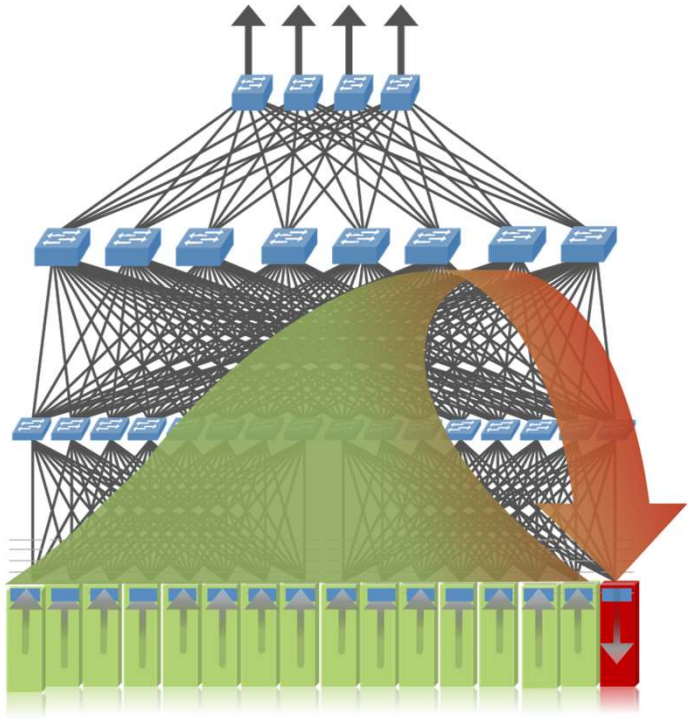
Remote Priority Flow Control
towards Source Flow Control (SFC)

P4 Workshop Taiwan, Dec 21, 2021

Presenter: Jeremias Blendin (jeremias.blendin@intel.com)
Intel team: Jeongkeun Lee, Jeremias Blendin, Yanfang Le, Grzegorz Jereczek, Ashutosh Agrawal, Rong Pan

Incast Congestion in Data Centers

- Cause: many-to-one traffic pattern
  - Congestion mostly at the last hop switches
  - Governs max/tail latency
  - Perf/scale impact on application workloads
- Incast in RDMA workloads
  - State of the art RDMA: RoCEv2 with DCQCN
  - Senders start sending at line rate
  - Incast requires fast sub-RTT reaction time
  - RTT = congestion-free minimum, nic-to-nic round-trip-time

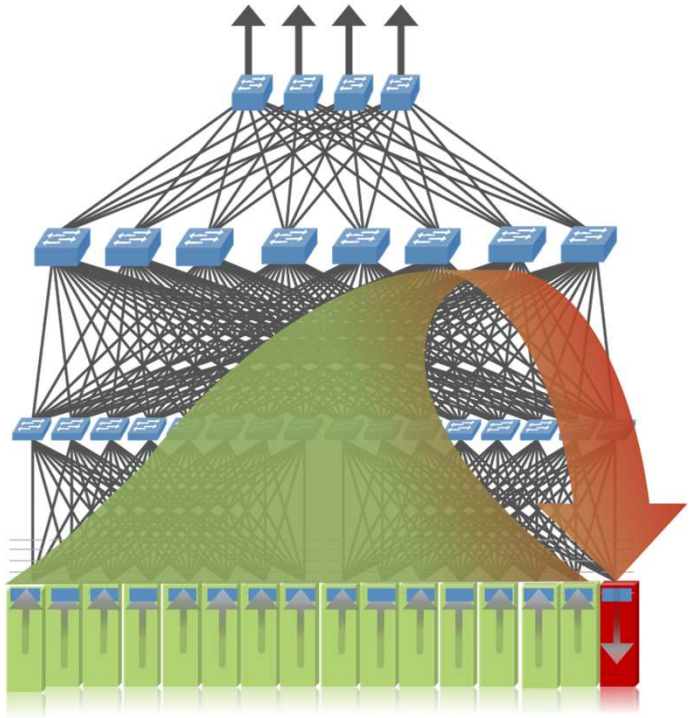


Solution space

- End-to-end (e2e) congestion control
  - Detect congestion in e2e path and adjust TX rates/cwnd
  - Congestion 'signaling' coupled w/ on-going congestion
  - Need many RTTs (100us to ms) to 'flatten the curve'

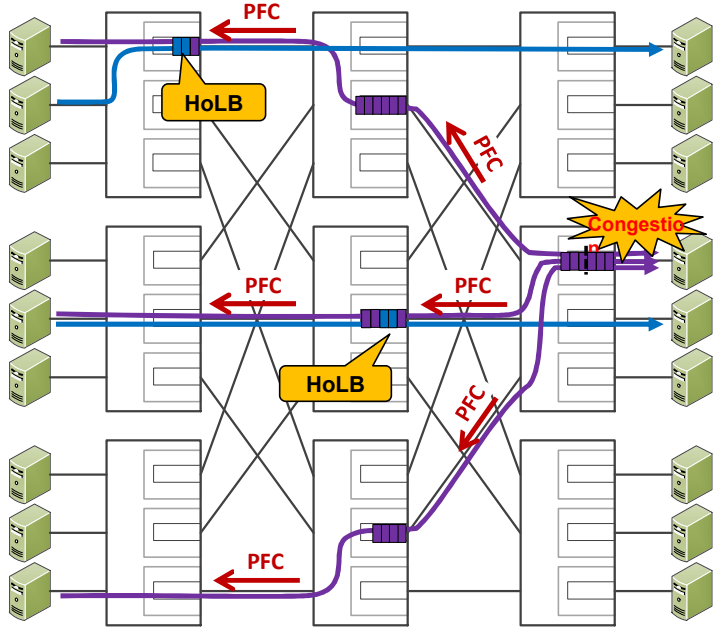
    e.g., cut rate by half

    16:1 incast → 8:1 → 4:1 → 2:1 → 1:1 → … → 0
- Hop-by-hop L2 flow control, e.g., IEEE 802.1 Qbb PFC
  - Low-latency xon/xoff (less than 1us) to previous hop queue
  - Designed to prevent packet loss
  - Slows down the fabric at scale; operational side-effects (details on next slide)



802.1Qbb - Priority-based Flow Control (PFC)

- Operational concerns
  - Head-of-Line blocking
  - Congestion spreading
  - Buffer Bloat, increasing latency
  - Increased jitter reducing throughput
  - Deadlocks with some implementations



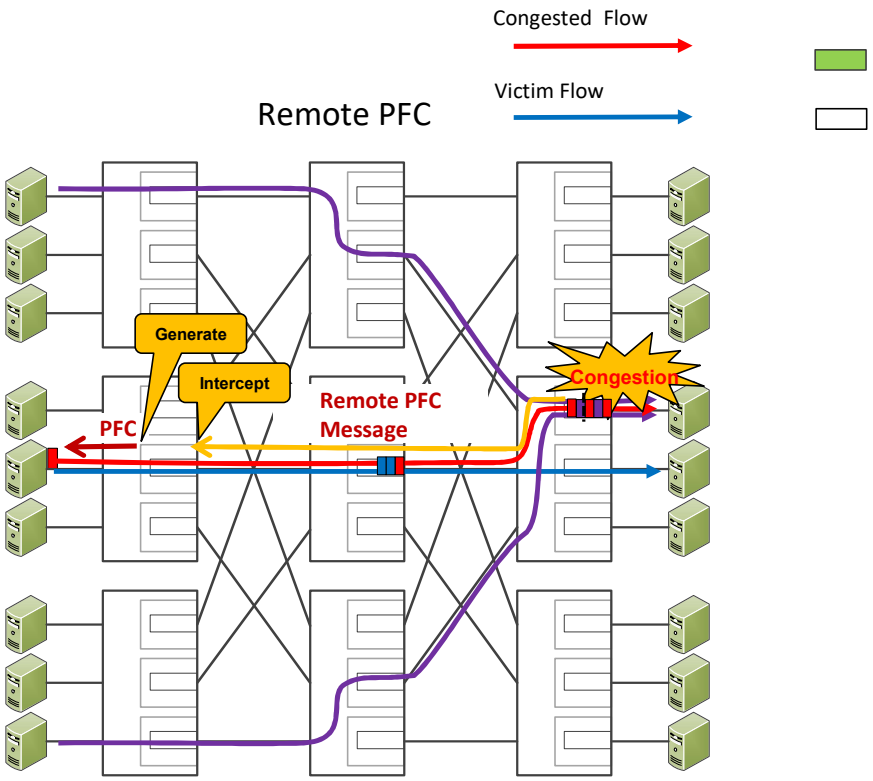
Source: https://mentor.ieee.org/802.1/dcn/21/1-21-0068-01-ICne.pdf

Need for a new, layer-3 flow control mechanism!

- At congested switch
  - Detect queue built-up
  - Compute the minimal time needed to drain the congested queue
  - L3 signal this info backwards towards the incast senders
- Flow control reaction either by
  1. Sender-side ToR switch converts it to standard PFC to sender NIC
     → "Remote PFC" or “Source PFC (SPFC)”
  2. Sender NIC/host directly pauses the source flow
     → ”Source Flow Control (SFC)”

Remote PFC Overview

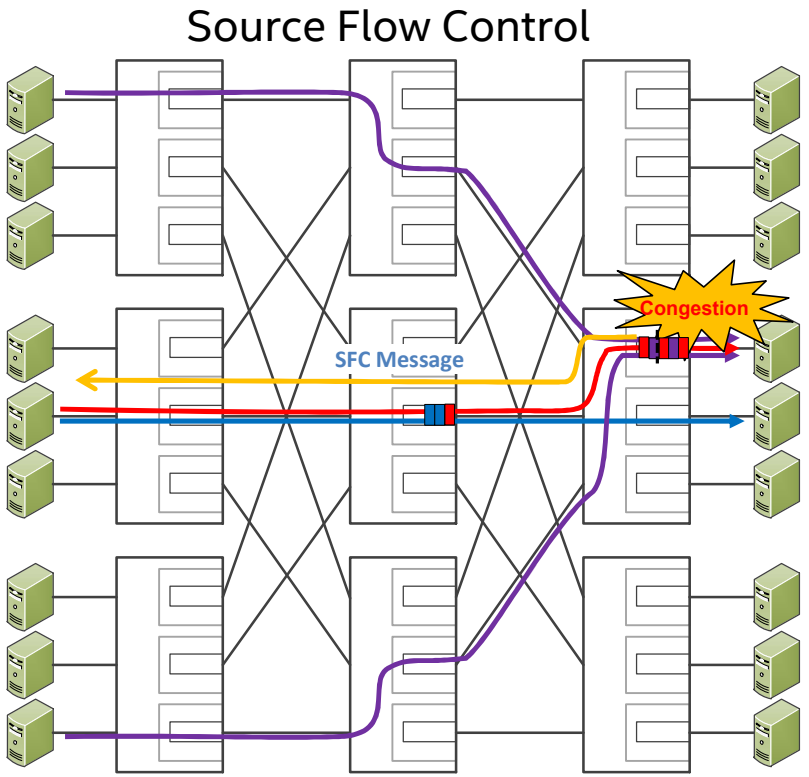
- Edge-to-edge signaling using L3 message
- Existing PFC generated at last hop
- Removes head-of-line blocking of the core network
- Works with today's RDMA NICs
- A small chance for head-of-line blocking remains at the sender NIC



Source: https://mentor.ieee.org/802.1/dcn/21/1-21-0068-01-ICne.pdf

Remote/Source PFC vs Source Flow Control

- Difference
  - Remote PFC = remote generation of PFC at the source ToR
  - SFC = pause at the flow level
- SFC signaling message direct to transport protocol end-point
- Removes head-of-line blocking completely from the network
- Requires next-generation RDMA NICs

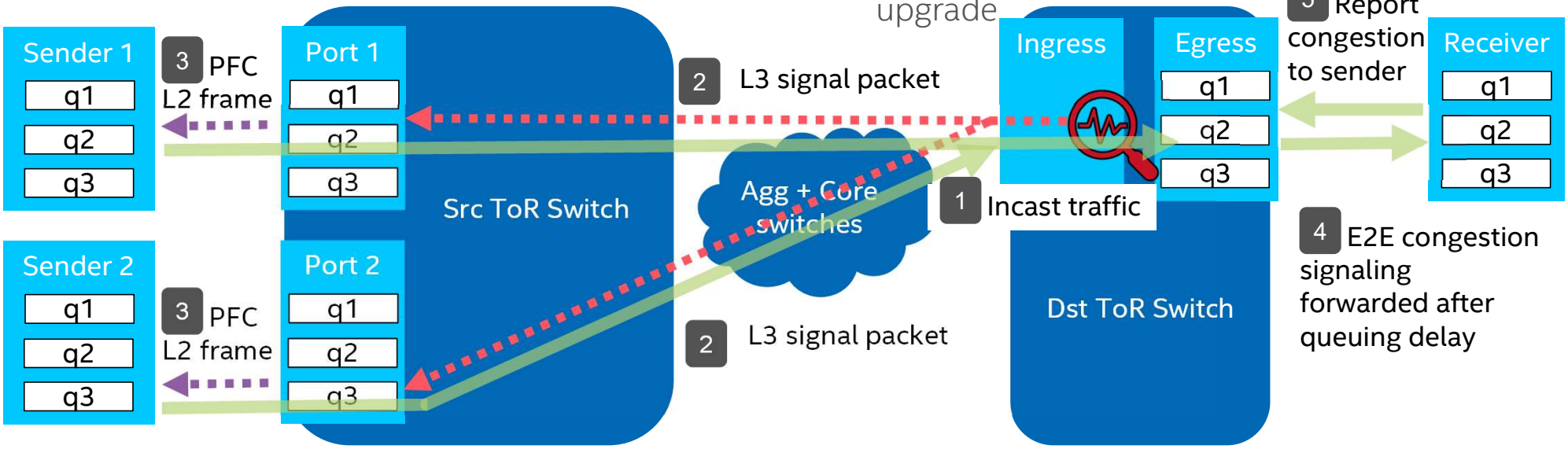


https://mentor.ieee.org/802.1/dcn/21/1-21-0068-01-ICne.pdf

A Closer Look at Remote PFC

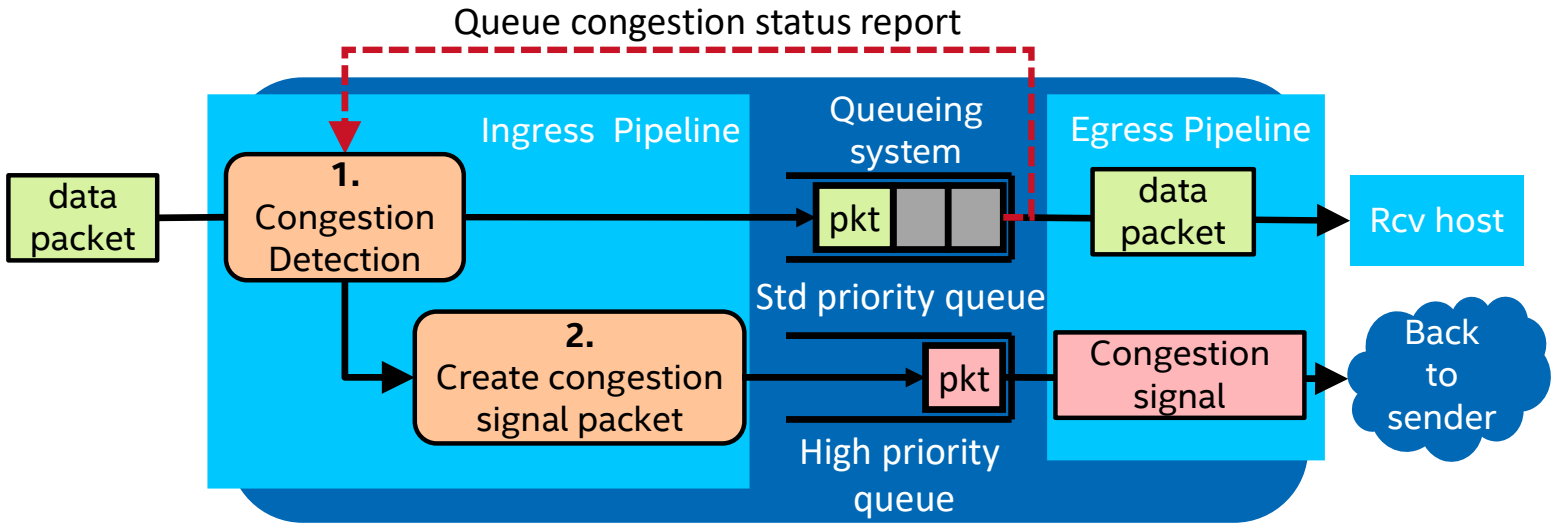
Remote PFC: Edge-to-Edge View

- What is Remote PFC?
  - Edge-to-Edge signaling of congestion
  - Flow control that instantly 'flattens the curve'
  - Signaling + source flow ctrl all in sub-RTT
  - RTT = congestion-free base RTT
- Remote PFC does not target/does target
  - ~~aim 100% lossless~~ → min switch buffering
  - ~~e2e congestion ctrl~~ → NIC flow ctrl
  - ~~Pause Agg/Core switches~~ → no PFC side effects
  - ~~Must greenfield deployment~~ → ToR-only upgrade

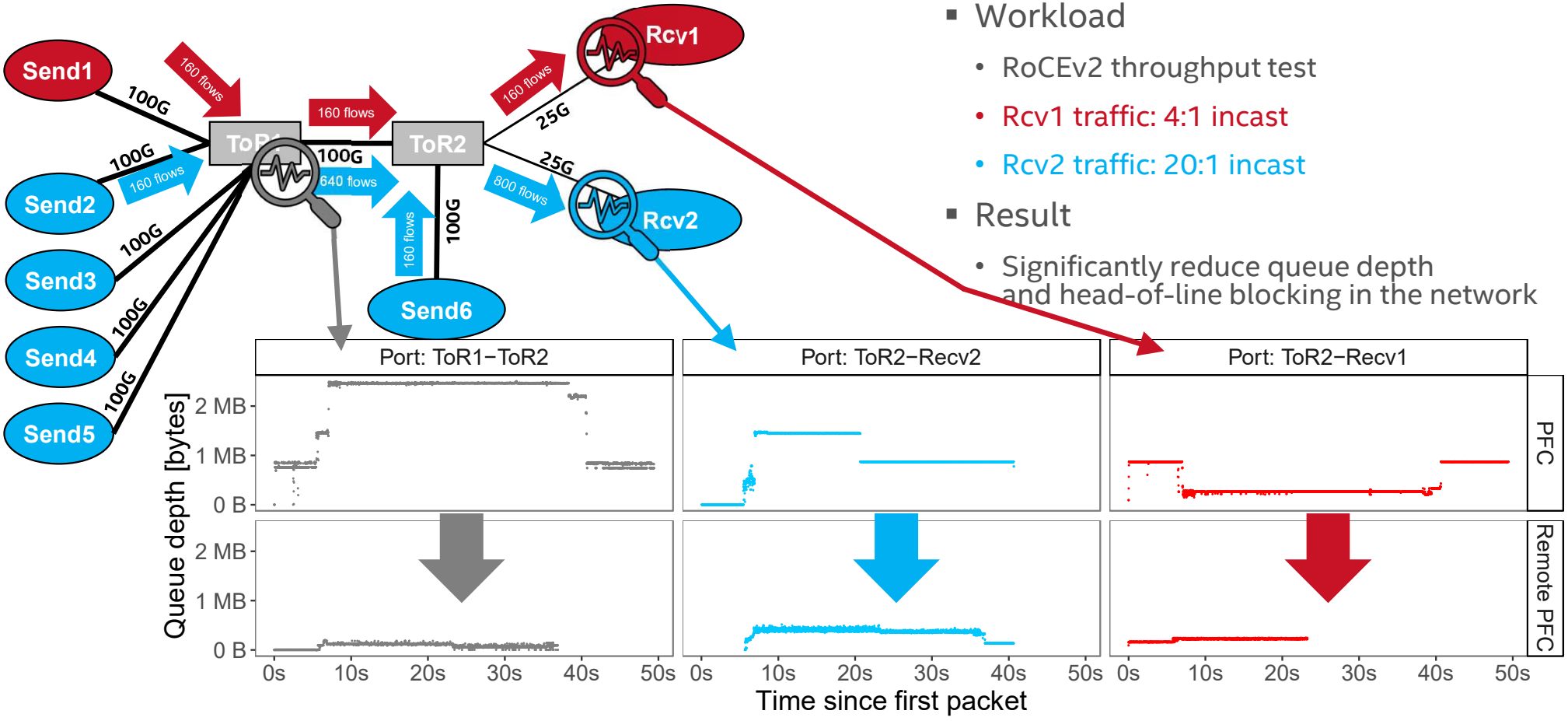


Intelligent Congestion Detection

1. The programmable logic checks the congestion status of an outgoing queue before enqueuing a packet
2. If congestion is detected, a signaling packet is created that skips the congestion and is sent directly back to the sender
    1. Redundant signaling back to the same sender/flow is suppressed temporally



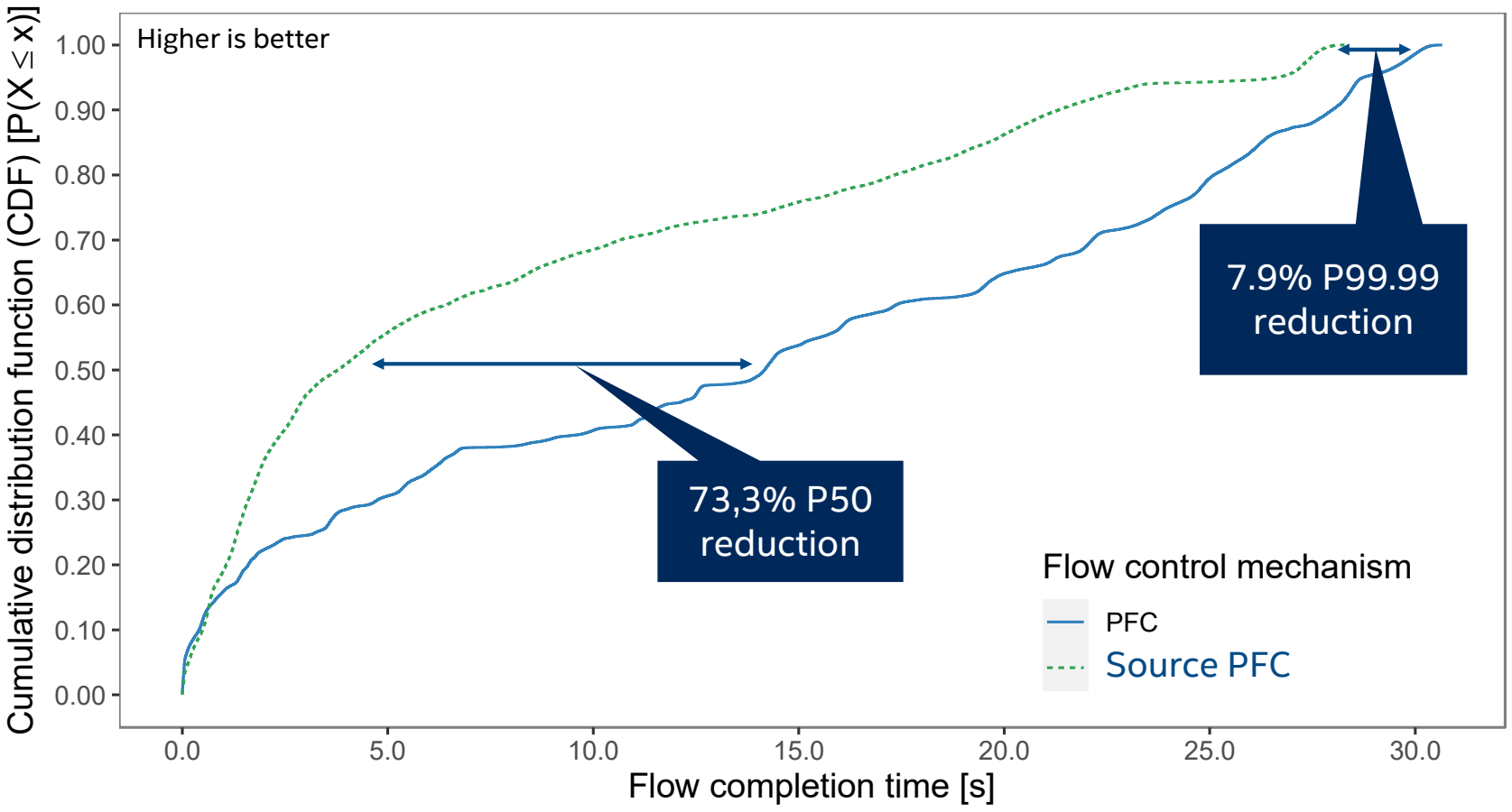
Remote PFC's Effect on Queue Depth



- Workload
  - RoCEv2 throughput test
  - Rcv1 traffic: 4:1 incast
  - Rcv2 traffic: 20:1 incast
- Result
  - Significantly reduce queue depth and head-of-line blocking in the network

See backup for workloads and configurations. Results may vary.

Remote PFC's Effect on Flow Completion Time



See backup for workloads and configurations. Results may vary.

Summary

- Remote PFC
  - Flattens the buffer utilization curve for incast workloads in data centers
  - Leverages the programmability of Intel® Tofino™ 2/Tofino™ 3-based ToR switches for sub-RTT edge-to-edge congestion signaling
  - Compatible with standard NICs that support IEEE 802.1Qbb PFC
- Future
  - Ongoing efforts to standardize Remote PFC at IEEE 802.1
  - Plan to upstream to OCP Switch Abstraction Interface (SAI)
  - Source Flow Control (SFC)
    - Pause directly at the flow level in next-gen RDMA NICs

Notices and Disclaimers

- Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.
- Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.
- Your costs and results may vary.
- Intel technologies may require enabled hardware, software or service activation.
-


intel

‘Source’ Flow Control (SFC) = pause at flow level

- Either at SW stack or modified RDMA HW stack
  - E.g., On-Ramp @ NSDI’21 implemented at qdisc
- How does differ from ICMP Source Quench (SQ, deprecated RFC)?
  - SQ didn’t specify which info to signal or how to react
    - SFC carries pause time duration, and immediately pause the source flow
  - SQ was for WAN Internet
    - SFC is for data center with single administrative domain
- How does differ from IEEE QCN?
  - QCN is *Layer-2* congestion control btw switches and senders, needing multiple RTTs to ‘flatten the curve’
  - Note) RoCEv2 DCQCN is a L3 adoption of QCN, using ECN for e2e congestion control signal

Q/A

- How is the protocol secured? concerns of spoofing the control messages
  - For a single-domain data center of trusted switching devices
  - Signaling between switches (for SPFC) ~= LLDP or BGP
    - Note) BGP encryption may stop a man-in-the-middle attack; but doesn't solve the problem of a malicious or poorly implemented router
  - SFC signaling to sender transport ~= ECN marking
  - ACL at domain boundaries can block signal pkts coming from NIC/host/outside
- Is there another use case for this besides RoCEv2?
  - RDMA is the primary use case of SPFC, making RDMA (regardless of transport) to scale on standard Ethernet fabric
    - See backup for the case with ML training
  - SFC can be applied to non-RDMA use cases; evaluation WiP
- Edge-to-Edge signaling delay will be proportional to RTT, solution for large DC?
  - Cache per-dstIP pause time at sender-ToR or NIC; instant flow control new senders towards the incast dst IP

Slide 17

LJ7 [@Agrawal, Ashutosh] this is based on your input.
Lee, Jeongkeun, 11/6/2021

Switch Config

	Switch Config1 (Remote PFC “off”, PFC “on”)	Switch Config2 (Remote PFC “on”, PFC “off”)
Test by	Intel	
Test date	04/08/2021	
SUT Setup		
Platform	Accton AS9516 32d-r0	
# Switches	2 (ToR1, ToR2)	
HWSKU	Newport	
Ethernet switch ASIC	Intel® Tofino™ 2 Programmable Ethernet Switch ASIC	
SDE version	9.5.0-9388-pr	
OS	SONiC.master.111-dirty-20210201.022355	
Buffer Pool allocation	Ingress Lossless pool size is 7.6MB and lossy pool size is 7.6MB. Egress lossless pool size is 16.7MB, and lossy pool size is 6.4MB.	
Remote PFC threshold	N/A	100KB
PFC threshold	Headroom size is 184KB, dynamic threshold is 4.	N/A

Server Config

	Two server models (A and B) are used at the same time in the testbed	
Server model	Model A	Model B
Test by	Intel	Intel
Test date	04/08/2021	04/08/2021
Server Setup		
Platform	Intel S2600WFT	Supermicro X10DRW-i
# Nodes	3 (Send 6, Recv 1, 2)	5 (Send 1, 2, 3, 4, 5)
# Sockets	2	2
CPU	Intel(R) Xeon(R) Gold 6240 CPU @ 2.60GHz	Intel(R) Xeon(R) CPU E5-2620 v4 @ 2.10GHz
Cores/socket, Threads/socket	18/36	8/16
Microcode	0x5003003	0xb000038
HT	On	On
Turbo	On	On
Power management (disabled/enabled)	enabled	enabled
# NUMA nodes per socket (1, 2, 4...)	2	2
Prefetcher'e enabled (svr_info)	Yes	Yes
BIOS version	SE5C620.86B.02.01.0008.031920191559	3.0a
System DDR Mem Config: slots / cap / speed	6 slots / 16GB / 2934 (*)	8 slots / 32 GB / 2133
Total Memory/Node (DDR, DCPMM)	96, 0	256, 0
NIC	1x 2x100GbE Mellanox ConnectX-6 NIC	1x 2x100GbE Mellanox ConnectX-6 NIC
PCH	Intel C620	Intel C610/X99
Other HW (Accelerator)	RoCEv2 protocol engine in Mellanox ConnectX-6 NIC	RoCEv2 protocol engine in Mellanox ConnectX-6 NIC
OS	Ubuntu 20.04.2 LTS	Ubuntu 20.04.2 LTS
Kernel	5.4.0-66-generic	5.4.0-66-generic
Workload	Custom trace based on Homa (Sigcomm 2018) "Facebook Hadoop" dataset	Custom trace based on Homa (Sigcomm 2018) "Facebook Hadoop" dataset
Compiler	gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0	gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0
Libraries	MLNX_OFED_LINUX-5.1-2.5.8.0 (OFED-5.1-2.5.8)	MLNX_OFED_LINUX-5.1-2.5.8.0 (OFED-5.1-2.5.8)
NIC driver	mlx5_core	mlx5_core
NIC driver version	5.1-2.5.8	5.1-2.5.8
NIC Firmware version	20.28.2006 (MT_0000000224)	20.28.2006 (MT_0000000224)

*The memory population is per system. For server Model A only half of the memory channels are used per socket. This is a sub-optimal memory configuration compared to the best-known configuration where all memory channels are populated but is not a performance-critical issue. The performance-critical path for the workload runs in the RoCEv2 hardware engine of the RDMA NIC and accesses the memory controllers of the CPUs directly. The maximum network throughput on the NIC is limited to the port speed of 100Gbps. The maximum load on the memory controller is limited to 12.5GB/s and hence the memory controller is not a performance limiter.