Experiences in deploying in-network data caches

Alex Sim[1,*], *Ezra* Kissel[2,**], *Damian* Hazen[2,***], and *Chin* Guok[2,****]

[1]Lawrence Berkeley National Laboratory; 1 Cyclotron Road, Berkeley, CA 94720
[2]Energy Sciences Network; 1 Cyclotron Road, Berkeley, CA 94720

Abstract. Data caches of various forms have been widely deployed in the context of commercial and research and education networks, but their common positioning at the Edge limits their utility from a network operator perspective. When deployed outside the network core, providers lack visibility to make decisions or apply traffic engineering based on data access patterns and caching node location. As an alternative, in-network caching provides a different type of content delivery network for scientific data infrastructure, supporting on-demand temporary caching service. We will describe the status of in-network caching nodes deployed within ESnet in support of the US CMS data federation. We will describe the container and networking architecture used to deploy data caches within ESnet, and update on the evolving tooling around service management lifecycle. An analysis of cache usage will also be provided along with an outlook for expanding the in-network cache footprint.

1 Introduction

With advances in instruments and computing hardware, scientific experiments and simulations generate an increasing amount of data, and that data is frequently shared among geographically distributed users. Existing datasets and the cost of the storage hardware and its maintenance limit the number of data sources, and data distribution requires higher network bandwidth for timely data delivery. Some datasets are popular among users and transferred multiple times to the same institution or institutions in the same region for the different users as well as for the same user for various reasons. Also, users move the data from the data sources to multiple computing locations depending on when the computing jobs run for the user. Some type of content delivery network or hierarchical data distribution infrastructure can accommodate sharing data among geographically distributed users by pre-staging popular datasets that many users might use in the same region or same institution.

In-network caching provides a different type of content delivery network for scientific data infrastructure, supporting on-demand temporary caching service. It also enables a network provider to design data hotspots into the network topology, and to manage traffic movement and congestion by data-driven traffic engineering. A regional in-network caching strategy would also reduce the data access latency for the users and increase the overall computing application performance. For network providers, in-network caching service would decrease

*e-mail: asim@lbl.gov
**e-mail: kissel@es.net
***e-mail: dhazen@es.net
****e-mail: chin@es.net

traffic bandwidth demands on busy links. This is especially relevant to the High Energy Physics (HEP) community with the LHC instrument at CERN and Tier-1 sites for ATLAS at Brookhaven National Laboratory and CMS experiments at Fermi National Accelerator Laboratory.

For example, the Southern California Petabyte Scale Cache (SoCal Cache) [1] based on XCache [2] consists of 23 data cache nodes with approximately 2PB of storage space, supporting client computing jobs for High-Luminosity Large Hadron Collider (HL-LHC) analysis in Southern California. The SoCal Cache has cache nodes at the ESnet junction in Sunnyvale, at Caltech, and at UCSD. The system has been used by the CMS collaboration for real CMS data analysis as part of the Caltech and UCSD Tier-2 center production infrastructure. We observed that 85.5% of the requests and 94% of the requested bytes are cache hits, sharing datasets in the regional data cache from Sep. 2022 to July 2023. Studying the characteristics of the data access patterns [3] has enabled new strategies for how the needed resources such as compute, storage, and network can be allocated.

In this paper, we build upon our prior analysis work [3] by providing an update on the existing caching infrastructure, and we include sample analysis results from HEP jobs using the SoCal Cache. We introduce DTN-as-a-Service (DTNasS) as our approach for hosting in-network caching and storage services for scientific datasets within ESnet, and we describe our methodology and experiences in deploying new caching hardware within ESnet points-of-presence. Finally, we discuss future deployment opportunities driven by our data-driven analysis and flexible provisioning framework.

2 Background

2.1 Energy Sciences Network (ESnet)

The Energy Sciences Network (ESnet) is the US Dept of Energy (DOE) Office of Science's high-performance network user facility, delivering highly-reliable data transport capabilities optimized for the requirements of large-scale science. ESnet is stewarded by the Advanced Scientific Computing Research Program (ASCR) and managed and operated by the Scientific Networking Division at Lawrence Berkeley National Laboratory (LBNL). ESnet acts as the primary data circulatory system for science by interconnecting the DOE's national laboratory system, dozens of other DOE sites, and 150 research and commercial networks around the world. It allows tens of thousands of scientists at DOE laboratories and academic institutions across the country to transfer vast data streams and access remote research resources in real-time. ESnet exists to provide the specialized networking infrastructure and services required by the national laboratories, large science collaborations, DOE user facilities, and the DOE research community. All together, ESnet provides a foundation for the nation's scientists to collaborate on some of the world's most important scientific challenges, including energy, biosciences, materials, and the origins of the universe. Science data traffic across its network has grown at around 60% each year, and traffic has exceeded an exabyte per year since 2019.

2.2 High Energy Physics (HEP)

The HEP experiments have been generating large volume of data [4], especially from the LHC in Switzerland. Experiments such as ATLAS and CMS have thousands of globally geographically distributed collaborations, and the efficient data distribution infrastructure has been explored for a long time, delivering Petabytes of data. The LHC community has been preparing the increase in annual data volume, and one of the approaches is the data replication between regional "Data Lakes" [5] and the mixture of remote access and caching within those lakes. A regional data lake is expected to serve multiple computing centers within that region.

2.3 DTN-as-a-Service (DTNaaS)

Both the Open Science Grid (OSG) [6] and broader HEP community have embraced software containers as part of their data and analysis pipelines where container deployment solutions such as HELM+Kubernetes [7] and SLATE-CI [8] have become commonplace. As a result, a number of existing service container workflows (often using Docker and Docker Hub) are being maintained in public repositories and container registries. Open source software and tooling around containers is also feature-rich and improving with time. In contrast with virtual machines (VMs), containers are generally a lighter-weight option that support the execution environment for single services, and have benefits when ease of software packaging, distribution, and upgrade cycles are considered.

The deployment we describe makes use of a platform known as DTN-as-a-Service (DTNaaS) [9] being developed within ESnet, which provides a tailored container orchestration framework focused on the configuration and tuning options relevant to high-performance data movement services. Underpinning the design philosophy was a recognition that more general, feature-rich platforms such as Kubernetes may not always provide an ideal interface for single-service container instances where automated resource scaling and migration are not primary concerns. The ability to control specific features at the container level, such as attaching multiple network interfaces (and interface types), and dual-stack configurations for IPv4 and IPv6 at a fine granularity with minimal setup cost, were primary design goals. While the initial focus has been on traditional Data Transfer Node (DTN) software endpoints, the framework has been designed to be flexible enough to accommodate many types of software containers that have high-throughput and advanced networking configuration requirements. Thus, a DTNaaS solution that can support performance-oriented data mover services with modest support overheads was an ideal fit for deploying OSG XCache software containers from an "in-the-network" perspective.

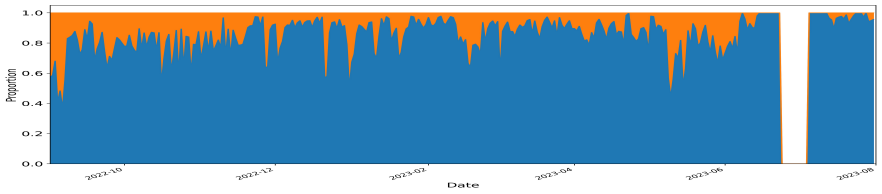
3 Cache Utilization

We collected data access measurements from the SoCal Cache between Sep. 2022 and July 2023, where HL-LHC analysis jobs requested data files for users. The SoCal Cache has 23 cache nodes regionally at Caltech, UCSD and ESnet, consisting of approximately 2PB of storage. We studied how much data is shared, how much network traffic volume is consequently saved, and how much the in-network data cache contributes to the resource management and performance. Additionally, we analyzed data access patterns, and it may show a few characteristics of the data sharing, cache utilization, and network utilization where shared data directly contributes to the network traffic savings.

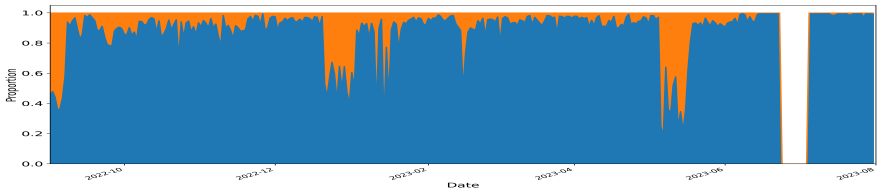
Table 1: Summary statistics for data accesses from Sep. 2022 to July 2023

	# of accesses	cache hit size (TB)	cache miss size (TB)	number of cache hits	number of cache misses
Total	5,889,264	10,824.09	690.75	5,038,749	850,515
Daily	18,233	33.41	2.14	15,599	2,633

Table 1 shows the basic statistics on the data access activities for all caching nodes during the study period (from Sep. 2022 to July 2023). "Cache hits" in Table 1 represent when a data access request could be satisfied with a file in the cache and the data file was shared from the cache. The shared data accesses correspond to the network traffic savings. "Cache misses" mean that any caching nodes in the SoCal Cache did not have the data, resulting in a data transfer from the remote data source to one of the caching nodes. In Table 1, cache hits show about 85.5% of the file requests and 94.0% of the total data volume. The percentage of

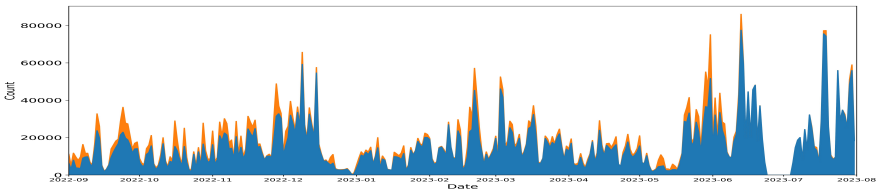


(a) Daily proportion of cache hit counts and cache miss counts. Overall, 85.5% of the total accesses has been satisfied by the cache.

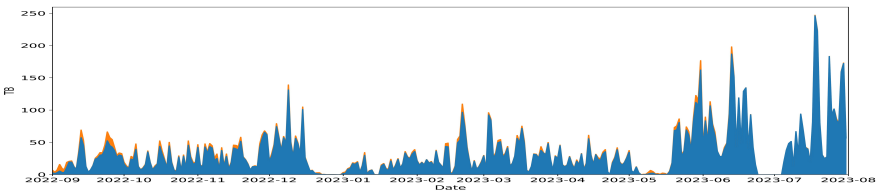


(b) Daily proportion of cache hit volume and cache miss volume. Overall, 94% of the total traffic volume has been saved from the cache.

Figure 1: Daily proportion of cache hits (in blue) and cache misses (in orange) from Sep. 2022 to July 2023.



(a) Daily counts of cache hits and cache misses. The number of accesses went over 50,000 on some days.



(b) Daily volume of cache hits and cache misses. The data volume went over 100TB on some days.

Figure 2: Daily accesses of cache hits (in blue) and cache misses (in orange) from Sep. 2022 to July 2023.

cache hits volume depends on the caching policy and the storage capacity. We have observed previously that the percentage of cache hits volume decreases when new cache nodes are added because the requests would fill the new cache nodes first by the policy [10]. Also, the percentage of the cache hits volume would be maintained at a certain level after the cache nodes are full.

Figure 1a shows the daily rates, based on the data request counts, of cache hits (in blue) and cache misses (in orange) from Sep. 2022 to July 2023. Figure 1b shows the daily proportion of the cache misses (in orange) and cache hits (in blue), and also shows that there are a large portion of the cache hits in the daily accesses indicating network traffic savings.

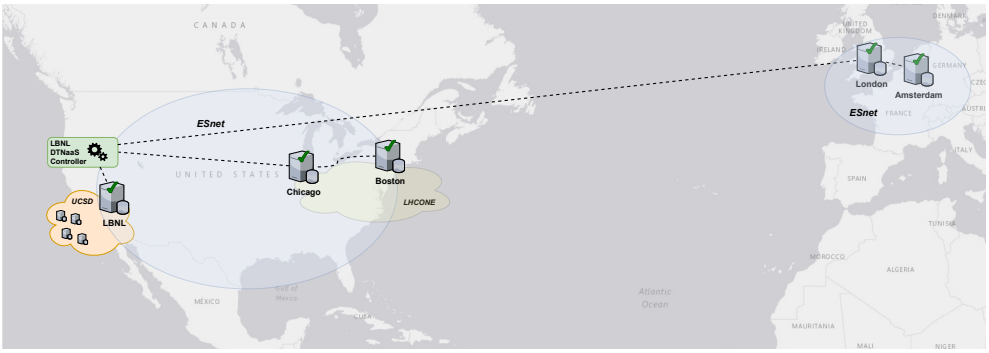


Figure 3: Deployment of caches within ESnet with three initial caching node deployed in the US at LBNL, Boston, and Chicago. The London and Amsterdam deployments were completed in 2023. Container services are managed by a DTNaaS controller instance located at ESnet's LBNL datacenter.

Figure 2 shows more information about the data requests during the study period. Figure 2a shows that the daily number of file requests separating into cache hits (in blue) could be satisfied with files in the cache and cache misses (in orange) require wide-area data transfers. Figure 2b shows the daily volume of data requests.

Monitoring issue was observed for a few days in June and July of 2023, and also we observe that the majority of requests is cache hits and much higher data accesses are observed during the last two months of the study period. Understanding the impact of these usage spikes in the longer term would be good for provisioning future deployment of caches.

4 Cache Node Deployments

Two additional caching nodes have been deployed within ESnet to supplement the existing Sunnyvale node within the broader SoCal Cache (Fig. 3). These nodes were installed as physical servers at ESnet points of presence (PoP) in Boston and Chicago to serve as regional caches for CMS. The specifications of each node is as follows: dual-socket Intel Xeon Gold 5220S CPUs, 384GB RAM, 12x Micron 9300 PRO 15.36TB NVMe SSDs, and Mellanox ConnectX-5 100G network interface cards (NICs). The node storage was configured to provide an effective 165TB of available capacity each while providing suitable performance to match the expected XCache usage given 100G network connectivity.

Flexible tooling which enables deployments of diverse higher performance containerized workloads is one of the design goals of the DTNaaS service effort. This ability is demonstrated in two additional node deployments in Amsterdam and London. The nodes will initially support regional caching for the Laser Interferometer Gravitational-Wave Observatory (LIGO)[1] which has detectors located in Hanford, Washington and Livingston, Louisiana. The systems have 20 15.36 TB drives with a usable cache capacity of 280 TB and 512GB of RAM. The CPUs and NICs for the system builds are the same as those in the Boston and Chicago nodes. The Amsterdam and London deployments run the Open Science Data Federation (OSDF) StashCache[2] software. The nodes are integrated into the OSDF caching federation which authorizes cache access by maintaining Virtual Organizations of project collaborators. In addition to running the containerized caching software, the deployment includes the OSG Shoveler [3] software which was developed to improve logging reliability.

[1] LIGO: https://www.ligo.caltech.edu

[2] StashCache: https://osg-htc.org/docs/data/stashcache/run-stashcache-container

[3] Shoveler: https://osg-htc.org/docs/data/xrootd/install-shoveler

Part of our strategy has been to explore the logistics of hosting data movement services within an international science network such as ESnet where *networking* services, not *application* services, have been the traditional offering. The development of DTNaaS is one ongoing effort to help bridge this technology gap, and below we describe some of the architectural and technical decisions that have been put into practice.

4.1 Network Connectivity and Security

A typical DTNaaS deployment relies on a centralized controller that contacts one or more nodes to provision and manage service containers. The controller runs in a data center environment while the DTNaaS service nodes are located at a wide-area network point-of-presence. The low-bandwidth communication between the controller and service nodes forms the management network and is achieved using a secured, dedicated control interface as diagrammed in Figure 4. A number of agent processes are run in containers to provide the DTNaaS management functionality over this control connection, and their access is facilitated by using a standard Docker bridge network with processes binding to localhost ([::1]). A reverse proxy provides encrypted external endpoints to these locally bound processes via the host networking namespace.

The high-speed dataplane is realized through physical connections from the node's Mellanox NIC ports directly to ESnet routers, and flows originating from or terminating at the XCache service container make use of these links. Key requirements for these connections included effective network isolation from the baremetal host OS and low virtualization overhead to achieve line rate performance. To that end, *macvlan* container networks were configured in 802.1q trunk bridge mode and attached to the parent VLAN-tagged service interfaces instantiated in the host namespace. The DTNaaS controller was then able to map the dataplane networks to the XCache container configuration and automatically expose the *macvlan* interfaces to each running container as appropriate.

Generally, the Linux *macvlan* sub-interface type provides isolation from the host networking namespace at layer 2. From an external ESnet switch or router perspective, a container with a macvlan interface will appear as another unique host on the network with a unique MAC and/or IP address. For the baremetal host to communicate with the running container, another *macvlan* sub-interface must be created on the host and local routing adjusted for any configured layer 3 subnet(s). Another container may also communicate within the *macvlan* subnet if attached to the same container network. In this manner DTNaaS service containers provide separation between networks allocated for each service container and reduce the complexity of a single filtering rule set maintained alongside the host network namespace.

The DTNaaS management network is secured using a combination of the reverse proxy service and host-based filtering using standard *iptables* rules. In addition, our deployment makes use of network ACLs that are implemented on the ESnet router platform to provide ingress and egress filtering for both the control and dataplane interfaces on each node. The routing instances exposed to the XCache container via the *macvlan* networks have unique ACLs applied; for example, only the XCache TCP port is allowed for ingress in the LHCONE instance. This ability to rely on the network infrastructure to provide packet filtering simplifies any otherwise necessary host and container configuration considerably.

Finally, our deployment involves the requirement that our XCache containers are running both in a dual-home and dual-stack configuration. As shown in Figure 4, two separate routing instances are exposed to the containers via *macvlan* interfaces: 1) a global routing service to provide a default route and support common networking tasks such as name resolution, and 2) an LHCONE L3VPN service to provide connectivity to other HEP data sources and peers. In

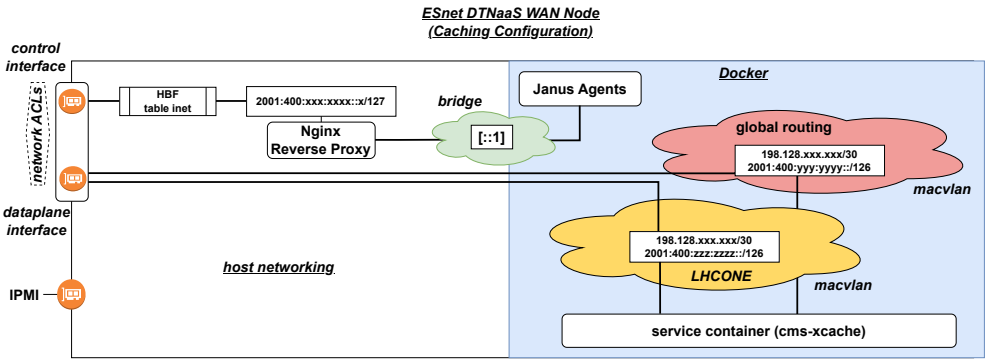


Figure 4: The DTNaaS networking configuration for cms-xcache deployments. Each service container is dual-homed and dual-stacked (IPv4, IPv6) using *macvlan* interfaces for the high-speed dataplane. The control interface provides infrastructure and service management connectivity.

each instance, both IPv4 and IPv6 addressing is configured in the container networks. To support this connectivity model, the DTNaaS controller was extended so that multiple networks (i.e. interfaces) could be mapped to service profiles and dual-stack addressing specified for each network instance, allowing our multi-node deployment to be managed in a centralized and automated manner.

4.2 Container Management

The CMS XCache container images managed by DTNaaS for this deployment are sourced from the OSG Docker Hub Community Organization[4]. The DTNaaS framework additionally makes use of an internal container registry that performs security vulnerability scanning using Trivy[5] as part of a larger constant integration (CI) pipeline for managed images. Automation ensures we pull the latest OSG-maintained images from Docker Hub and have them pass through ESnet's CI infrastructure before being instantiated on the physical infrastructure nodes.

Our DTNaaS approach includes tooling to allow for rapid startup or shutdown of distributed caching instances as well as the ability to maintain a history of image revisions for rollback to a prior version if necessary. Both command line and web interfaces expose the most sysadmin-friendly mechanisms for common operational tasks, while a controller API is also available for more programmatic integration.

As of this writing, the installation and verification of the two caching nodes in Boston and Chicago has been completed and we are awaiting final integration into the CMS workflow. The additional caching infrastructure will form the nucleus of Midwestern (Chicago) and Eastern (Boston) regional CMS Data Lakes to improve the data accessibility of nearby computing facilities, and their placement within the ESnet production network is anticipated to aid in delivering data to peering sites with minimal friction.

5 Conclusion and Future Work

We have described our design, deployment, and experience with DTN-as-a-Service (DTNasS) as our approach for supporting in-network caching and storage services within ESnet.

[4]cms-xcache image: https://hub.docker.com/r/opensciencegrid/cms-xcache
[5]Welcome to Trivy: https://aquasecurity.github.io/trivy

We also described observations on the data access trends in the existing SoCal Cache for HEP analysis jobs at Caltech and UCSD. In-network data cache reduces repeated data transfers enabling network traffic savings. During the study period, 94% of the cache hits volume was observed in the SoCal Cache leading to about 10.8PB of network traffic savings, and the study opens other leads to the network engineering.

As future work, we plan to extend this analysis to observe longer term caching trends, and we plan to study data-driven network traffic engineering and traffic volume forecasting as well as locally customized caching policy in the future. Future directions for our DTNaaS approach include considering the positioning and flexible service deployments in support of science applications over the entirety of the ESnet footprint, where geographic boundary considerations along with data locality requirements will influence placement.

6 Acknowledgments

This work was supported by the Office of Advanced Scientific Computing Research, Office of Science, of the US Department of Energy under Contract No. DE-AC02-05CH11231.

References

[1] E. Fajardo, A. Tadel, M. Tadel, B. Steer, T. Martin, F. Würthwein, Journal of Physics: Conference Series **1085**, 032025 (2018)

[2] L. Bauerdick, K. Bloom, B. Bockelman, D. Bradley, S. Dasu, J. Dost, I. Sfiligoi, A. Tadel, M. Tadel, F. Wuerthwein et al., Journal of Physics: Conference Series **513** (2014)

[3] C. Sim, K. Wu, A. Sim, I. Monga, C. Guok, F. Wurthwein, D. Davila, H. Newman, J. Balcas, *Effectiveness and predictability of in-network storage cache for Scientific Workflows*, in *IEEE International Conference on Computing, Networking and Communication* (2023)

[4] B. Brown, E. Dart, G. Rai, L. Rotman, J. Zurawski, University of California, Publication Management System Report LBNL-2001281, Energy Sciences Network (2020), `https://www.es.net/assets/Uploads/20200505-NP.pdf`

[5] X. Espinal, S. Jezequel, M. Schulz, A. Sciabà, I. Vukotic, F. Wuerthwein, EPJ Web of Conferences **245**, 04027 (2020)

[6] R. Pordes, D. Petravick, B. Kramer, D. Olson, M. Livny, A. Roy, P. Avery, K. Blackburn, T. Wenaus, F. Würthwein et al., Journal of Physics: Conference Series **78**, 012057 (2007)

[7] J. Spillner, CoRR **abs/1901.00644** (2019), `1901.00644`

[8] J. Breen, L. Bryant, G. Carcassi, J. Chen, R.W. Gardner, R. Harden, M. Izdimirski, R. Killen, B. Kulbertis, S. McKee et al., *Building the SLATE Platform*, in *Proceedings of the Practice and Experience on Advanced Research Computing* (Association for Computing Machinery, New York, NY, USA, 2018), PEARC '18, ISBN 9781450364461, `https://doi.org/10.1145/3219104.3219144`

[9] E. Kissel, *Janus: Lightweight Container Orchestration for High-performance Data Sharing*, in *The Fifth International Workshop on Systems and Network Telemetry and Analytics (SNTA 2022)* (2022)

[10] R. Han, A. Sim, K. Wu, I. Monga, C. Guok, F. Wurthwein, D. Davila, J. Balcas, H. Newman, *Access Trends of In-network Cache for Scientific Data*, in *5th ACM International Workshop on System and Network Telemetry and Analysis* (2022)