

Daniele De Sensi<sup>1</sup>, Salvatore Di Girolamo<sup>1</sup>, Kim McMahon<sup>2</sup>,  
Duncan Roweth<sup>2</sup>, Torsten Hoefler<sup>1</sup>

<sup>1</sup>ETH Zurich, Switzerland  
<sup>2</sup>HPE Cray

# An In-Depth Analysis of the Slingshot Interconnect



**Hewlett Packard  
Enterprise**



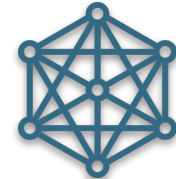
All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

**Adaptive Routing**



**Quality of Service**



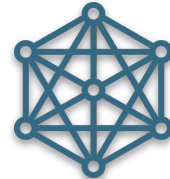
All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

Adaptive Routing



**Quality of Service**



All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

**Adaptive Routing**

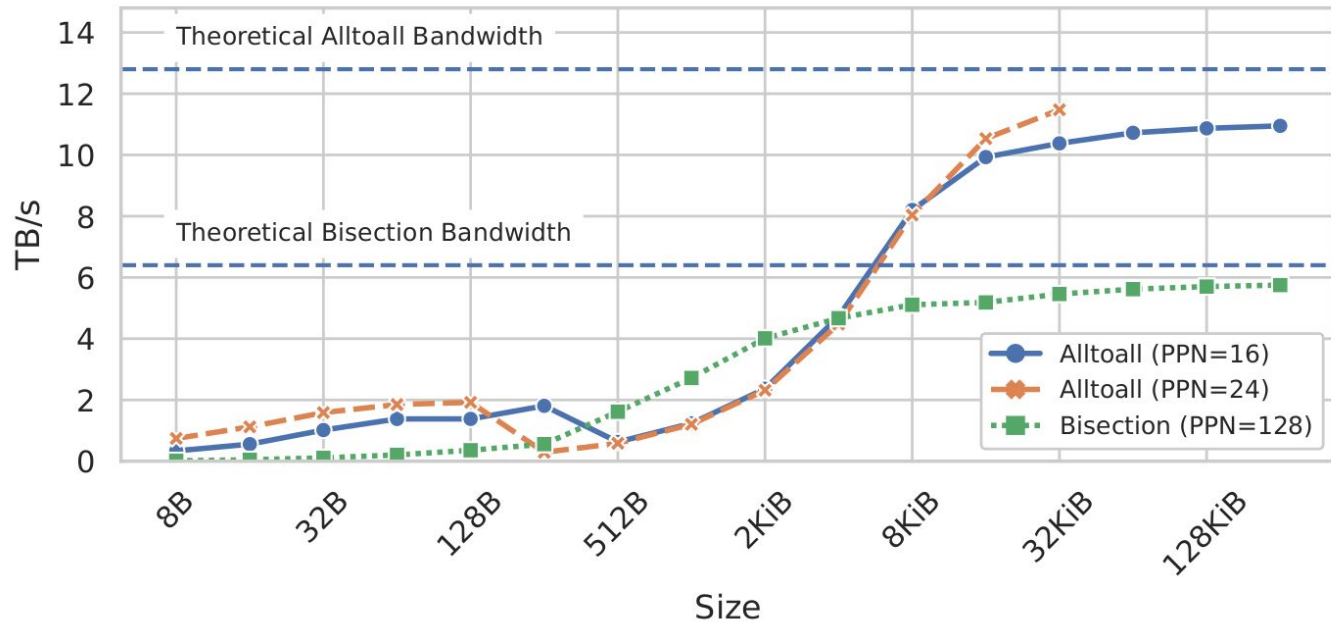


**Quality of Service**

# ETHERNET ENHANCEMENTS

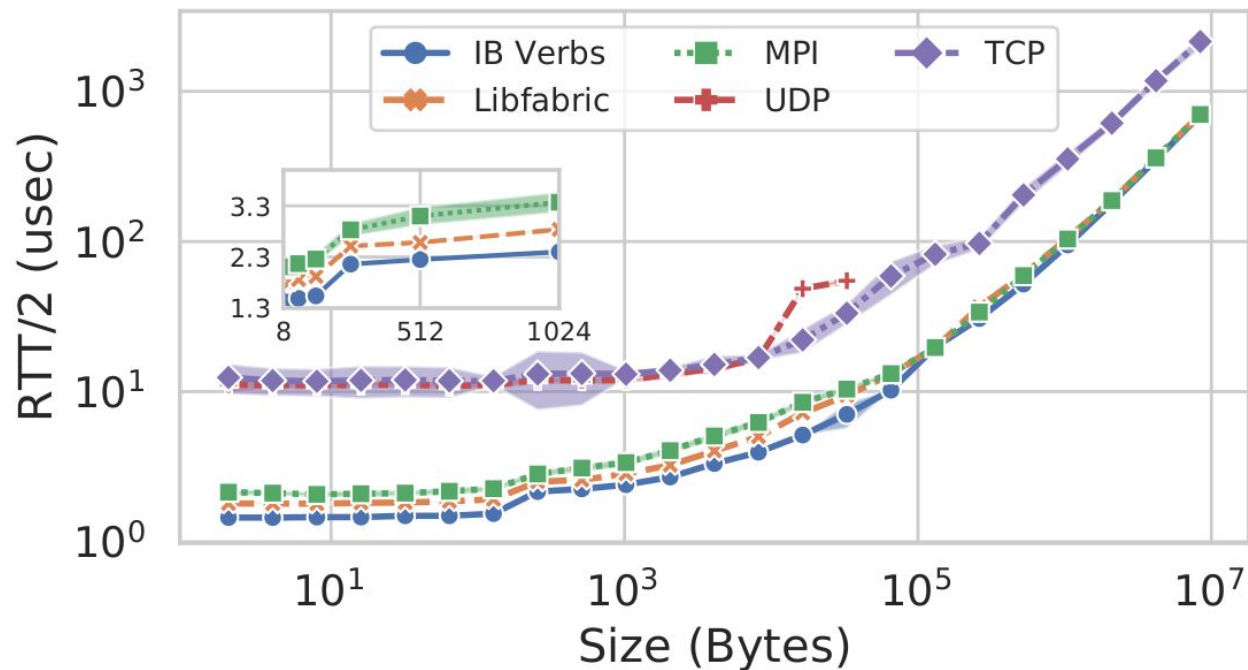
Can process both **standard** and **enhanced** Ethernet packets

1024 nodes



# SOFTWARE STACK

Standard **TCP/IP** stack or **libfabric**





All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

**Adaptive Routing**



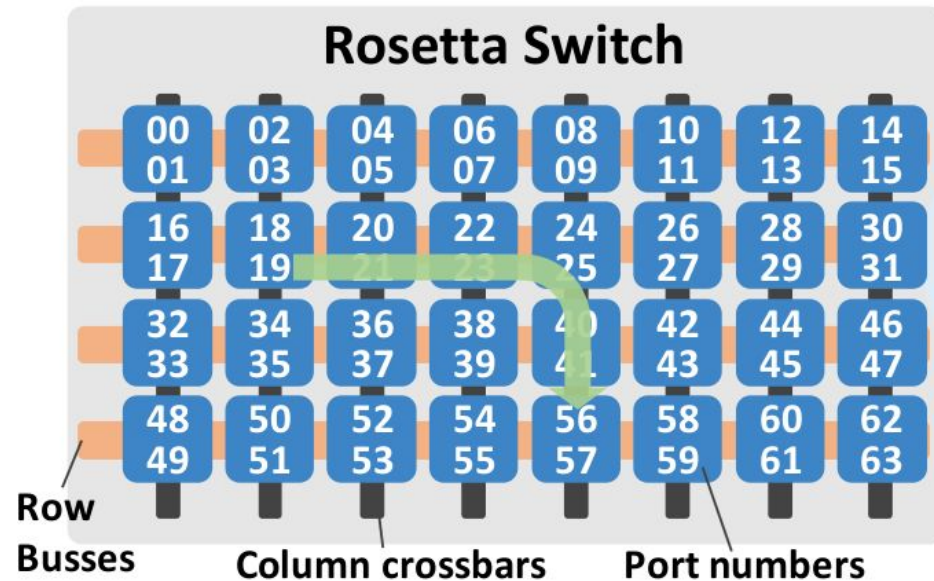
**Quality of Service**



# SWITCH - ROSETTA

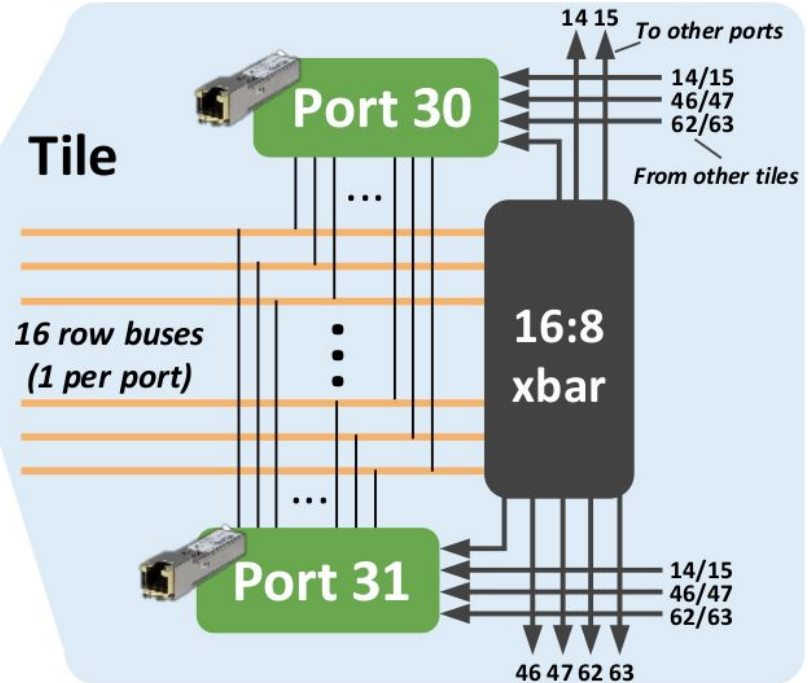
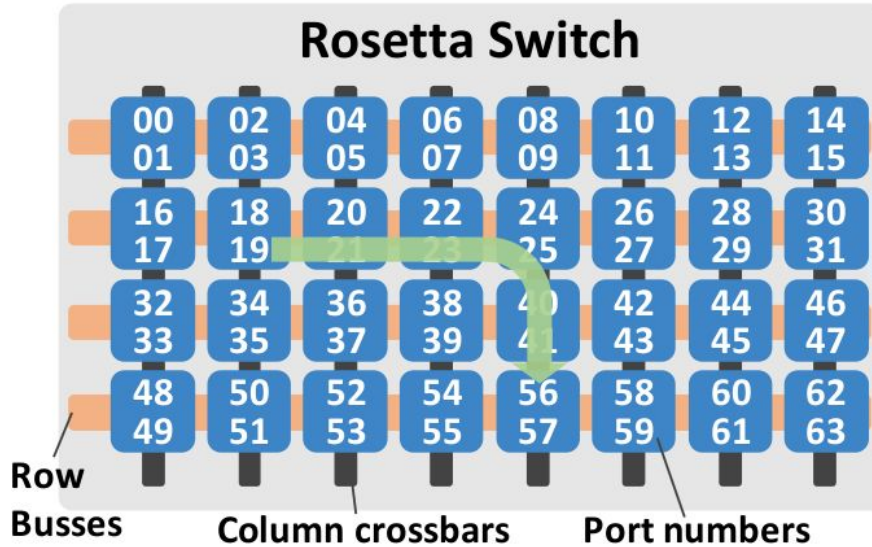
64 x 200Gb/s ports

32 tiles





## SWITCH - ROSETTA





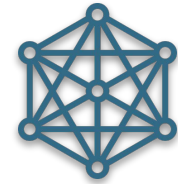
All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

**Adaptive Routing**

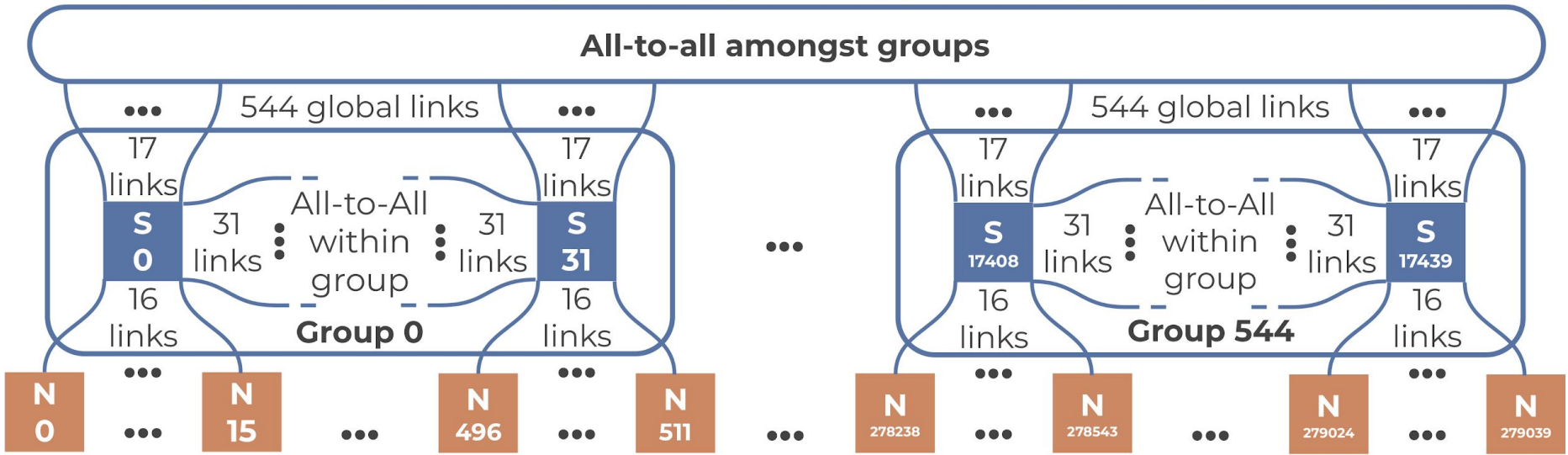


**Quality of Service**

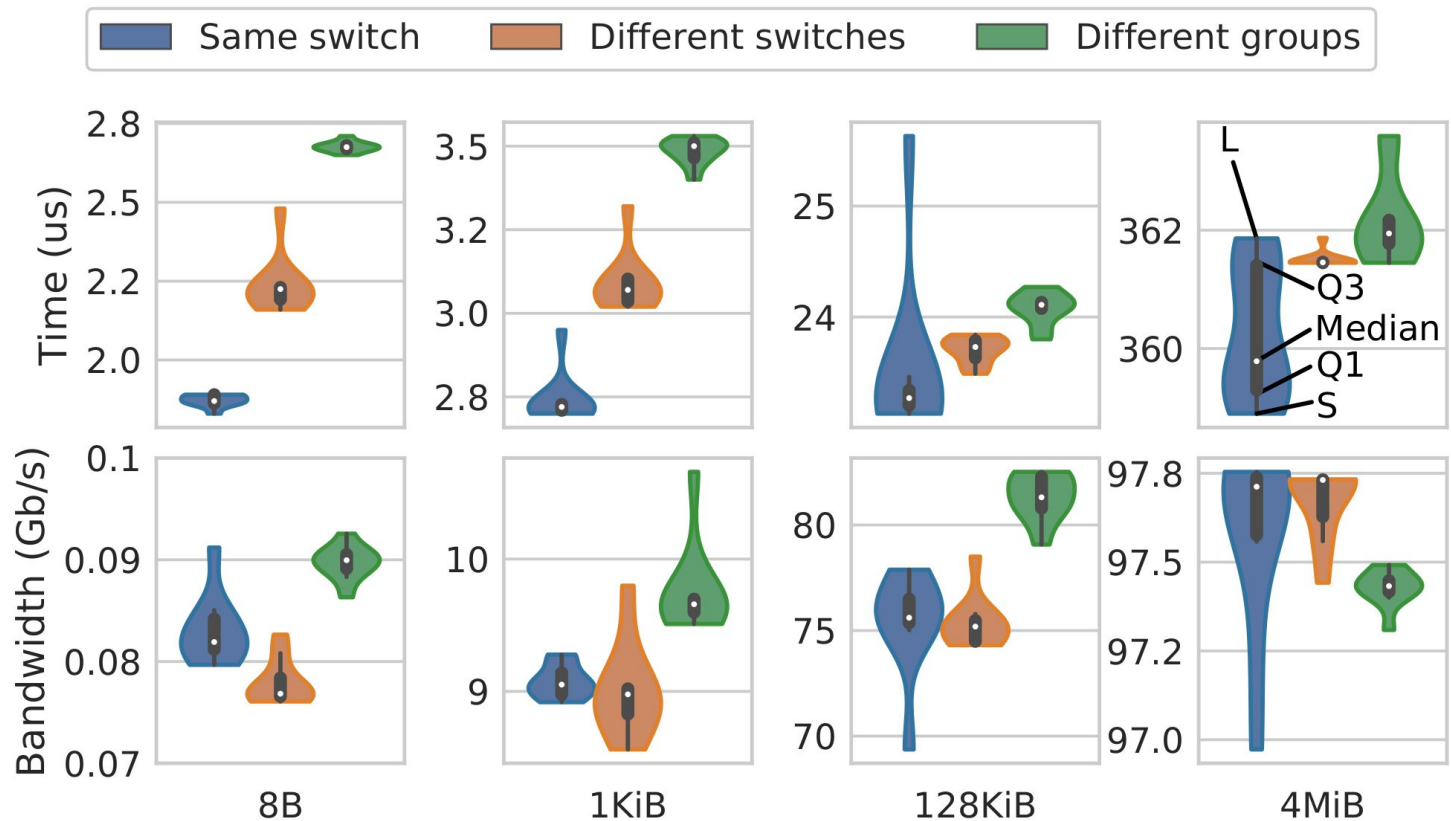
# SLINGSHOT TOPOLOGY

Switches can be connected into **arbitrary topologies**

**Dragonfly** is the default topology

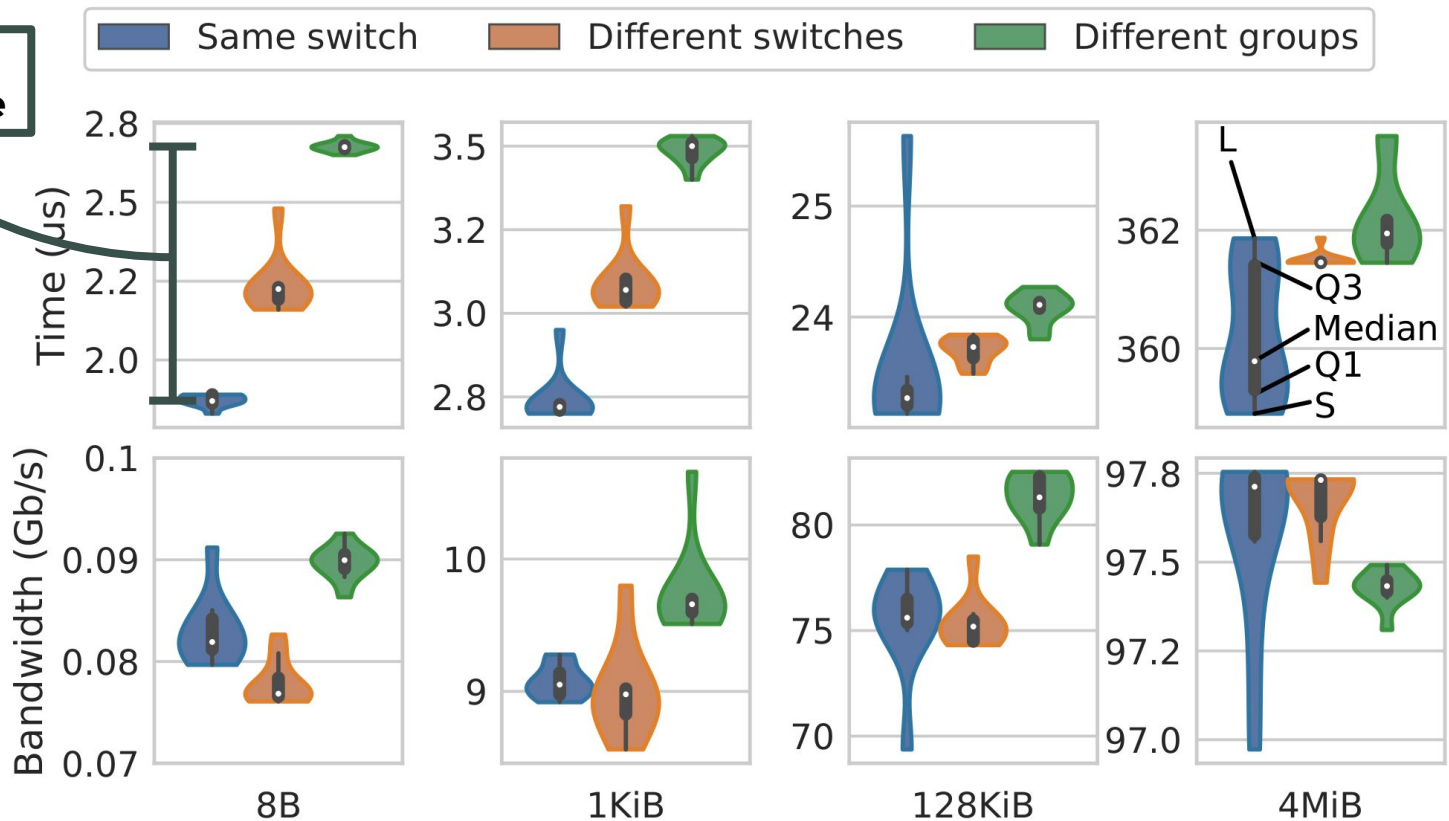


# SLINGSHOT TOPOLOGY - LATENCY & BANDWIDTH



## SLINGSHOT TOPOLOGY - LATENCY &amp; BANDWIDTH

~40%  
difference





All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

**Adaptive Routing**



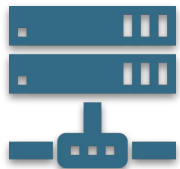
**Quality of Service**



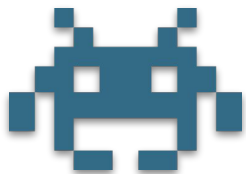
# CONGESTION CONTROL



ECN/QCN **hard to tune** and **slow to converge**



**Tracks** the traffic between **every pair of endpoints**



Slows down **offending traffic** only



**Improves** average and tail **latencies**

# CONGESTION CONTROL TESTS

Run two concurrent jobs: **victim** and **aggressor**



**Microbenchs.**

MILC

HPCG

LAMMPS

FFT



**silobench**

sphinx

xapian

img-dnn

resnet-proxy

**Tailbench**

**incast**

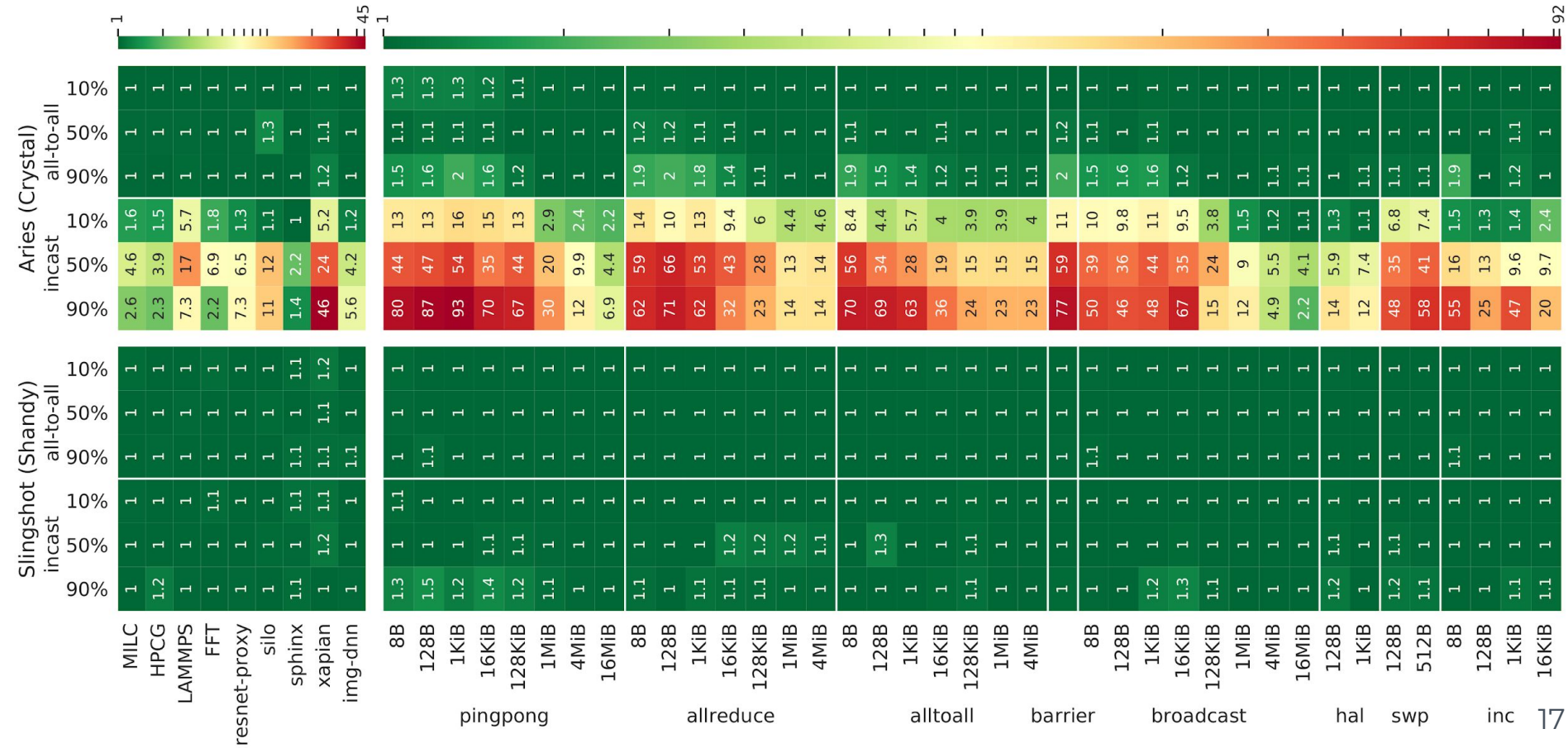
(intermediate congestion)

**all-to-all**

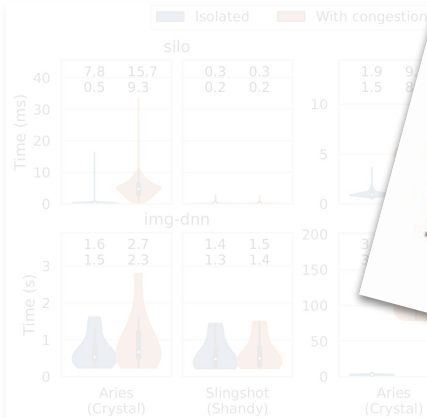
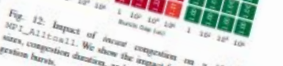
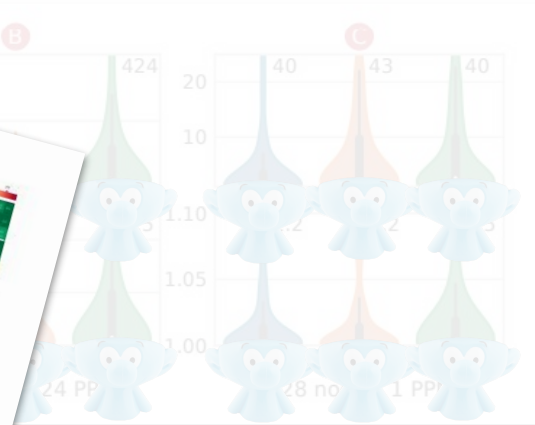
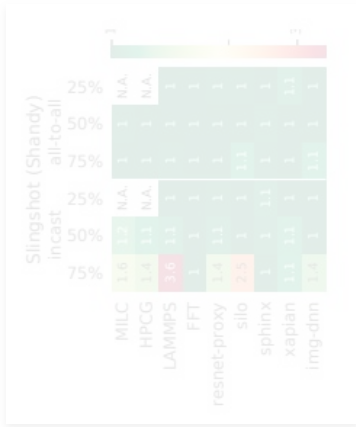
(intermediate congestion)



## CONGESTION IMPACT - 512 NODES



# CONGESTION IMPACT - ADDITIONAL ANALYSIS



**Fig. 10** impact gives us 154 (Figure 10) to 40 (Figure 10) fewer nodes by the lower generated traffic (aggressors have more bandwidth). On SHANDY, the same experience makes that the SLINGSHOT is less affected by congestion, even when varying the system size and the number of allocated nodes. The results of Figure 11 show the congestion impact on the applications when using all the 1024 nodes on SHANDY. We can observe that even in full system scale the congestion is controlled effectively through applications from congestion, with nodes are allocated to the aggressor composition, with and HPCG with a 29%/75% aggressor/victim ratio in making on a number of nodes which is a power of two.

We complete our analysis on the effect of congestion by analyzing the impact of heavy congestion SLINGSHOT in the previous experiments, generated by sending messages with a fixed size of 128KiB during the entire victim execution. To analyze the impact of heavy congestion, we evaluate a 128 byte `MD5.All11=1` microbenchmark (executed with an incast aggressor). This is one of the cases where we observed the highest congestion impact on SHANDY (see Figure 12).

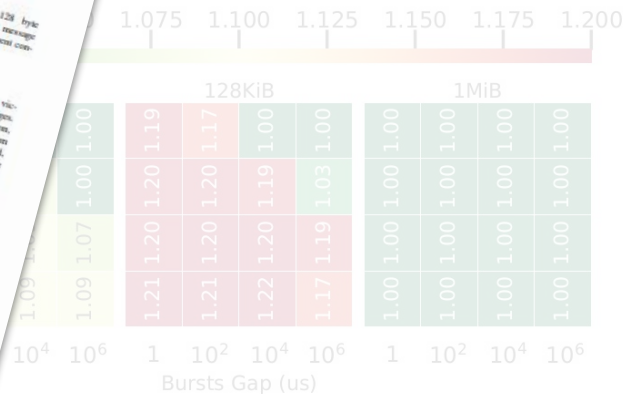
We run this test on all the MAJIC nodes, splitting them equally between aggressor and victim, with an unbalanced allocation strategy.

We report the results of this analysis in Figure 12. Each heatmap corresponds to a different message size for the incast aggressor. On each heatmap we report the congestion impact when varying the number of messages in a burst (y-axis, on the Y-axis) and the time between two subsequent congestion bursts (burst gap, on the x-axis). For example, the bottom-left element in the first heatmap represents the case where the aggressor sends  $10^5$  consecutive messages, each containing 8 bytes. Before sending the next burst of  $10^5$

messages, the aggressor will wait 1 millisecond. We observe that the incast aggressor does not affect the victim when sending low small messages or too large messages. Indeed, small messages do not generate enough congestion, while large messages do not generate enough congestion, fully lacks in and therefore the congestion control algorithm for medium size messages, some congestion builds up before the congestion control algorithm detects and reacts to it. We observe an increase in the congestion impact up to 1.21. However, as we shown in Figure 12, this is negligible when compared to what happens on other types of systems. We also observe no differences between subsequent bursts. We infer that both persistent congestion, and heavy and short-lived congestion.

**8. Traffic Classes**

We now evaluate the ability of SLINGSHOT to provide performance guarantees to jobs starting by using traffic classes. It is worth remarking that traffic classes and congestion control are orthogonal concepts. Traffic classes can be used to protect





All HPC traffic layered over **RoCEv2**

Efficient **software stack**



High-Radix **Switches**

Low-Diameter **Topology**



**Congestion Control**

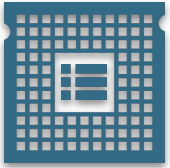
**Adaptive Routing**



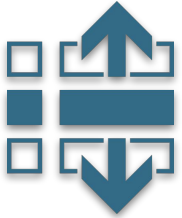
**Quality of Service**



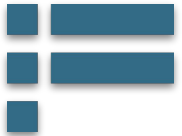
# QUALITY OF SERVICE



Each traffic class occupies **hardware resources** in the switches



**Tunable** priority, ordering, minimum/maximum bandwidth, ...



Jobs can be assigned to a small number **traffic classes**



Traffic class can be changed on a **per-packet** basis

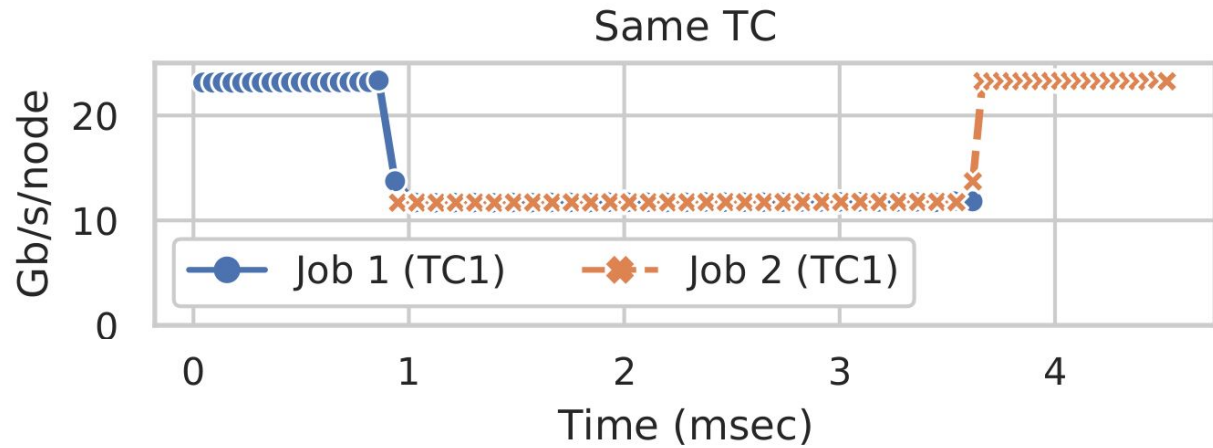
# QOS TESTS

25% bandwidth  
tapering

2 jobs running **bisection**  
**bandwidth** tests

**TC1**: 80% minimum  
bandwidth

**TC2**: 10% minimum  
bandwidth



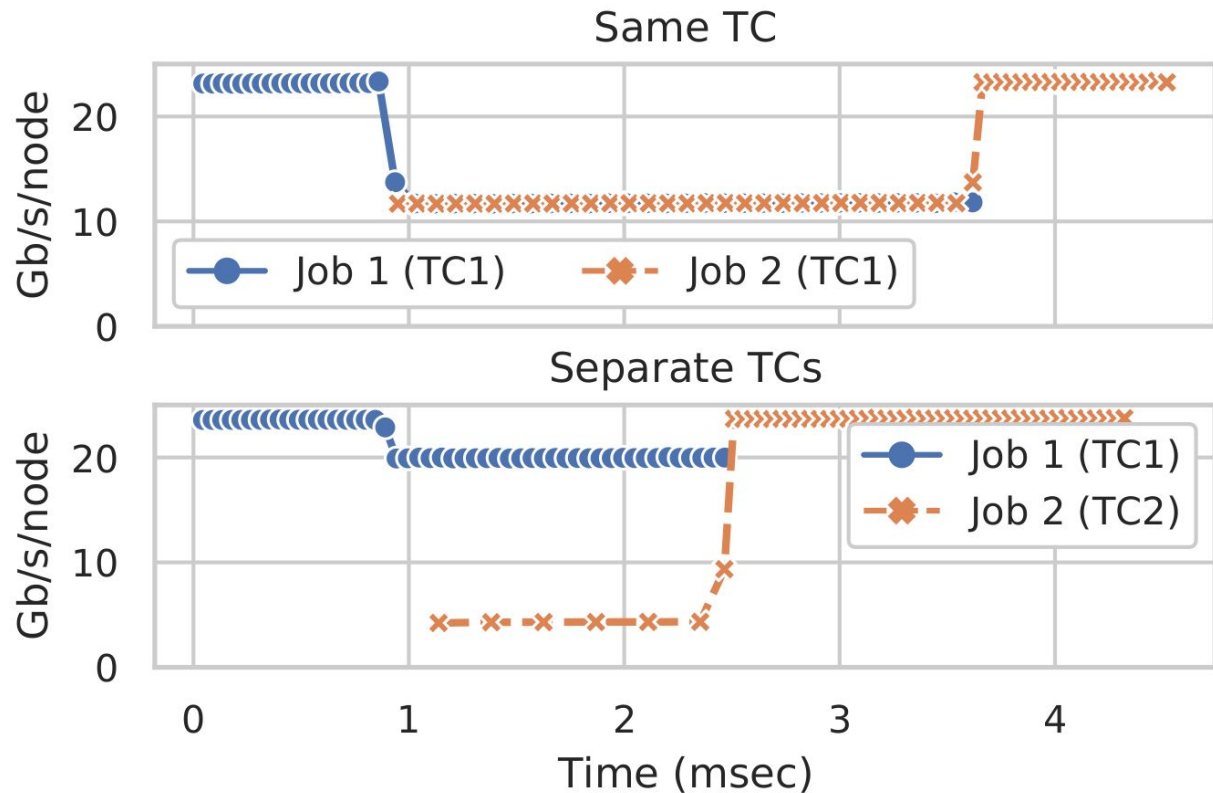
# QOS TESTS

25% bandwidth tapering

2 jobs running **bisection bandwidth** tests

**TC1**: 80% minimum bandwidth

**TC2**: 10% minimum bandwidth





# CONCLUSIONS

# CONCLUSIONS

ETH zürich spcl.inf.ethz.ch @spcl\_eth SPCL

- All HPC traffic layered over **RoCEv2**
- Efficient **software stack**
- High-Radix **Switches**
- Low-Diameter **Topology**
- Congestion Control**
- Adaptive Routing**
- Quality of Service**

3

