EPJ Web of Conferences **245**, 04031 (2020) https://doi.org/10.1051/epjconf/202024504031
CHEP 2019

Third-party transfers in WLCG using HTTP

Brian Bockelman[6,*], *Andrea* Ceccanti[3], *Fabrizio* Furano[1], *Paul* Millar[2], *Dmitry* Litvintsev[5], and *Alessandra* Forti[4]

[1]European Organization for Nuclear Research (CERN)
[2]Deutsches Elektronen-Synchrotron (DESY)
[3]INFN
[4]University of Manchester
[5]Fermi National Accelerator Laboratory
[6]Morgridge Institute for Research

Abstract. Since its earliest days, the Worldwide LHC Computational Grid (WLCG) has relied on GridFTP to transfer data between sites. The announcement that Globus is dropping support of its open source Globus Toolkit (GT), which forms the basis for several FTP client and servers, has created an opportunity to reevaluate the use of FTP. HTTP-TPC, an extension to HTTP compatible with WebDAV, has arisen as a strong contender for an alternative approach.
In this paper, we describe the HTTP-TPC protocol itself, along with the current status of its support in different implementations, and the interoperability testing done within the WLCG DOMA working group's TPC activity. This protocol also provides the first real use-case for token-based authorisation for this community. We will demonstrate the benefits of such authorisation by showing how it allows HTTP-TPC to support new technologies (such as OAuth, OpenID Connect, Macaroons and SciTokens) without changing the protocol. We will also discuss the next steps for HTTP-TPC and the plans to use the protocol for WLCG transfers.

1 Introduction

The primary driver for wide-area data movement for all LHC experiments is bulk data movement between storage services. This bulk data movement serves to pre-stage data to be processed by production systems or to increase data replication to make it more available for analysis. The technique to perform these transfers is third-party copy (TPC); in TPC, a central entity (the 'third party') contacts a source and destination storage endpoint to facilitate a transfer from the source to the destination. This provides for central management and coordination of transfers but allows for data to move directly between the storage systems. The high-level concept is illustrated in Figure 1.

In 2017, Globus announced the retirement of the Globus Toolkit, which served as the reference implementation for GridFTP protocol [3, 19]; this has increased interest into a number of alternatives such as HTTP [10]. HTTP is protocol that underpins the World Wide Web, making it one of the most common protocols on the planet - meaning there is a large

*e-mail: bbockelman@morgridge.org

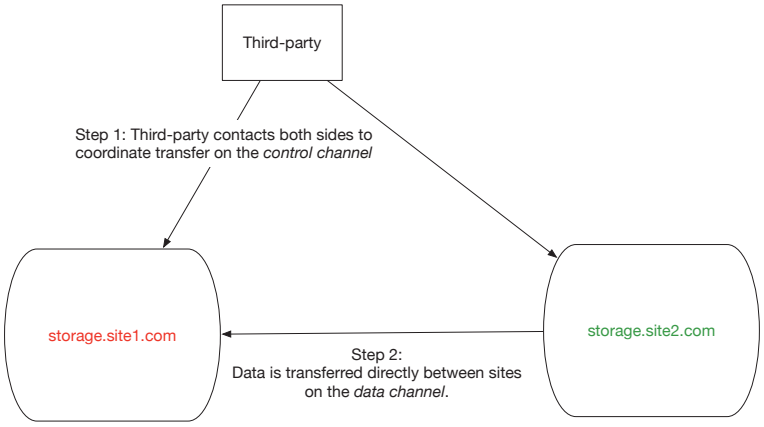

Figure 1. The basic concept behind TPC: an entity independent of either storage service coordinates direct transfer of data between the source and destination.

community of experts and many mature implementations. Unlike GridFTP, utilizing HTTP does not expose the WLCG community to the risks of relying on a specialized protocol.

The initial work to adopt HTTP as a third party transfer protocol within the WLCG community was outlined in [9], following activities descripted in [14]; since then, the protocol has evolved and matured into what we term "HTTP Third Party Copy" (`HTTP-TPC`), as described in Section 2. Further, the WLCG community has formed a working group around Third Party Copy as part of the *Data Organization, Management, and Access* (DOMA) initiatives; as outlined in Section 3, this working group is testing and developing both HTTP-TPC and third party copy extensions for XRootD [15]. This has allowed for the continued growth of the activity - both in maturing implementations so they can be used in production and further evolving the protocol as described in Section 4.

2 Background

In preparation for the Run 1 of the LHC, a number of transfer protocols were considered by the LHC community before it eventually settled on GridFTP with a reference implementation provided by the Globus Toolkit [4] (at least one other production-quality implementation has been written by the dCache project [13]). This transfer protocol was augmented with the Storage Resource Management (SRM) protocol [23] which helped manage load-balancing between servers and the storage end-points. Overall, the GridFTP protocol has served the community faithfully for nearly 15 years.

By the end of Run 2, several events transpired that motivated the community to re-evaluate its use of GridFTP as a TPC protocol. First, many sites began to retire their SRM endpoints as unique space management features of SRM were largely never used, GridFTP could be used directly and native load-balancing solutions were introduced. Second, the Globus organization's retirement of the Globus Toolkit [11] meant the implementation of GridFTP in use by several of the storage systems had no original developer support. This led to the formation of the WLCG DOMA TPC working group during the WLCG DOMA face-to-face at CHEP 2018, charged with examining alternative options and growing nascent ecosystems.

For the HTTP-TPC, as explained in [9], the key concept is the use of the WebDAV `COPY` verb. The client sends an HTTP request using `COPY` to the *active endpoint* of the transfer

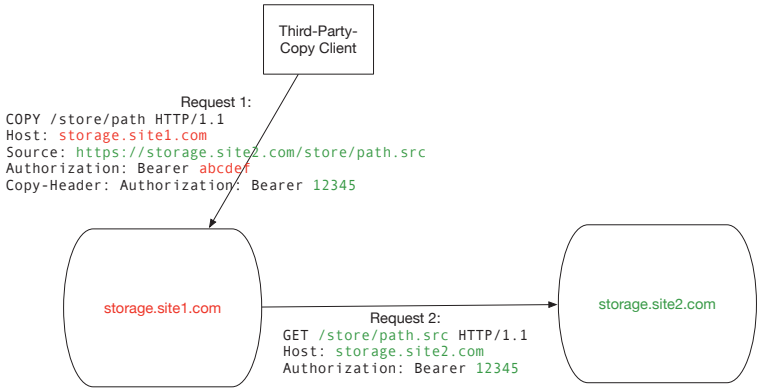

Figure 2. The basic mechanism of HTTP-TPC, reproduced from [9]. Here, we illustrate "pull mode", where the third party copy client contacts the destination site and issues a request that the destination downloads (*pulls*) the data from the source over HTTP.

along with another URL in an HTTP header. For *pull mode*, the active endpoint is given a `Source` header; for *push mode*, the active endpoint is given a `Destination` header. The active endpoint then downloads or uploads, respectively, from the *passive endpoint*. Features of note for the HTTP-TPC protocol include:

- Separation between the "framing" and the "transfer" protocol. The URL sent to the active endpoint does not have to use `https://`; for instance, the dCache implementation has shown that the HTTP-TPC active endpoint can be given a GridFTP URL to move the data over GridFTP (potentially useful for enabling a transition from GridFTP).
- The active endpoint sends continuous performance markers back to the TPC client, allowing the client to monitor progress (cancelling the transfer as necessary).
- Ability for transfers to be load-balanced using HTTP's built-in redirection response.
- When in 'pull' mode (the active endpoint is the destination), multiple pipelined `GET` requests can be load-balanced across multiple parallel TCP streams, allowing a single transfer to proceed faster compared to when a single TCP stream is used.

As of March 2020, there are four independent implementations of the HTTP-TPC protocol, in the dCache, DPM, StoRM [1], and XRootD software products. Further, as XRootD often forms the basis of other storage services in the WLCG community, services like EOS also have HTTP-TPC support without needing a separate implementation.

3 Building the HTTP-TPC Community

We found that the key to maturing the use of HTTP-TPC beyond initial specification and implementation is to build a user community. The primary mechanism thus far has been the WLCG DOMA TPC working group; in the 18 months that followed the work done in [9], this group (with co-leads Bockelman and Forti) has coordinated the development and finalization of HTTP-TPC, helped deploy a test-bed for HTTP-TPC, and organized a testing infrastructure. Within the testing infrastructure, the working group operates three types of tests: nightly, integration, and full scale.

The nightly "smoke tests" [22], demonstrate compliance and functionality with the HTTP-TPC protocol by performing a small transfer against a known working endpoint in

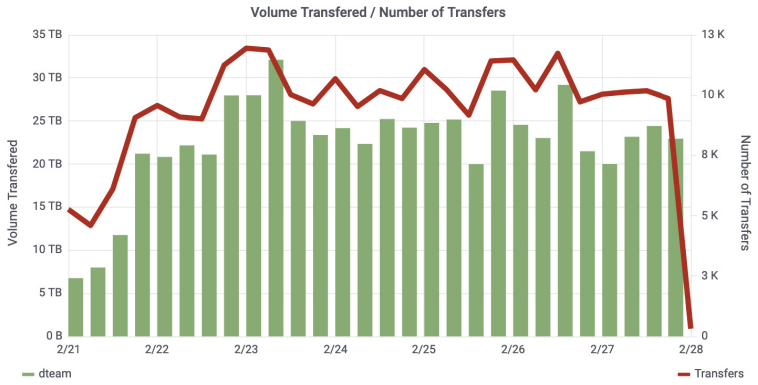

Figure 3. Example graph from the CERN monitoring system displaying the number and volume of HTTP-TPC transfers in the scale test-bed per 6 hour interval. Note that maintaining 20TB of data movement every 6 hours for a week adds up to 560TB of data transferred.

addition to simple tests for acquiring a transfer token from the endpoint (Section 4). These tests are driven by a simple script and are meant to be easily reproducible by developers or administrators.

As of March 2020, the HTTP-TPC 'smoke test' test-bed has 49 participating endpoints. To catch bugs and issues as early in the development process as possible, we include endpoints from across the full software development life cycle: from endpoints on production sites to integration test-beds to developer instances. This approach allows us to test across multiple versions (e.g., stable releases transferring against the latest nightly builds) in addition to across multiple implementation.

The continuous integration tests the 'transfer matrix' between any two protocol endpoints in the system at a small-scale (2-3 GB per hour), allowing the working group to understand behavior at a modest scale and to monitor for a broader set of pairwise issues. These are driven by a dedicated instance of Rucio [20], the data management solution used by the ATLAS experiment. Unlike the smoke tests, these integration tests include the full stack of WLCG transfer utilities. While failures are more difficult to reproduce in this environment, these tests are far more representative of a production transfer activity. For example, failures due to the interaction of the FTS3-based [7] TPC client and the storage service may require a full FTS3 server to reproduce.

Finally, full-scale tests are also driven by Rucio; other than scale, these are identical to the continuous integration tests. These scale tests are driven by a driver script that uploads a randomly-generated one terabyte dataset to an endpoint, then generates a rule to replicate the dataset to all other endpoints in the system. Once the dataset is completely transferred, the driver script triggers a deletion of the replicated datasets; after the deletion is complete, the same rule is installed again. With this setup, we have demonstrated the ability to transfer approximately half a petabyte of data per week. See Figure 3. This allows us to monitor for issues at scales similar to the production system.

4 Evolving Approach

One of the strongest motivations for using HTTP as the base protocol in TPC is that it allows for a number of authorization schemes. The WLCG has historically used the Grid Security

Infrastructure (GSI) [12] with various extensions; at its core, GSI is based on X.509 PKI. As GSI is based on X.509, its infrastructure can be used to associate a TLS session for HTTPS and authenticate a client and the TLS level. Previously in [21], work was done to delegate a grid proxy to the HTTPS-based active endpoint; with the client's delegated identity, the active endpoint could then authenticate its transfer request to the passive endpoint.

However, this use of GSI suffers from the same issue as GridFTP: the retired Globus Toolkit was the reference implementation. The WLCG DOMA TPC working group has instead settled on using bearer tokens to authenticate transfers. A number of bearer token based schemes have been proposed, including SciTokens [24] and the WLCG Common JWT profile [5]; these will eventually allow a transfer to be performed completely without the use of a X.509 client credential. As a transition mechanism, we have defined a way for clients to request a token from the storage endpoint, provided the request is made over a HTTPS connection that is GSI-authenticated.

First, the client must perform OAuth2 metadata discovery [18] against the storage endpoint to determine the associated (storage-specific) token endpoint. Then, an access token request is made against the token endpoint using the client credentials flow [16]. Unlike a typical client credential flow, when the client authenticates via an HTTP header (such as the `Authorization` header), this request must be done over the GSI-authenticated HTTPS channel. Based on this token request and the client's authorization, the storage endpoint will issue an access token that can be used as part of the HTTP-TPC infrastructure. Although the token format is considered opaque (implementations have been done both based on JWT [17] and Macaroons [8]), the client must ask for one or more of an agreed-upon set of scopes of the form `$ACTIVITY:$PATH`. If permitted, the returned token will permit the bearer to perform the specified activity (`$ACTIVITY`) for any resource inside the normalized `$PATH`. The defined authorizations (based upon the work done in dCache for its initial Macaroon support [6]) are:

- `UPLOAD`: Authorization to create new and upload contents, provided that existing data at the endpoint is not altered.
- `DOWNLOAD`: Authorization to read data.
- `DELETE`: Authorization to delete resources from the endpoint.
- `MANAGE`: Change file metadata at the storage endpoint and perform operations that may overwrite existing data.
- `LIST`: List the contents of a directory resource at the storage endpoint.

Beyond authorization, the experience gained in the WLCG scale tests has shown that, while HTTP-TPC can be run in either *push* or *pull* mode, pull mode has become preferred. Pull mode allows the active endpoint to download with the HTTP `GET` requests; as `GET` is idempotent, the endpoint can issue numerous requests in a pipeline or partition them over a number of TCP streams to improve overall throughput. Further, the active endpoint is the most natural entity in the system to manage a queue of transfer requests. Given the ability to write to disk is considered a more scarce resource than reading, sites have preferred the pull mode.

5 Conclusions

Not only is third party copy an essential technique in the WLCG infrastructure, it is how the majority of the LHC data is transferred. Over the past several years, the HTTP-TPC protocol has emerged as viable replacement for the venerable GridFTP protocol. While the broad outlines of the protocol have been used by the community for years, the protocol has evolved

based on operational experience and the community evolution away from using X.509 client credentials. These activities have been led by the WLCG DOMA TPC working group which organizes several types of test-beds. A key activity in the coming months will be to evaluate fully token-based data transfers, with authorization following the rules defined by the WLCG JWT profile [5] and leveraging the integration with the WLCG IAM token issuer [2].

As we go beyond tests, LHC experiments are beginning to consider the use HTTP-TPC in their production infrastructure; we expect to see significant, at-scale tests in the lead-up to Run 3.

Acknowledgements

This material is based upon work supported by the National Science Foundation under Grant No. 1836650.

References

[1] The StoRM WebDAV service. https://github.com/italiangrid/storm-webdav.

[2] The WLCG IAM instance. https://wlcg.cloud.cnaf.infn.it.

[3] W Allcock. GridFTP: Protocol Extensions to FTP for the Grid, April 2003. URL: https://www.ogf.org/documents/GFD.20.pdf.

[4] Allcock, W., Bresnahan, J., Kettimuthu, R., and Link, M. The Globus Striped GridFTP Framework and Server. *SC '05: Proceedings of the 2005 ACM/IEEE Conference on Supercomputing*, 2005. doi:{10.1109/SC.2005.72}.

[5] Altunay, Mine, Bockelman, Brian, Ceccanti, Andrea, Cornwall, Linda, Crawford, Matt, Crooks, David, Dack, Thomas, Dykstra, David, Groep, David, Igoumenos, Ioannis, Jouvin, Michel, Keeble, Oliver, Kelsey, David, Lassnig, Mario, Liampotis, Nicolas, Litmaath, Maarten, McNab, Andrew, Millar, Paul, Sallé, Mischa, Short, Hannah, Teheran, Jeny, and Wartel, Romain. WLCG Common JWT Profiles (Version 1.0), 2019. doi:10.5281/zenodo.3460258.

[6] Ashish, A, Millar, P., Mkrtchyan, T, Fuhrmann, P., Behrmann, G., Sahakyan, M., Adeyemi1, O, Starek, J, Litvintsev, D, and Rossi, A. dCache, towards Federated Identities & Anonymized Delegation. *J. Phys.: Conf. Ser.*, 898, 2017. doi:10.1088/1742-6596/898/10/102009.

[7] Ayllon, A, Salichos, M, Simon, M, and Keeble, O. FTS3: New Data Movement Service For WLCG. *J. Phys.: Conf. Ser.*, 513:032081, 2014. doi:10.1088/1742-6596/513/3/032081.

[8] Arnar Birgisson, Joe Gibbs Politz, Ulfar Erlingsson, Ankur Taly, Michael Vrable, and Mark Lentczner. Macaroons: Cookies with contextual caveats for decentralized authorization in the cloud. In *NDSS*, 2014.

[9] Bockelman, Brian, Hanushevsky, Andrew, Keeble, Oliver, Lassnig, Mario, Millar, Paul, Weitzel, Derek, and Yang, Wei. Bootstrapping a new LHC data transfer ecosystem. *EPJ Web Conf.*, 214:04045, 2019. doi:10.1051/epjconf/201921404045.

[10] R. Fielding, J. Gettys, J. Mogul, H. Frystyk, and T. Berners-Lee. Hypertext Transfer Protocol – HTTP/1.1. Internet Requests for Comments, January 1997. URL: https://tools.ietf.org/pdf/rfc2068.pdf.

[11] Ian Foster. *Support for open source Globus Toolkit will end as of January 2018*, 2017 (accessed 28 November 2018). URL: https://github.com/globus/globus-toolkit/blob/4c88c9ca1423e2af806714a2eca54f6eb5d9fd4e/support-changes.md.

[12] Foster, Ian, Kesselman, Carl, Tsudik, Gene, and Tuecke, Steven. A security architecture for computational grids. *In Proceedings of the 5th ACM conference on Computer and Communications Security*, 1998.

[13] Fuhrmann, P. and Guelzow, V. dCache, Storage System for the Future. *Euro-Par 2006 Parallel Processing*, 4128, 2006. doi:10.1007/11823285_116.

[14] Furano, Fabrizio, Devresse, Adrien, Keeble, Oliver, and Hellmich, Martin. Towards an HTTP Ecosystem for HEP Data Access. *J. Phys.: Conf. Ser.*, 2014. URL: http://iopscience.iop.org/1742-6596/513/3/032034, doi:10.1088/1742-6596/513/3/032034.

[15] Andrew Hanushevsky. *Third Party Copy Protocol TPC Version 2.0 Reference*, 2020 (accessed March 7, 2020). URL: https://xrootd.slac.stanford.edu/doc/dev49/tpc_protocol.htm.

[16] Hardt, D. The OAuth 2.0 Authorization Framework. Internet Requests for Comments, October 2012. URL: https://tools.ietf.org/html/rfc6749.

[17] Jones, M., Bradley, J., and Sakimura, N. JSON Web Token (JWT). Internet Requests for Comments, May 2015. URL: https://tools.ietf.org/html/rfc7519.

[18] Jones, M, Sakimura, N, and Bradley, J. OAuth 2.0 Authorization Server Metadata. Internet Requests for Comments, June 2018. URL: https://tools.ietf.org/html/rfc8414.

[19] I Mandrichenko. GridFTP Protocol Improvements, July 2003. URL: https://www.ogf.org/documents/GFD.21.pdf.

[20] Martin Barisits, Thomas Beermann, Frank Berghaus, Brian Bockelman, Joaquin Bogado, David Cameron, Dimitrios Christidis, Diego Ciangottini, Gancho Dimitrov, Markus Elsing, Vincent Garonne, Alessandro Di Girolamo, Luc Goossens, Wen Guan, Jaroslav Guenther, Tomas Javurek, Dietmar Kuhn, Mario Lassnig, Fernando Lopez, Nicolo Magini, Angelos Molfetas, Armin Nairz, Farid Ould-Saada, Stefan Prenner, Cedric Serfon, Graeme Stewart, Eric Vaandering, Petya Vasileva, Ralph Vigne, and Tobias Wegner. Rucio: Scientific data management. *Computing and Software for Big Science*, 3, 2019. doi:10.1007/s41781-019-0026-3.

[21] Andrew McNab. The GridSite Web/Grid security system. *Software: Practice and Experience*, 35:827–834, 2005. doi:10.1002/spe.690.

[22] Paul Millar. *General utilities for working with HTTP Third-Party-Copy (TPC)*, 2020 (accessed March 7, 2020). URL: https://github.com/paulmillar/http-tpc-utils.

[23] Sim, Alex and Shoshani, Arie. *The Storage Resource Manager Interface Specification Version 2.2*, 2009 (accessed March 7, 2020). URL: https://sdm.lbl.gov/srm-wg/doc/SRM.v2.2.html.

[24] Alex Withers, Brian Bockelman, Derek Weitzel, Duncan Brown, Jeff Gaynor, Jim Basney, Todd Tannenbaum, and Zach Miller. Scitokens: Capability-based secure access to remote scientific data. *PEARC '18: Proceedings of the Practice and Experience on Advanced Research Computing*, 2018. doi:10.1145/3219104.3219135.