Towards Managed Terabit/s Scientific Data Flows

Artur Barczyk, Azher Mughal, Harvey Newman, Iosif Legrand, Michael Bredel, Ramiro Voicu, Vlad Lapadatescu
California Institute of Technology, 1200 E. California Blvd, 91125 Pasadena, CA, USA
{artur.barczyk, iosif.legrand, michael.bredel, ramiro.voicu, vlad.lapadatescu}@cern.ch, {azher, newman}@hep.caltech.edu

Tony Wildish, Princeton University, Princeton, NJ 08544, USA
awildish@princeton.edu

Abstract - Scientific collaborations on a global scale, such as the LHC experiments at CERN [1], rely today on the presence of high performance, high availability networks. In this paper we review the developments performed over the last several years on high throughput applications, multilayer software-defined network path provisioning, path selection and load balancing methods, and the integration of these methods with the mainstream data transfer and management applications of CMS [2], one of the major LHC experiments. These developments are folded into a compact system capable of moving data among research sites at the 1 Terabit per second scale. Several aspects that went into the design and target different components of the system are presented, including: evaluation of the 40 and 100Gbps capable hardware on both network and server side, data movement applications, flow management and the network-application interface leveraging advanced network services. We report on comparative results between several multi-path algorithms, the performance increase obtained using this approach, and present results from the related SC'13 demonstration.

I. INTRODUCTION

The LHC physics program relies on highly reliable continental and transoceanic networks to support the global distribution, processing and analysis of multi-terabyte to petabyte datasets at hundreds of sites, with ongoing data aggregate flows of several petabytes per week. Compared to the highly successful data taking Run 1 period of 2009-12 that led to many groundbreaking results including the Higgs boson discovery, the challenges of the upcoming Run2 (2015-18) and beyond are even greater including larger data flows, processing and storage requirements. In response to these challenges, the future trends are towards greater "location independent" data access with caching in addition to strategic pre-placement of datasets, managed data movement and load balancing as needed, and the development of agile systems able to exploit and coordinate the use of globally distributed heterogeneous computing resources.

An end-to-end computing system relies on several components: the high-performance storage and computing hardware, server-side network equipment, switching and routing elements in the local and wide-area networks, and the software tools for data transfer and management. The work presented here has focused on several of these components.

We present our system in a bottom-up manner, starting with the network infrastructure. In Section II we describe our multi-path approach, how we leverage the capabilities of OpenFlow-enabled network elements for this purpose, the path-selection algorithms used, and the interplay with the end-host based MP-TCP protocol. In Section III we give a detailed description of the data transfer management toolkit used by one of the LHC experiments, and in particular the interface with the Bandwidth on Demand (BoD) provisioning system. We conclude in Section IV with a description and results of the demonstration done during the SC'13 conference.

II. MULTIPATH WITH OPENFLOW

In today's networks, forwarding is usually constrained to a single path by route selection or spanning tree topology, or at best limited by a simple multipath mechanism that operate on a hop-by-hop basis. Meeting the challenge of increasing data transfer volumes of the major science programs, which have reached several petabytes per week in the case of the LHC experiments, requires better optimization and management capabilities, rather than costly over-provisioning of capacity [4]. As part of the OLiMPS (Openflow Link-layer MultiPath Switching) project [5], we have investigated a logically centralized traffic engineering solution with multipath forwarding, thus removing the constraints induced by a spanning-tree topology. By extending the Floodlight [6] OpenFlow controller, we implemented several path allocation algorithms and evaluated their performance. In addition, we provide an API which enables data movement applications to provide additional information, such as the volume of data to be transferred, to the controller. The latter may use this information to perform traffic optimization. Furthermore, we deployed Multipath-TCP (MP-TCP) [7] [8] on the end-hosts, and demonstrated how it can benefit from an intelligent path allocation.

A. Multipath Algorithms

In our study we have compared several path selection strategies, such as hash-based, random, round-robin path selection, as well as one assigning flows to the path with least number of flows installed. Details are described in [9]. One particularly interesting selection algorithm is using additional information from the data transfer application. The application provides the amount of data to be transferred, and using this information, the controller calculates a virtual finishing time *T($link_i$)* for each link *i,* calculated as

$$T(link_i) = \sum_{j=1}^{J} \frac{\max\{0, S(j) - D(j)\}}{w * C(link_i)}$$

where J is the total number of flows on link and *j* is the index of a specific flow. *S*(*j*) is the amount of data for flow *j* as announced by the application, while *D*(*j*) is the amount of data that has already been transferred. We estimate *D*(*j*) by using OpenFlow flow statistics as well as *S*(*j*) and the time elapsed since the start of the flow. *C*(*link*$_i$) is the capacity of link *i* and *w* is an arbitrary weight that can be used, e.g. in the case of varying link capacities. *T*(*link*$_i$) is calculated for all links on all possible paths, and the flow is assigned to the path with the smallest virtual finishing time

We have used a dedicated four switch testbed emulating the US LHCNet transatlantic network to evaluate the algorithms. The inter-switch connections were a full mesh with 2x1 Gbps between any switch-pair. Consequently, we had a total of 6 link-disjoint paths between each pair of servers. The uplinks to the servers were at 10 Gbps, and the servers were able to fully utilize the network capacity. Each transfer transmitted a total of approximately 500 GBytes of data using several sequential TCP flows with Zipf-distributed file sizes between 1 and 40 Gbytes. Each site initiates 1-15 parallel data transfers. The inter-transfer waiting times are exponentially distributed with an expected waiting time equal to half the mean transfer time.

We measured the average transfer times between pairs of servers with an increasing number of parallel transfers. The smaller the transfer time, the greater the network utilization and the better the load-balancing algorithm. Given an optimal path allocation, we expect the first 6 parallel transfers will to finish in approximately 100 minutes. Once the number of parallel transfers exceeds the number of link-disjoint paths, we expect the transfer time to increase linearly.

Figure 2 depicts the normalized mean transfer times obtained in our experiments on the testbed, with an increasing number of parallel transfers. We find that for the hash-based and random path allocation algorithms, the normalized average transfer times are longer then the optimal duration, while the more intelligent algorithms, that take the link utilization and application information into account, closely match the expected optimum closely. Moreover, while the random path allocation approaches show a substantial variation in the transfer times, the more intelligent approaches have a more deterministic behavior, with smaller variations.

For the MP-TCP experiments, we combined the in-network load balancing with Multipath-TCP. MP-TCP leverages multiple available paths between source and destination by creating a set of TCP sub-streams for each connection. As opposed to the multipath algorithms described above, MP-TCP runs in the linux kernel on the end-hosts, and is agnostic to the network topology. We performed experiments similar to the baseline experiments, however, using an MP-TCP enabled Linux kernel with the *ndiffports* path-manager configured to generate 3 sub-flows per transfer. By increasing the number of TCP flows for each transfer, we find a performance improvement in case of a small number of parallel transfers, as depicted in Figure 2.

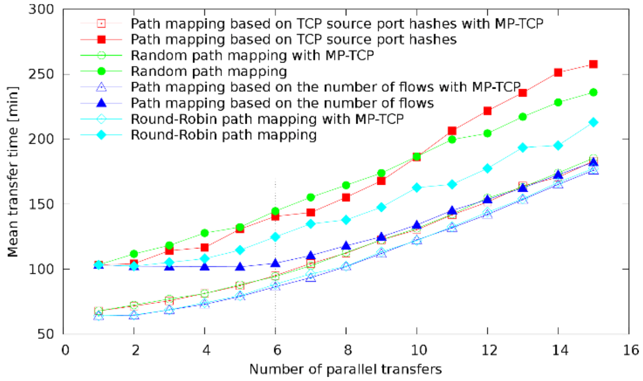

Figure 1: A comparison between in-network load balancing mechanisms with and without MP-TCP enabled. We find that intelligent in-network load balancing algorithm match the theoretical optimum closely. Moreover, all algorithms can benefit from MP-TCP.

One interesting observation is that the choice of algorithm is somewhat less important when MP-TCP is used on the end-hosts. That is, as long as we deploy per-flow multipath forwarding, even the randomized path selection approaches, which worked poorly without MP-TCP, achieve relatively good results. This is a direct result of MP-TCP's optimization algorithm which dynamically adjusts the sending rate on per individual TCP sub-flow basis.

The OLiMPS controller is currently being ported to the OpenDaylight [9] framework.

III. ANSE, PhEDEx and dynamic circuits

The ANSE project [9] has been focusing on interfacing "advanced network services" with scientific data and workflow management applications such as those being used in the LHC experiments. These services allow the applications to either observe or react to the status of the network, e.g. through interfaces to the MonALISA [3] [10] [11] and/or PerfSONAR [12] monitoring systems, or to execute some level of control, e.g. as provided by capacity allocation systems such as OSCARS [13], or NSI [14] based systems such as OpenNSA [15]. In this section we present the work on dynamic circuit allocation, and the PhEDEx [16] toolset in the CMS [2] experiment.

PhEDEx is the set of data placement management tools used by the CMS experiment at the LHC that manages the scheduling of all large-scale wide area network transfers, in order to ensure reliable delivery of the data.

The ANSE project (in the context of CMS) has made significant improvements to PhEDEx, driven by the need for more predictable performance of data transfers, and for more effective co-scheduling of jobs with the arrival of the data to be processed. This has been achieved by making PhEDEx aware of the network status, and giving it the means to allocate guaranteed bandwidth on demand through the use of dynamic virtual circuits. A prototype [17] that has demonstrated the value of this approach was successfully field tested between CERN and Amsterdam, and is the basis of a production ready version which will soon be deployed in CMS.

A. *PhEDEx architecture*

PhEDEx consists of an Oracle database, a website/data-service, a set of central agents and a set of site agents for each PhEDEx site. The central agents run at CERN and deal with routing, request-management, bookkeeping and other activities. The site agents process the transfers which were queued by the central agents. PhEDEx operates in a data-pull mode: the destination pulls the data to itself when it is ready.

The FileDownload Agent (FDA) is perhaps the most important site agent. It executes file-transfers in bulk, copying many files with each transfer (job). Each job contains the source and destination Physical File Names (PFNs). The agent receives only Logical File Names (LFNs) plus the name of the chosen source site. It builds PFNs from the LFNs and a lookup-table per-site, which each site maintains and uploads to the database.

B. *Implementation details*

There were several alternatives explored, when trying to integrate the control and use of dynamic circuits in PhEDEx. The most basic approach is the integration at the individual file transfer level, essentially asking for a new circuit for each new transfer. For our first prototype, we decided to go for a more advanced integration which uses an FDA to manage the circuits for an entire site. In this approach, the FDAs remain active across multiple transfers, and persist as long as needed according to the transfer-queue at each site.

The prototype included an implementation of the circuit management software directly in the FDA code. In order to transfer data over alternative paths (other than the one specified by the lookup-table), the original hostname/IP in each PFN is replaced in the FDA with the source IP and destination IP, each time a new path is selected.

For our production version we created a ‘CircuitAgent’, which extends the FileDownload agent base class. This lets us switch between the CircuitAgent and the FileDownload agent, with minimal impact on the infrastructure.

The CircuitAgent checks the workload every minute. It estimates the remaining work per node-pair based on the size of the download queue and past transfer rates. It then decides if it’s worthwhile to request a circuit for that pair. If so, a request will be made to the ‘CircuitManager’.

Before a transfer task is passed to the transfer backend we call the CircuitManager to check if a circuit exists between the endpoints. If so, it updates the PFNs with the source/destination IPs of the new path.

The CircuitManager receives a request from the CircuitAgent, uses one of the pluggable backends to pass it to a circuit-capable infrastructure, then manages the circuit on behalf of the CircuitAgent. Requests and teardowns are forwarded via a backend to a circuit infrastructure’s API of the user’s choice. Currently we support only OSCARS (via Dynes [18] [19]) and ODL calls (via MonALISA). This can be extended via a plug-in system.

We present in Figure 3 a simplified version of the sequence diagram of our software.

The next step is to complete the production version of this framework (mostly stress testing and bug fixing). Longer term, we will move this functionality to a central ‘CircuitManagement’ entity, only one of which would exist for the whole PhEDEx instance. This can then make more informed decisions about which transfers would actually merit and benefit from the creation of a new path/circuit.

IV. SC’13 RESULTS

During the Supercomputing conference 2013 (SC13) in Denver Colorado, Caltech along with international team of researchers designed and demonstrated the first LHC Terabit network Hub in the Caltech booth. The Terabit network hub consisted of four 100G WAN connections and 1 Tbps DWDM optical connection between Caltech and Vanderbilt booths. High speed SSD based disk servers with 40GE NICs were used as the end point systems. In addition, for the first time a multipath WAN network controlled by the SDN controller was demonstrated, which provided smooth data flows balanced across network paths with varying network speeds. Figure 4 shows the SC13 WAN and show floor network layout.

The network was designed using high speed optical and Ethernet switching devices. Key hardware components used during the demonstration are: Mellanox MLXe-16 Ethernet switch with 4 x 100GE, 40 x 40GE ports and 8 x 10GE ports; OpenFlow enabled Dell-Force10 Z9000 Ethernet switches; Mellanox SX6036 Ethernet switches; 40GE Network Interface Cards from Mellanox along with active optical cables and Padtec optical DWDM equipment for inter-booth data transfer at 1Tbps

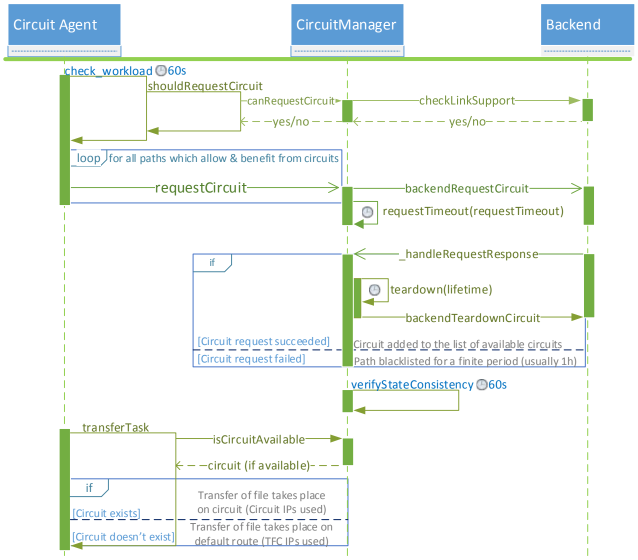
Circuit Agent
CircuitManager
Backend
check_workload 60s
shouldRequestCircuit
canRequestCircuit
checkLinkSupport
yes/no
yes/no
loop
for all paths which allow & benefit from circuits
requestCircuit
backendRequestCircuit
requestTimeout(requestTimeout)
_handleRequestResponse
if
teardown(lifetime)
backendTeardownCircuit
[Circuit request succeeded]
[Circuit request failed]
Circuit added to the list of available circuits
Path blacklisted for a finite period (usually 1h)
verifyStateConsistency 60s
transferTask
isCircuitAvailable
circuit (if available)
if
[Circuit exists]
[Circuit doesn’t exist]
Transfer of file takes place on circuit (Circuit IPs used)
Transfer of file takes place on default route (TFC IPs used)

Figure 2: Sequence diagram of the circuit management framework

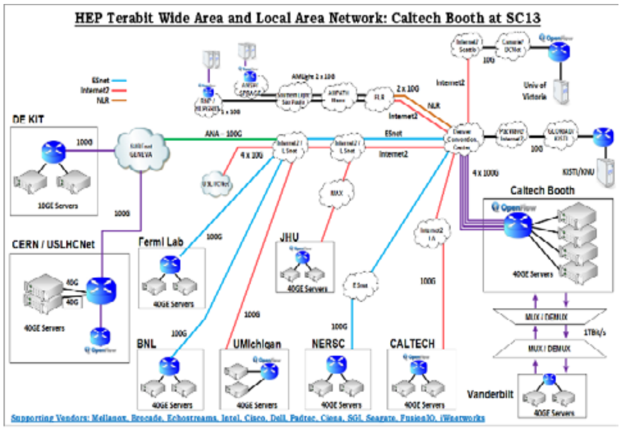

Figure 3: Caltech 2013 - WAN and Inter Booth network layout.

Data was transferred from the show floor to several LHC end sites around the globe. Figure 5 shows both the inter booth and the WAN data transfers. In total, average data transfer rates of 750 Gbps with peaks at 850Gbps were achieved.

The following points provide a summary of data transfer results achieved between Caltech booth on the show floor and the various individual LHC end sites. This summary also includes the challenges faced on each path, and the techniques used to meet the challenges.

SC13 – DE-KIT (Germany, via ANA transatlantic link): 75Gbps from disk to disk was achieved. DE-KIT used multiple 10GE servers while two servers were used at the show floor.

SC13 – BNL over ESnet: 80Gbps achieved over two pair of hosts at each end site. Only memory to memory tests were performed due to non-availability of disk based servers

SC13 – NERSC over ESnet: Packet loss was encountered initially due to the usage of data center grade Ethernet switches having relatively small buffers in the WAN path. However the path became clean once those switches were removed from the picture. A consistent 90 Gbps throughput was then achieved by reading from two SSD hosts at NERSC facility sending to a single host at the booth with multiple 40GE network cards.

SC13 – FNAL over ESnet: The wide area path showed packet loss. It was not clearly identified which network, router, end hosts or NIC firmware had issues. A single stream TCP session could reach up to 5Gbps. However a single UDP stream could go up to 15Gbps per flow. Later on, Linux traffic shaper tools 'tc' were used to pace the TCP flows, led to single stream throughputs of up to 15 Gbps. However multiple streams were still a problem to FNAL. This seemed to indicate that something in the path, most probably a router or a switch with small buffers, was causing packets to be dropped.

SC13 – Pasadena over Internet2 AL2S: 80Gbps transfer rates were reached by reading from the disks on the show floor and writing on servers at the Caltech Tier2 center. This was a disk to memory transfer because the link was lossy in the other direction.

SC13 – CERN over ESnet (ANA-100 transatlantic link): A maximum of 75Gbps memory to memory was achieved by using two servers at CERN and two servers on the show floor. Disk to disk data throughput of 40Gbps was reached.

V. CONCLUSIONS

Fast and efficient data distribution and access, as required by modern distributed scientific instrumentation such as the LHC experiments’ computing infrastructures, rely on smooth interplay of many components. On top of the raw network capacity, the network architecture, switching equipment features and performance, end-system I/O architecture, the transfer applications and data management system software need to be tuned, and to some extent co-designed and co-developed, for frictionless operation.

We showcase the current status of what is achievable using the state-of-the-art components, aiming at demonstrating full Terabit/s data movement between nodes at the SC’14 exhibition floor as well as several LHC computing sites reachable over 100G WAN infrastructure.

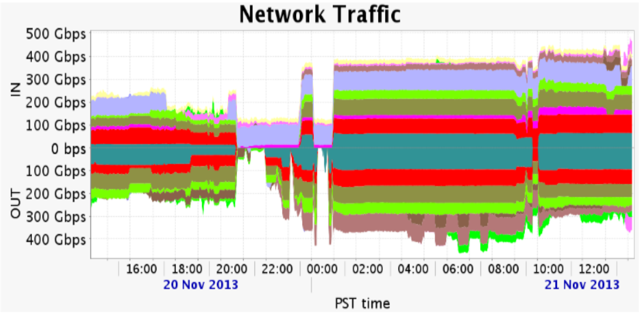

Figure 4: Total traffic flow from Caltech Booth.

VI. ACKNOWLEDGMENTS

We thank our commercial partners who made this demonstration possible through donations of the large ensemble of state-of-the art equipment used during SC’13: Mellanox, Brocade, Dell/Force10, Echostreams, Intel, Padtec and Extreme Networks.

The development work described in this paper was supported through the following grants: DOE DE-SC0007346, NSF 0958998, NSF 1246133 and Cisco Research grant 2014-128271.

We also thank the SCinet network team for their outstanding support over the past editions of the Supercomputing exhibition, culminating in this year’s infrastructure that will reach the Terabit/s scale.

VII. References

[1] CERN, [Online]. Available: http://home.web.cern.ch/.

[2] The CMS Collaboration, "The CMS experiment at the CERN LHC," in *JINST 3 S08004*, 2008.

[3] "Fast Data Transfer (FDT)," [Online]. Available: http://fdt.cern.ch/.

[4] S. Jain, A. Kumar et al, "Experience with a Globally-Deployed Software Defined WAN," in *Proceedings of ACM SIGCOMM*, 2013.

[5] "OpenFlow Link-layer MultiPath Switching," [Online]. Available: http://www.uslhcnet.org/projects/olimps/.

[6] Big Switch Networks, "Project Floodlight," [Online]. Available: http://www.projectfloodlight.org/floodlight/.

[7] A. Ford, C. Raiciu, M. Handley and O. Bonaventure, "IETF RFC 6824.

[8] C. Raiciu, C. Paasch et al., "How Hard Can It Be? Designing and Implementing a Deployable Multipath TCP," in *Proceedings of NSDI*, 2012.

[9] "LHCONE Point-to-Point Service Workshop, December 2012," [Online]. Available: http://indico.cern.ch/event/215393/session/1/contribution/8/material/slides/1.pdf.

[10] "MONitoring Agents using a Large Integrated Services Architecture (MonALISA)," [Online]. Available: http://monalisa.caltech.edu/.

[11] I. Legrand, H. Newman, R. Voicu, C. Cirstoiu, C. Grigoras, C. Dobre, A. Muraru, A. Costan, M. Dediu and C. Stratan, "MonALISA: An agent based, dynamic service system to monitor, control and optimize," *Computer Physics Communications*, vol. 180, no. 12, pp. 2472-2498, December 2009.

[12] B. Tierney, J. Metzger, J. Boote, E. Boyd, A. Brown, R. Carlson, M. Zekauskas, J. Zurawski, M. Swany and M. Grigoriev, "perfSONAR: Instantiating a Global Network Measurement Framework".

[13] C. Guok, D. Robertson, M. Thompson, J. Lee, B. Tierney, and W. Johnston, "Intra and Interdomain Circuit Provisioning Using the OSCARS Reservation System," *IEEE Broadband Communications Networks and Systems*, pp. 1-8, 2006,.

[14] T. Kudoh, G. Roberts and I. Monga, "Network Services Interface: An Interface for Requesting Dynamic Inter-datacenter Networks," *OFC/NFOEC Technical Digest*, 2013.

[15] "GitHUB repository for OpenNSA," [Online]. Available: https://github.com/NORDUnet/opennsa.

[16] R. Egeland, S. Metson and T. Wildish, "Data transfer infrastructure for CMS data taking," in *XII Advanced Computing and Analysis Techniques in Physics Research*, Erice, Italy, 2008.

[17] V. Lapadatescu, T. Wildish et al, "Integrating Network-Awareness and Network Management into PhEDEx," in *ISGC*, Taipei, Taiwan, 2014.

[18] Z. Jason, R. Ball et al., "The dynes instrument: A description and overview," in *Journal of Physics*, 2012.

[19] J. Zurawski, E. Boyd et al., "Scientific data movement enabled by the DYNES instrument," in *ACM*, 2011.