Dynamic Router's Buffer Sizing using Passive Measurements and P4 Programmable Switches

Elie Kfoury[1], Jorge Crichigno[1], Elias Bou-Harb[2], Gautam Srivastava[3]

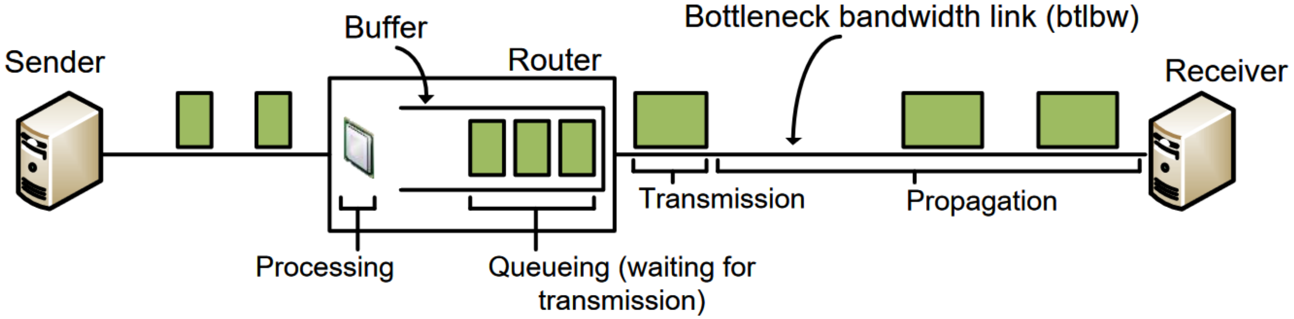
[1] University of South Carolina, SC
[2] The University of Texas at San Antonio, TX
[3] Brandon University, CA

Agenda

- Introduction
- Buffer sizing rules
- Stanford rule
- Programmable switches
- Proposed system
- Results and evaluation
- Conclusion

Introduction

- Routers and switches are designed to include packet buffer
- The size of buffers imposes significant implications on the performance of the network
- Large buffers -> excessive delays, Bufferbloat
- Small buffers -> packet drops, potential low link utilization



Buffer sizing rules

- General rule-of-thumb: Bandwidth-delay product
  - Buffer = $C * RTT$
  - C is the capacity of the port and RTT is the average round-trip time (RTT)
- Stanford rule:
  - Buffer = $\frac{C * RTT}{\sqrt{N}}$
  - N is the number of long (persistent over time) flows traversing the port
  - Statistical multiplexing
- Tiny buffer:
  - Few dozen KB
  - 10-20% drop in link utilization

Stanford rule applicability

- Setting the router's buffer size to $BDP/\sqrt{N}$ would require determining the current average RTT and the number of flows
- This could be achieved by passively capturing traffic crossing the router and forwarding it to a general-purpose CPU
- Cannot cope with high traffic rates, especially in high-speed networks
- Sampling techniques (e.g., NetFlow) cannot be applied either since they are not accurate enough and often lose measuring information[1]

---

[1] Spang, Bruce, and Nick McKeown. "On estimating the number of flows." *Stanford Workshop on Buffer Sizing*. 2019.

Overview P4 switches

- P4 switches permit programmer to program the data plane
- Customized packet processing
- High granularity in measurements
- Per-packet traffic analysis and inspection
- If the P4 program compiles, it runs on the chip at line rate

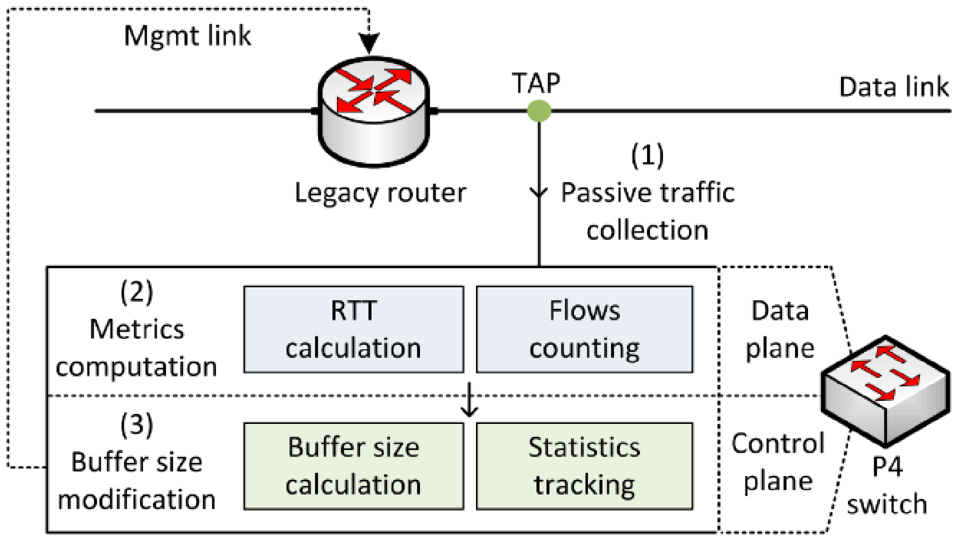
```
/*************************************************************************
*********************** P A R S E R  ***********************************
*************************************************************************/

    state parse_ethernet {
        packet.extract(hdr.ethernet);
        transition select(hdr.ethernet.etherType) {
            TYPE_IPV4: parse_ipv4;
            default: accept;
        }
    }

    state parse_ipv4 {
        packet.extract(hdr.ipv4);
        verify(hdr.ipv4.ihl >= 5, error.IPHeaderTooShort);
        transition select(hdr.ipv4.ihl) {
            5           : accept;
            default     : parse_ipv4_option;
        }
    }
```

P4 code

Programmable chip

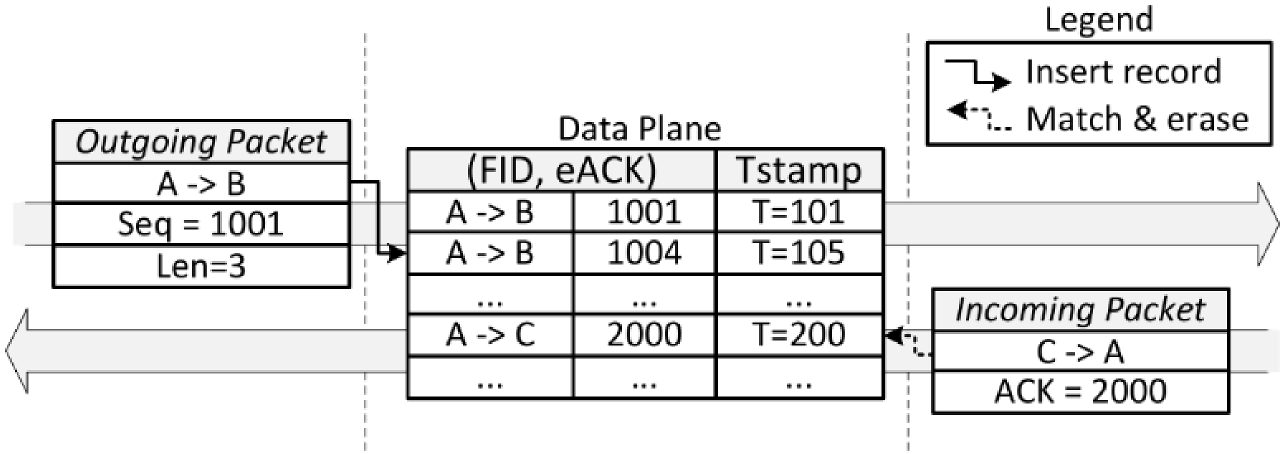
Proposed system

- Dynamically modify the buffer size of routers based on measurements collected on programmable switches
  1. Copy of the traffic is forwarded to a programmable switch by passively tapping on routers' ports
  2. The programmable switch identifies, tracks, and computes the RTT of long flows
  3. The programmable switch modifies the legacy router's buffer size



RTT calculation

- Relate the TCP sequence (SEQ) and acknowledgement (ACK) numbers of incoming and outgoing packets[1]
- The RTT can then be inferred by calculating the time difference between the two packets
- In reality, devices might not acknowledge every packet



[1]Chen, Xiaoqi, et al. "Measuring TCP round-trip time in the data plane." Workshop on Secure Programmable Network Infrastructure. 2020.

Long flows counting

- The number of flows in the buffer size formula (BDP / √N) -> long flows sharing the bottleneck link[1]
- Short flows on the other hand are not considered since they have very small effect on the buffer[1]
- Need to differentiate between the two

Algorithm 1: Long flows counting algorithm

input : Packet headers hdr; flow identifier FID; switch timestamp S_{tstamp}; flows timestamps $tstamps$, indexed by FID; packet counts $counts$, indexed by FID; current flow count C, time threshold T_THRESH, packet count threshold C_THRESH

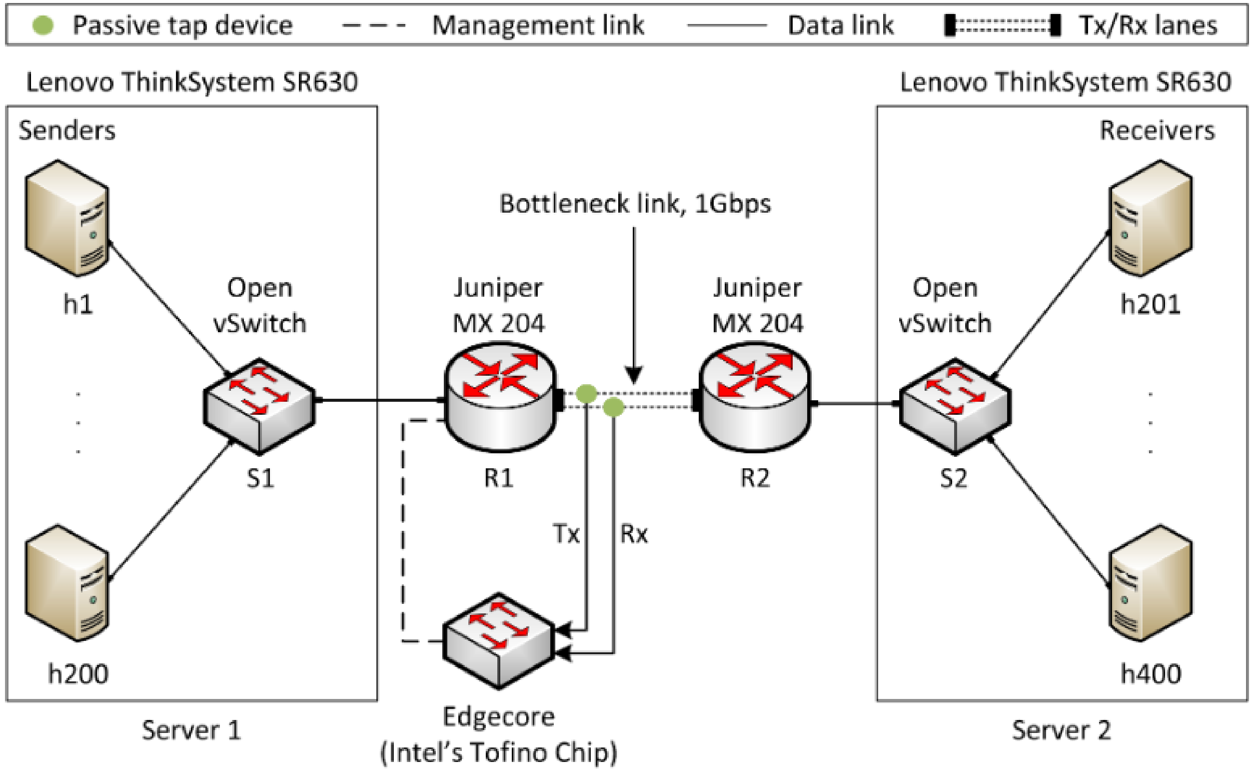
output: Updated long flow count

begin
- $prev\_tstamp \leftarrow tstamps[FID]$
- $tstamps[FID] \leftarrow S_{tstamp}$
- **if** $tstamps[FID] - prev\_tstamp < T\_THRESH$ **then**
  - **if** $counts[FID] = C\_THRESH$ **then**
    - $C \leftarrow C + 1$
  - **else**
    - $counts[FID] \leftarrow counts[FID] + 1$
- **else**
  - $counts[FID] \leftarrow 0$
- **if** $hdr.tcp.flags = FIN$ **then**
  - **if** $counts[FID] = C\_THRESH$ **then**
    - $C \leftarrow C - 1$

[1]Appenzeller, Guido, Isaac Keslassy, and Nick McKeown. "Sizing router buffers." *ACM SIGCOMM Computer Communication Review* 34.4 (2004)

Implementation and evaluation

- Topology and experimental setup
- Different congestion control algorithms[1]
- iPerf3
- Default buffer size of the router is 200ms[2]
- Edgecore Wedge100BF-32X, ASIC chip (Intel's Tofino)



[1]Mishra et al. "The great Internet TCP congestion control census," ACM on Measurement and Analysis of Computing Systems, 2019

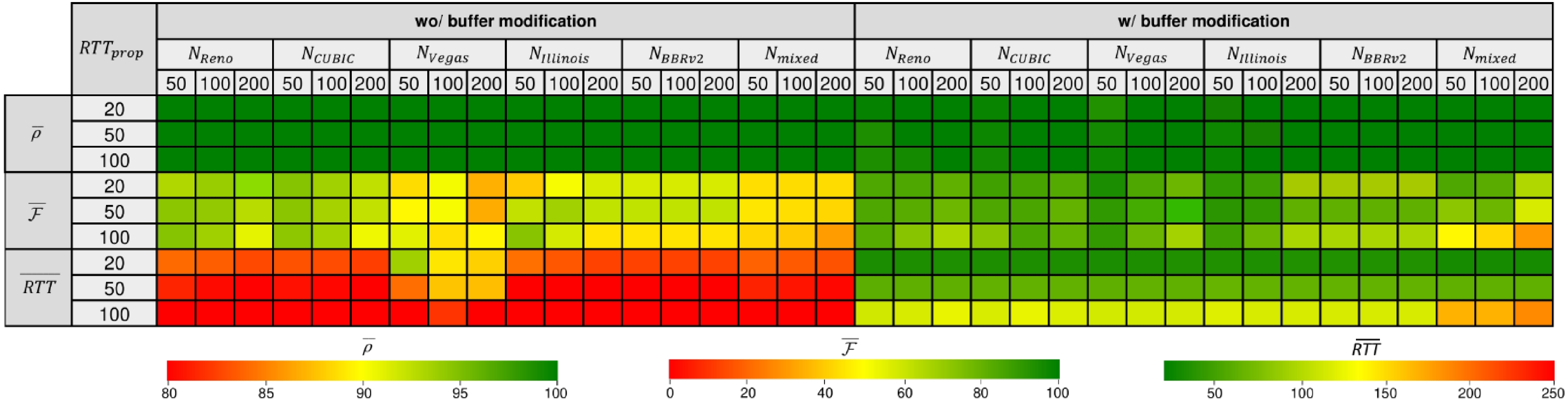
[2]N. McKeown et al. "Sizing router buffers (redux)," ACM SIGCOMM Computer Communication Review, vol. 49, no. 5

Implementation and evaluation

- Two scenarios are considered:
  1. Default buffer size on the router, without any dynamic modification
  2. P4 switch measures and modifies the buffer size of the router

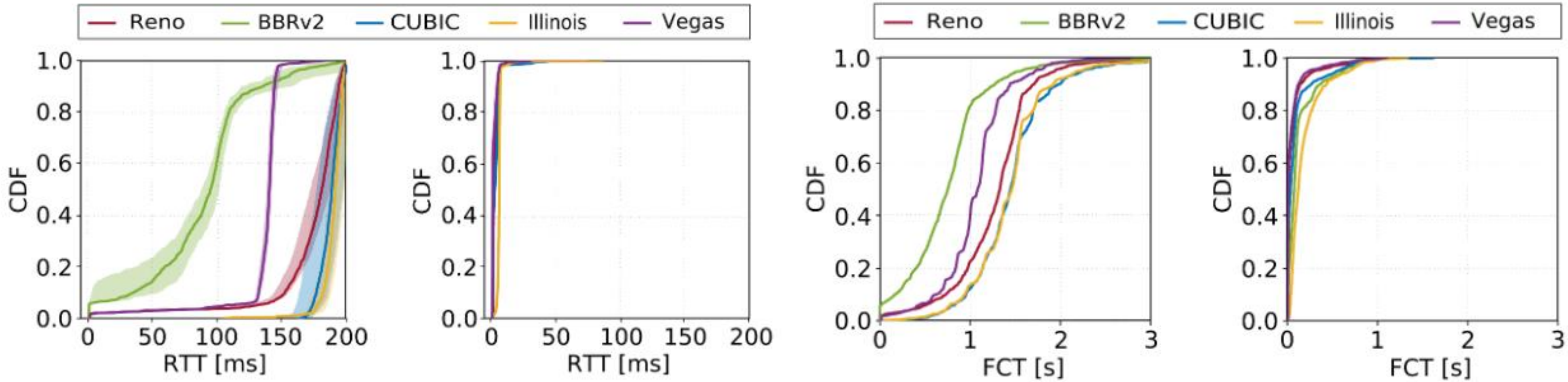
Results

- Multiple long flows, CCAs, and propagation delays
- Average link utilization $(\overline{\rho})$
- Average fairness index $(\overline{\mathcal{F}})$
- Average RTT $(\overline{RTT})$



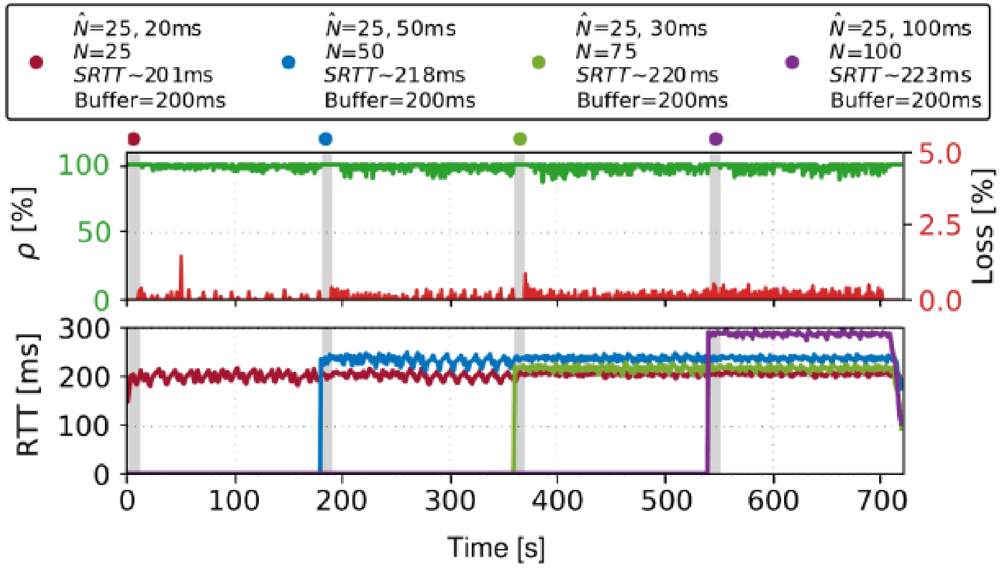
Results

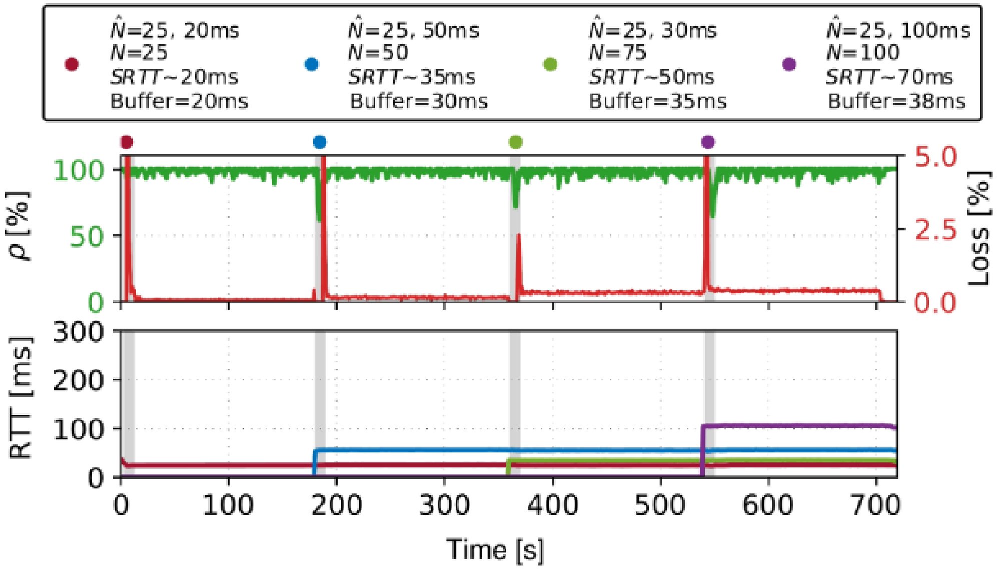
- Performance of short flows sharing the bottleneck with long flows
- 1000 short flows are arriving according to a Poisson process
- Flow size distribution resembles a web search workload (10KB to 1MB)
- Background traffic: 200 long flows, propagation delay = 50ms



Results

- Long flows with different emulated propagation delays
- 100 long flows, divided into four groups of each 25 flows each
- Each group starts three minutes after the other
- Cubic congestion control algorithm





Conclusion

- This paper presented a scheme that dynamically modifies the size of the router’s buffer
- The scheme uses passive measurements collected by programmable P4 switches
- Experiments conducted on real hardware demonstrate the improvements in the RTT, packet loss rate, fairness, and FCT of flows

Acknowledgement

- Thanks to the National Science Foundation (NSF)!
- Activities in the CI Lab at the UofSC are supported by NSF, Office of Advanced Cyberinfrastructure (OAC), awards 1925484 and 2118311

Thank You

- Contact info for further questions
  - ➢ ekfoury@email.sc.edu
  - ➢ jcrichigno@cec.sc.edu

- CyberInfrastructure Lab (CI Lab) website
  - http://ce.sc.edu/cyberinfra/