MAY 18-20 2021 P4 Workshop

Hosted by ONF

Realizing One Big Switch Performance Abstraction using P4

Jeongkeun "JK" Lee
Principal Engineer, Intel Barefoot

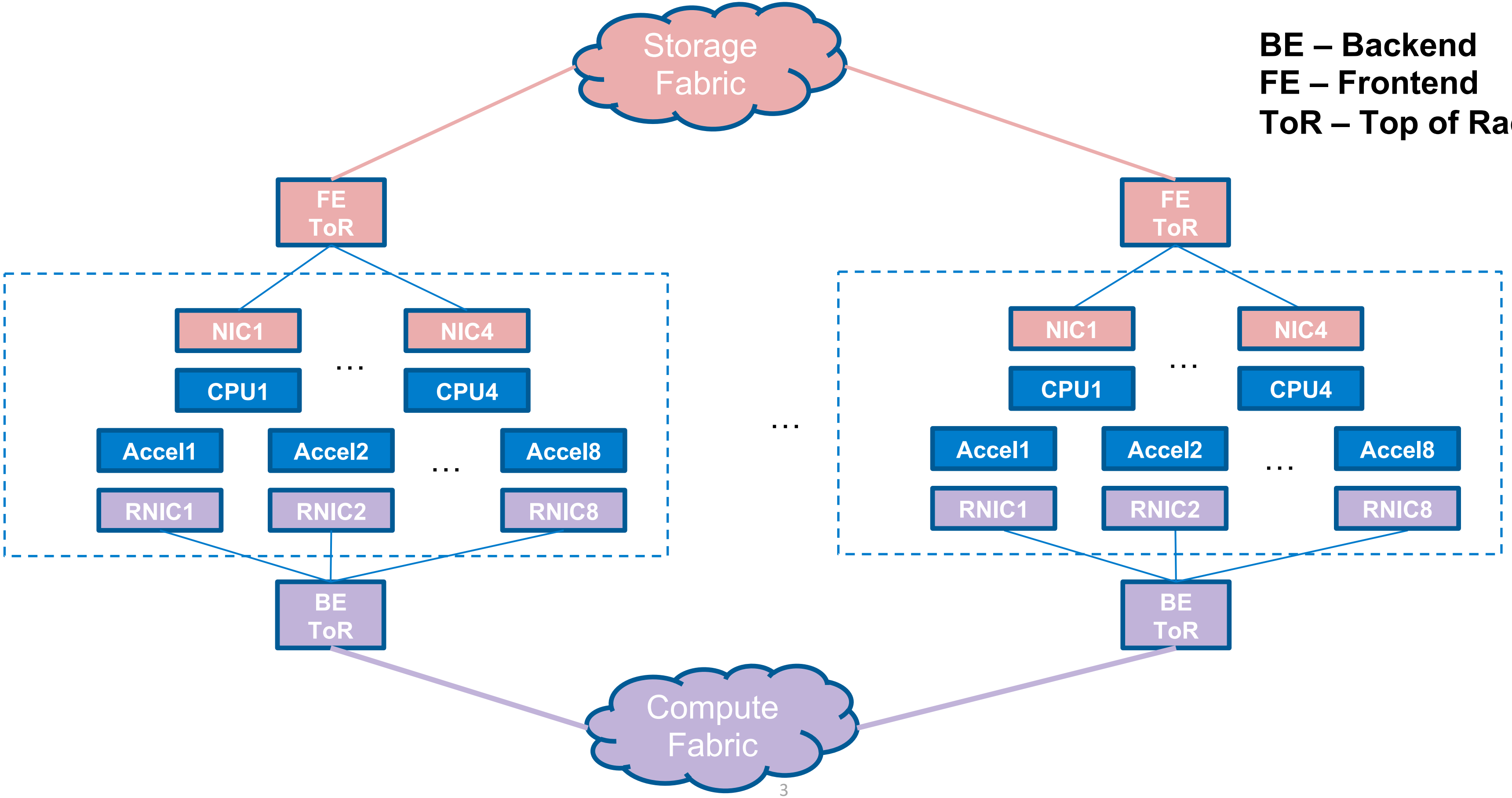
Petr Lapukhov
Network Engineer, Facebook

Agenda

- Need for high-performance cluster for ML training
- Vision: One Big Switch with Distributed VoQ
- Problem: network congestion
- Switch-side building blocks and P4 constructs

- Contributions from
  - Anurag Agrawal, Jeremias Blendin, Andy Fingerhut, Grzegorz Jereczek, Yanfang Le, Georgios Nikolaidis, Rong Pan, Mickey Spiegel

A Training Cluster Primer



BE – Backend
FE – Frontend
ToR – Top of Rack

Distributed ML training

- Combines data & model parallelism
- Bulk-synchronous parallel realization (BSP)
- Collective-style communications
- All trainers arrive to a barrier in each cycle

Ref: https://arxiv.org/pdf/2104.05158.pdf

The problem statement

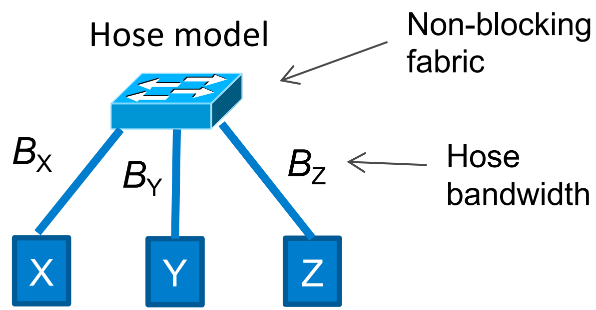
- Synchronous training cycle = [Compute + Network]
- Network Collectives: All-to-all, All-Reduce,…
- Collective produces a combination of flows
- Objective: min(collective completion time)
- Large messages – O(1MB)

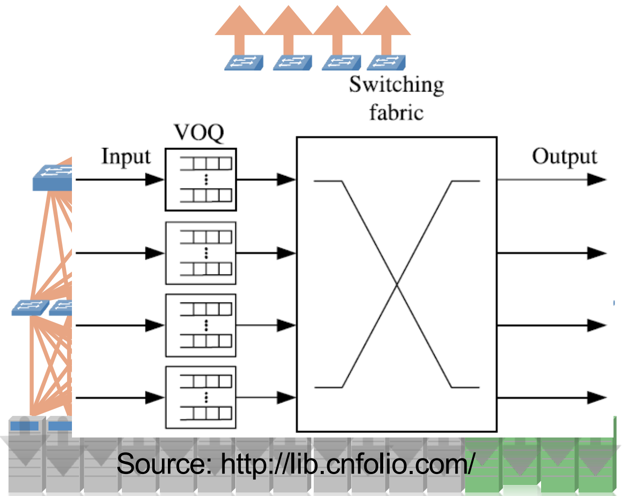
What’s the challenge?

- Accelerators require HW offloaded transport
- All-to-all is bisection BW hungry
- All-reduce may cause incast; flow entropy varies
- Large Clos fabric requires tuning with RoCE
- Small scale is fine – single large crossbar switch

Vision: One Big Switch with Distributed VoQ

- "Network as OBS" model has been used in
  - SDN for network-wide control plane and policy program
  - OVS/OVN network virtualization
  - Service Mesh for uServices
- OBS performance model = hose model
  - Non-blocking fabric
  - Congestion only at network egress (output queue)
- *Distributed* VoQ (Virtual Output Queue)
  - VoQ reflects egress congestion & BW towards receivers
  - Move queueing from egress to ingress
  - OBS ingress = 1st-hop switches or senders

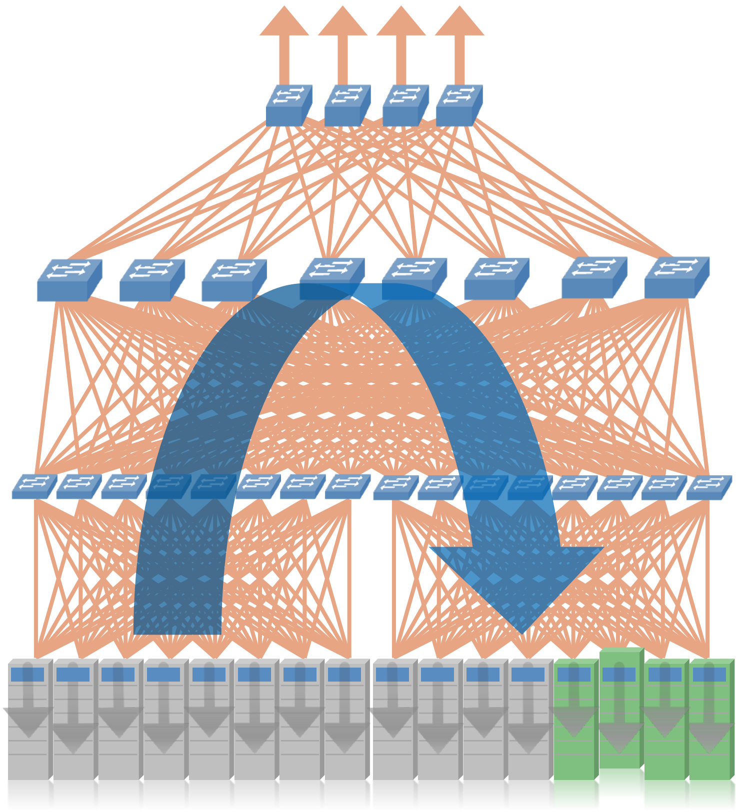




Source: http://lib.cnfolio.com/

Reality: Congestions in DC CLOS

- Uplink/core ← *challenge for non-blocking fabric*
  - Cause: ECMP/LAG hash collision
  - Worse at oversubscribed networks
- Incast ← *challenge for VoQ*
  - Cause: many-to-one traffic pattern
  - Congestion surge at the last-hop
  - Slows down e2e signal loop
- Receiver NIC ← *challenge for VoQ*
  - Cause: slow software/CPU, PCIe bottleneck



Switch-side building blocks

1. Provide rich congestion, BW metrics
   ➔ for applications to acquire available BW asap while controlling congestion
2. Fine-grained load-balancing w/ minimal or no out-of-order delivery
   ➔ For non-blocking, full-bisection fabric
3. Cut-payload signaling to receivers
   ➔ For NDP-style receiver pulling
4. Sub-RTT signaling back to senders
   ➔ Sudden change of congestion/BW state
5. React to rx NIC congestion
   ➔ Leverage switch programmability where smart NIC is not available or applicable

1. Provide Congestion and BW metrics

- Goal: switch provides rich info, for sender/receiver to consume and control

- For **whom**: incoming data pkts, control pkts (RTS/CTS solicitation)
- **What** to provide: queue depth, drain time, TX rate or avail BW, arrival rate (incast rate), # of flows, congestion locator (node/port/Q IDs)
- **How**: in-band on forwarding pkts, separate signal pkt back to sender, or signal pkt receiver
- **Where**: last-hop or core switch, ingress or egress
- **When**: threshold crossing, bloom-filter suppression, when switch metric is off from in-pkt metric, when NIC sends PFC, upon pkt drops, …

P4 Primitives (w/ PSA/TNA/T2NA)

- For **whom**: incoming data pkts, control pkts (RTS/CTS solicitation)

  ➔ P4 parser

- **What** to provide: queue depth, drain time, per-port or -q TX rate, arrival rate (incast rate), # of flows, congestion locator (node/port/Q IDs)

  ➔ standard/intrinsic metadata, register with HW reset, meter or LPF extern

- **How**: in-band on forwarding pkts, separate signal pkt back to sender, or signal pkt receiver (at high-priority like NDP cut-payload)

  ➔ modify, mirror, recirculate, multicast actions

- **Where**: last-hop or core switch, ingress or egress, depending on use case

  ➔ T2NA: ingress visibility of egress queue status (P4 Expert Round Table 2020)

- **When**: threshold crossing, bloom-filter suppression, when switch metric is off from in-pkt metric, when NIC sends PFC, upon pkt drops, …

  ➔ register, meter, P4 processing of RX PFC frame, TNA: Deflect on Drop

2. Fine-grained Load Balancing

- For non-blocking fabric
- Sender NIC may spray packets by changing L4 src port number (e.g., AWS SRD)
  - Pros: ECMP switch doesn't need to change; better handle brownfield w/ path tracking at NIC
  - Cons: transport & congestion control change; 5tuple connection ID may alter
- Switch alternative 1: flowlet switching (e.g., Conga)
  - Pros: no change to transport, no OOO delivery
  - Cons: flow scale is limited
- Switch alternative 2: DRR (Deficit Round Robin) packet spraying
  - Pros: DRR minimizes load imbalance, hence OOO delivery window; DRR possible in P4
  - Cons: need greenfield (at least ToR layer); need packet re-ordering at RX NIC/host

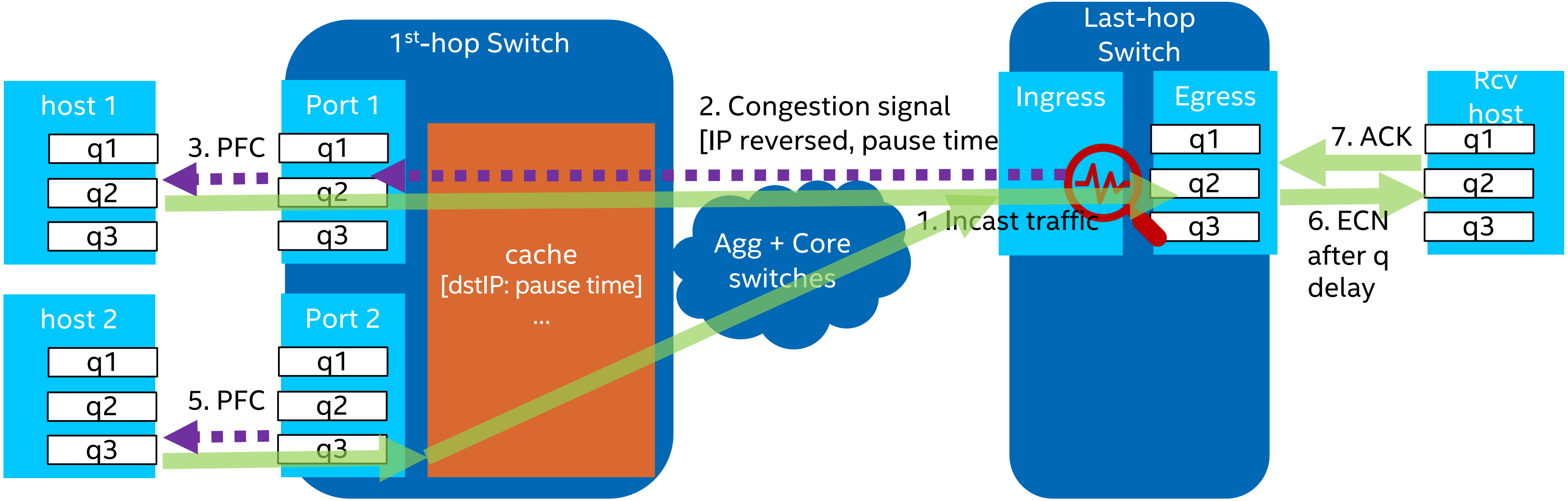
3. Cut-Payload to Receiver

- Signal last-hop switch congestion & drop events to receiver at high-priority
- NDP P4 design by Correct Networks and Intel, using
  - ingress meter + mirror
  - deflect-on-drop + multicast
  - on Tofino1
- P4 source will be posted soon at *https://github.com/p4lang/p4-applications*

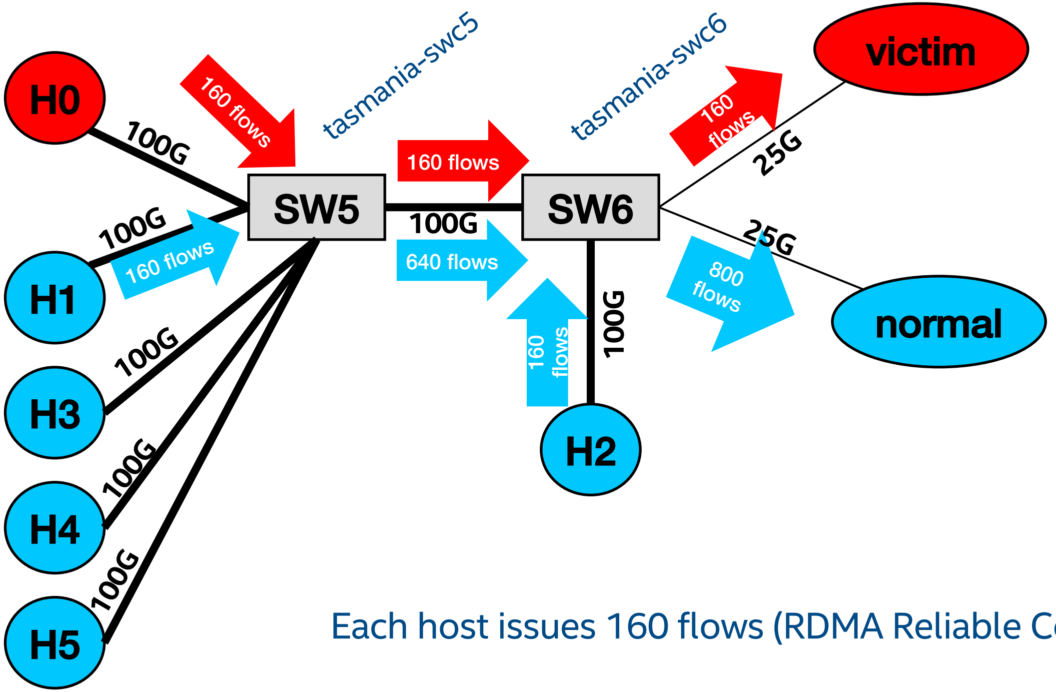
Check out “NDP with SONiC-PINS: A low latency and high performance data-center transport architecture integrated into SONiC.“ by Rong P. and Reshma S.

4. Sub-RTT, L3 Signaling back to sender

- Edge-to-Edge signaling of congestion info
- 5. able to carry rcv NIC congestion signal (e.g., NIC-to-switch PFC) to sender hosts
- Senders can use the signal (VoQ) in various ways, e.g., instant flow control to ‘flatten the curve’
  - Example below: **SFC (Source Flow Control)** = new L3 switch-to-switch signaling + PFC as existing flow control at sender NIC. (No PFC btw switches)
  - Future work: fine-grained per-receiver VoQ at sender

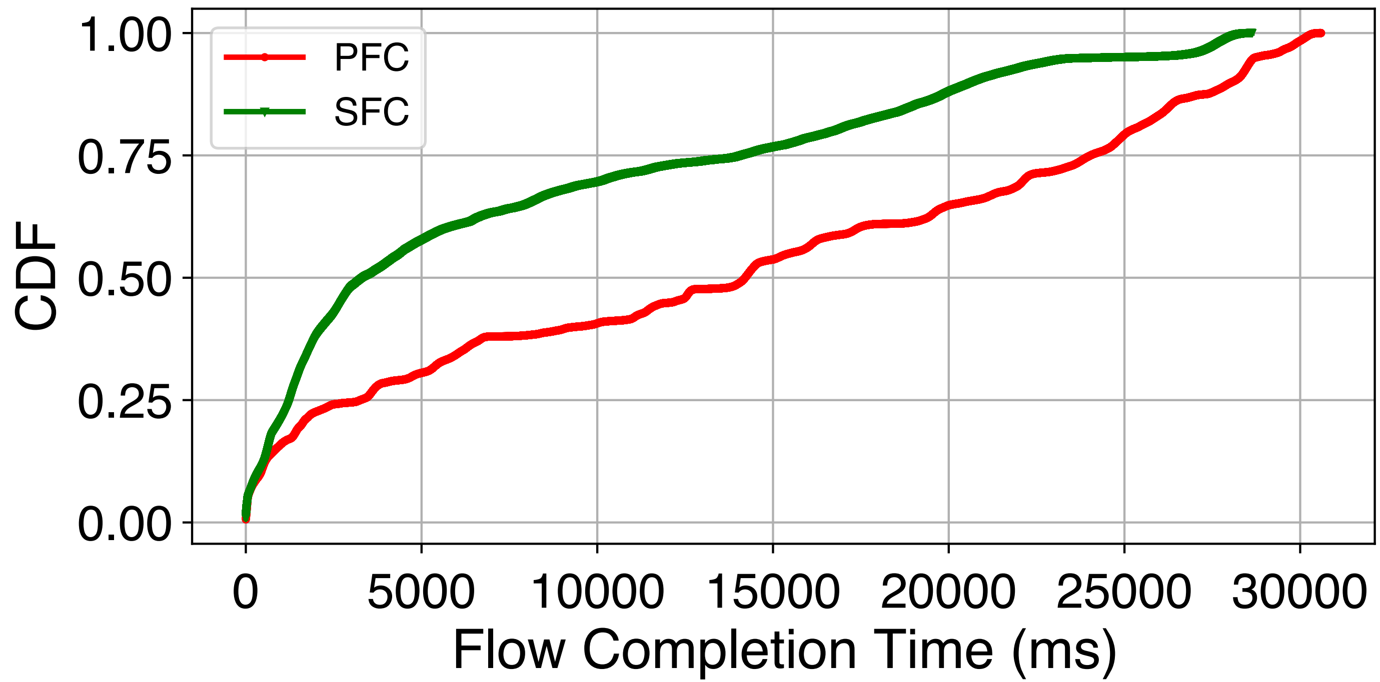


SFC: System Demo

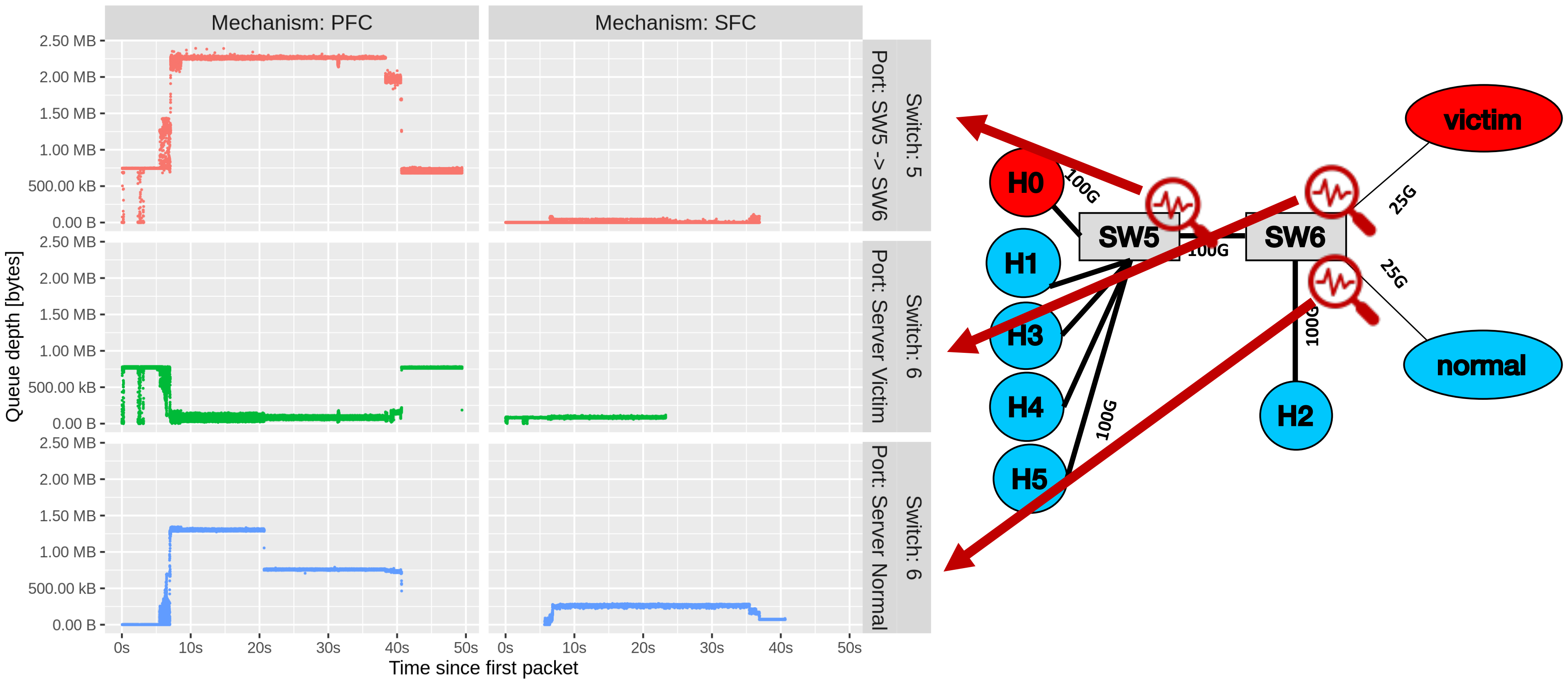


Each host issues 160 flows (RDMA Reliable Connection).

Flow Completion Time



Queue Depth



Mechanism: PFC
Mechanism: SFC
Queue depth [bytes]
2.50 MB
2.00 MB
1.50 MB
1.00 MB
500.00 kB
0.00 B
0s
10s
20s
30s
40s
50s
Time since first packet
Port: SW5 -> SW6
Switch: 5
Port: Server Victim
Switch: 6
Port: Server Normal
Switch: 6
victim
H0
100G
SW5
SW6
100G
25G
25G
H1
H3
H4
H5
100G
100G
normal
H2

Summary

- True One-Big-Switch with performance is possible
- P4 primitives can build
  - non-blocking CLOS fabric
  - sub-RTT signaling of congestion & BW metrics

- Research questions
  - What is the scope of OBS; granularity of distributed VoQ
    - From last-hop switch queue to rxNIC port to service node (ML worker or µService)
    - Should consider major incast bottleneck point and VoQ scale
  - NIC-side building blocks to enable distributed VoQ at senders
  - How QoS infra and congestion control benefit from VoQ

MAY 18-20
2021
P4
Workshop
Hosted by ONF

Thank You

jk.lee@intel.com
petr@fb.com

Simulation setup

Custer: 3-tier, 320 servers, full bisection, 12us base RTT

Switch buffer: 16MB, Dynamic Threshold

Congestion control: DCQCN+window, HPCC

SFC Parameters

- SFC trigger threshold = ECN threshold = 100KB, SFC drain target = 10KB

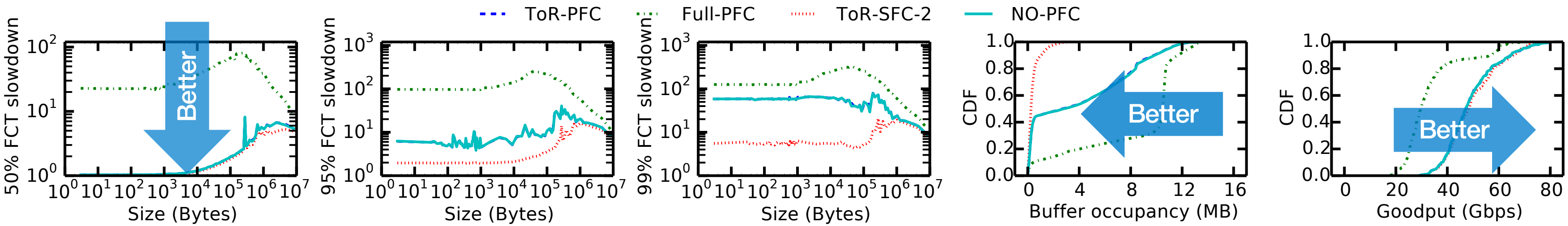
Workload: RDMA writes

- 50% Background load: shuffle, msg size follows public traces from RPC, Hadoop, DCTCP
- 8% incast bursts: 120-to-1, msg size 250KB, synchronized starts within 145us

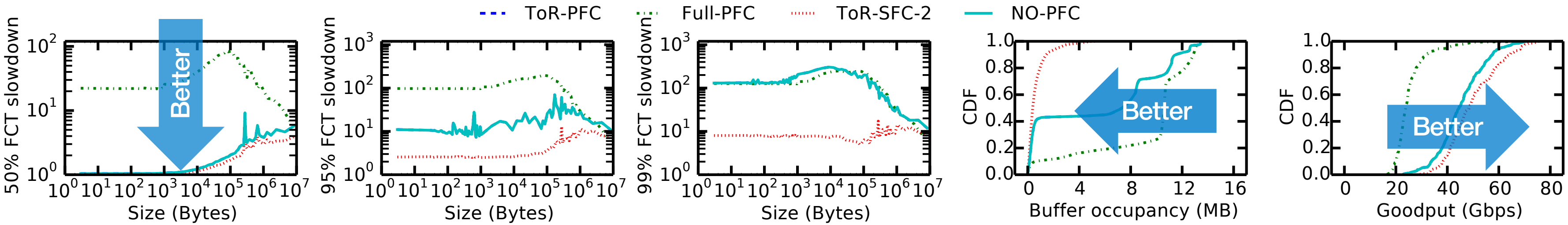
Metrics

- FCT slowdown: FCT normalize to the FCT of same-size flow at line rate
- Goodput, switch buffer occupancy

Simulation with RPC-inspired msg size dist.



DCQCN+Window



HPCC