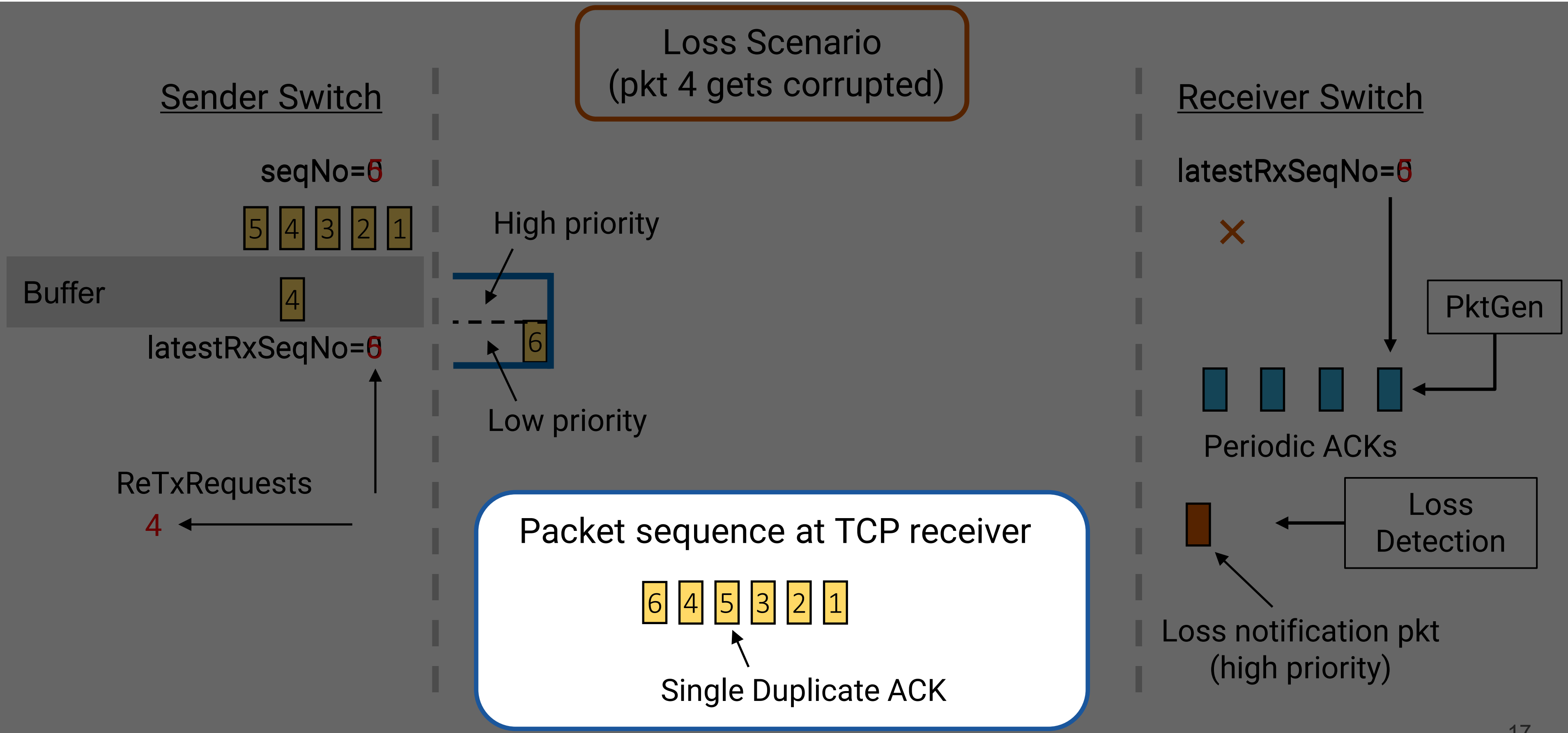
Activate LinkGuardian

LinkGuardian protocol runs entirely in the dataplane

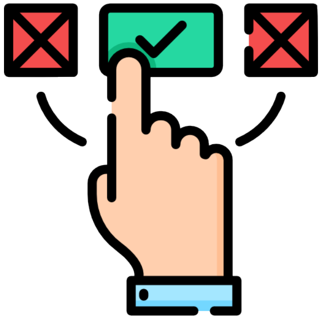
LinkGuardian Protocol



LinkGuardian Protocol

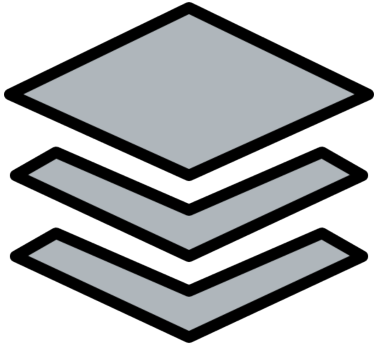


LinkGuardian Protocol



Loss detection at the receiver switch

Comparing seq # of current packet and the latestRxSeqNo



Packet buffering at the sender switch

Currently achieved through recirculation

Refer to the paper for more details

Evaluation

Implementation

- Intel Tofino ~900 lines of P4

Evaluation setup

- 10 Gbps 3-stage Clos network: Intel Tofino switches and Xeon servers
- Random generator on Tofino: emulate corruption packet loss (different rates)
- TCP flows: CUBIC and DCTCP

Compare

- No mitigation
- Mitigation using Wharf [NetCompute ’18]
  - Provides link-local redundancy

Mitigate impact on throughput-sensitive flows

TCP Throughput in Gbps

Loss rate →	**0 (baseline)**	**10^{-5}**	**10^{-4}**	**10^{-3}**	**10^{-2}**
CUBIC	9.49	9.48	7.28	3.43	1.33
+ Wharf	n/a	9.13	9.13	9.13	7.91
+ LinkGuardian	9.47	9.47	9.47	9.46	9.28

* similar results for DCTCP (see paper)

Wharf: FEC redundancy overhead for ***all*** packets
LinkGuardian: retransmits only the lost packets

Mitigate impact on latency-sensitive flows

FCT distribution (in μs) for 45KB “affected” DCTCP flows

Loss rate →	0	10^{-3}	
	Baseline	Loss	Loss + LinkGuardian
min	113	193	143
mean	197	707	375
50%	180	419	258
90%	311	2421	424
95%	↑ 10x 315	3216	455 ↓ ~85%
99%	329	4192	3540

* more results in the paper

Overheads

Packet buffering at the sender switch

- Packet buffer: 5.44 KB (3-4 packets) @10 Gbps

Periodic ACK packets

- Link capacity overhead: 1%

Overall, the overheads for LinkGuardian remain low.

Future directions

Scalability

- Currently, works for a 100 Gbps link, as long as individual TCP flows <= 10 Gbps
- Investigate: can support individual TCP flows > 10 Gbps

Preserve packet ordering (ordered in-network retransmission)

- Beneficial for end-point transports that may not be reordering tolerant (e.g. RDMA)

Buffering packet copies without recirculation

- Next gen programmable switches (Intel Tofino2) provide primitives with which this seems plausible

Conclusion

Packet Corruption Loss

- Can be significant in datacenter networks. Affects application network performance

LinkGuardian

Fast out-of-order Recovery
in the network dataplane

→

Loss → reorder
for end-host TCP

→

Prevent performance degradation
By avoiding end-host based recovery & reduction of cwnd

Preliminary results → promising approach for mitigating pkt corruption

Thank you!