Shape-shifting Elephants: Multi-modal Transport for Integrated Research Infrastructure

Nik Sultana
Illinois Institute of Technology
USA

Yatish Kumar
ESnet
USA

Chin Guok
ESnet
USA

James B Kowalkowski
Fermilab
USA

Michael H L S Wang
Fermilab
USA

ABSTRACT

Data Acquisition (DAQ) workloads form an important class of scientific network traffic that by its nature (1) flows across different research infrastructure, including remote instruments and supercomputer clusters, (2) has ever-increasing throughput demands, and (3) has ever-increasing integration demands—for example, observations at one instrument could trigger a reconfiguration of another instrument. Today's DAQ transfers rely on UDP and (heavily tuned) TCP, but this is driven by convenience rather than suitability. The mismatch between Internet transport protocols and scientific workloads becomes more stark with the steady increase in link capacities, data generation, and integration across research infrastructure.

This position paper argues the importance of developing specialized transport protocols for DAQ workloads. It proposes a new transport feature for this kind of elephant flow: *multi-modality* involves the network actively configuring the transport protocol to change how DAQ flows are processed across different underlying networks that connect scientific research infrastructure. Multi-modality is a layering violation that is proposed as a pragmatic technique for DAQ transport protocol design. It takes advantage of programmable network hardware that is increasingly being deployed in scientific research infrastructure. The paper presents an initial evaluation through a pilot study that includes a Tofino2 switch and Alveo FPGA cards, and using data from a particle detector.

CCS CONCEPTS

• **Networks → Transport protocols**; • **Applied computing → Physical sciences and engineering**.

KEYWORDS

DAQ Workloads,Transport Protocol,Scientific Networking

ACM Reference Format:
Nik Sultana, Yatish Kumar, Chin Guok, James B Kowalkowski, and Michael H L S Wang. 2024. Shape-shifting Elephants: Multi-modal Transport for Integrated Research Infrastructure. In *The 23rd ACM Workshop on Hot Topics in Networks (HOTNETS '24), November 18–19, 2024, Irvine, CA, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3696348.3696855

1 INTRODUCTION

As instruments become more precise, they produce more data. We can only process a fraction of data from large instruments. For example, the Large Hadron Collider (LHC) *generates* data at more than 600Tbps but around 40Tbps is currently *acquired* [52]—that is, read out of the instrument. Table 1 lists examples of recent and under-development experiments, and their *data acquisition* (DAQ) rates.

Large instruments—such as those in Table 1—have a *DAQ network* that connects the instruments' sensors to a small downstream processing facility. From there, data is transferred over other networks to reach large-scale processing facilities. Often, the DAQ network is an Ethernet built using commodity equipment. Traffic consists of elephant flows with a regular shape (size and arrival rate)—detailed further in §2.

In addition to producing more data, research infrastructure is becoming more *integrated* [47]—we will see the example where a neutrino detector in South Dakota instructs a state-of-the-art optical telescope in Chile on the arrival direction of photons from a supernova burst. How we transport DAQ data is key to enabling further integration, as explained next.

Fig. 1 sketches the flow of instrument-generated data across key stages. At the DAQ stage ❶, preprocessing involves identifying interesting data in the DAQ stream—such as evidence of particle collisions. Then a time window of such readings from the instrument is sent over a WAN ❷ to downstream cloud or high-performance computing (HPC) clusters ❸—such as national labs—where the bulk of analysis happens,

Experiment	DAQ rate
CMS L1 Trigger	63Tbps [77]
DUNE	120Tbps [68]
ECCE detector	100Tbps [13]
Mu2e	160Gbps [29]
Vera Rubin	400Gbps [38]

Table 1: *Data Acquisition* (DAQ) rates for examples of large instruments. CMS, Mu2e and ECCE are high-energy physics experiments that study observations from collisions generated artificially by particle accelerators; Vera Rubin is a telescope; DUNE is both driven by a particle accelerator and by natural neutrino sources (including our sun, cosmic rays, and from supernova bursts).

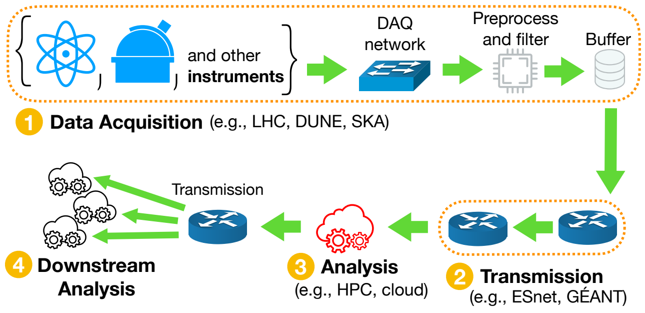
and other instruments
DAQ network
Preprocess and filter
Buffer
1 Data Acquisition (e.g., LHC, DUNE, SKA)
Transmission
4 Downstream Analysis
3 Analysis (e.g., HPC, cloud)
2 Transmission (e.g., ESnet, GÉANT)

Figure 1: Dataflow for large-scale instruments: ❶ DAQ data crosses a ❷ WAN to reach ❸ large-scale storage and processing facilities—often consisting of supercomputers and specialized clusters. ❹ Products derived from this data are made available over a WAN to other researchers.

and for permanent storage. From there, data reaches downstream sites ❹ such as universities, for local caching and analysis. Sometimes, data must go straight from ❶ to ❹ for rapid coordination between scientists—such as the *multi-domain alerts* from a telescope to astronomers around the world [57].

Today's approach to transporting DAQ data evolved by using existing Internet protocols, particularly TCP (§4). This evolution was driven by convenience at the cost of complexity and performance overhead. This overhead accrues from TCP's general design and the termination of streams across ❶-❹. To mitigate overhead, TCP is heavily tuned to support high data rates [22, 28, 43, 73], which introduces complexity.

But a TCP-based transport is fundamentally unsuitable for DAQ. TCP's bytestream abstraction, mechanisms for capacity discovery and congestion avoidance, and abstraction of the underlying network create an impedance mismatch with DAQ traffic and the type of networks on which that traffic flows: **(1)** DAQ traffic consists of time-stamped messages with well-defined boundaries, making an ordered bytestream unsuitable; **(2)** end-to-end is the wrong processing model for this network environment and workload, where we can use a hop-by-hop model to reduce the retransmission latency, reduce buffering, and improve flow-completion time; **(3)** TCP strives to be network-oblivious, but in this environment we can take advantage of input from the network to influence the transport's behavior (a feature we call *multi-modality*) to better work in integrated research infrastructure; **(4)** DAQ traffic for each experiment has a well-known shape, and flows over infrastructure that is capacity-planned for that experiment.

This position paper argues for the development of specialized transport protocols for DAQ workloads. Through multi-modality, a DAQ stream would not support retransmission while in the DAQ network, but obtains this feature before it crosses into the WAN. Along its end-to-end path, the protocol changes modes as the underlying network resources change—for example, it can use a more "recent" (lower RTT) retransmission buffer to avoid retransmission from the source.

This argument builds on the shortcomings of today's DAQ transport approach (§4.1) and charts a different direction from related work (§4.2) which includes specialized variants of TCP for "long fat networks"[31, 39] in scientific computing.

This paper makes the following contributions: **(1)** A design outline of a transport protocol that is tailored for DAQ workloads (§5.2). This protocol is *multi-modal*—its behavior can change according to the network segment over which it is currently travelling. **(2)** An example of how multi-modality benefits from in-network processing support (§5.3) from commodity, programmable network hardware that is increasingly being deployed across scientific computing infrastructure. In this work, the use of programmability is limited to header processing, making it suitable for P4-programmable hardware. **(3)** A pilot study (§5.4) that includes a Tofino2 switch and Alveo FPGA cards, and which uses data from a particle detector. **(4)** To help spur wider debate and research on this topic, we describe open challenges and future work in §6.

2 DAQ WORKLOAD CHARACTERISTICS & INTEGRATED INFRASTRUCTURE

Fig. 1 outlines the creation and consumption of Data Acquisition (DAQ) workloads. DAQ workloads consist of digitizations of analogue waveforms from physical phenomena—such as the interactions of electrons with a sensor.

A DAQ workload starts by crossing a *DAQ network* ❶ that is designed to meet the needs of a specific experiment [7, 23, 24, 59]. Each experiment has a known data acquisition rate—such as the examples in Table 1. The DAQ rate is based on the precision of the instrument's sensors, the frequency and precision of the analogue-to-digital conversion, and the observations made by the sensors—that is, a maximum number of events would be expected to be observed in a given time window. DAQ networks tend to use commercial-off-the-shelf (COTS) equipment because of simpler procurement and development. For example, both Vera Rubin and DUNE use Ethernet-based DAQ networks.

Data in these workloads is then filtered and buffered in a small computing cluster that forms part of the instrument. Data is then transported over a (10-100ms RTT) WAN ❷ to reach analysis and storage sites ❸. For example, Vera Rubin data flows from Chile to California, and DUNE data flows from South Dakota to Illinois. Constraints on costs and deployability of computing and network hardware [49] impede building large processing facilities with each instrument. For example, DUNE's detector is located one mile underground (in a decommissioned salt mine) and Vera Rubin is located on a mountain at the edge of a desert. These locations are chosen to obtain the best-quality data by reducing the presence of interfering signal sources.

From ❸, DAQ data is transported to storage tiers and analysis facilities ❹ around the world over dedicated circuits—typically 100Gbps and higher—over terabit scale links [3, 5]. Finally, data reaches a *Data Transfer Node* (DTN) located in the "Science DMZ" [51] of the destination network. DTNs are placed in the DMZ to avoid the overhead of traversing perimeter appliances such as firewalls.

2.1 Traffic profile

Across stages ❶→❷→❸→❹, DAQ workloads usually consist of elephant flows carried in a sequence of jumbo Ethernet frames. The MTU of each hop is configured to remove any fragmentation. Workloads can also travel *directly* from ❶ to ❹, to reach researchers sooner. The Vera Rubin telescope uses this stream to alert other telescopes and researchers of interesting overhead observations [12]. For that instrument, this stream is expected to burst to 5.4Gbps, and takes place alongside the nightly 30TB capture from the telescope [37].

2.2 Programmable Networking

The ecosystem outlined in Fig. 1 enthusiastically adopts technologies that can improve scaling, as evidenced in the work by the Global Network Advancement Group (GNA-G) and its Data Intensive Science and AutoGOLE/SENSE working groups [53]. There is *production use* of P4-programmable hardware in several instances of stage ❷. For example, ESnet's high-touch framework [40] relies on Alveo FPGA NICs; AmLight uses over a dozen Tofino switches [14, 15]; and GÉANT [55] uses Tofino switches for network monitoring. As an example of stage ❹, P4Campus [35] monitors large science transfers among other flows coming into the campus. There is also growing *research interest* in using programmable networking across stages ❶-❹ for monitoring [45], analysis [56, 64], and experimentation [65]. All cases that we are aware of use P4-programmable hardware.

3 DAQ TRANSPORT REQUIREMENTS

Present and future expectations for DAQ workload transfers boil down to these DAQ transport protocol requirements: The protocol must **(Req 1)** Operate across different types of networks (Fig. 1). In most cases these networks are IP networks. In some cases, the DAQ transport must work directly over layer 2 (§4). **(Req 2)** Enable high-capacity transfers of the data produced by an instrument. The transport protocol must minimize overhead on network equipment and the end-host DTNs. **(Req 3)** Ensure the timely handling of data to satisfy near-real-time analysis—such as for Vera Rubin's alert distribution system (§2.1). Different transport sessions need *not* be treated independently of one another, and the DTN and network elements may prioritize some transfers over others. **(Req 4)** Be reliable, to ensure that all the instrument's data is analyzed. **(Req 5)** Ensure security of data and resources [19]. For example, Vera Rubin alerts must be encrypted to ensure that security-sensitive observations are not inadvertently leaked [54]. **(Req 6)** Take advantage of deployed in-network processing capabilities (§2.2). **(Req 7)** Provide a message-based abstraction since DAQ workloads consist of discrete, time-stamped measurements. **(Req 8)** Support instrument partitioning: detectors may be partitioned for different simultaneous experiments by different researchers [6], therefore the protocol must indicate which "slice" of the instrument produced the data. **(Req 9)** Be reusable across DAQ networks, science domains, and experiment equipment—including lower-throughput, dispersed sensors [20]. Large instruments can also require reusability across their components—for example, DUNE's four detectors each have specific headers but they all share a top-level DAQ header [68]. **(Req 10)** Support the *integration* of DAQ workload generation, processing, and storage across different instruments and facilities [47]. Scientific instruments are huge, multi-year investments whose value is increased by collaboration and reuse. Integration would also support low-latency coordination through multi-terabit infrastructure that is available to publicly-funded testbed resources [3] and across the energy research community [2]. For example, a supernova burst detected in DUNE would alert Vera Rubin on where to expect photons to arrive from—since neutrinos escape the collapsing star before photons are emitted [72]. Depending on the type of star, the time interval between emission of neutrinos and photons could range from around a minute to several days [36].

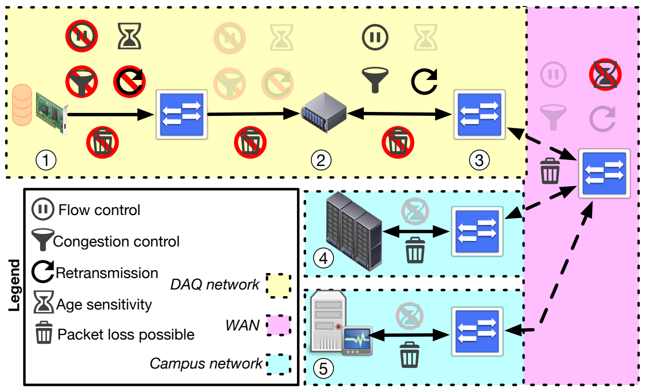

Figure 2: DAQ data is propagated to non-DAQ networks. Today this requires using different transports for different networks, and heavily relies on TCP tuning for high-capacity transfers. To aid the readability of this figure, transport features that do not change between steps appear faded. "Age sensitivity" means that the aging of transported data follows a pre-determined policy. This diagram is explained in §4.

4 TRANSPORTING DAQ DATA TODAY

Today's DAQ transfers evolved to make heavy use of Internet transports because of the convenience from using mature implementations of familiar protocols. §4.1 describes how this convenience is traded-off against performance overhead and against the complexity of several stages of connection termination, buffering, and protocol tuning.

Fig. 2 zooms in on the transport facets of Fig. 1. The yellow area in Fig. 2 shows an instrument's DAQ network. At the originating sensor ①, DAQ data is not buffered for retransmission, and there is no risk of congestion in the DAQ network (§2). When a transport is used in a DAQ network, it is usually UDP (as done in DUNE) but DAQ data might be carried directly over an Ethernet frame (as done in Mu2e). DAQ data is buffered at the first line of servers ② and streamed using TCP through a border router ③ to a WAN through which it reaches a longer-term storage and processing facility ④.

The WAN is shared with other traffic, but it is carefully capacity planned to avoid congestion. It can occasionally lose packets from corruption. As indicated by the icons in Fig. 2, TCP provides flow control, congestion control, and re-transmission between ② and ④. DAQ data is accessed by researchers at various locations. This access might take place soon after the data is gathered. Upon access, this data is streamed using TCP through a WAN from ④ to reach the researchers ⑤. Across all stages where it is used, TCP is heavily tuned to support high data rates [22, 43, 73].

4.1 Shortcomings of today's approach

Today's DAQ transfers incur configuration complexity and performance overhead because of the reliance on TCP. Configuration complexity is incurred from TCP tuning to support high data rates [22, 43, 73]—around 30Gbps for a single stream [46]. Recent work has achieved 55Gbps single-stream transfers over TCP in a testbed environment [66]. The degree of tuning and specialization of TCP in this environment attests to the tension between convenience and overhead.

Even with tuning, there are performance overheads that result from TCP's fundamental unsuitability for transporting DAQ data: **(1)** DAQ traffic consists of discrete, time-stamped messages with well-defined boundaries. Similar to the motivating observations for SCTP [70], TCP's strict, ordered bytestream (a) requires DAQ peers to use message delineation in the bytestream, and (b) causes unnecessary head-of-line blocking when part of the bytestream arrives later. **(2)** End-to-end transport across the "long fat" (huge bandwidth-delay product) segment of DAQ's journey (in the WANs, with a RTT of 10-100ms and throughput of at least 100Gbps) increases retransmission latency and buffering requirements, and these increase flow-completion time. As link capacities and scientific data generation increase, so does the processing overhead for concurrent TCP streams [31]. Moreover, TCP termination and buffering at ④ is unsuitable for rapid inter-instrument coordination—such as online processing of Vera Rubin alerts to reach end-users at the time-scale of milliseconds. Further, this setup is unsuitable for multi-domain alerts [57]—in which alerts from DUNE can provide early-warning about a supernova burst to an optical telescope like Vera Rubin. **(3)** Even if TCP offload is used aggressively—and "in the service of very specific goals" [48]—to reduce processing overhead, TCP's abstraction of the underlying network hides signalling and programmability (§2.2) opportunities. **(4)** DAQ traffic is unlike Internet traffic that TCP evolved to transport: while the arrival rate of DAQ traffic can exceed that of a well-behaving TCP sender, DAQ's "users" are unlike general Internet users. In this environment, resource reservation and capacity planning forestall the potential harm from misbehaving peers [9].

Today's approach meets the following requirements from §3: it is reliable (Req 4), can be secured through TLS (Req 5), and it is reusable across experiments (Req 9). It can transfer data at 30Gbps for single streams, and up to 100Gbps for multiple streams [46] (Req 2) but modern DTNs are being installed with 400GbE NICs [42]. It does not meet the remaining requirements: it does not work directly on layer 2 (Req 1), no timeliness is built into the protocol (Req 3), it relies on termination for processing (Req 6), its stream abstraction requires dissecting by the receiver (Req 7), and instrument metadata is encapsulated in the bytestream, and not available for header processing (Req 8 and Req 10).

4.2 Related work

Specialized transports that were developed for high-performance networks in datacenters [8, 32, 34, 50, 61, 62, 75, 76] are designed to work in a single administrative domain with very low RTT. In contrast, DAQ workloads traverse network segments in different administrative domains, and that offer different latency and throughput guarantees. In contrast to Performance-Enhancing Proxy (PEP) middleboxes [16], the properties of these segments are not necessarily abstracted from communicating peers or other network operators. We envision that exchanging control messaging about multi-modal transports can provide a foundation for reasoning about end-to-end behavior in terms of hop-by-hop behavior, for improving network performance and transparency.

Earlier research on scientific research networks proposed specialized transports for "long fat networks" [30, 31, 39], a custom middlebox for TCP processing in ScienceDMZs [18], and tuned congestion control [41]. In comparison, this paper proposes **(1)** the design of specialized protocols for DAQ—rather than general streaming protocols—and **(2)** the use of programmable networking to provide in-network support for DAQ workloads. The protocol design (1) is extensible—consisting of a core protocol and parametrized extensions. This extensibility is similar to that of BEEP [63], but this work focuses on the transport layer. An important consequence of (1) is that we can take advantage of DAQ's predictable generation of traffic (§2). DAQ's demands on the whole network can be planned in advance. Use of in-network support (2) is inspired by "protocol boosting" [44, 74] but adapts that idea to the specific needs of DAQ workloads and the networks that those workloads traverse.

5 MULTI-MODAL TRANSPORT PROTOCOL FOR DAQ

This section sketches a transport protocol that is tailored for DAQ workloads (§3). We start by describing our end-to-end behavior goals (§5.1) before outlining the protocol header (§5.2), its hop-by-hop processing (§5.3), and the pilot study (§5.4) to develop and evaluate this further.

The protocol operates in a *mode*, in which a combination of *features* are activated and *configured*—features such as retransmission, pacing, and timeliness; and configurations such as where to retransmit from, what pace to set, and the delivery deadline. The mode may be changed by programmable hardware as the transported packets traverse network segments.

Using this paradigm, a DAQ stream would not support retransmission while in the DAQ network, but will take up this feature before it crosses into the WAN. Along its end-to-end path, the protocol changes modes if its features or their configuration changes—for example, if another retransmission buffer becomes available, we would then avoid the need to retransmit from the source, to reduce flow-completion time because of the shorter RTT. This is explained further in §5.1.

Thus a single stream using this transport protocol may be *multi-modal* by opportunistically making use of on-path network resources (§2.2). The protocol's hop-by-hop processing involves conservative, header-based processing, using features that existing P4 hardware supports well [25].

Why create a new protocol? Using UDP would still require additional fields to capture DAQ metadata (§5.2) and add support for reliability (§5.3). We also considered adding custom headers to IPv6, but hop-by-hop headers are not currently reliably supported in today's hardware [33, 58], and extension headers cannot be updated in flight [21]. Moreover, we would still need a transport protocol on top of IP.

5.1 Goal scenario

Fig. 3 outlines the behavior of DAQ transfers using a custom protocol. ① Detectors stream out data encapsulated in the protocol's header, even if directly over layer 2—see (Req 1) in §3. Instruments may be partitioned for different simultaneous experiments by different researchers [6], and the protocol header (§5.2) indicates which "slice" of the detector produced the data. ② The first line of servers buffers and analyzes the data, encapsulates DAQ data in a network protocol, and ③ programmable network hardware across the different networks reference retransmission buffers and packet processing facilities. This buffering reduces the flow-completion time since a re-transmission would originate from a closer source, rather than from ②. Further, these programmable resources can be configured to track the age of information as it crosses the network. This tracking is done to enforce a deadline on the expected arrival of data. Finally, if an element ③ receives signals of downstream congestion or loss, it can relay a back-pressure signal to the sender ②. Streams can be duplicated in the network ③ to reach several downstream researchers directly, ensuring that they get rapid access to fresh data. ⑤ The same protocol can be used to relay streams from storage nodes to other integrated research infrastructure.

5.2 Header structure

We envision instrument sensors supporting this protocol from source, therefore the core header is kept very simple. The core header contains 3 fields: (1) an 8-bit configuration identifier—essentially a version field for interpreting the values of the next field. (2) 24 bits of configuration data—these specify options that are activated for this protocol during the current segment of the transmission. The combination of fields 1 and 2 indicate the transport's *mode*. The configuration data bits activate protocol features such as flow or congestion control, or describe the acknowledgement scheme—if any—used in a network segment. (3) A 32-bit experiment ID. Some

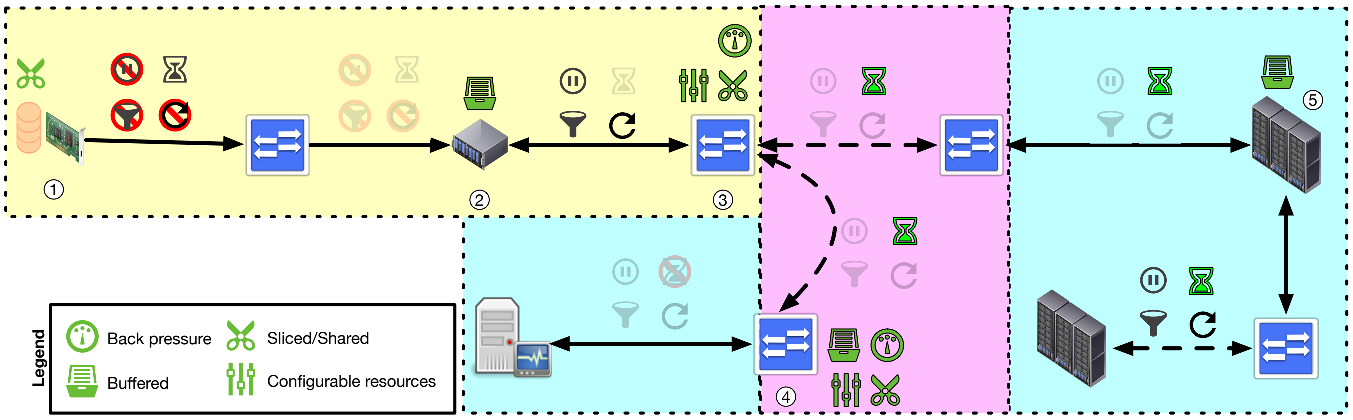

Figure 3: DAQ workloads benefit from multi-modal transport through tailored use of network resources along the segments traversed by the DAQ traffic—a pragmatic layering violation. This figure builds on Fig. 2 and is explained in §5.

of these bits can be used to describe which part of a partitioned instrument produced the data, to satisfy **(Req 8)**.

Additional modes are activated downstream. Activating a mode involves updating the core header and adding mode-specific extension headers, as described below. To keep the implementation simple, the protocol transports discrete datagrams, not bytestreams—following **(Req 7)** from §3.

After the core header, there is a variable number of fixed-size, optional fields (in a fixed order) that depend on the activated features (configuration bits). These fields provide configuration values for different features. For example, if the mode supports retransmission then there is a field that specifies the IP address where to send request for retransmission. If the mode supports timeliness, then there is a field that specifies the delivery deadline and where (IP address) to send a notification if that deadline is exceeded.

5.3 DAQ Transport Behavior

This section outlines the design for a multi-modal transport—the various modes of the protocol are still work-in-progress. In *mode 0*, this transport protocol simply identifies the experiment whose DAQ data is being transported. DAQ data starts out in mode 0 at the sensor ① in Fig. 3. The protocol works both directly on Ethernet and on IP, to satisfy **(Req 1)**.

As DAQ workloads traverse networks that provide different latency and reliability guarantees, the transport's mode is changed by on-path network elements. Changing modes activates transport features and uses resources to ensure that end-to-end latency and reliability guarantees are upheld. Key features, including transport reliability, are described below. The simple core and extensibility of the multi-modal design is intended to satisfy reusability **(Req 9)** and integration **(Req 10)** across scientific research infrastructure.

The protocol is designed to support line-rate transfers **(Req 2)** and minimize network overhead and processing overhead. Processing overhead is minimized through simplicity of logic. Simplicity is achieved by specializing the protocol for the workloads and networks in which it will operate. For example, timely-behavior **(Req 3)** is ensured by explicit transport deadlines that provide a signal for congestion and an input to active queue management. The timeliness mode involves providing an IP address to which "deadline exceeded" messages are sent, to alert the source. On the open Internet, this scheme would be a vector for a Denial-of-Service attack, but in this "limited domain" [17] we can prioritize the processing of age-sensitive data as it travels away from ①. For security **(Req 5)**, we retain the current practice of encrypting the payload using existing third-party software or hardware.

Reliability **(Req 4)** relies on a a re-transmission scheme that generalizes the hop-by-hop behavior of X25 (albeit at a higher layer) by providing an explicit source (IP address) where to request the retransmission. This is closer to a short-term publish-subscribe behavior [60], rather than defaulting to the source for retransmission as done in TCP. We hypothesize that this transport does not require sophisticated congestion control, since data transfers across scientific networks are usually capacity-planned and scheduled to ensure that suitable transmission capacity is available. This hypothesis will be tested in a later deployment of the pilot (§5.4).

The logic for using in-network resources **(Req 6)**—such as switches ③ and ④—and changing modes is not yet fully developed. As described in the next section, we start with a simple 3-mode setup that pre-supposes knowledge of in-network resources at system start, inspired by EJ-FAT [67]. Developing a general approach is an open challenge (§6) that forms a key part of future work.

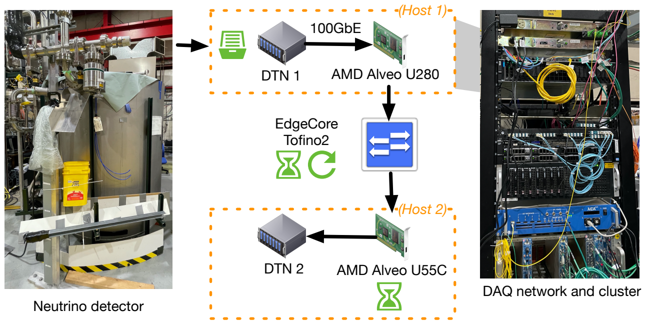

Figure 4: Pilot study. Icons are explained in Fig. 3, and DTN is explained in §2. The first DTN represents the processing and buffering stage in the DAQ network.

5.4 Pilot Study

A pilot study using this transport is outlined in Fig. 4. Two versions of the pilot were developed: the first uses lower-performance, virtual hardware on the FABRIC testbed [10], and the second uses physical hardware and saturates 100 GbE links. The FABRIC prototype was developed for easier design exploration, and used CREASE [4] to manage the virtual resources. In the physical testbed, the Alveo cards were managed using the ESnet smartNIC platform [1].

There are two data sources: (1) the ICEBERG DUNE prototype that uses a liquid argon time projection chamber (LArTPC) detector to generate DAQ data, and (2) synthetic DUNE DAQ data [69] that simulates the neutrino generation by different physical events. In this pilot, data is streamed between (Data Transfer Nodes) DTN 1 and DTN 2, as shown in Fig. 4. The pilot design features three modes: (1) unreliable transport from the sensor to DTN 1; (2) age-sensitive and recoverable-loss transport between DTN 1 and DTN 2; (3) timeliness check at the destination. The transport's mode is changed as the data flows through different segments.

Changing modes involves changing the protocol header, done entirely in network elements. Age-sensitivity involves tracking a time budget as DAQ data travels through the network, and loss-recoverability involves requesting re-transmission from DTN 1. Age-sensitivity is handled entirely in network elements. An element updates an "age" field, and it additionally updates an "aged" flag if a maximum age threshold was exceeded by the time the packet reached that network element. Recovering lost packets involves support from network elements and from data buffers (DTN 1). Network elements add a sequence number to loss-recoverable streams and identify DTN 1 as the nearest buffer; both operations are implemented in network elements. DTN 2 then uses this information to detect loss, and to prepare a NAK to restore the missing packets.

6 OPEN CHALLENGES AND FUTURE WORK

This paper motivates broader research into network support for scientific computing. We outlined the requirements of a DAQ transport protocol and sketched a candidate solution. But, much further work remains. Designing a transport protocol for DAQ workloads that uses in-network support touches upon open research problems that require a community-scale effort: **(1)** We initially envisage having a map of in-network programmable resources that DAQ workloads can use. This map is shared between network operators—perhaps by piggy-backing on BGP messages—to describe their programmable infrastructure and its capabilities. It is an open problem how to discover programmable resources in the network, distribute work to them, and coordinate their activity. Prior work [27, 71, 78] provides potential starting points. **(2)** Beyond header processing, how do we integrate payload processing along the path? For example, DPDK-capable or FPGA resources could be used to generate multi-domain alerts from raw DAQ data (§3) or transcode into other formats, such as HDF5 [26] which is ubiquitously used for storage in scientific computing. **(3)** Osmotic computing [11] uses a large number of distributed sensors, instead of a few large instruments. Sensors lack a DAQ network—instead they rely on cell networks and backhaul. We believe that TCP is adequate for these low-volume streams (over telecom networks), but finding suitable transport modes would better integrate these sensors with other research infrastructure.

ACKNOWLEDGMENTS

We thank the anonymous reviewers, Adrian Farrel, Ezra Kissel, Bruce Mah, Jeff Mogul, Richard Mortier, Chris Neely, Harvey Newman, and Grace Worm for feedback. We're especially grateful to Nishanth Shyamkumar for setting up parts of the pilot study; to Ron Rechenmacher and Bonnie King for providing the ICEBERG traffic samples, Kurt Biery and Wes Ketchum for explaining the DUNE packet format, and David Christian for permission to use photos of the ICEBERG cryostat and its DAQ rack. Joe Mambretti, Ken Miller, and Jason Zurawski helped us understand DTN's state of the art, Ryan Rivera explained the Mu2e packet format, and Niko Neufeld helped us understand the LHC DAQ rate. This work was supported by a URA Visiting Scholars Program Award #24-S-23, by the Fermi Research Alliance LLC under Contract No. DEAC02-07CH11359 with the United States Department of Energy, and by the National Science Foundation (NSF) under award 2346499. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of funders.

REFERENCES

[1] [n. d.]. ESnet SmartNIC platform. https://github.com/esnet/esnet-smartnic-hw/. Accessed: 2024-01-26.

[2] 2022. ESnet Launches Next-Generation Network to Enhance Collaborative Science. https://www.es.net/news-and-publications/esnet-news/2022/esnet-launches-next-generation-network-to-enhance-collaborative-science/. Accessed: 2024-01-26.

[3] 2022. NSF FABRIC project announces groundbreaking high-speed network infrastructure expansion. https://learn.fabric-testbed.net/knowledge-base/nsf-fabric-project-announces-groundbreaking-high-speed-network-infrastructure-expansion/. Accessed: 2024-01-26.

[4] 2024. CREASE Project: Causal REasoning and Attestation for Scientific Experimentation. http://crease.cs.iit.edu/. Accessed: 2024-10-10.

[5] 2024. ESnet Announces First Trans-Atlantic Subsea Spectrum Agreement. https://www.es.net/news-and-publications/esnet-news/2024/esnet-announces-first-trans-atlantic-subsea-spectrum-agreement/. Accessed: 2024-01-26.

[6] B. Abi and 965 others. 2020. Deep Underground Neutrino Experiment (DUNE) Far Detector Technical Design Report, Volume I. Introduction to DUNE. *Journal of Instrumentation* 15, 08 (aug 2020), T08008–T08008. https://doi.org/10.1088/1748-0221/15/08/t08008

[7] A. Abed Abud, C. Batchelor, K. Biery, J. Brooke, G. Crone, D. Cussans, P. Ding, A. Earle, E. Flumerfelt, J. Freeman, A. Habig, R. Halsall, P. Hamilton, N. Ilic, A. Kaboth, W. Ketchum, B. King, J. Klein, P. Lasorak, G. Lehmann Miotto, D. Newbold, D. Oliva, M. Roda, R. Sipos, A. Tapper, A. Thea, and S. Trilov. 2023. DAQ Design document. In *Final Design Review of the DUNE Data Acquisition (DAQ) System*. https://edms.cern.ch/document/2812882/1

[8] Mohammad Alizadeh, Albert Greenberg, David A. Maltz, Jitendra Padhye, Parveen Patel, Balaji Prabhakar, Sudipta Sengupta, and Murari Sridharan. 2010. Data Center TCP. *SIGCOMM Comput. Commun. Rev.* 40, 4 (aug 2010), 63–74. https://doi.org/10.1145/1851275.1851192

[9] M. Allman. 2003. RFC 3465: TCP Congestion Control with Appropriate Byte Counting (ABC). https://www.rfc-editor.org/rfc/rfc3465.html. Accessed: 2024-01-26.

[10] Ilya Baldin, Anita Nikolich, James Griffioen, Indermohan Inder S. Monga, Kuang-Ching Wang, Tom Lehman, and Paul Ruth. 2019. FABRIC: A National-Scale Programmable Experimental Network Infrastructure. *IEEE Internet Computing* 23, 6 (2019), 38–47. https://doi.org/10.1109/MIC.2019.2958545

[11] Daniel Balouek-Thomert, Eduard Gibert Renart, Ali Reza Zamani, Anthony Simonet, and Manish Parashar. 2019. Towards a computing continuum: Enabling edge-to-cloud integration for data-driven workflows. *The International Journal of High Performance Computing Applications* 33, 6 (2019), 1159–1174. https://doi.org/10.1177/1094342019877383 arXiv:https://doi.org/10.1177/1094342019877383

[12] Eric Bellm, Robert Blum, Melissa Graham, Leanne Guy, Željko Ivezić, William O'Mullane, Maria Patterson, John Swinbank, and Beth Willman. 2020. LDM-612 Plans and Policies for LSST Alert Distribution. https://docushare.lsst.org/docushare/dsweb/Get/LDM-612

[13] J.C. Bernauer and 200 others. 2023. Scientific computing plan for the ECCE detector at the Electron Ion Collider. *Nuclear Instruments and Methods in Physics Research Section A: Accelerators, Spectrometers, Detectors and Associated Equipment* 1047 (Feb. 2023), 167859. https://doi.org/10.1016/j.nima.2022.167859

[14] Jeronimo Bezerra. 2023. AmLight 2.0: Flexible control, deep visibility, and programmability Tbps. https://sc23.supercomputing.org/wp-content/uploads/2023/11/SC23-NRE-019-JeronimoBezarra-AmLight_2.0_Flexible_control_deep_visibility_and_programmability_at_Tbps.pdf

[15] Jeronimo Bezerra, Italo Brito, Renata Frez, and Julio Ibarra. 2023. An adaptive and efficient approach to detect microbursts leveraging per-packet telemetry in a production network. In *NOMS 2023-2023 IEEE/IFIP Network Operations and Management Symposium*. 1–6. https://doi.org/10.1109/NOMS56928.2023.10154390

[16] J. Border, M. Kojo, J. Griner, G. Montenegro, and Z. Shelby. 2001. RFC 3135: Performance Enhancing Proxies Intended to Mitigate Link-Related Degradations. https://www.rfc-editor.org/rfc/rfc3135. Accessed: 2024-01-26.

[17] B. Carpenter and B. Liu. 2020. RFC 8799: Limited Domains and Internet Protocols. https://www.rfc-editor.org/info/rfc8799. Accessed: 2024-06-10.

[18] Joaquin Chung, Wojciech Zacherek, AJ Wisniewski, Zhengchun Liu, Tekin Bicer, Rajkumar Kettimuthu, and Ian Foster. 2022. SciStream: Architecture and Toolkit for Data Streaming between Federated Science Instruments. In *Proceedings of the 31st International Symposium on High-Performance Parallel and Distributed Computing* (Minneapolis, MN, USA) *(HPDC '22)*. Association for Computing Machinery, New York, NY, USA, 185–198. https://doi.org/10.1145/3502181.3531475

[19] Eli Dart, Lauren Rotman, Brian Tierney, Mary Hester, and Jason Zurawski. 2013. The Science DMZ: a network design pattern for data-intensive science. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis* (Denver, Colorado) *(SC '13)*. Association for Computing Machinery, New York, NY, USA, Article 85, 10 pages. https://doi.org/10.1145/2503210.2503245

[20] S. Datta-Barua, Y. Su, K. Deshpande, D. Miladinovich, G. S. Bust, D. Hampton, and G. Crowley. 2015. First light from a kilometer-baseline Scintillation Auroral GPS Array. *Geophysical Research Letters* 42, 10 (2015), 3639–3646. https://doi.org/10.1002/2015GL063556 arXiv:https://agupubs.onlinelibrary.wiley.com/doi/pdf/10.1002/2015GL063556

[21] S. Deering and R. Hinden. 2017. RFC 8200: Internet Protocol, Version 6 (IPv6) Specification. https://www.rfc-editor.org/rfc/rfc8200.html. Accessed: 2024-06-18.

[22] ESnet. [n. d.]. ESnet Data Transfer Nodes. https://fasterdata.es.net/performance-testing/DTNs/. Accessed: 2024-01-26.

[23] R. Acciarri et al. 2017. Design and construction of the MicroBooNE detector. *Journal of Instrumentation* 12, 02 (feb 2017), P02017. https://doi.org/10.1088/1748-0221/12/02/P02017

[24] The DUNE Collaboration et al. 2022. Design, construction and operation of the ProtoDUNE-SP Liquid Argon TPC. *Journal of Instrumentation* 17, 01 (jan 2022), P01005. https://doi.org/10.1088/1748-0221/17/01/P01005

[25] Andy Fingerhut. 2020. Why doesn't P4 have floating point types? https://github.com/jafingerhut/p4-guide/blob/master/docs/floating-point-operations.md. Accessed: 2022-08-06.

[26] Mike Folk, Gerd Heber, Quincey Koziol, Elena Pourmal, and Dana Robinson. 2011. An overview of the HDF5 technology suite and its applications. In *Proceedings of the EDBT/ICDT 2011 Workshop on Array Databases* (Uppsala, Sweden) *(AD '11)*. Association for Computing Machinery, New York, NY, USA, 36–47. https://doi.org/10.1145/1966895.1966900

[27] Jiaqi Gao, Ennan Zhai, Hongqiang Harry Liu, Rui Miao, Yu Zhou, Bingchuan Tian, Chen Sun, Dennis Cai, Ming Zhang, and Minlan Yu. 2020. Lyra: A Cross-Platform Language and Compiler for Data Plane Programming on Heterogeneous ASICs. In *Proceedings of the Annual Conference of the ACM Special Interest Group on Data Communication on the Applications, Technologies, Architectures, and Protocols for Computer Communication* (Virtual Event, USA) *(SIGCOMM '20)*. Association for Computing Machinery, New York, NY, USA, 435–450. https://doi.org/10.1145/3387514.3405879

[28] GEANT. [n. d.]. Optimising DTN Configurations. https://wiki.geant.org/display/DTN/Optimising+DTN+Configurations. Accessed: 2024-06-18.

[29] A. Gioiosa, R. Bonventre, S. Donati, E. Flumerfelt, G. Horton-Smith, L. Morescalchi, V. O'Dell, E. Pedreschi, G. Pezzullo, F. Spinella, L. Uplegger, and R. A. Rivera. 2021. Prototype Data Acquisition and Slow Control Systems for the Mu2e Experiment. *IEEE Transactions on Nuclear Science* 68, 8 (2021), 1862–1868. https://doi.org/10.1109/TNS.2021.3094130

[30] Yunhong Gu and Robert L. Grossman. 2007. UDT: UDP-based data transfer for high-speed wide area networks. *Computer Networks* 51, 7 (2007), 1777–1799. https://doi.org/10.1016/j.comnet.2006.11.009 Protocols for Fast, Long-Distance Networks.

[31] Nathan Hanford, Vishal Ahuja, Matthew K. Farrens, Brian Tierney, and Dipak Ghosal. 2018. A Survey of End-System Optimizations for High-Speed Networks. *ACM Comput. Surv.* 51, 3, Article 54 (jul 2018), 36 pages. https://doi.org/10.1145/3184899

[32] Keqiang He, Eric Rozner, Kanak Agarwal, Yu (Jason) Gu, Wes Felter, John Carter, and Aditya Akella. 2016. AC/DC TCP: Virtual Congestion Control Enforcement for Datacenter Networks. In *Proceedings of the 2016 ACM SIGCOMM Conference* (Florianopolis, Brazil) *(SIGCOMM '16)*. Association for Computing Machinery, New York, NY, USA, 244–257. https://doi.org/10.1145/2934872.2934903

[33] Geoff Huston. 2022. Hop-by-hop extension headers. https://blog.apnic.net/2022/04/22/hop-by-hop-extension-headers/. Accessed: 2024-01-26.

[34] Glenn Judd. 2015. Attaining the Promise and Avoiding the Pitfalls of TCP in the Datacenter. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*. USENIX Association, Oakland, CA, 145–157. https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/judd

[35] Hyojoon Kim, Xiaoqi Chen, Jack Brassil, and Jennifer Rexford. 2021. Experience-driven research on programmable networks. *SIGCOMM Comput. Commun. Rev.* 51, 1 (mar 2021), 10–17. https://doi.org/10.1145/3457175.3457178

[36] Matthew D. Kistler, W. C. Haxton, and Hasan Yüksel. 2013. Tomography of Massive Stars from Core Collapse to Supernova Shock Breakout. *The Astrophysical Journal* 778, 1 (nov 2013), 81. https://doi.org/10.1088/0004-637X/778/1/81

[37] Ron Lambert, Jeff Kantor, Mike Huffer, Chip Cox, Paul Wefel, Matt Kollross, Albert Astudillo, Sandra Jaque, Mark Foster, Richard Hughes-Jones, and Stuart McLellan. 2021. LSE-78 (rel6.0) Observatory Network Design. https://docushare.lsst.org/docushare/dsweb/Get/LSE-78

[38] Ron Lambert, Jeff Kantor, Mike Huffer, Chip Cox, Paul Wefel, Matt Kollross, Albert Astudillo, Sandra Jaque, Mark Foster, Richard Hughes-Jones, and Stuart McLellan. 2021. LSE-78: Vera C Rubin Observatory Network Design. https://docushare.lsst.org/docushare/dsweb/Get/LSE-78

[39] Yee-Ting Li, Douglas Leith, and Robert N. Shorten. 2007. Experimental evaluation of TCP protocols for high-speed networks. *IEEE/ACM Trans. Netw.* 15, 5 (oct 2007), 1109–1122. https://doi.org/10.1109/TNET.2007.896240

[40] Zhang Liu, Bruce Mah, Yatish Kumar, Chin Guok, and Richard Cziva. 2021. Programmable Per-Packet Network Telemetry: From Wire to Kafka at Scale. In *Proceedings of the 2021 on Systems and Network Telemetry and Analytics* (Virtual Event, Sweden) *(SNTA '21)*. Association for Computing Machinery, New York, NY, USA, 33–36. https://doi.org/10.1145/3452411.3464443

[41] Imtiaz Mahmud, George Papadimitriou, Cong Wang, Mariam Kiran, Anirban Mandal, and Ewa Deelman. 2023. Elephants Sharing the Highway: Studying TCP Fairness in Large Transfers over High Throughput Links. In *Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis* (Denver, CO, USA,) *(SC-W '23)*. Association for Computing Machinery, New York, NY, USA, 806–818. https://doi.org/10.1145/3624062.3624594

[42] Joe J Mambretti. 2024. Personal communication.

[43] Harry Mangalam. 2017. How to transfer large amounts of data via network. http://moo.nac.uci.edu/~hjm/HOWTO_move_data.html. Accessed: 2024-01-26.

[44] William S. Marcus, Ilija Hadzic, Anthony J. McAuley, and Jonathan M. Smith. 1998. Protocol boosters: applying programmability to network infrastructures. *IEEE Commun. Mag.* 36, 10 (1998), 79–83. https://doi.org/10.1109/35.722140

[45] Ali Mazloum, Jose Gomez, Elie Kfoury, and Jorge Crichigno. 2023. Enhancing perfSONAR Measurement Capabilities using P4 Programmable Data Planes. In *Proceedings of the SC '23 Workshops of The International Conference on High Performance Computing, Network, Storage, and Analysis* (Denver, CO, USA,) *(SC-W '23)*. Association for Computing Machinery, New York, NY, USA, 819–829. https://doi.org/10.1145/3624062.3624596

[46] Kenneth Miller. 2024. Personal communication.

[47] William L. Miller, Deborah Bard, Amber Boehnlein, Kjiersten Fagnan, Chin Guok, Eric Lançon, Sreeranjani Ramprakash, Mallikarjun Shankar, Nicholas Schwarz, and Benjamin L. Brown. 2023. Integrated Research Infrastructure Architecture Blueprint Activity (Final Report 2023). (7 2023). https://doi.org/10.2172/1984466

[48] Jeffrey C. Mogul. 2003. TCP Offload Is a Dumb Idea Whose Time Has Come. In *9th Workshop on Hot Topics in Operating Systems (HotOS IX)*. USENIX Association, Lihue, HI. https://www.usenix.org/conference/hotos-ix/tcp-offload-dumb-idea-whose-time-has-come

[49] Jeffrey C. Mogul and John Wilkes. 2023. Physical Deployability Matters. In *Proceedings of the 22nd ACM Workshop on Hot Topics in Networks* (Cambridge, MA, USA,) *(HotNets '23)*. Association for Computing Machinery, New York, NY, USA, 9–17. https://doi.org/10.1145/3626111.3628190

[50] Matthew K. Mukerjee, Christopher Canel, Weiyang Wang, Daehyeok Kim, Srinivasan Seshan, and Alex C. Snoeren. 2020. Adapting TCP for Reconfigurable Datacenter Networks. In *17th USENIX Symposium on Networked Systems Design and Implementation (NSDI 20)*. USENIX Association, Santa Clara, CA, 651–666. https://www.usenix.org/conference/nsdi20/presentation/mukerjee

[51] Vasudevan Nagendra, Vinod Yegneswaran, and Phillip Porras. 2017. Securing Ultra-High-Bandwidth Science DMZ Networks with Coordinated Situational Awareness. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks* (Palo Alto, CA, USA,) *(HotNets '17)*. Association for Computing Machinery, New York, NY, USA, 22–28. https://doi.org/10.1145/3152434.3152460

[52] Niko Neufeld. 2024. Personal communication.

[53] Newman, Harvey. 2024. The Global Network Advancement Group: A Next Generation System for the LHC Program and Data Intensive Sciences. *EPJ Web of Conf.* 295 (2024), 07044. https://doi.org/10.1051/epjconf/202429507044

[54] William O'Mullane. 2021. DTMN-108 Security of Rubin Observatory data. https://dmtn-108.lsst.io/

[55] Damian Parniewicz, Tomáš Martínek, Federico Pederzolli, Damu Ding, Mauro Campanella, Ivana Golub, and Tim Chown. 2022. In-Band Network Telemetry Tests in NREN Networks. https://resources.geant.org/wp-content/uploads/2022/02/GN4-3_White-Paper_In-Band-Network-Telemetry.pdf

[56] Shivam Patel, Rigden Atsatsang, Kenneth M. Tichauer, Michael H. L. S. Wang, James B. Kowalkowski, and Nik Sultana. 2022. In-network fractional calculations using P4 for scientific computing workloads. In *Proceedings of the 5th International Workshop on P4 in Europe, EuroP4*

2022, Rome, Italy, 9 December 2022, Marco Chiesa and Shir Landau Feibish (Eds.). ACM, 33–38. https://doi.org/10.1145/3565475.3569083

[57] Maria Patterson, Eric Bellm, John Swinbank, and Spencer Nelson. 2020. DMTN-093: Design of the LSST Alert Distribution System. https://dmtn-093.lsst.io/. Accessed: 2024-01-26.

[58] S. Peng, Z. Li, C. Xie, Z. Qin, and G. Mishra. 2024. Operational Issues with Processing of the Hop-by-Hop Options Header. https://www.ietf.org/archive/id/draft-ietf-v6ops-hbh-09.html. Accessed: 2024-06-20.

[59] G N U Perdue, L Bagby, B Baldin, C Gingu, J Olsen, P Rubinov, , E C Schulte, R Bradford, W K Brooks, D A.M. Caicedo, and C M Castromonte. 2012. The MINERνA Data Acquisition System and Infrastructure. *Nucl.Instrum.Meth.A694:179-192,2012* 694 (9 2012). https://doi.org/10.1016/j.nima.2012.08.024

[60] Peter R. Pietzuch, Brian Shand, and Jean Bacon. 2003. A framework for event composition in distributed systems. In *Proceedings of the ACM/IFIP/USENIX 2003 International Conference on Middleware* (Rio de Janeiro, Brazil) *(Middleware '03)*. Springer-Verlag, Berlin, Heidelberg, 62–82.

[61] Pawan Prakash, Advait Dixit, Y. Charlie Hu, and Ramana Kompella. 2012. The TCP Outcast Problem: Exposing Unfairness in Data Center Networks. In *9th USENIX Symposium on Networked Systems Design and Implementation (NSDI 12)*. USENIX Association, San Jose, CA, 413–426. https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/prakash

[62] Costin Raiciu, Sebastien Barre, Christopher Pluntke, Adam Greenhalgh, Damon Wischik, and Mark Handley. 2011. Improving datacenter performance and robustness with multipath TCP. *SIGCOMM Comput. Commun. Rev.* 41, 4 (aug 2011), 266–277. https://doi.org/10.1145/2043164.2018467

[63] M. Rose. 2001. RFC 3080: The Blocks Extensible Exchange Protocol Core. https://datatracker.ietf.org/doc/html/rfc3080. Accessed: 2024-01-26.

[64] Ganesh C. Sankaran, Joaquin Chung, and Raj Kettimuthu. 2021. Leveraging In-Network Computing and Programmable Switches for Streaming Analysis of Scientific Data. In *2021 IEEE 7th International Conference on Network Softwarization (NetSoft)*. 293–297. https://doi.org/10.1109/NetSoft51509.2021.9492726

[65] Marcos Schwarz and Frederic Loui. 2023. Global P4 Lab.

[66] Marcos Schwarz, Brian Tierney, Kiran Vasu, Eli Dart, Christian Esteve Rothenberg, Jeronimo Bezerra, and Italo Valcy. 2024. Recent Linux Improvements that Impact TCP Throughput: Insights from R&E Networks. In *2024 Workshop on Innovating the Network for Data-Intensive Science (INDIS)*. In press. https://fasterdata.es.net/assets/INDIS_2024_final.pdf.

[67] Stacey Sheldon, Yatish Kumar, Michael Goodrich, and Graham Heyes. 2022. EJ-FAT Joint ESnet JLab FPGA Accelerated Transport Load Balancer. In *2022 IEEE/ACM International Workshop on Innovating the Network for Data-Intensive Science (INDIS)*. 32–40. https://doi.org/10.1109/INDIS56561.2022.00010

[68] Roland Sipos. 2023. The Ethernet readout of the DUNE DAQ system.

[69] E.L. Snider and G. Petrillo. 2017. LArSoft: toolkit for simulation, reconstruction and analysis of liquid argon TPC neutrino detectors. *Journal of Physics: Conference Series* 898, 4 (oct 2017), 042057. https://doi.org/10.1088/1742-6596/898/4/042057

[70] R. Stewart. 2007. RFC 4960: Stream Control Transmission Protocol. https://datatracker.ietf.org/doc/html/rfc4960. Accessed: 2024-01-26.

[71] Nik Sultana, John Sonchack, Hans Giesen, Isaac Pedisich, Zhaoyang Han, Nishanth Shyamkumar, Shivani Burad, André DeHon, and Boon Thau Loo. 2021. Flightplan: Dataplane Disaggregation and Placement for P4 Programs. In *18th USENIX Symposium on Networked Systems Design and Implementation (NSDI 21)*. USENIX Association, 571–592. https://www.usenix.org/conference/nsdi21/presentation/sultana

[72] NASA Webb Mission Team. 2024. Webb Finds Evidence for Neutron Star at Heart of Young Supernova Remnant. https://science.nasa.gov/missions/webb/webb-finds-evidence-for-neutron-star-at-heart-of-young-supernova-remnant/. Accessed: 2024-01-26.

[73] Brian Tierney, Eli Dart, Ezra Kissel, and Eashan Adhikarla. 2021. Exploring the BBRv2 Congestion Control Algorithm for use on Data Transfer Nodes. In *2021 IEEE Workshop on Innovating the Network for Data-Intensive Science (INDIS)*. 23–33. https://doi.org/10.1109/INDIS54524.2021.00008

[74] K.D. Underwood, K.S. Hemmert, A. Rodrigues, R. Murphy, and R. Brightwell. 2005. A hardware acceleration unit for MPI queue processing. In *19th IEEE International Parallel and Distributed Processing Symposium*. 10 pp.–. https://doi.org/10.1109/IPDPS.2005.30

[75] Balajee Vamanan, Jahangir Hasan, and T.N. Vijaykumar. 2012. Deadline-aware Datacenter TCP. *SIGCOMM Comput. Commun. Rev.* 42, 4 (aug 2012), 115–126. https://doi.org/10.1145/2377677.2377709

[76] Vijay Vasudevan, Amar Phanishayee, Hiral Shah, Elie Krevat, David G. Andersen, Gregory R. Ganger, Garth A. Gibson, and Brian Mueller. 2009. Safe and effective fine-grained TCP retransmissions for datacenter communication. *SIGCOMM Comput. Commun. Rev.* 39, 4 (aug 2009), 303–314. https://doi.org/10.1145/1594977.1592604

[77] Alexandre Zabi. 2022. Real time analysis with the CMS Level-1 Trigger. https://cms.cern/news/real-time-analysis-cms-level-1-trigger. Accessed: 2024-06-01.

[78] Changgang Zheng, Haoyue Tang, Mingyuan Zang, Xinpeng Hong, Aosong Feng, Leandros Tassiulas, and Noa Zilberman. 2023. DINC: Toward Distributed In-Network Computing. *Proc. ACM Netw.* 1, CoNEXT3, Article 14 (nov 2023), 25 pages. https://doi.org/10.1145/3629136