Moving beyond GridFTP (and GSI)

At the end of May 2017, the Globus Alliance announced that the Open source Globus toolkit would be no longer supported by the Globus team at the University of Chicago [1]. This announcement had obvious impact on WLCG, since the Global Cyberinfrastructure Security (GCS) and GridFTP lie at the core of the WLCG data management infrastructure, and discussions started in the appropriate forums for the search of alternatives. The DONA Technical Third-party Copy (TPC) Working Group [8] was established to investigate alternatives to the GridFTP protocol for bulk transfers across WLCG sites. This led to a requirement for all storage elements (SE) to support WebDAV-based or XrootD-based third-party transfers.

This poster describes the work done in the past months to comply with the above requirement.

The StoRM Lightweight Storage Element (SE)

StoRM [5] is a lightweight storage resource manager (SRM) solution developed at INFN, which powers the Italian Tier-1 data center at CNAF, as well as more than 30 other sites. StoRM implements the SRM version 2.2 data management specification and is typically deployed on top of a cluster file system like IBM GPFS. StoRM has a layered architecture (Figure 1), split between two main components: the StoRM frontend and backend services. The StoRM frontend service implements the SRM interface exposed to client applications and workflows. The StoRM backend service implements the actual storage management logic by interacting directly with the underlying file system.

Data transfer is provided by GridFTP, HTTP/WEBDAV and XrootD services accessing directly the file system underlying the StoRM deployment.

StoRM WebDAV besides HTTP data transfer functionality, also provides a WebDAV-based data management interface.

WebDAV-based third-party transfers

In November 2018, StoRM WebDAV was extended to support third-party transfer requests.

The protocol [6], already implemented in DPM, dCache and XrootD, relies on an extension of the semantics of the WebDAV COPY verb, typically used to copy resources local to the storage element, to also support remote transfers.

When starting a third-party transfer, the client can indicate whether a credential delegation should happen. Delegation is typically implemented using the GridKit delegation protocol but the protocol also specifies a mechanism to define how HTTP headers can be transferred between the original third-party transfer request and the actual HTTP file transfer request (GET or PUT) issued by the active SE.

Since StoRM WebDAV does not support GridKit delegation and we are moving away from X.509 certificates, we have decided to only support token-based delegation for the StoRM third-party transfers.

An example WebDAV-based third-party transfer is depicted in Figure 2. Here we assume that FTS, the client triggering the transfer, has already obtained two bearer tokens from SEs A and B. This is achieved today in the context of the DONA TPC testbed by exchanging a VOMS credential delegated by RUCIO (or an end user) with a so-called service issued token (SIT), i.e., a bearer token minted by Global Security that grants a subset the privileges granted by the VOMS credential used to obtain it. FTS includes the token obtained from SEs A and B in the Authorization HTTP header, following the HTTP bearer token standard and includes the token obtained from service B, i.e., the TransferHeaderAuthorization HTTP header. Any HTTP header in the request sent with the TransferHeaderAuthorization header will be copied into the actual file transfer request, with the TransferHeader prefix removed from the name.

This mechanism is used to forward to the active SE the token to use when contacting the passive SE in the transfer, as shown in Figure 2 for token T2.

A dedicated servlet filter was implemented and tested to serve third-party COPY requests. Transfer progress monitoring has been implemented following an approach equivalent to GridFTP performance markers.

On the client node, gfal-copy was used to request the file transfers, with the output of the transfer redirected to /dev/null. We measured the time taken to execute the gfal-copy command and used this time to compute an estimate of the transfer throughput. In this comparison, we looked only at download transfers.

We measured both single client transfers as well as parallel transfers (with 2 and 4 concurrent transfers). For each transfer type, we repeated the transfer 10 times and took the average of the throughput.

The tests results have been summarized in Figure 4.

Figure 4. The average transfer rate observed after running sequential and parallel token-based WebDAV transfers against GridFTP and StoRM WebDAV HTTP and XrootD endpoints, using gfal-copy to drive the transfers.

The first surprising result is the difference in throughput for single transfers between HTTP and GridFTP. We expected to have almost the same throughput between the two protocols, given both send data over an unencrypted channel, while the results in Figure 4 show plain HTTP transfers are ~150 MB/s faster than GridFTP.

We were not convinced by these numbers, so looked in more detail, and found that the difference lies in how gfal-copy handles an HTTP copy request. First, an HTTP HEAD request is sent to the server, followed by an HTTP GET to fetch the file. The time spent in the additional HEAD request explains the difference in throughput between plain HTTP and GridFTP. The overhead gets less noticeable with multiple concurrent transfers, due to the reduced bandwidth available to each transfer. By running a series of tests using curl instead of gfal-copy for plain HTTP transfers, we got throughput comparable to GridFTP, as shown in Figure 5.

Figure 5. The average transfer rate observed using curl as a client for plain HTTP transfers against the rate observed using gfal-copy for GridFTP transfers.

The second surprising result is how impactful is Conscrypt in HTTPS transfers. When Conscrypt is turned off, and the default JVM TLS stack is used, the maximum throughput we achieved was ~150 MB/s in all transfers. With Conscrypt turned on, the encryption overhead is ~20%, as expected.

Conclusions and future work

In this poster, we have described the work done to introduce third-party transfer support and token-based authorization in the StoRM WebDAV service.

We have also shown the results of an initial performance comparative analysis between Globus GridFTP and StoRM WebDAV, focusing on transfer throughput. Our initial results show that StoRM WebDAV represents a viable alternative to GridFTP for bulk data transfers.

In the future we will focus on improving the flexibility of token-based authorization by introducing support for the WLCG AWS profile file [7]. We will also extend our GridFTP comparative analysis to include upload transfers and a more detailed StoRM WebDAV scalability assessment.

Token-based authorization

We decided to reuse as much as possible existing standard and widely supported libraries to support token-based delegation in StoRM WebDAV. To leverage the latest version of the Spring Security libraries [4] providing native support for OAuth2 authorization and OpenID Connect authentication, StoRM WebDAV has been ported to the latest stable released version of the Spring Boot library.

A minimal OAuth authorization server has been implemented to support the exchange of a VOMS credential to a SIT, which for StoRM is a signed JSON Web Token. Support for external OpenID Connect providers has also been added, testing the integration against an INDIGO IAM [3] instance.

The coarse-grained authorization model supported by StoRM WebDAV for VOMS which provides flat, VO-based access to the namespae is also provided to external OpenID Connect providers, enabling a gradual transition between the two authorization approaches.

TL/OS performance issues

When initially testing the throughput of the StoRM WebDAV implementation we noticed that the service was much slower in serving content on HTTPS than over HTTP. While a ~20% decrease in performance was expected due to the encryption costs, we observed a much greater performance loss.

After some investigation, we found that the performance loss was caused by the inefficiencies of the OpenDKJ pure Java TLS implementation, a known issue for the OpenDKJ Java 8 VM.

To work around this JVM limitation, the StoRM WebDAV code was modified to add support for Conscrypt [2], a Java Secure Socket Extension (JSS) implementation developed by Google which delegates the handling of cryptographic operations to an embedded native library (BoringSSL, the Google fork of OpenSSL).

With this change, we noticed a much smaller encryption overhead over the transfers throughput, in line with our former expectations of a ~20% decrease in performance between HTTPS and plain HTTP transfers.

We were not convinced by these numbers, so looked in more detail, and found that the difference lies in how gfal-copy handles an HTTP copy request. First, an HTTP HEAD request is sent to the server, followed by an HTTP GET to fetch the file. The time spent in the additional HEAD request explains the difference in throughput between plain HTTP and GridFTP. The overhead gets less noticeable with multiple concurrent transfers, due to the reduced bandwidth available to each transfer. By running a series of tests using curl instead of gfal-copy for plain HTTP transfers, we got throughput comparable to GridFTP, as shown in Figure 5.

The second surprising result is how impactful is Conscrypt in HTTPS transfers. When Conscrypt is turned off, and the default JVM TLS stack is used, the maximum throughput we achieved was ~150 MB/s in all transfers. With Conscrypt turned on, the encryption overhead is ~20%, as expected.

Conclusions and future work

In this poster, we have described the work done to introduce third- 
party transfer support and token-based authorization in the StoRM WebDAV service.

We have also shown the results of an initial performance compa-
rative analysis between Globus GridFTP and StoRM WebDAV, fo-
cus ing on transfer throughput. Our initial results show that StoRM WebDAV represents a viable alternative to GridFTP for bulk data transfers.

In the future we will focus on improving the flexibility of token-based authorization by introducing support for the WLCG AWS profile file [7]. We will also extend our GridFTP comparative analysis to include upload transfers and a more detailed StoRM WebDAV scalability assessment.

References