Named Data Networking Strategies for Improving Large Scientific Data Transfers

Susmit Shannigrahi
Colorado State University
susmit@cs.colostate.edu

Chengyu Fan
Colorado State University
chengyu.fan@colostate.edu

Christos Papadopoulos
Colorado State University
christos@colostate.edu

***Abstract*—Current scientific workflows such as climate science and High Energy Particle Physics (HEP) routinely generate and use large volumes of observed or simulated data. Users of the data are geographically dispersed and often need to transfer large volumes of data over the network for replication, archiving, or local analysis. Scientific communities have built sophisticated applications and dedicated networks to facilitate such data transfers, and yet, users continue to experience failures, delay, and unpredictable transfer latency [19].**

Named Data Networking (NDN) is a new Internet architecture that can provide a much more flexible and intelligent network layer suitable for large data transfers. In this work, we use a real scientific data flow to demonstrate NDN's flexibility and versatility that can make it a suitable choice for large-data workflows. We use deadline-based data transfers as our driving example since they are widely used for HEP data flows [6]. We first discuss several NDN based forwarding strategies that can help such data flows. In addition to using standard forwarding strategies, we propose, at a high level, a bandwidth reservation protocol for NDN and an on-demand high-speed path creation mechanism. Using these as building blocks, we create a deadline-based data transfer protocol and show how NDN can simplify scientific data distribution that currently requires complex applications. Finally, we use a week-long HEP data log to evaluate our protocol analytically.

I. Introduction

Data-intensive science has transformed modern scientific research. Scientists now use observed and simulated data to translate abstract ideas into conclusive findings and concrete solutions. While large scientific datasets help modern scientific communities immensely, the ever-increasing size of these datasets creates a considerable data management burden. Since the data volume is very high, for example, the Large Hadron Collider (LHC) generates petabytes of data per year, accommodating all computation and storage needs at the data generation site is not feasible. Consequently, many scientific workflows routinely need to transfer a substantial amount of scientific data for remote storage, replication, or local analysis that range anywhere from tens of gigabytes to terabytes [14]. While available bandwidth in scientific networks is significant, it is still insufficient. Therefore, such transfers must complete before a deadline to free up network resources for subsequent requests. Today this is often accomplished using complex applications and/or manually orchestrated high-speed paths.

Even with these intelligent applications, completing these transfers within a deadline is challenging. The inherent limitations of TCP/IP networking [19] can slow down transfers and waste valuable network resources. Inability to use multiple data sources at the network layer, failure to reuse in-network data, and lack of access to in-network state are some of the limitations that make big data retrieval inefficient. Scientific communities have tried several approaches to solve this problem; modified congestion control algorithms, smart applications and bandwidth reservations over dedicated links are a few examples. However, these solutions are complex, often domain specific and lack support from the underlying TCP/IP network. Though previous works have looked at mathematically optimizing deadline-based data transfers [11] [6] [18], these solutions are hard to deploy due to the inflexibility of the TCP/IP network layer.

Named Data Networking (NDN) [20] is a new Internet architecture that directly addresses content instead of end-hosts. It offers several optimizations such as request aggregation, in-network caching, and multi-path retrieval, that can significantly enhance large-scale data distribution. Additionally, a forwarding strategy layer in NDN can actively measure network conditions, adapt to changes without involving user applications, and provide intelligent request forwarding. This new network intelligence removes a substantial burden from applications and makes them more straightforward to implement.

In this work, we use deadline-based High Energy Particle Physics (HEP) data transfers as a driving example to demonstrate NDN's flexibility and versatility at the network layer. Note that NDN provides other benefits to scientific data such as provenance and content-centric security; however, for the sake of brevity we do not discuss them here and concentrate solely on data transfer. First, we propose, at a very high level, two protocols for NDN-based bandwidth reservation and on-demand path creation. We demonstrate how NDN can dynamically create strategically placed, in-network caches that can reduce hot spots and network resource consumption. We use these newly-proposed protocols along with two NDN strategies to build a deadline based data transfer solution. Finally, we analyze a one-week long HEP data access log to demonstrate NDN can lower bandwidth consumption by orders of magnitude. In addition to providing a real use case to the NDN community, our work can help HEP communities to address their data distribution problems.

II. Background on Scientific Data Transfers

Previous work has classified scientific data transfers into two broad categories - bulk data transfer and interactive traffic [8]. In this section, we briefly discuss these two data transfer modes and discuss why we only consider bulk data transfer for our work. We then point out the major problems associated with bulk data transfers over TCP/IP networks. Later we use these shortcomings to motivate our NDN based solution.

A. Data transfer modes

Bulk data transfers move a considerable amount of data over long distance links. For some workflows, such data transfers can reach more than a Terabyte per day [2]. Researchers need to transfer data for archiving, replication, or local analysis. Researchers also pre-place data copies around the world for efficient, CDN-like access [2]. Data pre-placement has been particularly popular with communities such as the LHC [2], which routinely places a significant amount of data near the users. Bulk data transfers are not overly sensitive to RTT but require a significant amount of bandwidth for a long time and no packet loss. However, a substantial amount of dedicated bandwidth for an extended period is challenging to acquire on public networks. On these long-distance links, commercial as well as other scientific traffic compete for bandwidth, leading to congestion and packet loss.

The other type of scientific traffic is interactive and comes from applications such as data visualization and audio/video conferencing tools. However, such data flows may be short-lived, personalized, and are often generated on-demand, making caching, aggregation, or scheduling less useful for them. For this study, we do not consider interactive traffic.

B. Problems with traditional Bulk Data Transfers

Currently, there are two ways to download data over IP networks. When data are not very large or the data transfer does not need to meet a deadline, data is requested using HTTP or FTP over a shared network. The network then fetches the data with little guarantee of QoS or performance [19]. When it is critical that requested data reach the requester by a deadline, flows are separated from other traffic using reserved resources. Reservations include router resources such as queue capacity and interfaces as well as the capacity of network links. For this work, we refer this traffic as resource-reserved traffic. However, a reservation does not eliminate the underlying architectural shortcomings of TCP/IP. In this section, we discuss these shortcomings in the context of large scientific data flows.

End-to-end paradigm unable to efficiently use network resources: In IP, all data transfers are between two end hosts. This end-to-end model means two hosts must download the same data separately even if they share the same network path. A large number of duplicate packets wastes network bandwidth. The End-to-end networking paradigm cannot efficiently use reservations either. Though scientific communities use reserved bandwidth channels regularly to avoid congestion and packet loss, such channels are end-to-end tunnels, and neither the channels nor the data flowing through them are reusable. The inability to reuse network resources and data leads to lower network utilization.

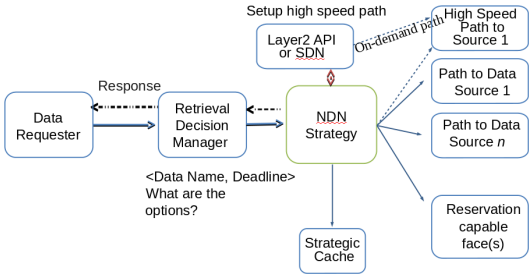

Fig. 1: Building blocks for our protocol

Stateless forwarding cannot adapt to changes: Traditional IP networks do not keep any state in the network but simply forward packets which means they are unable to react to changing network conditions. In case of failure or service degradation, data transfers continue to use the same path unless there is some external intervention. Scientific communities deployed intelligent applications and middleware solutions that keep track of the network performance. However, these solutions increase the complexity of the applications.

Inability to use multiple paths: Even when multiple data sources are available, IP cannot utilize them simultaneously at the network layer. Though the networking community has proposed workarounds such as multiple connections at the application layer or multipath TCP [3] at the transport layer, these approaches still require knowledge of the underlying network.

TCP congestion control interferes with transfer speed: For a large bandwidth link, losing even one in hundreds of thousands of packets can dramatically reduce transfer speed [5]. Long distance, best effort scientific traffic often encounters bottlenecks and congestion in the network. To circumvent congestion in public networks, scientific communities have built dedicated science networks such as the LHC Optical Private Network (LHCOPN) [12] and ESNet [1] that provide dedicated paths for science data. While dedicated networks can remove some of the uncertainties of public networks, scientific traffic flowing over these networks may still encounter congestion and packet loss, especially when other scientific flows are competing for resources.

III. Building Blocks of an NDN Based Large Data Transfer Protocol

In this section, we present a high-level overview of two new protocols, an NDN-based Bandwidth Reservation Protocol and an NDN-based circuit-creation protocol. Besides, we discuss two NDN forwarding strategies that we later use. Note that while we describe these strategies separately, they can be integrated into a single larger strategy. Descriptions of these constructs are deliberately high-level since the goal of this paper is not to present the protocol specifications but to simply demonstrate NDN's capabilities as the network layer.

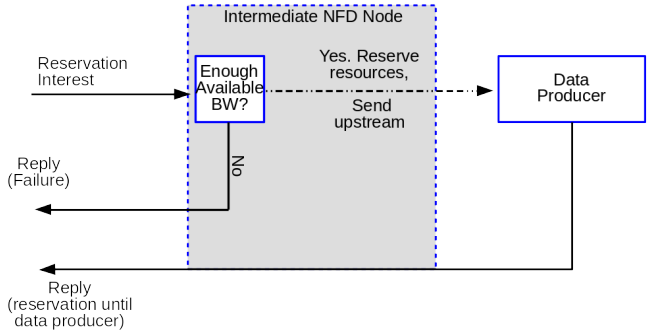

Fig. 2: Reservation with NDN

ReqID	Prefix	Requested StartTime	Deadline	BW
1	/xrootd	1463330393	1463355592	1Gbps
2	/xrootd	1463330519	1463355623	1Gbps

TABLE I: Scheduling Table

A. Bandwidth Reservation Protocol

Satisfying deadlines in a Grid environment often requires dedicated per-flow bandwidth allocation [6]. In this work, we propose a protocol to create hop by hop reservations in an NDN network. To set up such reservations an NDN node sends a special reservation Interest that is forwarded hop-by-hop upstream. The reservation request is represented by a tuple containing <data name, requested bandwidth, start_time, and a deadline>. Each node has a reservation manager that checks if the requested bandwidth is available during the requested period. If the request is successful, the reservation manager forwards the Interest upstream. Upon reaching a data producer or repo, the Interest brings back a reply with a success message. We show this protocol in Fig. 2. The reservation manager keeps track of the reservation using a reservation table similar to TABLE I. While similar to RSVP [21], there are two important differences between our reservation protocol and RSVP: (a) Our reservation is per name prefix, not end-to-end. Per-prefix reservation means transfers can share the reservation as long as they share some of the network paths, and (b) the NDN reservation manager can aggregate requests for the same content if they fall within a common deadline, a feature not currently available in RSVP.

B. Integrating Dynamic Path Creation with NDN Strategies

Sometimes existing bandwidth is simply not enough to satisfy a request deadline. At the same time, creating permanent, high-bandwidth paths between all sites might not be economically feasible. In addition, since scientific data flows are often bursty [17], creating such permanent paths might not be optimal. The scientific communities solve this short-term bandwidth shortage problem by creating temporary, high-bandwidth paths for the large data transfers. ESnet's On-Demand Secure Circuits and Advance Reservation System (OSCARS) [9] is a service that allows users to create such guaranteed bandwidth reserved paths. Authenticated users can request a reserved circuit between two end-points. However, users are still responsible for knowing the endpoints, creating paths, and scheduling transfers. Arbitrary creation of reserved paths may create conflict between users, and the network resources may not be optimally utilized. We argue that the network, not the users, should be in charge of creating network paths. A network-driven approach to reservations means the user does not need know how much bandwidth is needed. The user can simply specify a transfer deadline and the network then finds ways to fulfill the request. Our protocol allows NDN to provide a network-driven approach to path creation. In our implementation, an NDN strategy invokes a path creation mechanism when the available bandwidth on existing paths are not sufficient to meet a deadline. Our strategy works with OSCARs to create a high-speed path and if the path creation succeeds, modifies the FIB to add a new route. Since NDN consolidates requests for the same data, a high-speed path can potentially speed up other best-effort traffic if such flows are temporally close to the high-speed flow. Note that though we have used OSCARS for this study, strategies do not need to be constrained to using a specific lower layer protocol. They can interact with any lower or upper layer protocols; for example, our strategy can interact with an SDN controller to create a similar high-speed path.

C. Delay-based Forwarding Strategy

NDN contains forwarding strategies that record RTT on each outgoing link. On receiving the first Interest for a namespace (e.g., $/xrootd/data1$), the strategy sends it over all matching interfaces. Once data comes back, it records the RTT of each incoming Data packet before forwarding it downstream. It then ranks the faces based on RTT and uses that ranking for forwarding subsequent Interests. Periodically the strategy tries out other, lower ranked interfaces and as Interest/Data exchange continues the strategy adjusts the ranking based on observed RTT. At a given time, this strategy tries to choose the lowest latency path for data retrieval.

D. Multi-path Strategy

If multiple producers are reachable from a node, NDN can use these routes simultaneously for data retrieval. In our example, we created a strategy that can send Interests over multiple links. This strategy can also forward Interests based on the Interface ranking. For example, if two routes are available, NFD might forward 75% of the Interests to route one and the rest on route two based on their ranking. We use available bandwidth, and RTT as the metrics for initial Interface ranking and it is adjusted based on ongoing communication. Unlike the previous strategy, this strategy utilizes multiple links for data transfer.

E. Namespace

NDN names are hierarchical, which aligns well with existing scientific namespaces. We have shown in previous works [7] [13] that many scientific domains already use hierarchical names that we can use unmodified or with minor changes. In this work, we use xrootd [4], a data management

tool for High Energy Physics datasets (HEP) as an example. We assume data is published under a root prefix, e.g., $/xrootd$. Different sub-namespaces under the root prefix identifies data and various services. Fig. 3 shows such an example; actual data is served under $/xrootd/data$ while two other services are advertised under $/xrootd/query$ and $/xrootd/reservation$. The first service allows applications to query the current network state for the $/xrootd$ prefix. The second one provides a reservation service that a strategy can use to set up a reserved path for $/xrootd/data$.

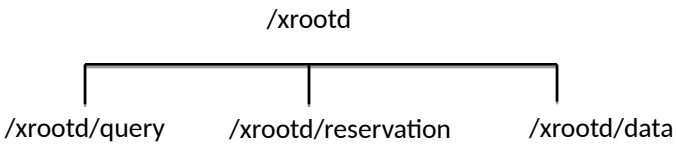

Fig. 3: Namespace Design

IV. A Deadline Based Data Transfer Protocol: Design and Implementation

In the previous section, we discussed several NDN based building blocks that can help create an intelligent data transfer protocol. In this section, we use those primitives to design a deadline-based data transfer protocol that can be used for both best effort and reserved bandwidth data transfers.

Our protocol has three main components in our reference implementation; a data requester/client, a per node retrieval decision manager, and custom NDN strategies. The forwarding strategy controls intelligent Interest forwarding decisions and if necessary, reserves bandwidth, creates in-network strategic caches and interacts with upper or lower layer protocols for dynamic path creation. The retrieval manager acts as an intermediary between the client and the strategy. In addition to communicating with strategies and the clients, the retrieval agent can work with a policy module for enforcing retrieval or reservation quotas. Policies are needed to ensure applications do not set impossible deadlines and force the network to use dedicated paths for all transfers.

A. Interaction between components

In our protocol, the client uses an Interest packet for sending the following parameters to the retrieval manager: <**name of data, the deadline for retrieval, size of the dataset, hard/soft deadline flag**>. The name of the dataset defines the requested dataset, retrieval deadline denotes the latest acceptable time for data retrieval, and a "hard deadline" flag means the deadline is non-negotiable while a "soft deadline" flag denotes best effort traffic. The Interest also tells the retrieval manager the size of the data. While estimating dataset size is not easy for general Internet traffic, scientific data sizes are usually recorded in a catalog and therefore, relatively easy to estimate. We have described such a distributed scientific catalog in our previous work [7].

If the retrieval manager sees two or more requests can be aggregated without violating the deadlines, it simply aggregates them and suggests a start time to the clients. Clients start sending Interests at the prescribed time. Otherwise, the retrieval manager talks to the local NFD about possible retrieval options using a special query namespace, $/xrootd/query$ in our study. For a request for $/xrootd/data1$, the retrieval manager sends a query Interest to $/xrootd/query$ along with the original request as a name component, e.g, $/xrootd/query/$ $</xrootd/data1/start\_time/deadline/deadline\_type>$.

B. Fulfilling requests with soft deadlines

Figure 4 shows the decision path for such Interests. If the deadline type is soft, our custom NDN strategy looks up the list of faces in the FIB that can be used for retrieving $/xrootd/data1$. The strategy may have several options for completing the transfers. It can use a single path or multiple paths simultaneously for the transfer. If the Interest is under a namespace that was not previously used for data retrieval, the strategy fetches a few chunks using each matching face and records the following information for each face: <FaceID, RTT, Max Bandwidth>. The strategy then estimates if it is possible to retrieve the data within the given deadline and sends back the information to the retrieval manager, which in-turn notifies the client. Instead of sending binary yes/no response to the client, our protocol replies with more detailed return values along with a suggested start time; for our implementation they are (a) the request can be satisfied, at which point the client starts retrieval immediately; (b) the retrieval can not be satisfied within the given deadline, but the data can be retrieved with an extended deadline; if the new deadline is acceptable, the client adjusts the deadline and requests again; (c) the retrieval can be aggregated, so the client starts retrieval at the suggested time; and (d) the request can not be satisfied. Such finer-grained information enables the clients to make intelligent decisions about retrieving data, a feature that is not available today.

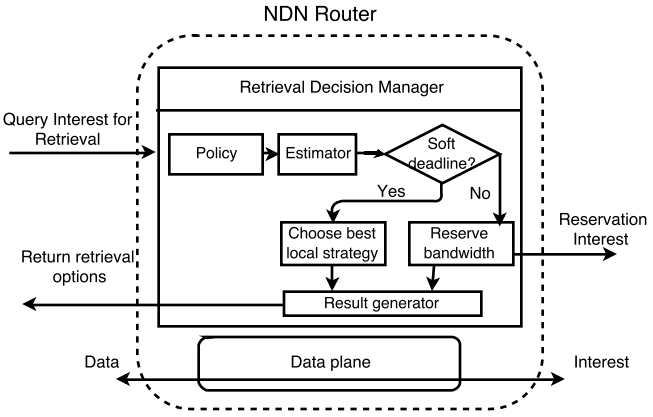

Fig. 4: Strategy Decision

C. Reserved bandwidth path for hard deadlines and Strategic in-network caching

While strategies such as multi-path transfer work well for best-effort traffic, the only way to guarantee timely

completion of large data transfers is to create a reserved bandwidth path [19]. In case the deadline is "hard", the strategy must create a reserved path to a data source or a cache. In our implementation, we send a reservation Interest using a special namespace,$/xrootd/reservation/$. One such reservation Interest looks like $/xrootd/reservation/$ $</xrootd/data1/start\_time/deadline/bandwidth>$. On receiving this Interest, a node reserves the appropriate amount of bandwidth for future incoming transfer. If the reservation is successful, it then forwards the reservation Interest upstream. If the node is the data source, then it returns a success message.

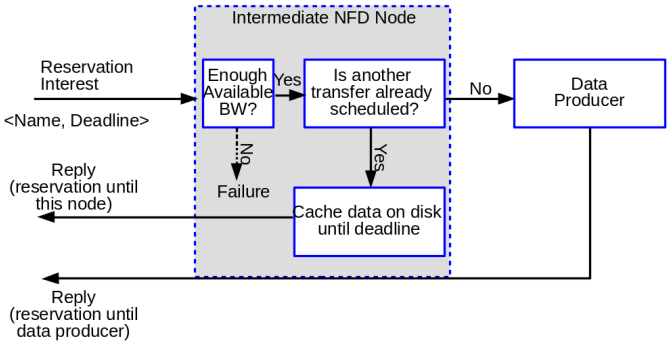

Fig. 5: Using Reservation with Strategic Caching

The reservation protocol can be tweaked to create intelligent data dissemination strategy. We demonstrate this with an example in Fig. 5. If a new request overlaps with an existing one, our strategy merges the requests and sends back a reply indicating success and the time of the reservation. Note that in this case, the reserved path is created only between the client and the replying node. In addition to performing this simple aggregation, an intelligent strategy can create a temporary strategic cache in the intermediate nodes. For example, a node may decide to cache data for $/xrootd$ until $t_{deadline}$ if there are n requests scheduled between now and time $t_{deadline}$. Since scientific datasets show a high degree of temporal locality [15], we can assume in-network strategic caching would be helpful for these data flows.

Our method has two benefits for scientific datasets; unlike today, an end-to-end per-client path reservation is not required thus freeing up network resources. Second, our strategy can dynamically create in-network caches without requiring prior planning and operator intervention.

D. Interacting with other layers

If none of the existing options are fast enough for a data transfer, a new high-speed path must be created for the request to be fulfilled. We built an NDN strategy that works with Layer 2 protocols (or SDN) to create such a path. In this work, our prototype implementation interfaces with OSCARS [10] to set up reserved paths. However, our strategy does not depend on OSCARs and can use any upper or lower layer API to create a high-speed path. Once the high-speed path is set up, the strategy uses it as simply another available link.

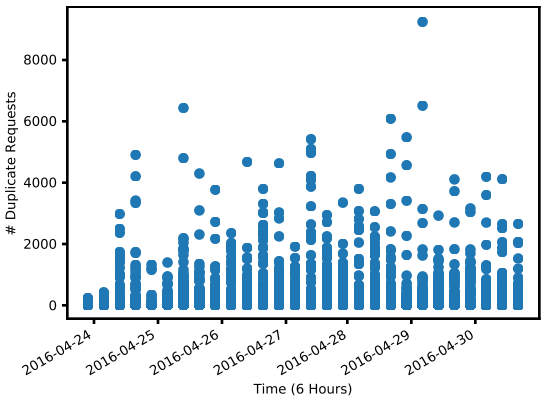

Fig. 6: Duplicate requests for individual datasets over time. Each dot represents a unique dataset in a 6 hour window

V. Evaluation

The deadline-based data transfer protocol we described optimizes network usage by aggregating requests, caching, and intelligent request scheduling. We are working on implementing and evaluating our strategy we described in the previous sections. In this section, we analytically explore how much bandwidth our NDN-based protocol can save for a sample HEP data flow. We analyze a xrootd [4] access log recorded over seven days from Apr 23th, 2016 to Apr 30, 2016. The logs had 114K unique requests from 267 users, recorded at a ten-minute interval. The access logs showed a high degree of duplicate requests; users requested only 1871 unique datasets over 114K requests; so on average, each dataset was requested sixty times.

Duplicate requests can occur in xrootd for popular datasets or if transfers fail, which then automatically triggers another request for the same data. NDN can optimize HEP data flow by de-duplicating requests using our deadline based protocol. For combining the requests, we introduce a "scheduling window"; requests falling within this window can potentially be combined.

To investigate temporal locality within our scheduling window, we first separated the requests in (arbitrary) six-hour bins. The actual window will depend on network capacity, storage, and individual workflows. Fig. 6 shows the number of duplicate requests over time; each dot represents the number of requests for a specific dataset in a six-hour window. We find that many datasets were requested several thousand of times and over 50% of the requests have one or more follow-up request(s) within 10 minutes. Having so many duplicate requests in the log is good news for our strategy since they can be effectively combined.

We assume each request was for a 2GB file, the average file size in xrootd [16]. Intelligent request scheduling allows NDN to retrieve only one copy of the data that satisfies all requests for the same data. To compare IP's bandwidth consumption with NDN we first calculate the amount of bandwidth needed for each request and then calculate the total aggregate bandwidth required to serve all requests over six-hour periods.

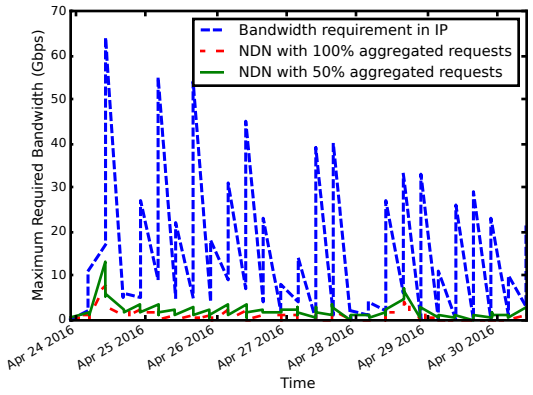

Fig. 7: Reduced Bandwidth Consumption with NDN

Fig. 7 shows that the total bandwidth requirement for xrootd is very high, with peaks at approximately 64 Gbps. We also calculated how much bandwidth NDN could save compared to IP when requests are aggregated. We take the same requests over 6 hours, de-duplicate the requests and calculate the total bandwidth requirement.

Fig. 7 shows overall bandwidth demand of the system in two scenarios. First one is the best case; if we can aggregate all duplicate requests over a 6-hour period, the max bandwidth requirement drops from 64 Gbps to around 8.2 Gbps, an 85% reduction. However, we acknowledge that not all requests can be aggregated; some requests might have very tight deadlines and need to be served immediately. Even if we assume only 50% of the duplicate requests can be aggregated using our protocol, the bandwidth requirement comes down to 13.2 Gbps, a 79% reduction. This result shows that even some degree of de-duplication can go a long way in reducing resource consumption for HEP data flows.

While our result demonstrates the improvements an NDN based intelligent network layer can bring, we did not evaluate bandwidth reservation or strategic caching in this work. We are optimistic that incorporating these features will improve our example data flow even more. We will investigate them in future work.

VI. Conclusions and Future Work

In this work, we used deadline-based data transfers as the driving example to demonstrate that an NDN based network layer can be more flexible and versatile compared to the current IP networks. Additionally, we proposed an RSVP-like bandwidth reservation protocol for NDN. Using in-network strategic caching and bandwidth reservation up to the cache, we show that NDN does not always require end-to-end reserved paths for high-speed transfers. Instead, a reserved path to the nearest cache can speed up transfers and at the same time, improve network utilization. We also present a mechanism that enables the NDN strategies to interact with lower layer protocols to set up dedicated paths for large data transfers. Finally, we used a real HEP access log to demonstrate that our NDN based protocol can potentially cut down bandwidth consumption by over 75% for this particular example. We acknowledge that our study is preliminary and can be improved in several ways. We are working on improving the implementation to include in-network strategic caching and the NDN-based reservation protocol. Once these pieces are implemented, we plan to evaluate our complete protocol using a real topology and more detailed access logs.

References

[1] ESNet. https://fasterdata.es.net.

[2] A. Barczyk. World-wide networking for lhc data processing. In *National Fiber Optic Engineers Conference*, pages NTu1E–1. Optical Society of America, 2012.

[3] S. Barré, C. Paasch, and O. Bonaventure. Multipath tcp: from theory to practice. *NETWORKING 2011*, pages 444–457, 2011.

[4] L. Bauerdick, K. Bloom, B. Bockelman, D. Bradley, S. Dasu, I. Sfiligoi, A. Tadel, M. Tadel, F. Wuerthwein, and A. Yagil. Xrootd monitoring for the cms experiment. In *Journal of Physics: Conference Series*, volume 396, page 042058. IOP Publishing, 2012.

[5] Brian Tierney and Joe Metzger, ESnet. High Performance Bulk Data Transfer. https://fasterdata.es.net/assets/fasterdata/JT-201010.pdf.

[6] B. B. Chen and P. V.-B. Primet. Scheduling deadline-constrained bulk data transfers to minimize network congestion. In *Cluster Computing and the Grid, 2007. CCGRID 2007. Seventh IEEE International Symposium on*, pages 410–417. IEEE, 2007.

[7] C. Fan, S. Shannigrahi, S. DiBenedetto, C. Olschanowsky, C. Papadopoulos, and H. Newman. Managing scientific data with named data networking. In *Proceedings of the Fifth International Workshop on Network-Aware Data Management*, page 1. ACM, 2015.

[8] I. Foster, M. Fidler, A. Roy, V. Sander, and L. Winkler. End-to-end quality of service for high-end applications. *Computer Communications*, 27(14):1375–1388, 2004.

[9] C. Guok. A user driven dynamic circuit network implementation. *Lawrence Berkeley National Laboratory*, 2009.

[10] C. Guok, E. N. Engineer, and D. Robertson. Esnet on-demand secure circuits and advance reservation system (oscars). *Internet2 Joint*, 2006.

[11] W. Lu, Z. Zhu, and B. Mukherjee. Optimizing deadline-driven bulk-data transfer to revitalize spectrum fragments in eons. *J. Opt. Commun. Netw.*, (12):B173–B183, 2015.

[12] E. Martelli and S. Stancu. Lhcopn and lhcone: Status and future evolution. In *Journal of Physics: Conference Series*, volume 664, page 052025. IOP Publishing, 2015.

[13] C. Olschanowsky, S. Shannigrahi, and C. Papadopoulos. Supporting climate research using named data networking. In *Local & Metropolitan Area Networks (LANMAN), 2014 IEEE 20th International Workshop on*, pages 1–6. IEEE, 2014.

[14] J. Rehn, T. Barrass, D. Bonacorsi, J. Hernandez, I. Semeniouk, L. Tuura, and Y. Wu. Phedex high-throughput data transfer management system. *Computing in High Energy and Nuclear Physics (CHEP) 2006*, 2006.

[15] S. Shannigrahi, C. Fan, and C. Papadopoulos. Request aggregation, caching, and forwarding strategies for improving large climate data distribution with ndn: A case study. In *To Appear in ICN17*, 2017.

[16] S. Shannigrahi, C. Papadopoulos, E. Yeh, H. Newman, A. J. Barczyk, R. Liu, A. Sim, A. Mughal, I. Monga, J.-R. Vlimant, et al. Named data networking in climate research and hep applications. In *Journal of Physics: Conference Series*, volume 664, page 052033. IOP Publishing, 2015.

[17] A. Shoshani and D. Rotem. *Scientific Data Management: Challenges, Technology, and Deployment*. CRC Press, 2009.

[18] S. M. Srinivasan, T. Truong-Huu, and M. Gurusamy. Flexible bandwidth allocation for big data transfer with deadline constraints. In *Computers and Communications (ISCC), 2017 IEEE Symposium on*, pages 347–352. IEEE, 2017.

[19] B. Tierney, E. Kissel, M. Swany, and E. Pouyoul. Efficient data transfer protocols for big data. In *E-Science (e-Science), 2012 IEEE 8th International Conference on*, pages 1–9. IEEE, 2012.

[20] L. Zhang, A. Afanasyev, J. Burke, V. Jacobson, P. Crowley, C. Papadopoulos, L. Wang, B. Zhang, et al. Named data networking. *ACM SIGCOMM Computer Communication Review*, 44(3):66–73, 2014.

[21] L. Zhang, S. Deering, D. Estrin, S. Shenker, and D. Zappala. Rsvp: A new resource reservation protocol. *Network, IEEE*, 7(5):8–18, 1993.