Approaching Incast Congestion with Multi-host Ethernet Controllers

Grzegorz Jereczek, Giovanna Lehmann-Miotto
CERN
Geneva, Switzerland
{grzegorz.jereczek|giovanna.lehmann}@cern.ch

David Malone
Maynooth University
Maynooth, Ireland
david.malone@nuim.ie

Miroslaw Walukiewicz
Intel Corporation
Gdansk, Poland
miroslaw.walukiewicz@intel.com

***Abstract*—The bursty many-to-one communication pattern, typical for data acquisition systems, but also present in datacenter networks, is particularly demanding for commodity TCP/IP and Ethernet technologies. We expand our study of building incast-resistant networks based on software switches running on commercial-off-the-shelf servers. In this paper we provide the estimates for costs and physical area required to build such a network. Our estimates indicate that our proposed design offers significant cost advantage over traditional solutions, but higher space utilisation. Next, we show how the latter can be improved with multi-host Ethernet controllers, as an alternative to typical network interface cards. This can also make software switching easier to adapt in datacenter as a solution for incast congestion. We confirm the capabilities for incast-avoidance by evaluating the performance of a reference platform.**

ATL-DAQ-PROC-2017-022
28 August 2017

I. Introduction

A many-to-one traffic pattern is particularly demanding for TCP/IP and Ethernet technologies and it is at the source of a network congestion problem. It leads to so-called *incast* congestion, which is perceived as a throughput collapse occurring when the number of servers sending data to clients increases past the ability of an Ethernet switch to buffer packets.

Incast congestion has been broadly studied in the context of datacenter networks (DCNs), but it is also present in data acquisition (DAQ) networks [1]. The latter are an important component of large-scale experiments, like the Large Hadron Collider (LHC) at CERN. A DAQ network collects the outputs from all the instruments to reconstruct a physical process.

In [2], [3] we presented software switches as an alternative approach to incast congestion in DAQ. We showed saturation at bidirectional bandwidth of 120 Gbps with a single software switch on real hardware under strong congestion, but also the possibility to scale to terabit networks. Aspects that have received little attention so far are costs and space consumption.

Here we show costs are better for software switching, however we see space usage can be an issue. It can be reduced using a new class of Ethernet devices — multi-host Ethernet controllers [4]. These devices combine a traditional network interface controller (NIC) with an Ethernet switch. However, since their performance characteristics are slightly different, we recheck that performance is still sufficient. Furthermore, we will explain how the shortcomings of software switching, limiting their application in DCNs, can be overcome with multi-host Ethernet controllers.

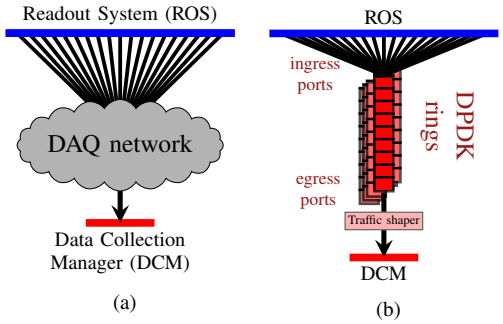


Fig. 1. Many-to-one pattern in a DAQ network (a). Data originating from the experiment are sent over a network to data collectors. Only one collector is drawn for clarity. The buffering mechanism providing a dedicated queue for each data collector is shown in (b).

We use the ATLAS experiment [5] as a case study. It is a general-purpose particle detector designed to study particle collisions at the LHC. Its DAQ network, having demanding traffic characteristics, is a good environment to evaluate candidate technologies. The conclusions apply, however, to other networks susceptible to incast.

This paper is structured as follows. Section II gives background information on the topics to be discussed here. The costs and the physical space for building large DAQ networks with software switches are discussed in Section III, whereas Section IV shows the performance of multi-host controllers under incast congestion. We conclude our work in Section V.

II. Background and Motivation

A. DAQ networks and many-to-one communication

In LHC nomenclature, particle collisions inside a detector are referred to as *events*. Their physical "fingerprints" are recorded by the detector, here being the ATLAS experiment [5], which is the data source for the DAQ system. Fragments corresponding to an event are usually buffered on multiple readout units, which constitute the Readout System (ROS). The DAQ network connects the ROS to a computer farm which retrieves, processes and filters the events (HLT — High Level Trigger). Every HLT node requests fragments of an event from the ROS. A number of independent worker processes run on a single filtering node, working on different events. However, there is only one process (DCM — Data Collection Manager)

per node that handles all the communication on behalf of the processing units.

The many-to-one traffic pattern, often accompanied by high burstiness [6], leads to *incast* congestion. It occurs when multiple nodes respond with data synchronously to a single requester, as pictured in Fig. 1a. It has been observed that these responses, although not very large on average, suffer from a high packet loss rate [7]. The scale of the problem increases with the number of responders, i.e. the size of the readout system in case of DAQ networks.

This incast pattern is particularly demanding for Ethernet switches and TCP/IP-based communication. The switch ports connected to data collectors are overcommitted by the data sent from many readout units. Switches with insufficient buffers must drop packets. The problem is systematic, so the built-in TCP retransmission mechanisms are not an effective solution [6]. They ensure reliability, but with a cost in performance. Another option is to provide enough buffering space within the network, to accommodate traffic spikes and bursts without the need for discarding data [8]. Large experiments, like those of the LHC, require particularly large packet buffers. For example, there are ~100 ROS nodes with 4x10GbE links and ~2000 HLT nodes with 1xGbE link to the ATLAS dataflow network. For this kind of network it was shown that a switch with a shared memory of 1.2 GB provides the best performance [9]. Only expensive, high-end switches or routers can be taken into consideration for use [6] as only those offer large enough memories. Furthermore, the current trend in the industry is to provide devices with smaller buffers [10] and not with deeper ones as the scaling needs of the experiments would require. The lack of buffering is the main obstacle in the use of a simple push architecture in DAQ [6], [9].

B. Software switches for incast-avoidance

In our prior work we showed that building a high-bandwidth lossless network based on software switches and a spine-leaf topology optimized for DAQ is feasible. It can be considered as a cheaper alternative to the current solutions based on expensive telecom-class routers with deep buffers or specialist technologies like InfiniBand.

We use Open vSwitch (OVS) [11], with a dedicated, throughput-oriented buffering algorithm [3] using DRAM memory. The overall idea is presented in Fig. 1b. For every data collector in the system there is a dedicated queue which is large enough to accommodate traffic bursts towards this DCM. This approach is flexible and scalable in terms of available buffers, which can be adjusted to system requirements. Also, traffic shaping can be performed on these queues in order to limit the outgoing flow, if there are any bottlenecks in the subsequent network stages.

Interconnected software switches can scale to terabit networks. This is achievable with the popular leaf-spine topology and flow distribution optimized for a given application. The ROS nodes and HLT racks can connect to a number of independent leaf-spine planes depending on the number of available uplinks and bandwidth requirements. An example of this topology with four planes and three *pods*, which contain six ROSes (R) and three filtering racks (H) each, is depicted in Fig. 2.

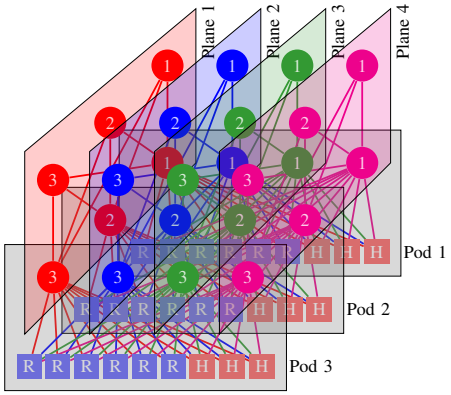


Fig. 2. A DAQ network in the parallel spine-leaf topology. Four parallel planes and three pods with ROS nodes (R) and filtering racks (H). Circles are the spine (upper layer) and leaf (lower layer) switches.

For more details on the design and performance of the proposed solution, please refer to [3].

C. Multi-host Ethernet controllers

An interesting alternative to using multiple network interfaces on commodity servers as software switches is replacing the former with a new class of Ethernet devices — *multi-host* Ethernet controllers. An example is Intel's FM10000 chip family [4], which offers up to 36 10GbE ports and a bandwidth of 200 Gbps over four PCIe gen3 interfaces. As such it provides the features of both a traditional hardware switch, with a fast ASIC offering large bandwidth for packet forwarding, and a flexible software switch, with large, but slower memory, that can be run in parallel.

The advantages in approaching incast congestion in this way are twofold. First, better port density can be reached. As we will see in Section III, an example server can be equipped with 40 10GbE ports and requires three rack units. In contrast, a device based on the FM10000 chip can fit 72 10GbE interfaces with four servers in four rack units [12]. Second, the possibility to offload only some of the packet processing tasks to the dedicated switching ASIC can make our approach to incast congestion more suitable for DCNs. In [2] we pointed to the potential limitations of software switches for general workloads, which normally do not apply to DAQ. Examples are forwarding performance for small packets or increased switching latency. If incast-sensitive flows were the only ones redirected to the software switch for buffering and traffic shaping, other flows would not be susceptible to those limitations. Furthermore, bufferbloat could be also eliminated.

D. Related work

The majority of approaches to incast focus on control of the packet injection rate so as not to overwhelm buffers in the network. The well-known TCP variant for datacenters is Datacenter TCP (DCTCP) [13]. However, it fails to avoid

incast if there are so many senders that the packets sent in the first round trip time overflow the buffers [13]. This situation is not uncommon for high-bandwidth low-latency DAQ networks under severe all-to-all incast congestion. For a full review of incast avoidance techniques see [3].

The systems at the LHC experiments follow their own strategies for congestion mitigation. The ATLAS experiment, as an example, relies on an application layer solution — traffic shaping, which is a simple credit-based system [9]. Despite the successful implementation of the traffic shaping algorithm in the large ATLAS DAQ network, it was shown in [9] that the buffering space available in a network plays a crucial role. Also, large buffers are still provided by costly telecom-class routers to avoid congestion in the network core. In our work we focus on providing a cost-effective alternative to them — software switches. An overview of work in software packet processing and performance proof is available in [2].

What is not well studied in the literature is a discussion on costs and space consumption of software switches when building terabit networks. Some indications for software routers are provided in [14]. This work considers, however, server platforms that could provide a bandwidth of 40 Gbps only. With this work we intend to provide an analysis and potential improvements at higher scale, using the ATLAS DAQ system as a use case.

III. Costs and physical space requirements

The total cost of building a DAQ network is one of the key factors when choosing a technology. As a matter of fact, this cost also includes the cost of developing better traffic shaping techniques in the given configuration, the cost of the inefficiencies introduced by network congestion, and the cost of operating a network. The latter includes also the cost of hiring or training experts for the technology of choice.

Here, we focus on the cost advantage when using a parallel leaf-spine topology with software switches instead of the traditional approach with high-end routers in the network core. Since both solutions are based on Ethernet, we will assume that the total network cost is the main differentiator.

A. Cost estimation

In case of the DAQ-adapted leaf-spine topology (see Fig. 2), subsequent leaves and planes are added incrementally (starting with a single plane with one leaf switch) in order to provide the required port count. The number of spine switches is predefined and determines the oversubscription factor at the leaves. 1:1 oversubscription is not required for full performance under specific traffic patterns. For example, in [3], we showed that only six spine switches are needed if there are seven HLT racks and 25 ROS nodes in each pod. This means that even an oversubscription of approximately 5:1 (32:6) is not going to reduce data acquisition performance.

The overall cost of the parallel leaf-spine network based on software switches is determined by the total number of servers used to build this network. We assume here that the same server is used for all the switches, which is the Supermicro SuperServer 6037R-TXRF system with ten PCIe 3.0 slots [15]. This server can be equipped, for example, with two E5-2697 v2 Intel Twelve-Core Xeon 2.7 GHz processors and ten quad-port 10GbE network adapters [4]. In this configuration it should be possible to provide a total of 40 10GbE ports and a bandwidth of 400 Gbps with a single software switch. The prices used are catalogue prices from [16], [17]. The total cost of a single server-switch is approximately £12 000. Unfortunately, we do not consider the costs of the alternative version with multi-host controllers as they are not known at the moment. We estimate, however, that these prices are comparable or even lower by comparing the costs of the network chips only. The cost of a single quad-port 10GbE Intel XL710 chip is $159.99 [4], whereas the FM10840 36-port chip with 200 Gbps bandwidth over PCIe costs $577.80 [4]. If both approaches are to provide similar bandwidth over PCIe with a single server switch (400 Gbps), ten quad-port chips at the cost of $1599.9 or two FM10840 at the cost of $1155.6 are needed.

The required number of servers $N_{servers}$ to provide N_{ports} ports to connect to the network is given by the total number of leaf switches $N_{L_{total}}$ and spine switches $N_{S_{total}}$. The number of servers is therefore given by

$$N_{servers} = N_{L_{total}} + N_{S_{total}} \,. \tag{1}$$

The number of spine switches N_S in each plane is predetermined and defines the oversubscription factor [18]. Another predefined value is the number of network ports available on every server-switch $N_{ports_{server}}$.

In order to fulfil the requirements a number of parallel leaf-spine planes is required. Each plane can offer no more than

$$N_{ports_P} = N_{pods} \cdot N_{ports_{pod}}$$

ports to connect the end-nodes. N_{pods} is the maximum number of pods per plane and $N_{ports_{pod}}$ is the number of available ports in each pod. Since each leaf switch in a plane is connected to every spine switch in this plane (see Fig. 2), it is given by

$$N_{ports_{pod}} = N_{ports_{server}} - N_S \,.$$

A single leaf switch connects also to a single pod. The maximum number of pods in a plane is therefore equal to the maximum number of leaf switches in a plane N_L. Since each spine switch has to be connected to every leaf switch in a plane, the maximum number of leaf switches in a plane N_L is determined by the number of ports available on a single switch. The maximum number of pods in a plane is therefore

$$N_{pods} = N_L = N_{ports_{server}}.$$

The number of fully filled planes is then calculated with

$$\begin{aligned} N_{P_{full}} &= \left\lfloor \frac{N_{ports}}{N_{ports_P}} \right\rfloor \\ &= \left\lfloor \frac{N_{ports}}{N_{pods} \cdot N_{ports_{pod}}} \right\rfloor \\ &= \left\lfloor \frac{N_{ports}}{N_{ports_{server}} \cdot (N_{ports_{server}} - N_S)} \right\rfloor \,. \end{aligned} \tag{2}$$

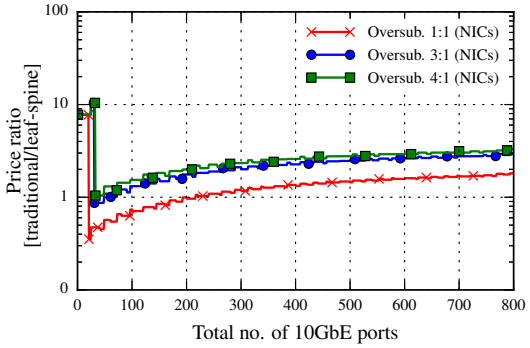


Fig. 3. Costs comparison for building a DAQ network with the traditional approach (large routers in the core) and the parallel leaf-spine topology (software switches) with different oversubscription factors at the leaf switches. The cost of cabling is not included.

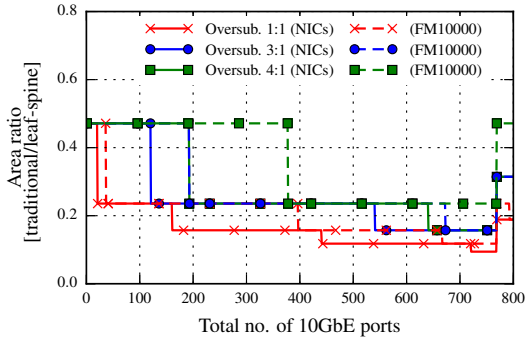


Fig. 4. Comparison of the total area required to build a DAQ network with the traditional approach and the parallel leaf-spine topology (software switches) with different oversubscription factors at the leaf switches.

The remaining ports are provided by a subset of $N_{L_{rem}}$ leaf switches only in another plane. The number of spine switches remains the same as the predefined value N_S. The number of leaf switches is equal to the number of pods required to provide the remaining ports $N_{P_{rem}}$ and can be calculated with

$$\begin{aligned} N_{L_{rem}} &= \left\lceil \frac{N_{ports_{rem}}}{N_{ports_{pod}}} \right\rceil \\ &= \left\lceil \frac{N_{ports_{rem}}}{N_{ports_{server}} - N_S} \right\rceil , \end{aligned} \tag{3}$$

where the number of remaining ports is given by

$$\begin{aligned} N_{ports_{rem}} =& N_{ports} - N_{P_{full}} \cdot N_{ports_P} = N_{ports} - \\ & N_{P_{full}} \cdot N_{ports_{server}} \cdot (N_{ports_{server}} - N_S) . \end{aligned} \tag{4}$$

Equation (1) can be now re-expressed as

$$\begin{aligned} N_{servers} &= N_{L_{total}} + N_{S_{total}} \\ &= N_{L_{rem}} + N_S + N_{P_{full}} (N_L + N_S) \\ &= N_{L_{rem}} + N_S + N_{P_{full}} (N_{ports_{server}} + N_S) \\ &= N_{L_{rem}} + N_S \left(N_{P_{full}} + 1\right) + N_{ports_{server}} , \end{aligned} \tag{5}$$

considering the servers used to build the planes with the maximum number of pods and the plane providing the remaining pods. In the end, equations (2) to (5) are sufficient to calculate the total number of servers required to build parallel leaf-spine fabrics offering N_{ports} ports with N_S spine switches in each plane and $N_{ports_{server}}$ ports on each server-switch.

1) The reference solution: The current ATLAS DAQ network architecture [9] is used as a reference solution. We assume that at least one Brocade MLXe 32 [19] chassis is used. Subsequent chassis and 10GbE modules (24-port) are added incrementally when increasing the total number of available network ports. The ROSes and HLT racks are connected directly to the router, so the oversubscription cannot be altered from 1:1. The catalogue prices[1] are used [20].

[1]The cost of the chassis is £62 808, the single 24-port module is £33 058, and the single switch fabric module is £4709.

2) Comparison: Fig. 3 shows that the DAQ network of the size of the ATLAS experiment using software switches can be three times cheaper than the current solution in the range from 100 to 800 ports. The advantage grows when increasing the oversubscription factor, which can be used to some extent without any performance degradation in the DAQ use case, as explained in Section III-A. For lower port counts, the ratio is larger due to the high cost of the reference router chassis.

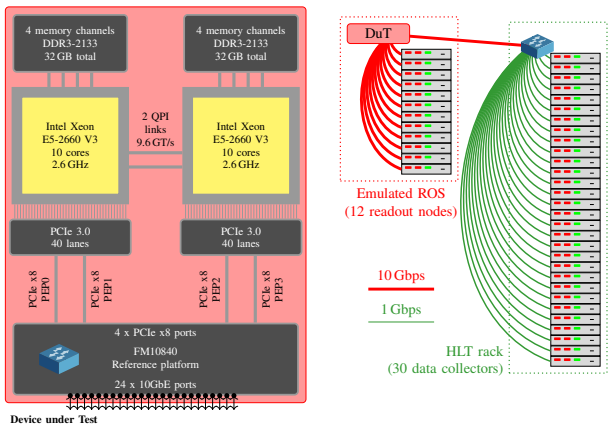
The case when the reference solution is cheaper is for an oversubscription of 1:1 and lower port counts. In few configurations, the traditional approach can be cheaper. This is caused by the fact that many spine switches are used for a small number of pods. This is, however, not a realistic configuration as the network would remain highly underutilized.

B. Physical space

The physical space required to house the entire set of network hardware is also an important factor when designing a DAQ or a datacenter network. This becomes particularly important in space-constrained areas, like at the LHC experiments. For this reason, we provide an estimate on the physical dimensions of a DAQ network based on software switches.

Following the same methodology as in the previous section the space requirements can also be discussed. Fig. 4 shows the ratio of the physical area required by the reference solution to the area required by the proposed topology. For the former, the area is given by the dimensions of the Brocade MLXe 32 chassis [19]. For the latter, the dimensions of racks required to fit all the server-switches determine the total physical area needed (see Section II-C). We use a reference rack from [21].

In this case, the traditional approach is advantageous. For higher oversubscription factors, server-switches in the leaf-spine fabrics require two to four times more physical area. Although the ratio remains in the single-digit range, this aspect has to be considered when designing a DAQ system. Increased space usage can increase the costs in the end. But Fig. 4 also shows a potential improvement considering the multi-host Ethernet controllers introduced in Section II-C. By replacing NICs with a device based on the FM10000 chip family [12] up to twice as many ports can fit the same physical



(a) (b)

Fig. 5. Block diagram of the DuT (a) and the evaluation setup (b).

space. Although the density offered by traditional network designs is still not reached, space utilisation is improved. For this reason, such devices could be considered an alternative to traditional NICs in the proposed design. However, since their performance characteristics are slightly different, in the following section we will confirm that the software switch performance demonstrated in our prior work is not impacted.

IV. Multi-host Ethernet controllers for incast-avoidance

As we explained in Section II-C, the limitations of software switches could be largely avoided by offloading incast-sensitive flows to the software switches themselves. In this section we evaluate the performance and discuss whether these devices are indeed suitable for incast-avoidance.

A. Device under test

The device under test (DuT), see Fig. 5a is a reference platform, being an Ethernet switch (FM10840 chip [4]) with 24 10GbE ports and four PCIe ports (*PEPs*). PEPs are connected with PCIe cables to four PCIe x8 gen3 slots of a COTS server. This arrangement is used in the following configurations.

*a) Traditional Ethernet switch (*PEPs:0*):* Only Ethernet ports are used and switching is performed by the FM10000 chip. PEP ports are not used. This configuration is used as a baseline (typical Ethernet switch). Congestion avoidance is implemented with a static TCP congestion window [1], the IEEE 802.3x PAUSE frame mechanism [22] at the switch and NICs, or with application layer traffic shaping [9], which limits the number of parallel data requests. Whenever possible, the algorithms are tuned to reach maximum performance.

*b) Traditional Ethernet switch with extra software queues (*PEPs:1|2|3|4*):* Those queues, which we refer to as *daqrings* [3], are used to accommodate many-to-one bursts. Here, packets incoming on Ethernet ports that match some predefined rules (like source/destination MAC/IP addresses, TCP ports, etc.) are redirected to the PEP ports (one, two, three or four PCIe ports can be used) and handled by a software switch running on the COTS server (Open vSwitch 2.4.0 with our custom patch [23]). This software switch is programmed to enqueue the packets in those daqrings in order to avoid incast congestion. On the other end, the packets are dequeued from daqrings (with optional traffic shaping) and sent back to the switching ASIC over the same PEP ports. There, they are switched to appropriate output Ethernet ports. This approach provides large buffers in the DRAM memory of the host and traffic shaping on a per daqring basis, so that a DAQ network is optimized and can provide lossless operation.

B. Traffic generation

For traffic generation we use an emulated ATLAS DAQ/HLT system in a scaled-down configuration, see Fig. 5b. Each of the twelve emulated ROS nodes provides 24 dummy event data fragments of 1 kB. These nodes are connected with a single unshared 10 Gbps link to the DuT. There are also 30 DCMs connected with a single unshared 1 Gbps link to a Top of Rack (ToR) switch. This switch connects to the DuT with a single 10 Gbps link. On each DCM node there are also twelve independent processing units, which request events from ROS systems in parallel. A single unit does not request another event before it receives all fragments of the previous one from all available ROSes. No event processing is performed so the DuT can be analysed in isolation from other factors.

The data collection latency of a single event is understood as the timespan between sending the first request to the ROS and receiving the last fragment from the ROS. Goodput (sustained load) is calculated as the raw event data bandwidth excluding all protocol overheads (Ethernet, TCP/IP, ATLAS protocol). Latency and goodput are averaged over approximately 120 s.

Unless otherwise stated, we disable dynamic TCP congestion control in all ROS hosts and instead use a static sender congestion window. The window is set to a very large value so that each ROS is not rate limited by TCP. This allows us to evaluate the performance of various approaches to incast without the influence of congestion control. This is also the best DAQ scenario, in which available data are pushed on the wire and the readout system can be simplistic. All tests are performed with an MTU of 1500 B.

C. Evaluation

In [3] we showed that optimized software switches can provide 15 % higher goodput than regular switches. Here, all solutions are expected to provide similar performance because of lower incast congestion. We aim to verify our solution remains valid on the multi-host controller.

The results are presented in Fig. 6. We compare ATLAS traffic shaping (PEPs:0), static TCP (PEPs:0, the sender congestion window is tuned to maximize the performance), PAUSE frames (PEPs:0), and daqrings with traffic shaping (PEPs:3), each in their optimum configuration. All approaches mitigate incast congestion and sustain the requested load (Fig. 6a). For daqrings, a rate limit of 344 Mbps is optimal to reach full load. For 1.38 Gbps the highest loads cannot be reached as there are too many overlapping traffic bursts and

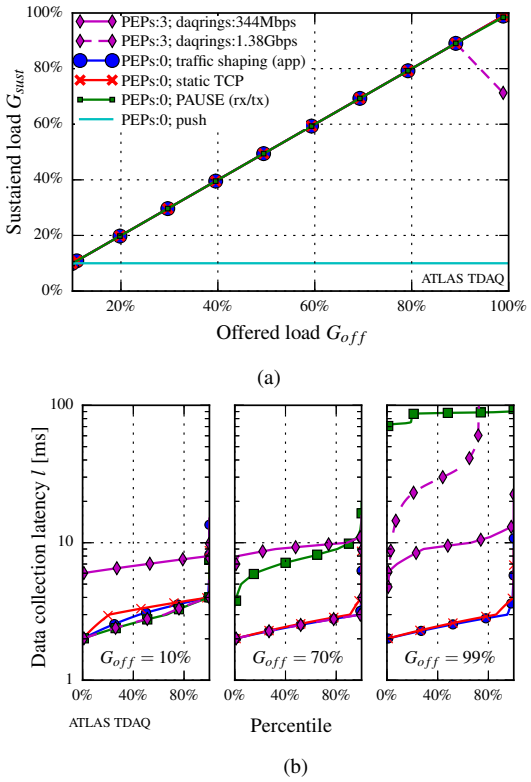


Fig. 6. Performance comparison of various incast avoidance techniques. Sustained load in (a) and the distribution of data collection latency in (b).

switch buffers are overflow. We also plotted the case without any congestion or flow control, using a simple *push* of all data over the network, in which it cannot operate at more than 10 % of the available bandwidth because of strong incast congestion.

Latency and jitter (Fig. 6b) are significantly increased for PAUSE frames because of the head-of-line blocking effect [24]. Higher latency is also true for daqrings, but with low jitter (flatter curve). Furthermore, at lower loads larger bursts can be tolerated by the network, so daqrings can be tuned and the rate limit can be increased to minimize latency. With the rate limit of 1.38 Gbps the lowest latency and jitter are achieved at a load of 70 %. In this configuration, daqrings provide lowest latency and jitter. At full load, a rate limit of 344 Mbps is required to sustain the load, but here the latency is higher than in case of traffic shaping and static TCP. With a rate limit of 1.38 Gbps all bursts can no longer be absorbed by the network, the configuration is not optimal and packet retransmissions occur, which results in increased data collection latencies.

V. Conclusion

In this paper we showed how incast-resistant networks based on application-optimized software switches can be built more cheaply than traditional approaches. Thus, it is not only the lossless operation under incast congestion, but also cost advantage, that are offered by software switches.

This comes, however, at the cost of higher space requirements. This can be mitigated by using multi-host Ethernet controllers instead of traditional network interfaces, which can improve space utilisation by a factor of two. We also pointed out that these devices can be potentially used as an alternative to other incast-avoidance techniques in datacenters, where limitations of typical software switches have stronger consequences than in data acquisition.

Finally, we performed an initial evaluation of a multi-host controller, using a reference platform. We confirmed that similar performance to other incast-avoidance techniques can be achieved. Thus, our evaluation showed that incast-avoidance in DAQ and DCN can be a new application area for multi-host Ethernet devices. Further study of the performance under strong incast congestion in various configurations is required in order to show the possible advantages in performance of this approach over other solutions. In the context of DCN, in particular, comparison with state-of-the-art TCP congestion control, DCTCP [13], is desirable.

Acknowledgment

This research project has been supported by a Marie Curie Early European Industrial Doctorates Fellowship of the European Community's Seventh Framework Programme under contract number (PITN-GA-2012-316596-ICE-DIP).

The authors thank Mikel Eukeni Pozo Astigarraga and Espen Blikra for help in preparing the evaluation testbed.

References

[1] G. Jereczek *et al.*, "Analogues between tuning TCP for data acquisition and datacenter networks," in *Proc. IEEE ICC*, 2015.
[2] G. Jereczek *et al.*, "A lossless switch for data acquisition networks," in *Proc. IEEE LCN*, 2015.
[3] G. Jereczek *et al.*, "A lossless network for data acquisition," *IEEE Transactions on Nuclear Science*, 2017.
[4] Intel. [Online]. Available: http://www.intel.com/
[5] The ATLAS Collaboration, "The ATLAS Experiment at the CERN Large Hadron Collider," *Journal of Instrumentation*, vol. 3, no. 08, 2008.
[6] N. Neufeld, "LHC trigger & DAQ — an introductory overview," in *Proc. IEEE-NPSS Real Time Conference*, 2012.
[7] S. Varma, *Internet Congestion Control*. Morgan Kaufmann, 2015.
[8] A. Phanishayee *et al.*, "Measurement and analysis of TCP throughput collapse in cluster-based storage systems." in *FAST*, vol. 8, 2008.
[9] T. Colombo *et al.*, "Data-flow performance optimisation on unreliable networks: the ATLAS data-acquisition case," *Journal of Physics: Conference Series*, vol. 608, no. 1, 2015.
[10] A. Vishwanath *et al.*, "Perspectives on router buffer sizing: Recent results and open problems," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 2, 2009.
[11] Open vSwitch. [Online]. Available: http://openvswitch.org/
[12] Adlink CSA-7400. [Online]. Available: http://www.adlinktech.com/
[13] M. Alizadeh *et al.*, "Data Center TCP (DCTCP)," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 40, no. 4, 2010.
[14] M. Dobrescu *et al.*, "Routebricks: Exploiting parallelism to scale software routers," in *Proc. ACM SOSP*, 2009.
[15] Supermicro. [Online]. Available: https://www.supermicro.com
[16] Broadberry. [Online]. Available: https://www.broadberry.co.uk
[17] Insight Direct UK. [Online]. Available: http://www.uk.insight.com/
[18] M. Al-Fares *et al.*, "A scalable, commodity data center network architecture," *ACM SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 4, 2008.
[19] Brocade MLX series. [Online]. Available: http://www.brocade.com/
[20] MCi Online. [Online]. Available: http://shop.mcidiventi.co.uk/
[21] 42U rack dimensions, cabinet size, & specifications. [Online]. Available: http://www.42u.com/42U-cabinets.htm
[22] "IEEE standard for Ethernet," *IEEE Std 802.3*, 2012.
[23] daq-software-switching — GitHub repository. [Online]. Available: https://github.com/gjerecze/daq-software-switching
[24] Y. Ren *et al.*, "A survey on TCP incast in data center networks," *International Journal of Communication Systems*, 2012.