OPENFABRICS ALLIANCE

13th ANNUAL WORKSHOP 2017

UBIQUITOUS ROCE: ROCE ON DATACENTER NETWORKS

Alex Shpiner, System Architect

Mellanox Technologies

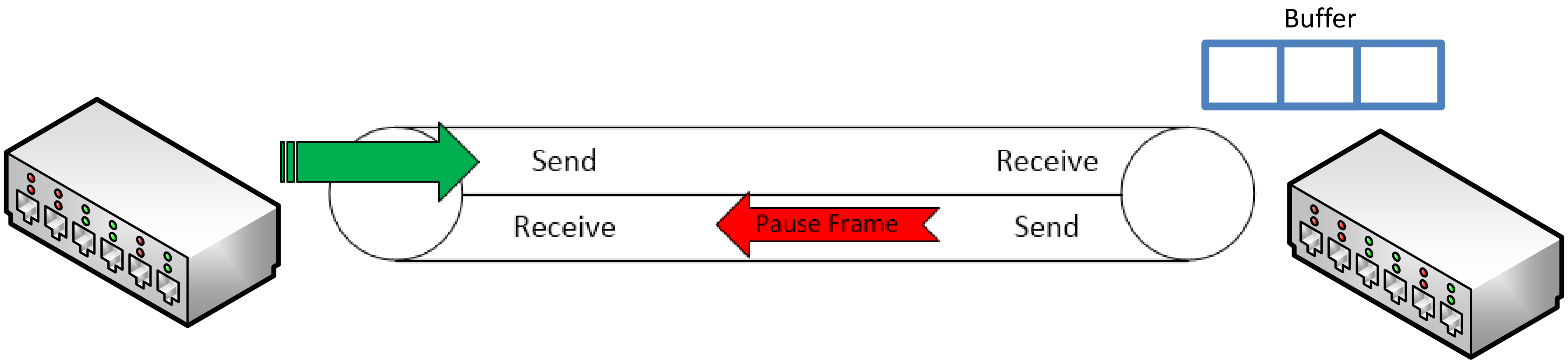
March, 2017

Mellanox TECHNOLOGIES

Connect. Accelerate. Outperform.™

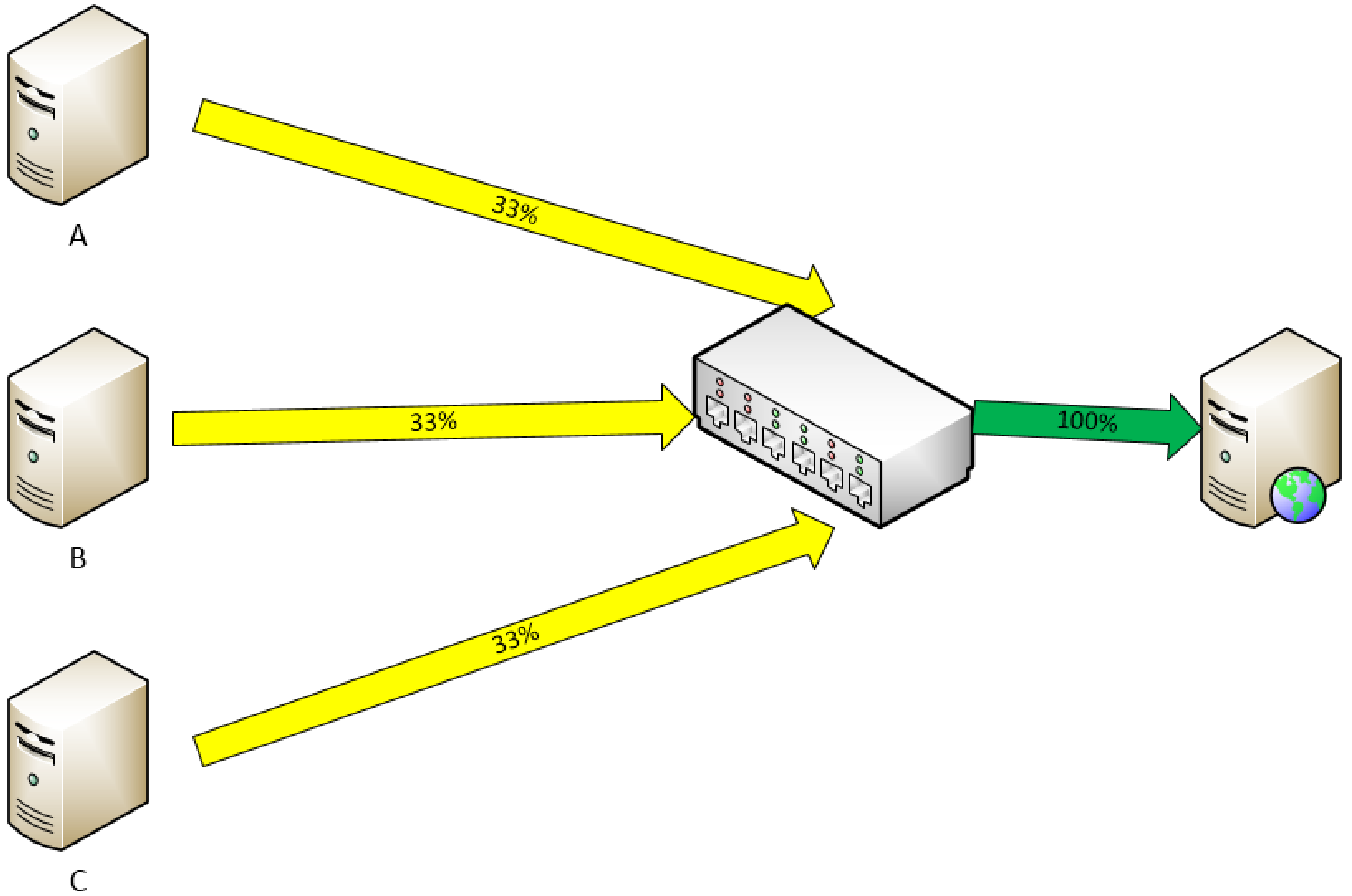
ROCE OVER LOSSLESS ETHERNET

- **RDMA becomes a crucial technology not only for HPC, but for the datacenters.**
- **RoCE was started as lossless network.**
  - InfiniBand legacy
  - Wasted bandwidth
  - Packet drops require complex transport handling
- **Mellanox ConnectX-3 RoCE is used by large installations over lossless network using Priority Flow Control.**



WHY CONGESTION CONTROL IS NEEDED?

- **Data center networks traditionally use Ethernet and their operators like lossy networks**
  - Less configuration
  - PFC deadlock by BGP or PIM
  - Less planned network and traffic
  - Legacy
- **Contrary to lossless network where congestion is not a killer the lossy networks drop packets on congestion**
- **Congestion control throttles rate of traffic injectors**
  - Aims to reduce queue lengths
  - Keep bottleneck link utilization
  - Keep fairness

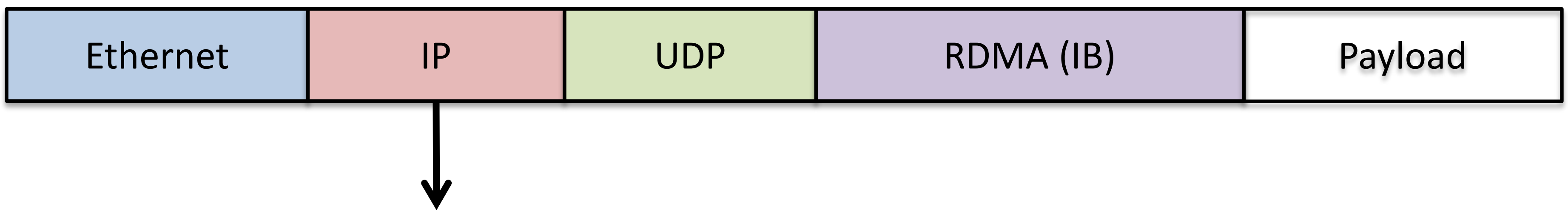


CONGESTION CONTROL ALGORITHM FOR ROCE

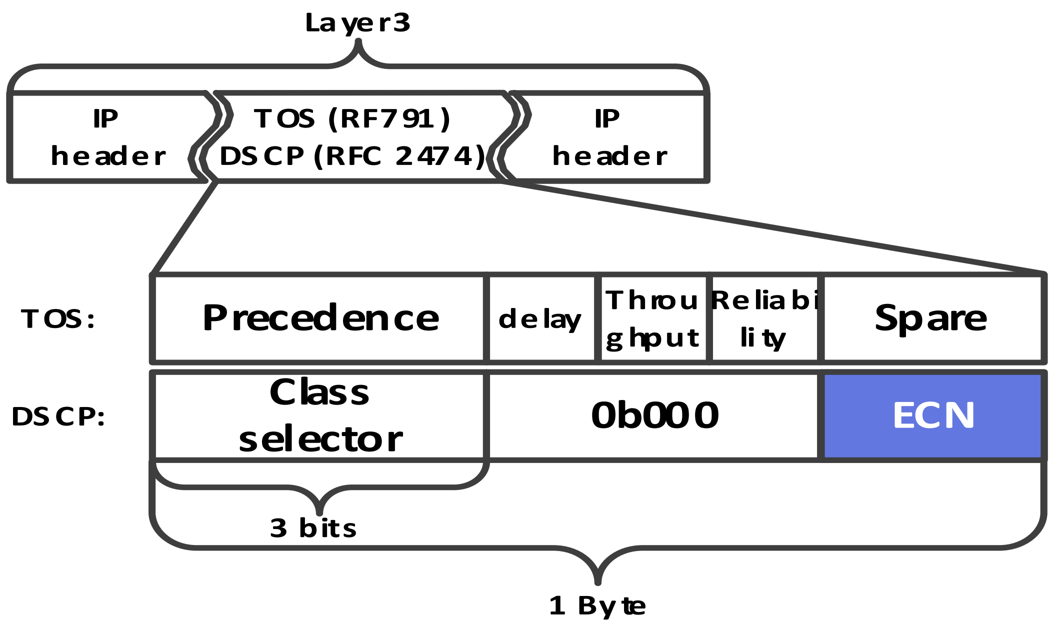
- **DCQCN (Data Center QCN (Quantized Congestion Notification))**
  - Based on combination of DCTCP (Data Center ~~TCP~~) and QCN (Quantized Congestion Notification) algorithms
  - Developed in collaboration with Microsoft
  - Documented in SIGCOMM'15 paper "Congestion Control for Large-Scale RDMA Deployments"

- **Was initially implemented in ConnectX3-Pro.**
  - Firmware-based implementation

ROCE PACKET FORMAT

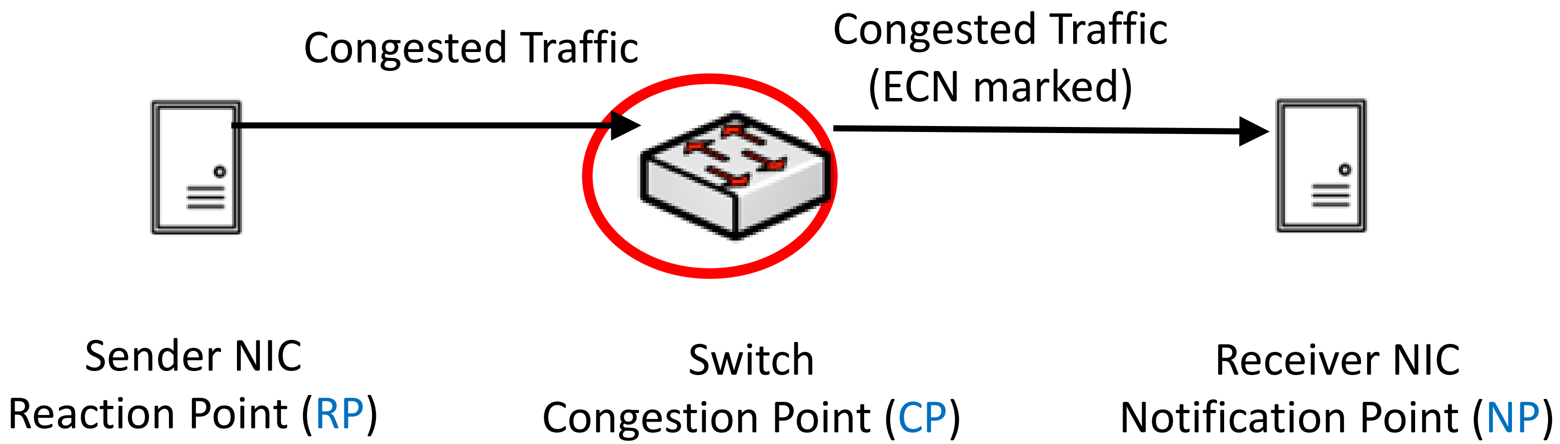


ECN field is used to mark congestion



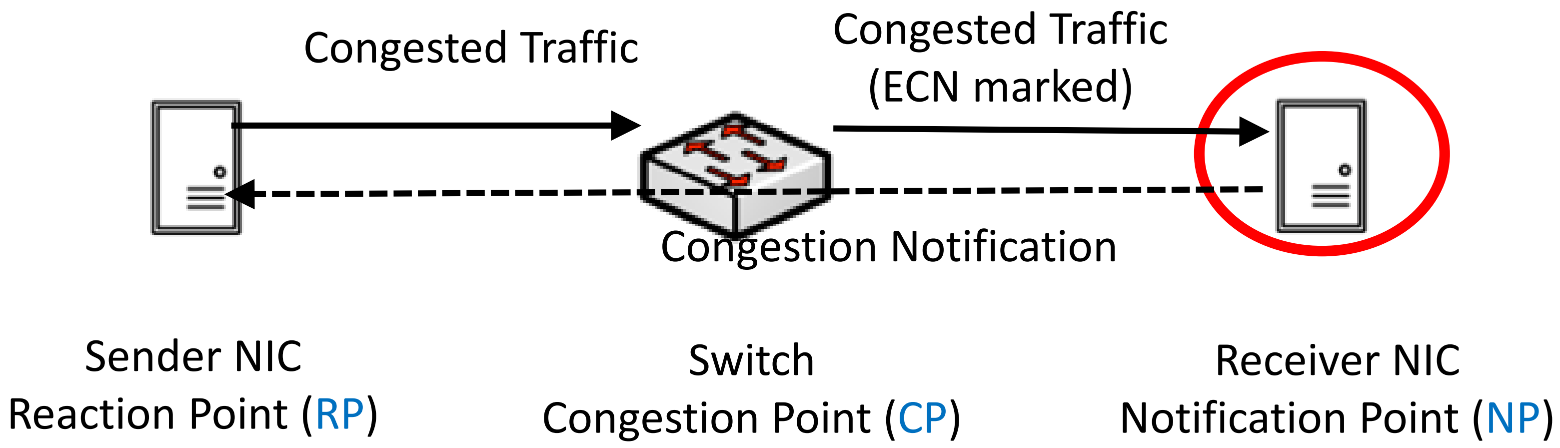
ROCE CONGESTION CONTROL ALGORITHM: CONGESTION POINT

- **Congestion Point (switch): marks ECN bits in packet header based on queue length**
- **Standard functionality supported by all commodity switches**
  - also used for TCP



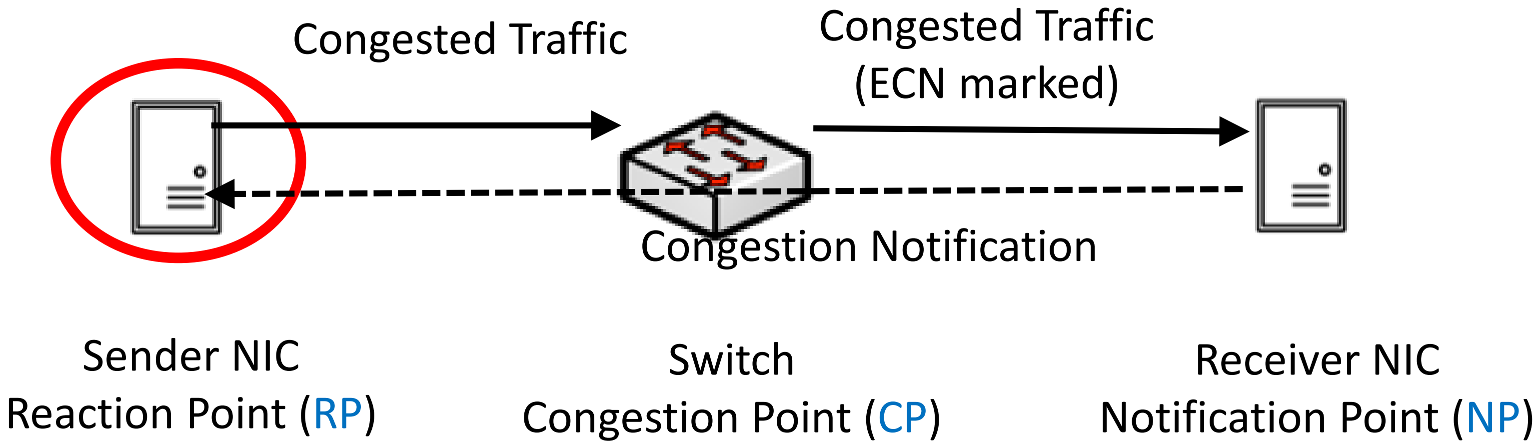
ROCE CONGESTION CONTROL ALGORITHM: NOTIFICATION POINT

- **Notification Point: If ECN-marked packet arrives, sends CNP (Congestion Notification Packet) back**
- **CNP generation is implemented by NIC HW**
  - HW implementation provides fast response
  - CNP can be delivered via low latency path (guaranteed QoS)



ROCE CONGESTION CONTROL ALGORITHM: REACTION POINT

- **Reaction Point: Throttles sending rate based on CNPs arrival**
  - Also based on packet drop (planned)
- **Implemented by HW**
  - Fast response to congestion notification

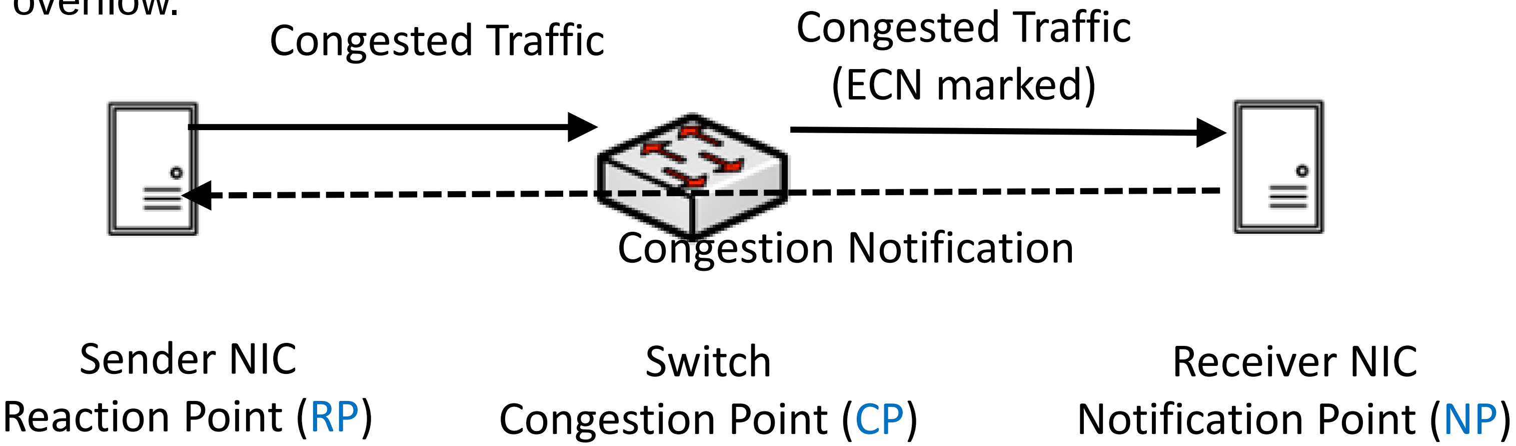


HARDWARE BASED CONGESTION CONTROL

- **The novelty in ConnectX4: Resilient RoCE announcement.**
- **Much faster than SW-based congestion control**
  - HW based: 10's nanosec.
    - Immediately on the entire posted queue
    - Does not require SW intervention
  - SW/FW based: 10's microsec and more
    - Might be much longer due to length of posted queue

~3 **orders of magnitude faster control loop**

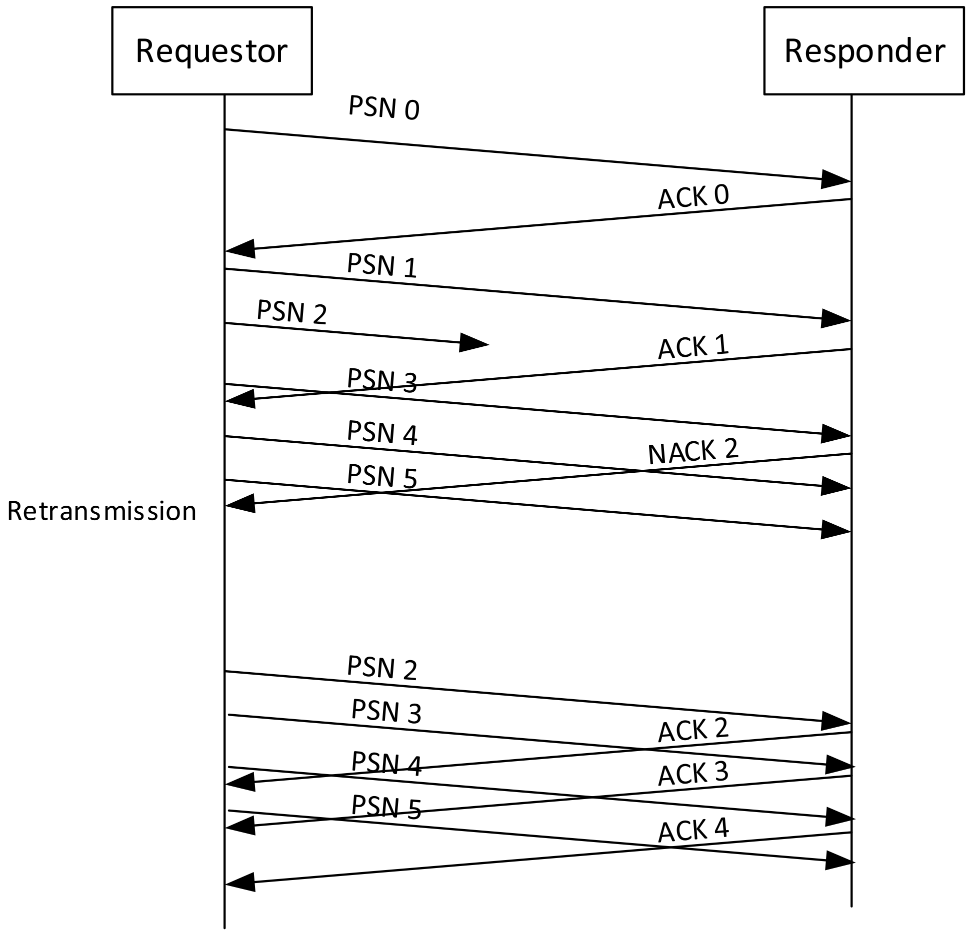
- **Fast reaction to congestion notification minimizes the network congestion time**
  - Congested switch buffers are less likely to overflow.



COPING WITH PACKET DROPS

- **RoCE uses InfiniBand transport semantics.**
- **InfiniBand transport is reliable!**
  - Packets are marked with sequence numbers (PSN)
  - On first packet arrived out of order, responder sends out-of-sequence (OOS) NACK.
  - OOS NACK includes the PSN of the expected packet.
  - Requestor handles OOS NACK by retransmitting all packets beginning from the expected PSN.

  - In previous ConnectX devices, OOS handling was relatively complex firmware flow
  - Each generation of ConnectX adds HW acceleration to handle packet loss events.



OPTIMIZING PERFORMANCE WITH NETWORK QOS

- **Resilient RoCE: RoCE works out-of-box!**
  - Requires only ECN configuration in the switch to make congestion control work.

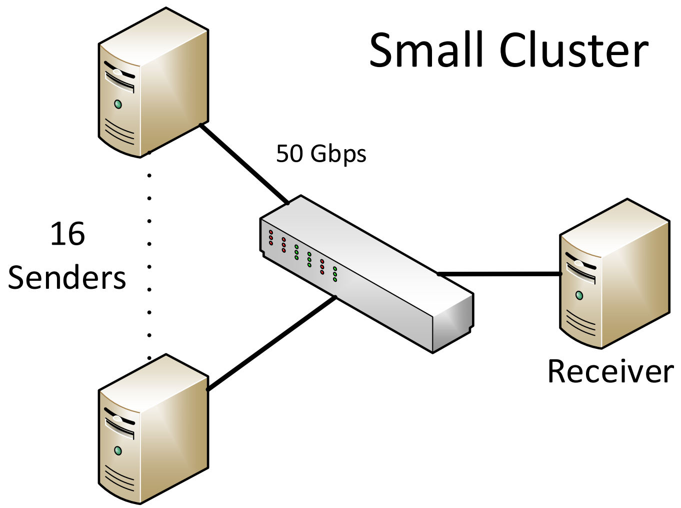
- **However, peak performance is achieved using network QoS configuration.**
- **Every additional layer of QoS configuration will improve RoCE performance:**

- **High priority traffic class separation of CNPs (congestion notification packets)**
  - Fast propagation over the network. Bypassing congested queues.
- **RoCE traffic priority isolation from other traffic (eg. background TCP, UDP)**
  - Avoid co-existence problems with non-controlled (or differently controlled) traffic
- **Flow Control (lossless network)**
  - Better to pause packets than drop packets

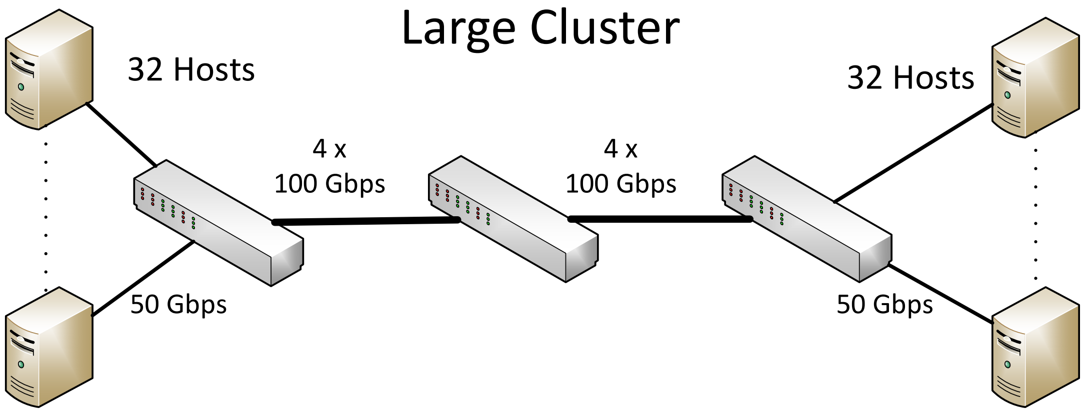
LAB EXPERIMENTS

LAB SETUP

- **Traffic Patterns:**
  - Many to One
  - All to All
- **Traffic/Network Configurations:**
  - RoCE over lossless network
  - RoCE over lossy network
  - RoCE + TCP with priority separation
  - RoCE + TCP without priority sepration
  - TCP only

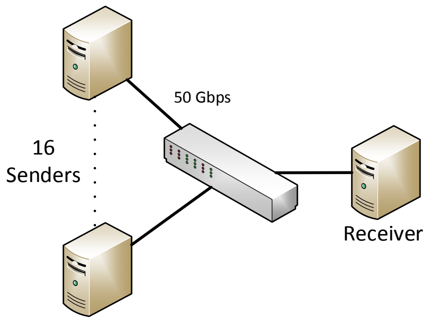


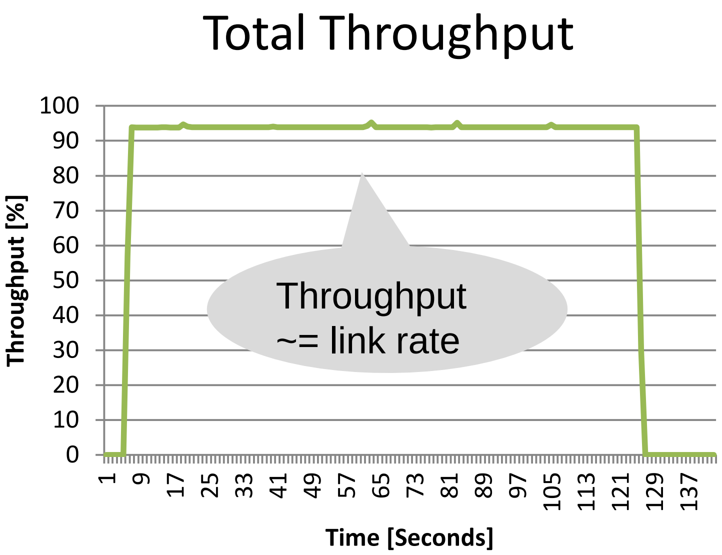
- **Tool: ib_write_bw / nd_perf**
  - Streaming continuous traffic of Write Requests
- **Driver: MLNX_OFED v. 4.0-1.6.1.0**
- **TCP stack: cubic (Linux Red Hat 7.0 defaults)**
- **Switch: Mellanox Spectrum**

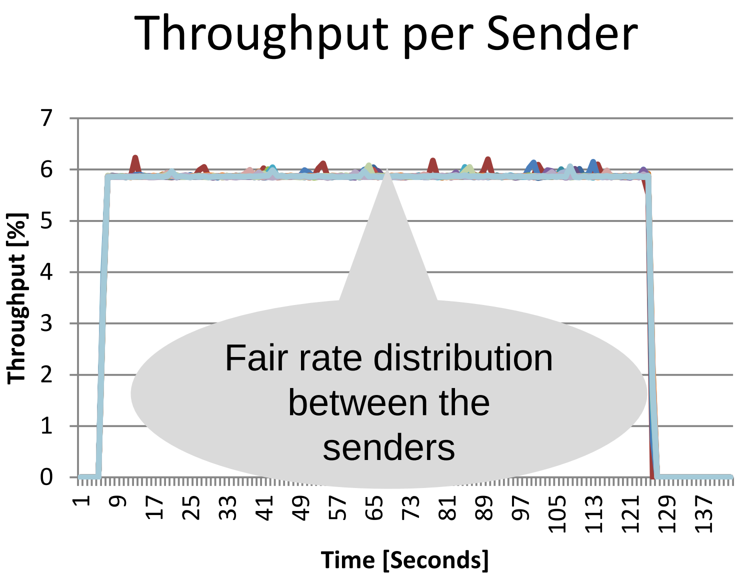


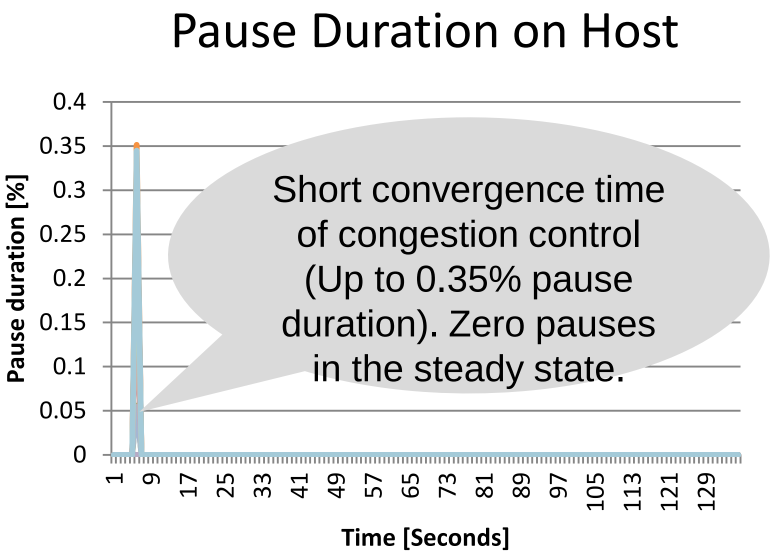
SMALL CLUSTER: LOSSLESS NETWORK

- **16 hosts to 1**
- **64 QPs per sender**



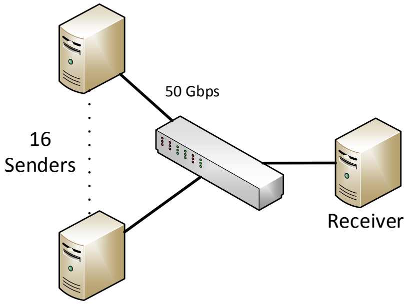




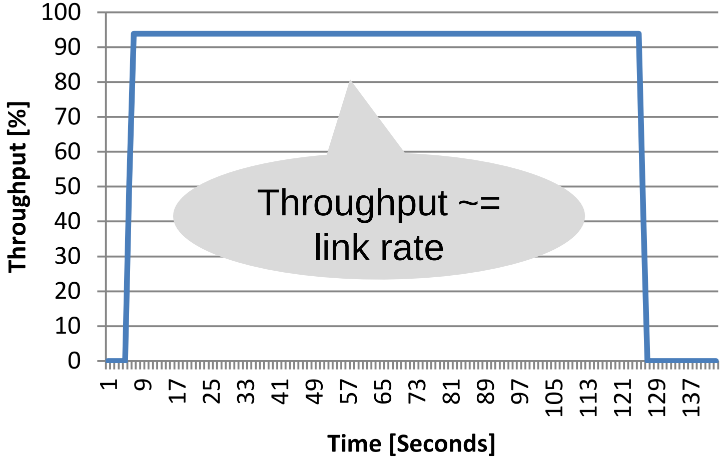


SMALL CLUSTER: LOSSY NETWORK

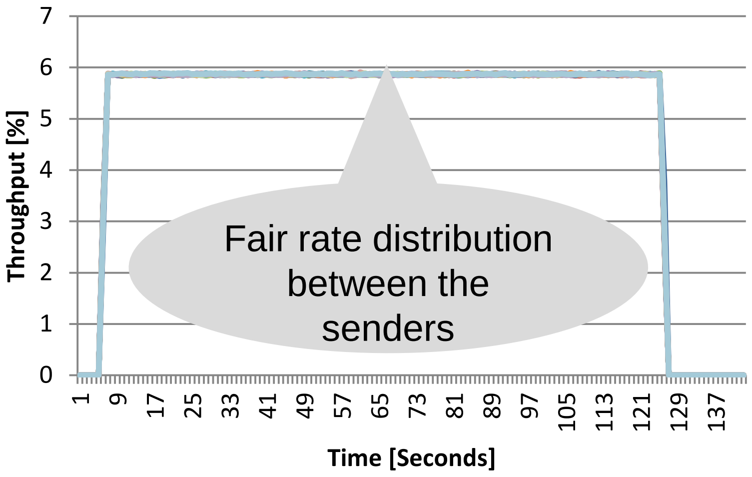
- **16 hosts to 1**
- **64 QPs per sender**



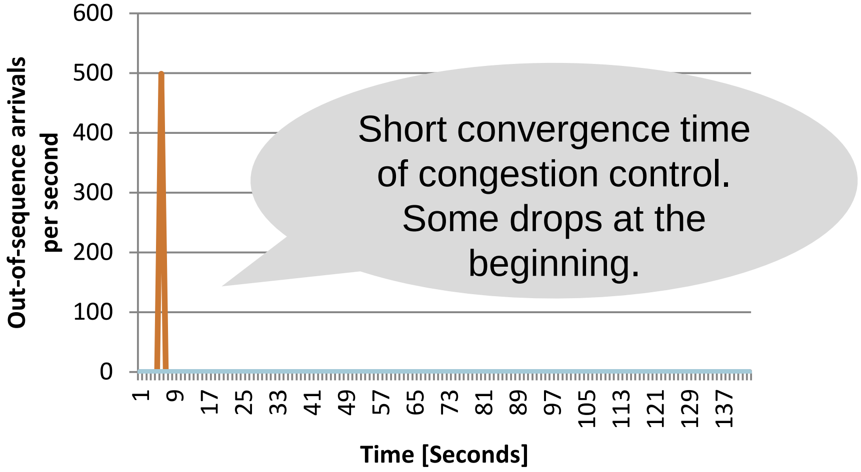
Total Throughput



Throughput per Sender



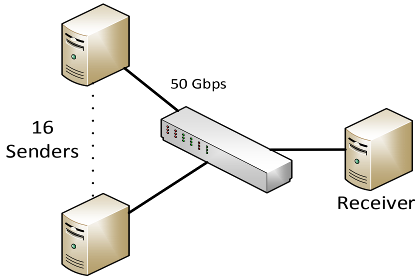
Out of Sequence Events
(indicates packet drops)



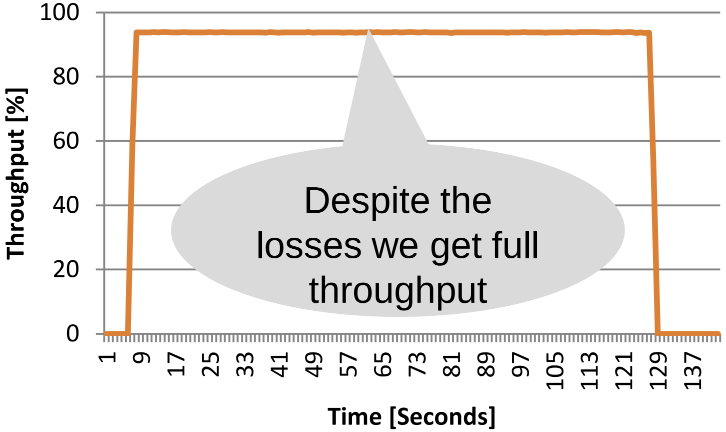
SMALL CLUSTER: LOSSY NETWORK UNDER HIGH LOAD

- **16 hosts to 1**
- **512 QPs per sender**

Large value was chosen to test system in stress

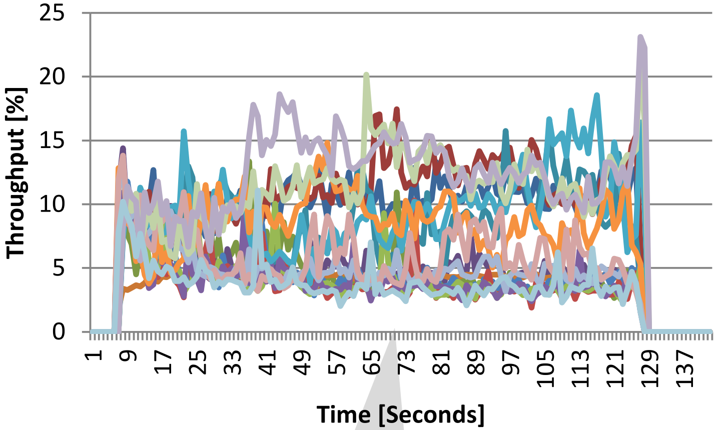


Total Throughput

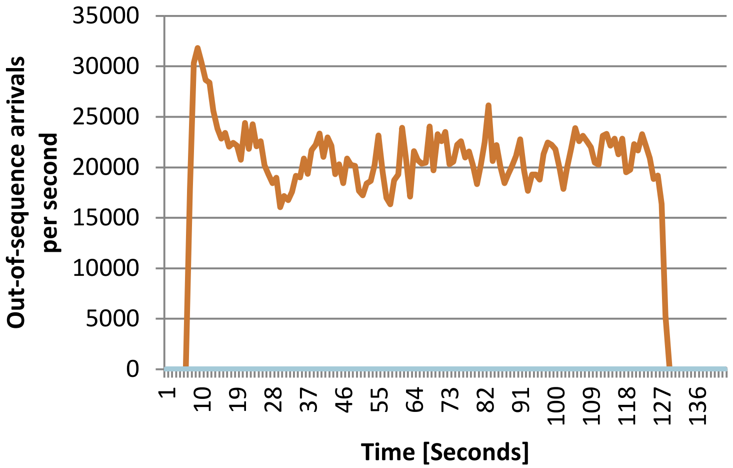


Despite the losses we get full throughput

Throughput per Sender



Out of Sequence Events (indicates packet drops)



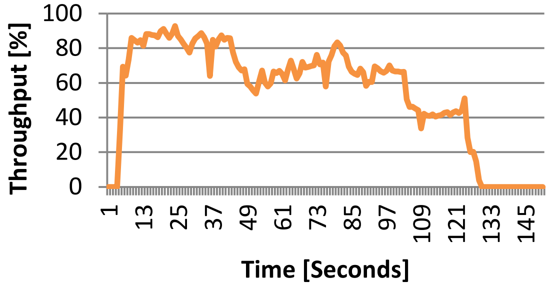
Losses cause unfair distribution. Issue to improve.

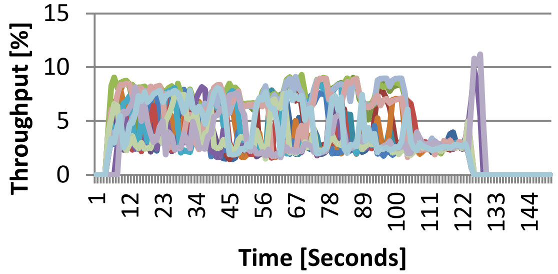
SMALL CLUSTER: ROCE VS TCP

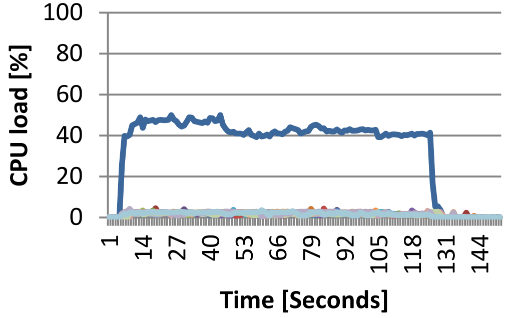
- **16 hosts to 1**
- **64 QPs per sender**
- **Lossy network**

Total Throughput | Throughput per Sender | CPU usage

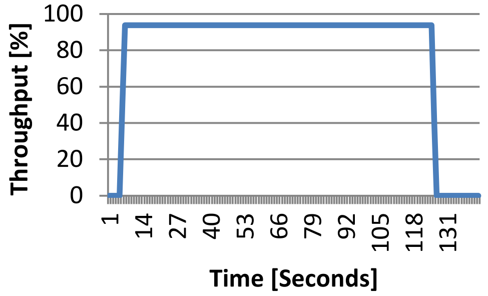
Experiment 1:
TCP

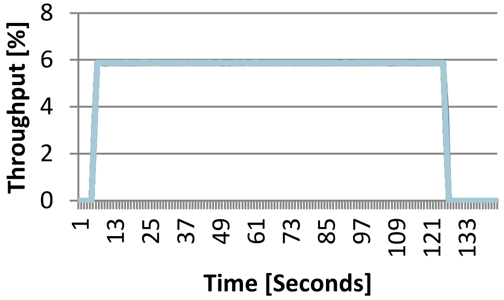


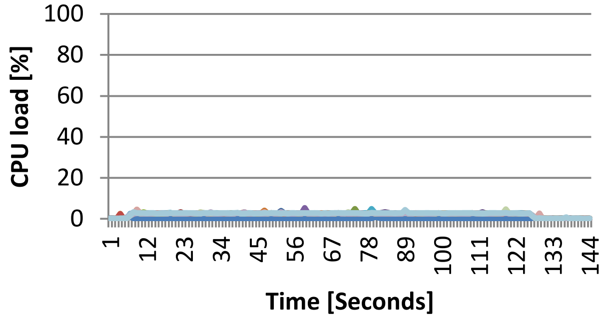




Experiment 2:
RoCE







Conclusions:

RoCE achieves almost twice larger throughput than TCP

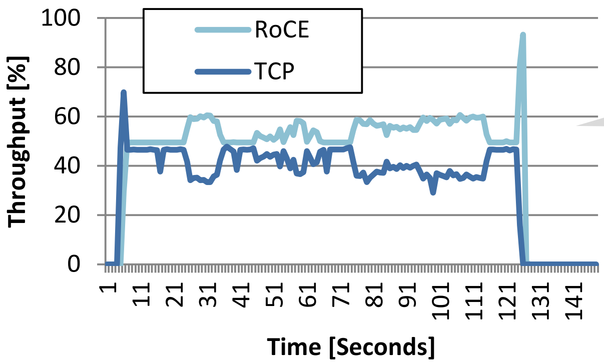
RoCE achieves better fairness and less fluctuations than TCP

TCP requires high CPU usage, while RoCE requires negligible CPU usage

HYBRID TRAFFIC (ROCE AND TCP), PRIORITIES ISOLATION

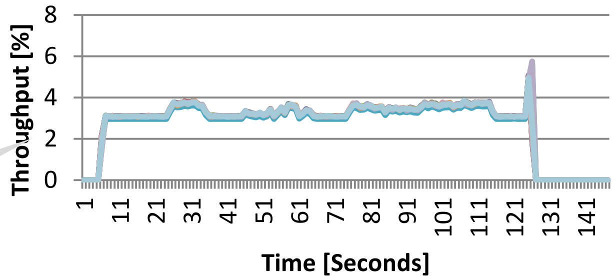
- **16 hosts to 1**
- **RoCE on lossless: 32 QPs per sender**
- **TCP on lossy: 32 flows per sender**

Total Throughput



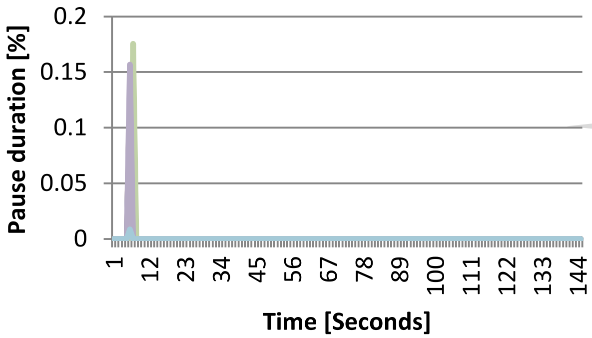
Fair share between two types of traffic: each one with ~half line rate BW
(oscillations due to under-flowing TCP queues)

RoCE Throughput per Sender



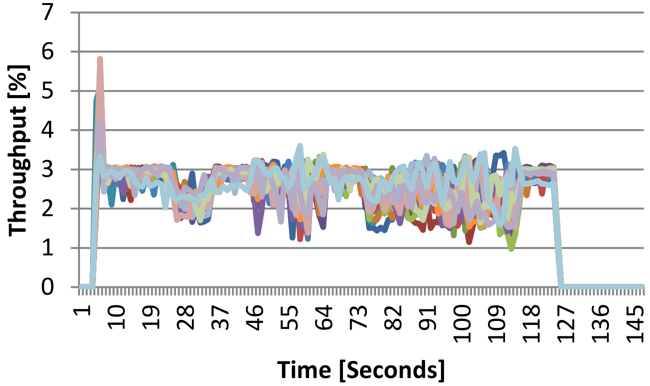
Inter-flow fairness in RoCE is preserved

Pause Duration on Host



No pauses from RoCE traffic during the stable stage of the test

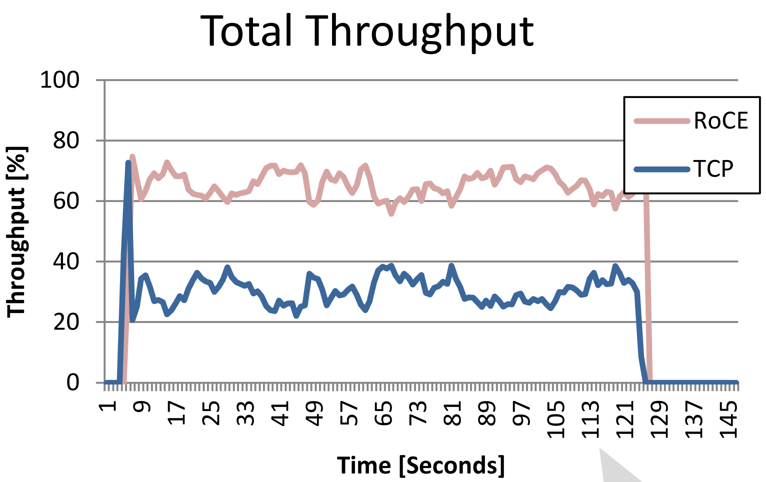
TCP Throughput per Sender

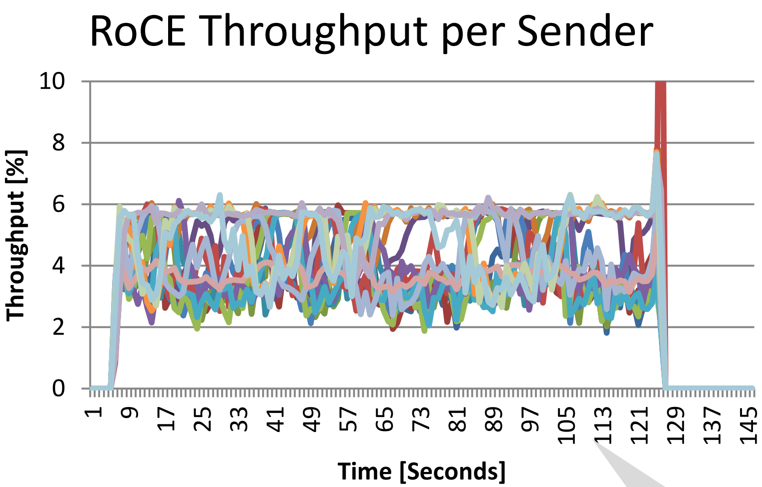


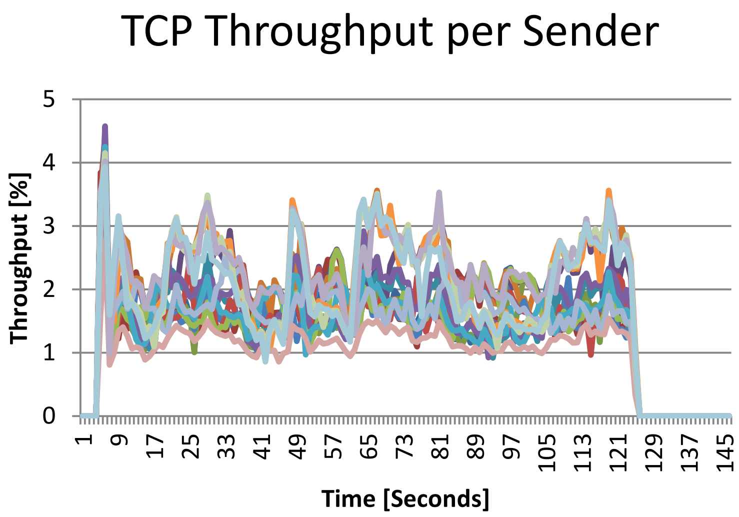
HYBRID TRAFFIC (ROCE AND TCP), NO PRIORITIES ISOLATION

- 16 hosts to 1

- RoCE 64QP / TCP 32 flows
- Lossy network







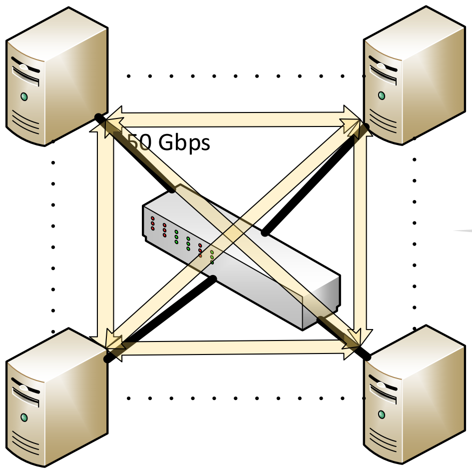
Achieved fairness by ratio of RoCE/TCP flows: more RoCE QPs, more RoCE BW

Background TCP traffic harms RoCE fairness

SMALL CLUSTER, ALL TO ALL

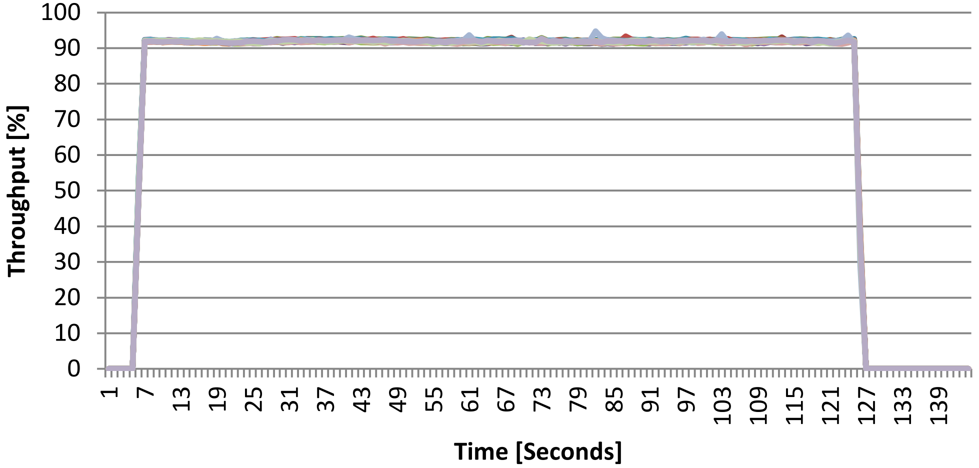
- **16 hosts, all to all**
- **16 QPs per pair of hosts**

16 hosts

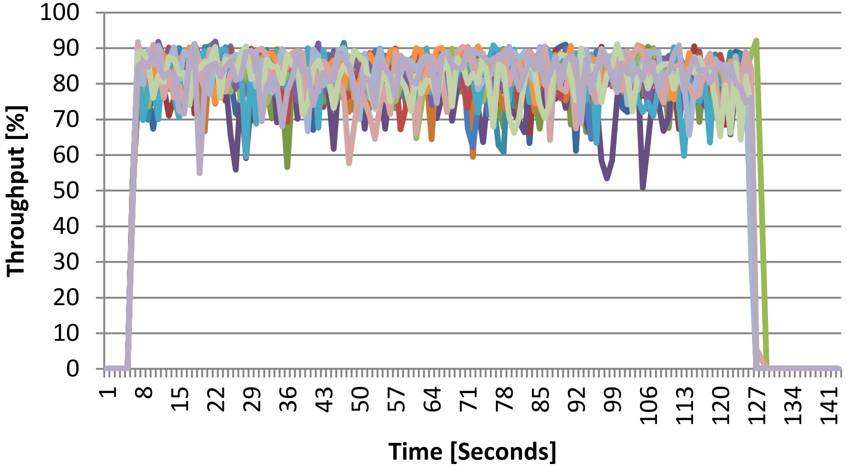


Switch is overloaded due to bursts and dis-synchronization

Lossless network

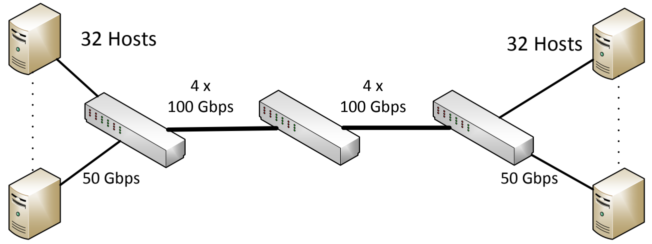


Lossy network

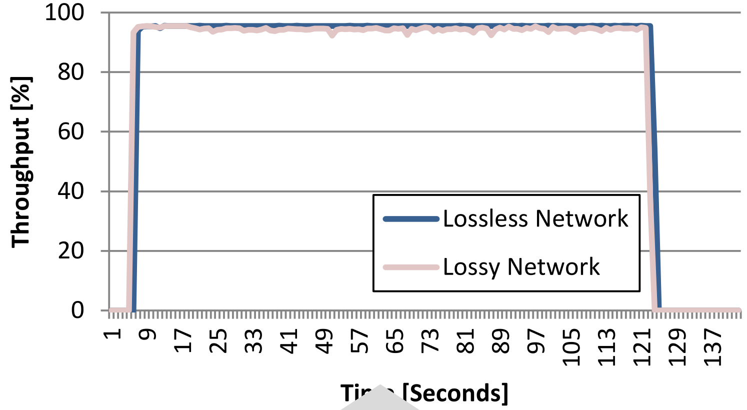


LARGE CLUSTER: MANY TO ONE

- **63 hosts to 1**
- **Lossless: 16QPs per sender.**
- **Lossy: 16 QPs per sender**



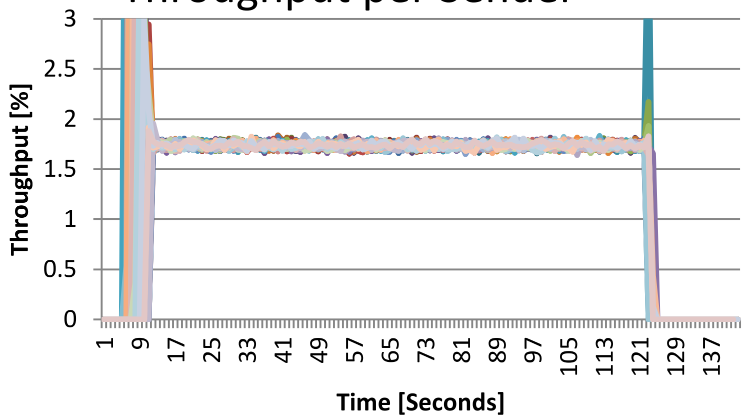
Total Throughput



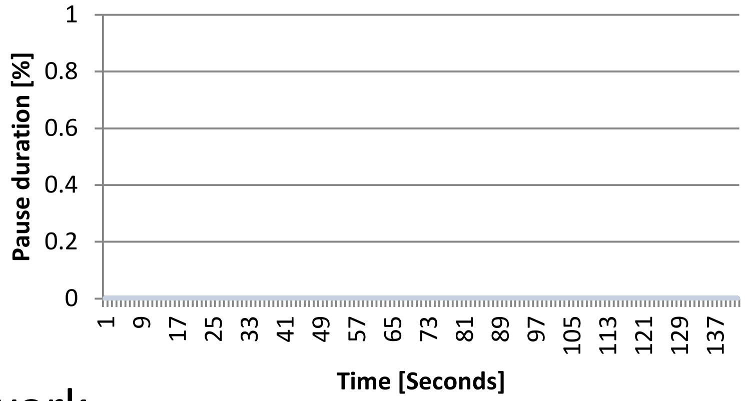
High throughput in both lossless and lossy

Lossless network

Throughput per Sender

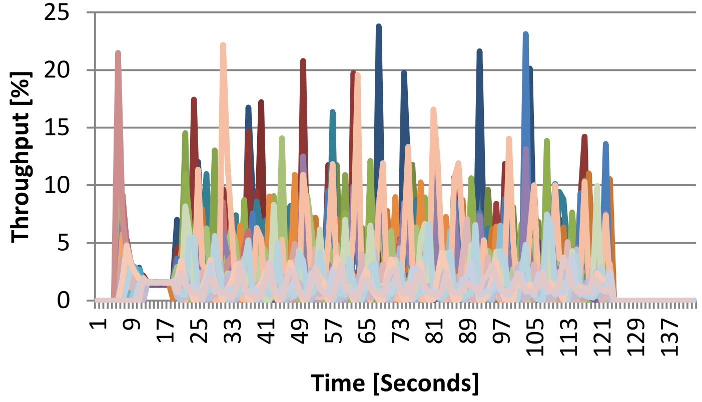


Pause Duration on Host

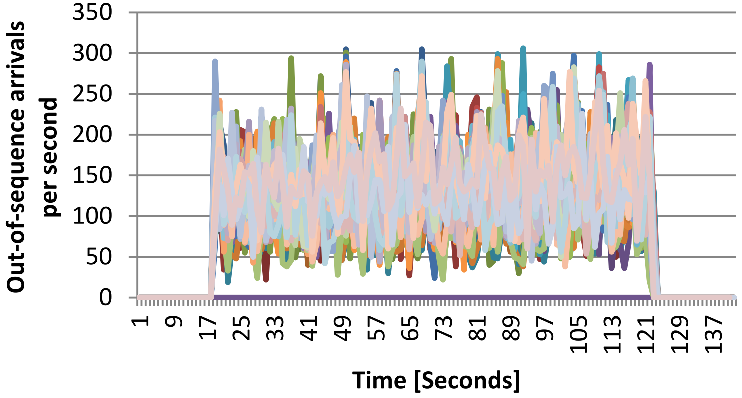


Lossy network

Throughput per Sender



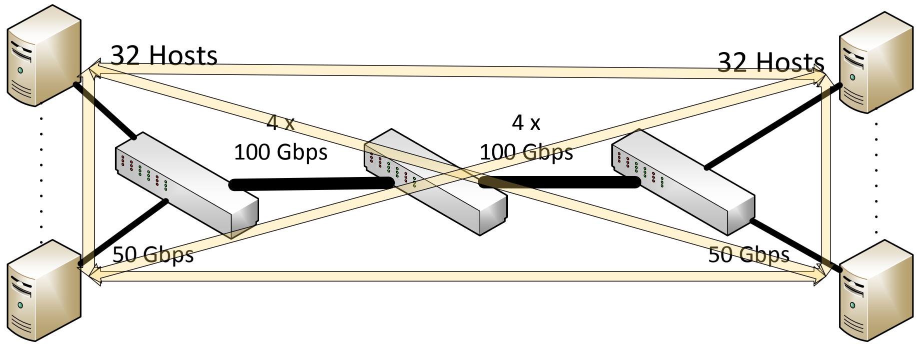
Out of Sequence Events
(indicates packet drops)



LARGE CLUSTER, ALL TO ALL

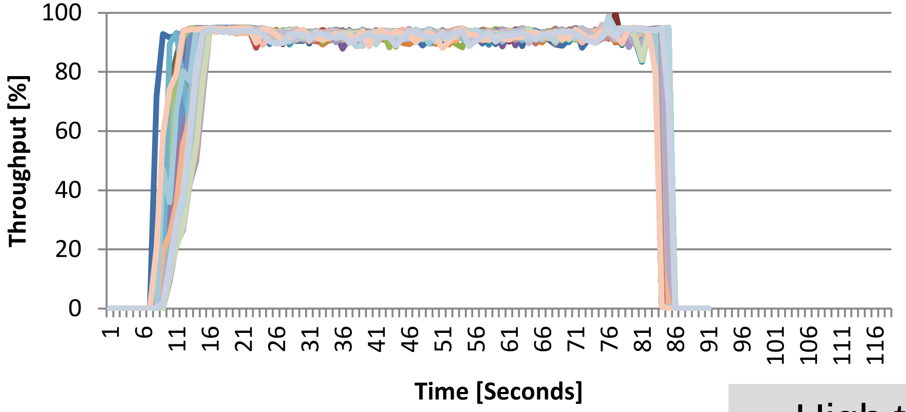
- **64 hosts, all to all**
- **4QPs per pair of hosts**

64 hosts



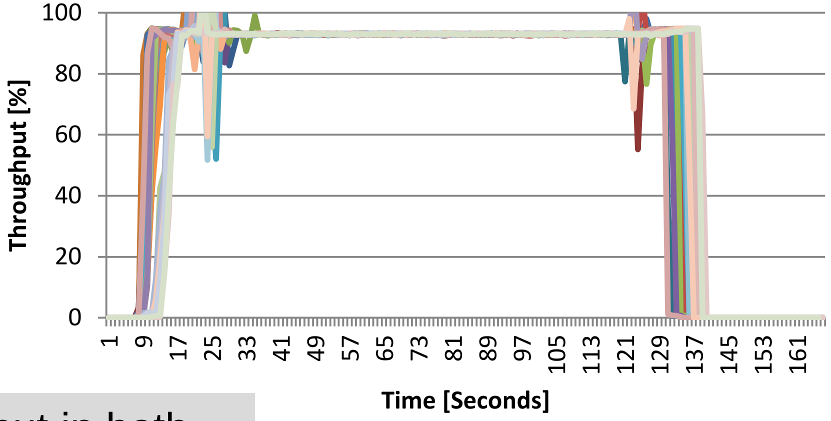
Lossless network

Throughput per Sender



Lossy network

Throughput per Sender



High throughput in both lossless and lossy

LARGE CLUSTER: ROCE VS TCP, MANY TO ONE

- **63 hosts to 1**
- **16 QPs per sender**
- **Lossy network**

Low TCP Throughput

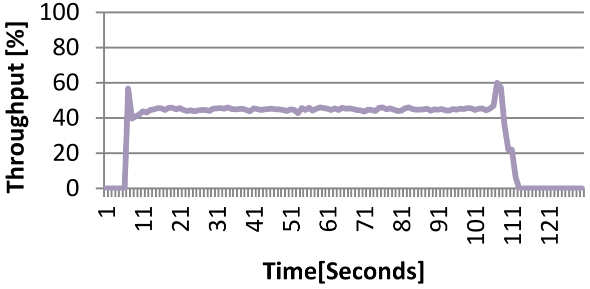
100% CPU load in the receiver

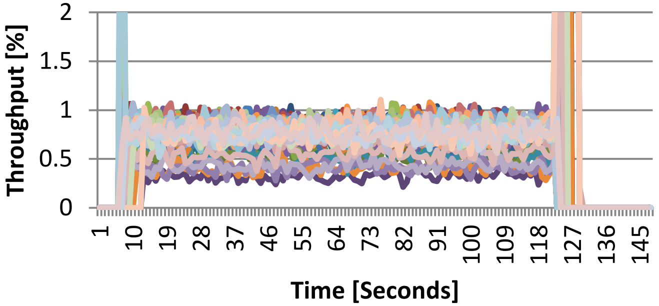
Total Throughput

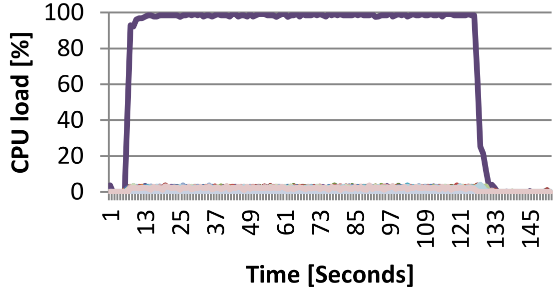
Throughput per Sender

CPU Load

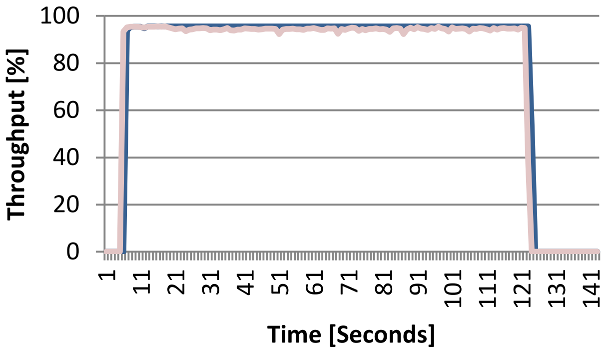
Experiment 1: **TCP**

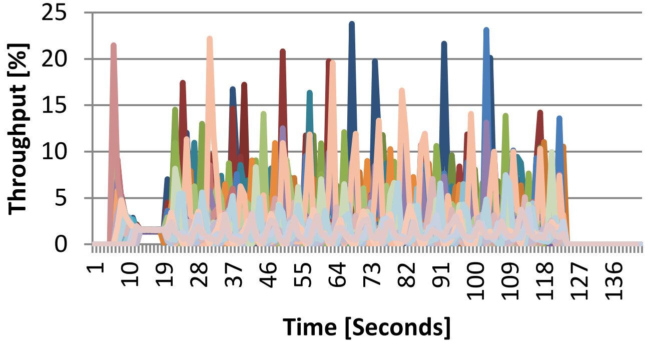


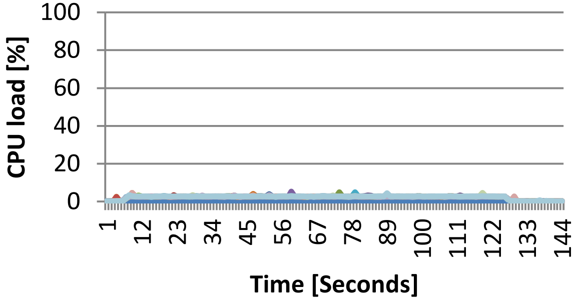




Experiment 2: **RoCE**







Conclusions:

RoCE achieves twice larger throughput than TCP

TCP requires high CPU usage, while RoCE requires negligible CPU usage

SUMMARY

- **Many data center deployments require lossy networks.**
- **Mellanox announced Resilient RoCE: running RoCE without flow control.**
- **ConnectX4 HW-based congestion control.**
- **Network QoS configuration for peak performance.**
- **Lab measurements of**
  - Lossless, lossy
  - Many to one , all to all
  - Co-existence with TCP
  - Comparison to TCP

- **Resilient RoCE works.**

13th ANNUAL WORKSHOP 2017

THANK YOU

Alex Shpiner, System Architect

Mellanox Technologies

Connect. Accelerate. Outperform.™

BACKUP SLIDES

RDMA FOR DATA CENTERS

- **Why?**
  - CPU utilization for non-communication computations.
  - Low latency communication for real-time applications.
  - High-bandwidth storage applications

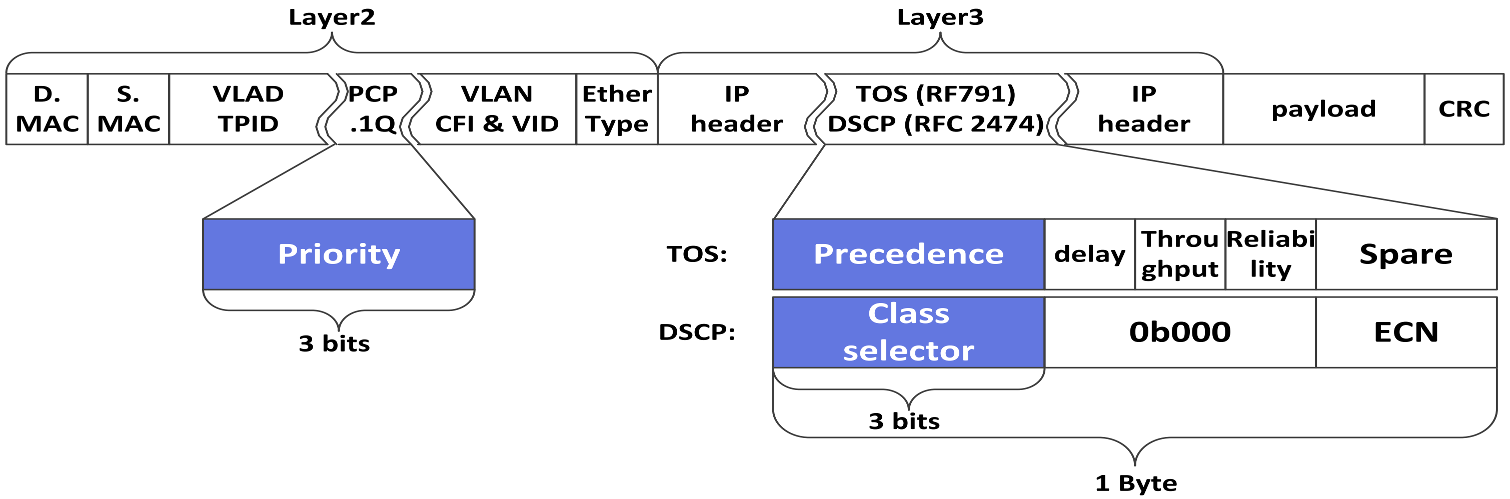
- **RDMA becomes a crucial technology not only for HPC, but for the datacenters**

CONGESTION CONTROL AND FLOW CONTROL

	Without flow control (PFC)	**With** flow control (PFC)
Without congestion control	Low performance due to many packet drops	Used in HPC (network optimized applications) - Congestion might spread - Not recommended for large scale data centers
With congestion control	Resilient RoCE - Easier to configure, but may cause slightly lower performance. - Congestion control alone reduces buffer overflows drops, but cannot prevent it.	Lossless RoCE - Recommended for large scale - Deployed today in production at scale

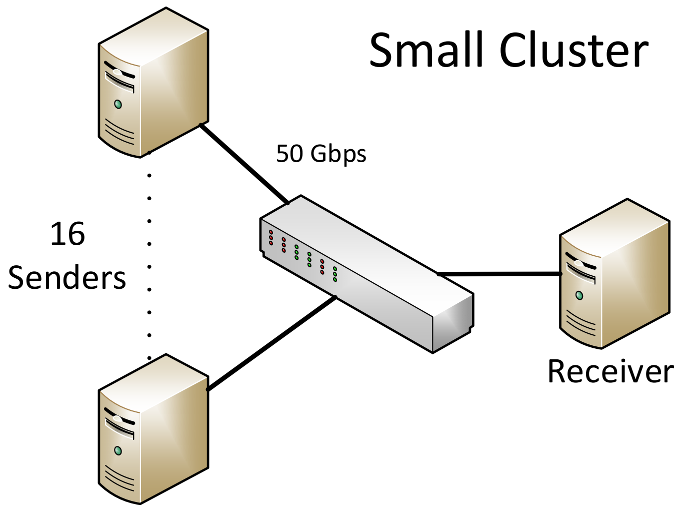
TRAFFIC CLASSIFICATION

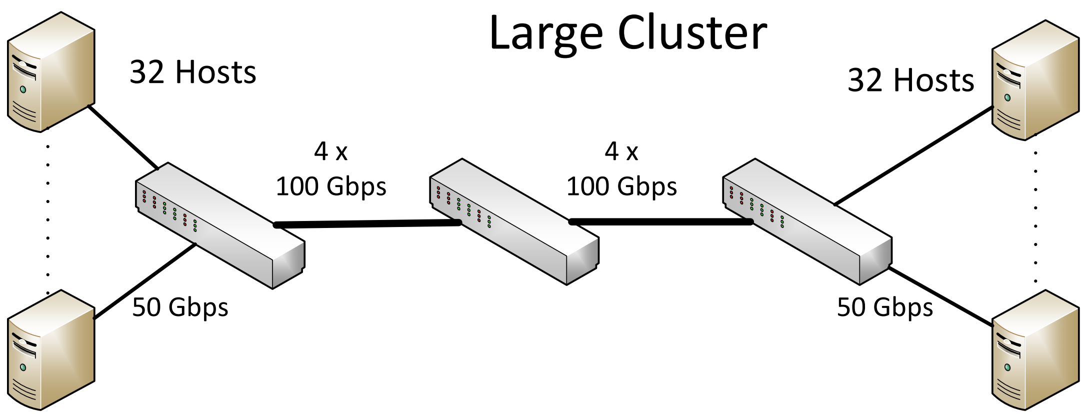
- **Classification used for:**
  - Scheduling (WRR, strict)
  - Buffer management
  - Lossless network: priority flow control
- **Per priority. Priority can be indicated by**
  - PCP (Priority Code Point) in the Vlan tag.
  - DSCP (Differentiated Service Code Point) in IP header.



LAB SETUP

- **Traffic Patterns:**
  - Many to One
  - All to All
- **Traffic/Network Configurations:**
  - RoCE over lossless network
  - RoCE over lossy network
  - RoCE + TCP with priority separation
  - RoCE + TCP without priority sepration
- **Tool: ib_write_bw / nd_perf**
- **Driver: OFED v. 4.0-1.6.1.0 / WinOF2 v. 1.60.16219.0**
- **TCP stack: cubic (Linux Red Hat 7.0 defaults)**
- **Switch: Mellanox Spectrum**
  - When QoS config used:
    - Two shared pools: lossy/lossless 3.5MB each
      - Egress alpha for lossy: 2
      - Ingress alpha for lossless: 2
    - Lossless ingress buffers of 94KB (Xoff 20KB)
    - Three traffic classes, with round-robin scheduling:
      - Lossy
      - Lossless
      - CNPs





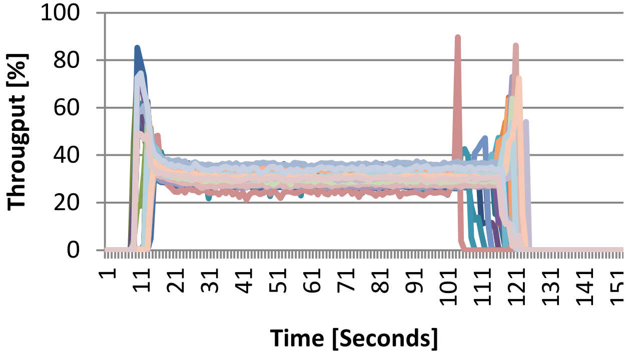
LARGE CLUSTER: ROCE VS TCP, ALL TO ALL

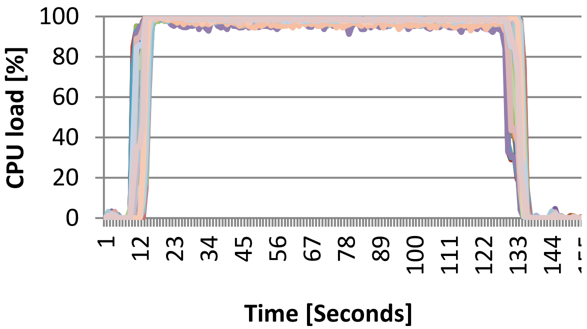
- **64 hosts all to all traffic**
- **4 QPs/flows per pair of hosts**
- **Lossy network**

Throughput per Sender

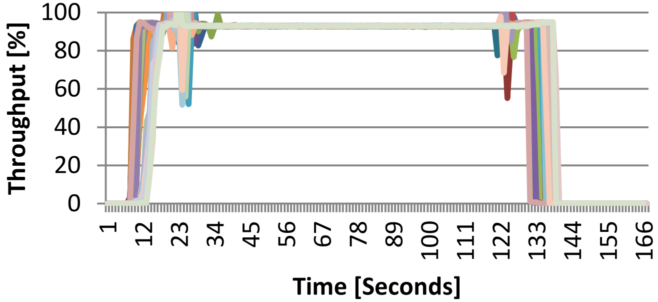
CPU Load

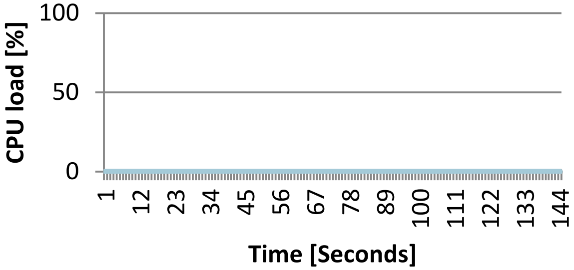
Experiment 1:
TCP





Experiment 2:
RoCE





Conclusion:

RoCE achieves almost twice larger throughput than TCP

TCP requires high CPU usage, while RoCE requires negligible CPU usage