A (New?) Transfer Ecosystem for the WLCG

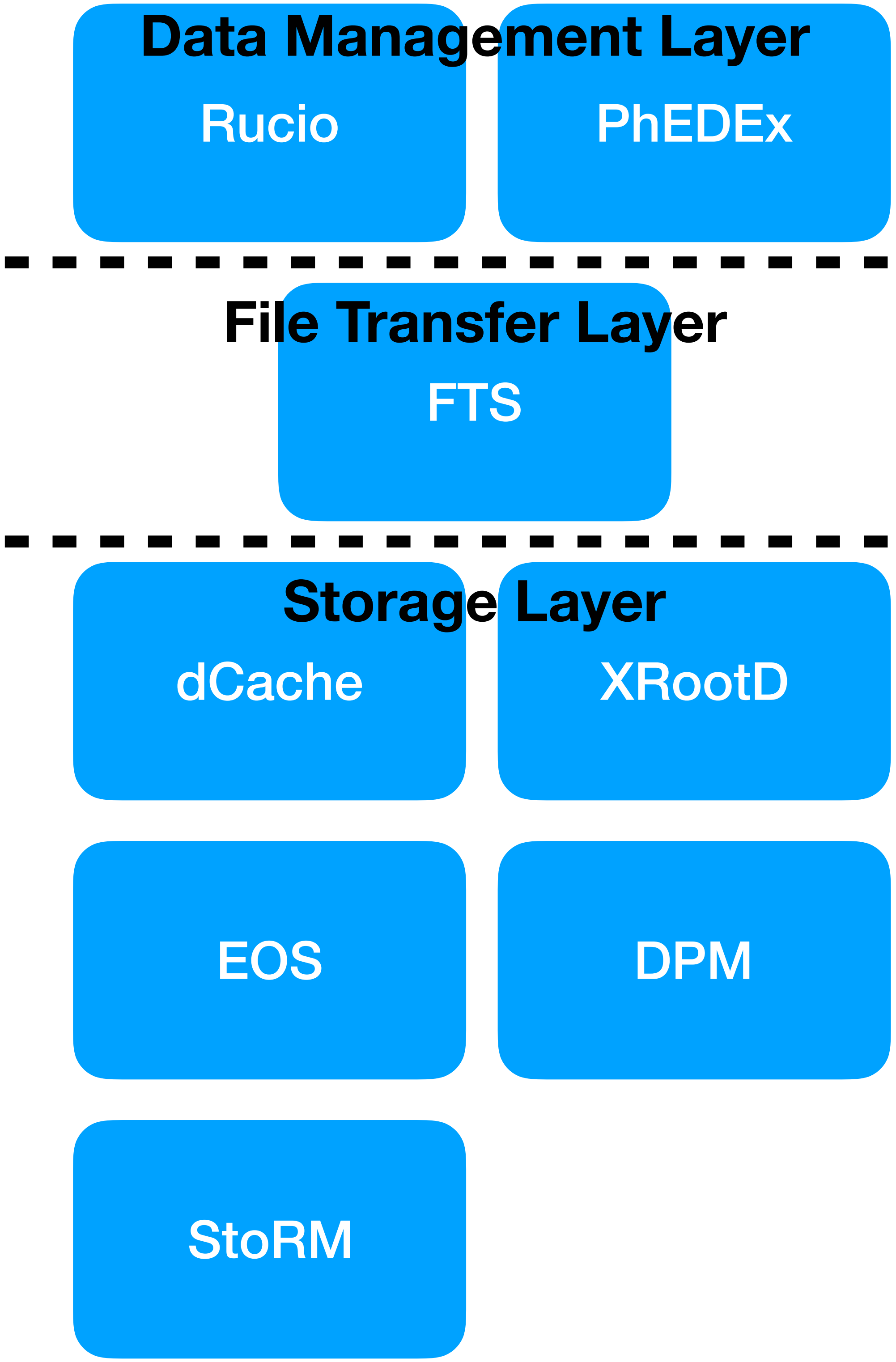
Brian Bockelman
WLCG / HSF Workshop 2018

Why am I here?

- The announcement in mid-2017 that Globus Toolkit support would end set off a flurry of activity.
    - Some of it was toward shorter-term collaborations around community support of this software. See https://gridcf.org
- This reinvigorated existing work around replacing various Globus Toolkit components; the most pressing are:
    - **Grid Security Infrastructure (GSI)**: An authentication and authorization infrastructure based around concepts of identity and X509 proxies.
    - **GridFTP**: A FTP-like transfer protocol that build on top of GSI, supports third-party-transfers, and multi-TCP-stream transfers.
- Luckily, there's a huge amount of prior effort to draw on, some dating back several years.

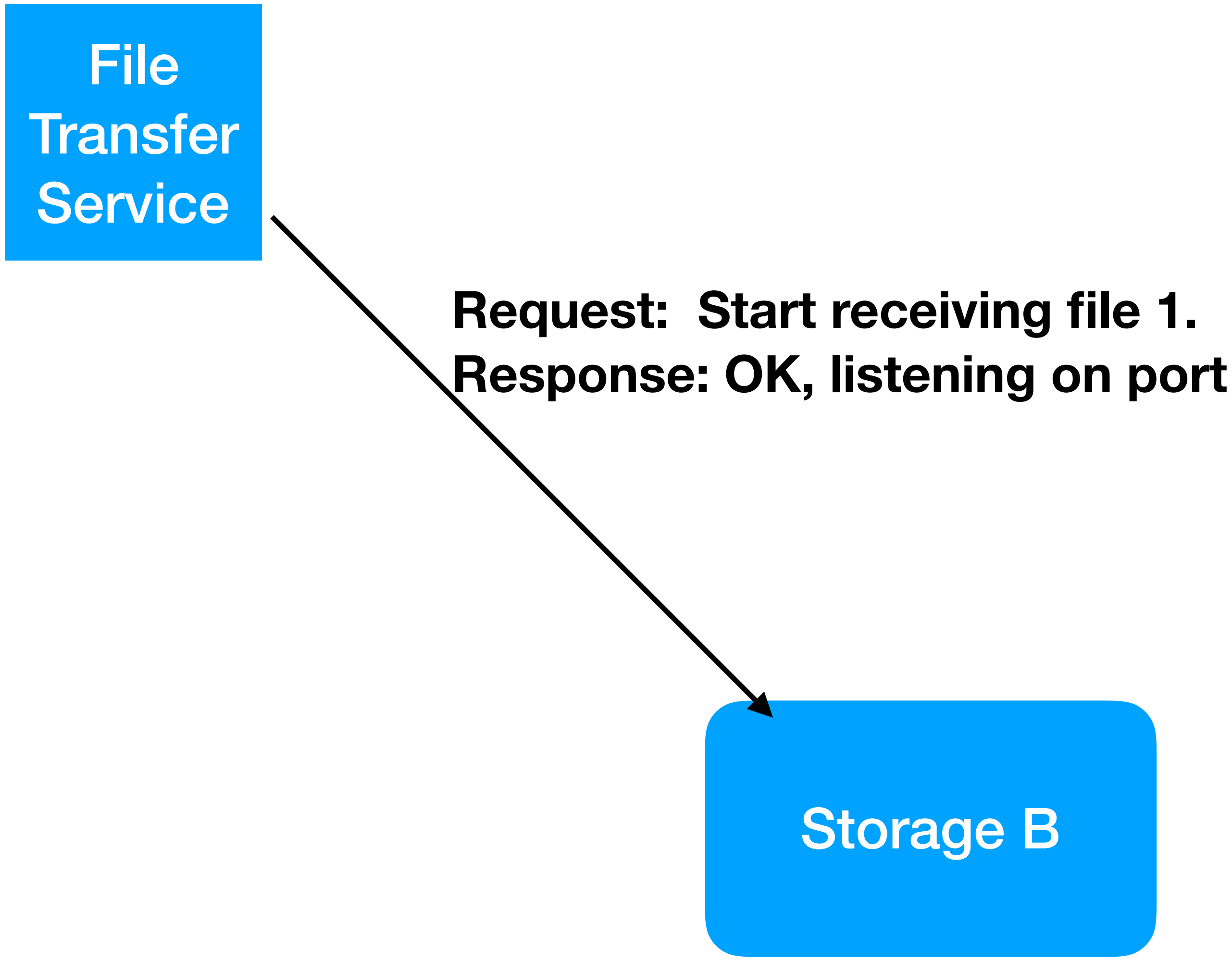
WLCG Transfer Ecosystem Demonstrator

- There's a need to organize the entire vertical stack to have a cohesive solution approach.
- We benefit little if multiple storage elements take mutually-incompatible approaches.
  - Same applies for moving across the data management / file transfer / storage layers.
- Put together a Google group to coordinate this activity and start to scale:
  - Feel free to join!
  - https://groups.google.com/forum/#!forum/wlcg-http-transfer



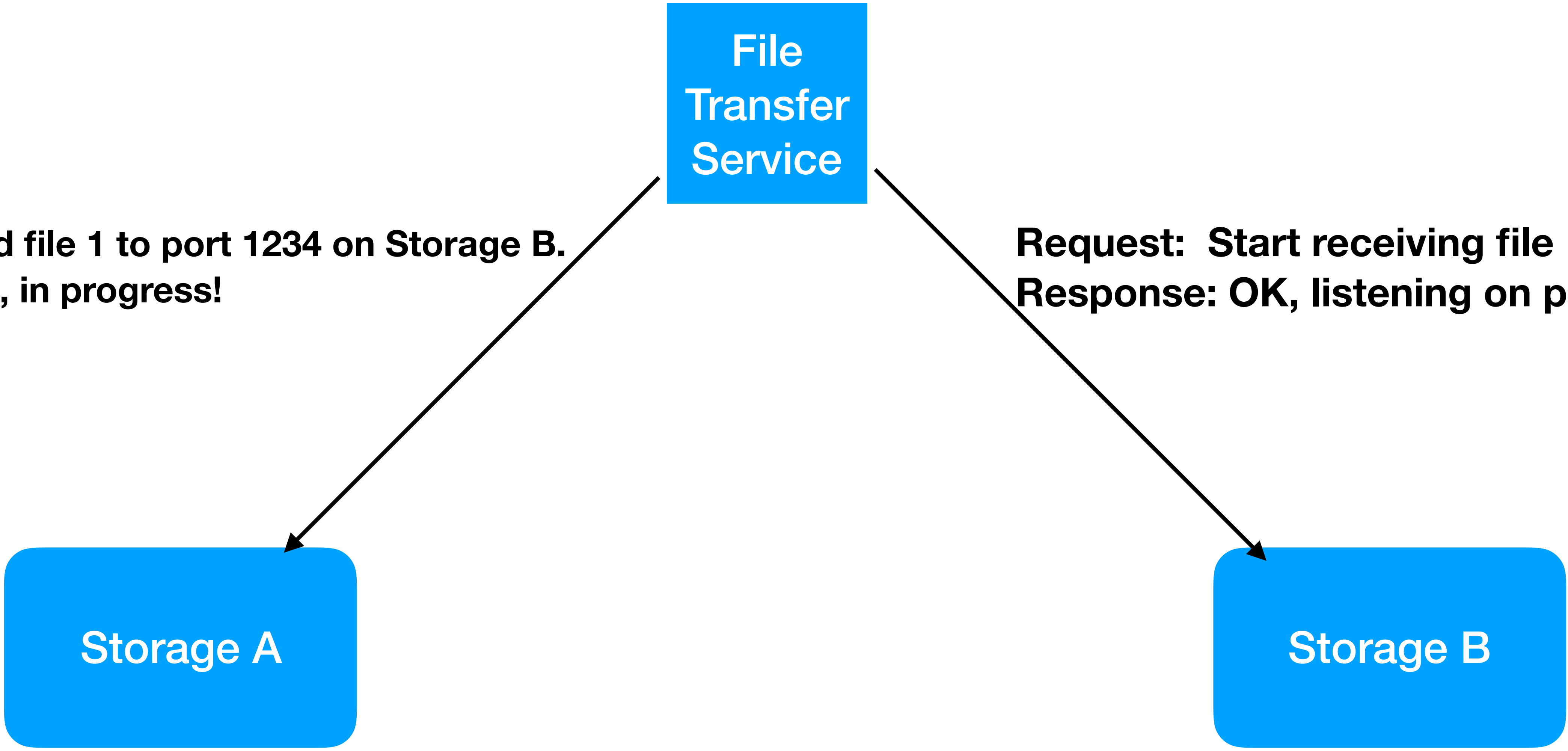
Transfers Under GridFTP - Where we are today!

Transfers Under GridFTP - Where we are today!



Request: Start receiving file 1.
Response: OK, listening on port 1234

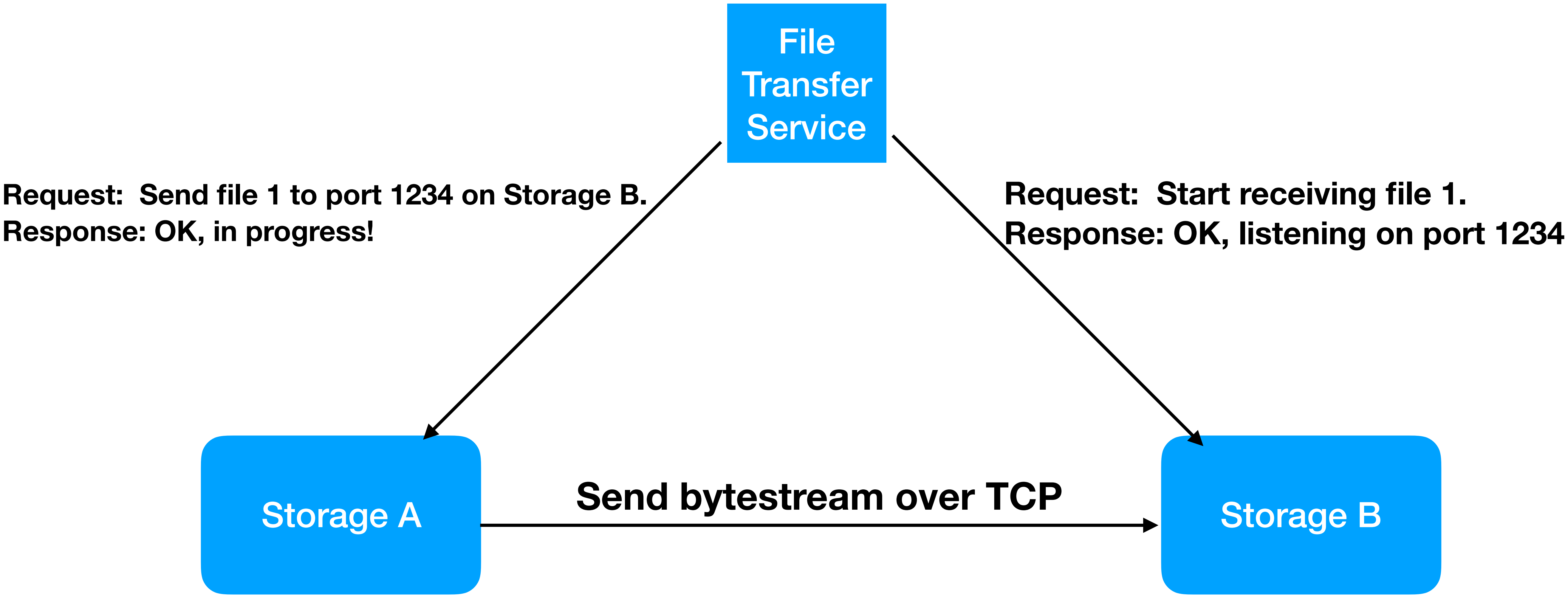
Transfers Under GridFTP - Where we are today!



Request: Send file 1 to port 1234 on Storage B.
Response: OK, in progress!

Request: Start receiving file 1.
Response: OK, listening on port 1234

Transfers Under GridFTP - Where we are today!



File Transfer Service

Request: Send file 1 to port 1234 on Storage B.
Response: OK, in progress!

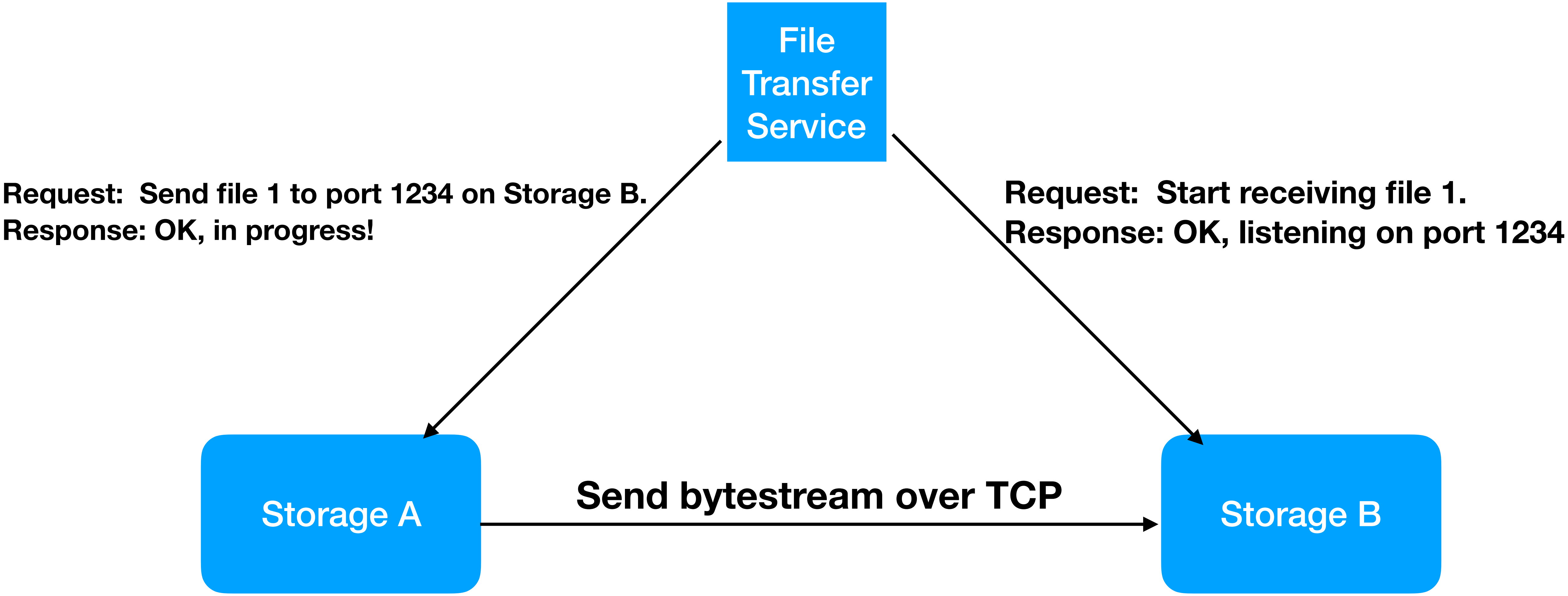
Request: Start receiving file 1.
Response: OK, listening on port 1234

Storage A

Send bytestream over TCP

Storage B

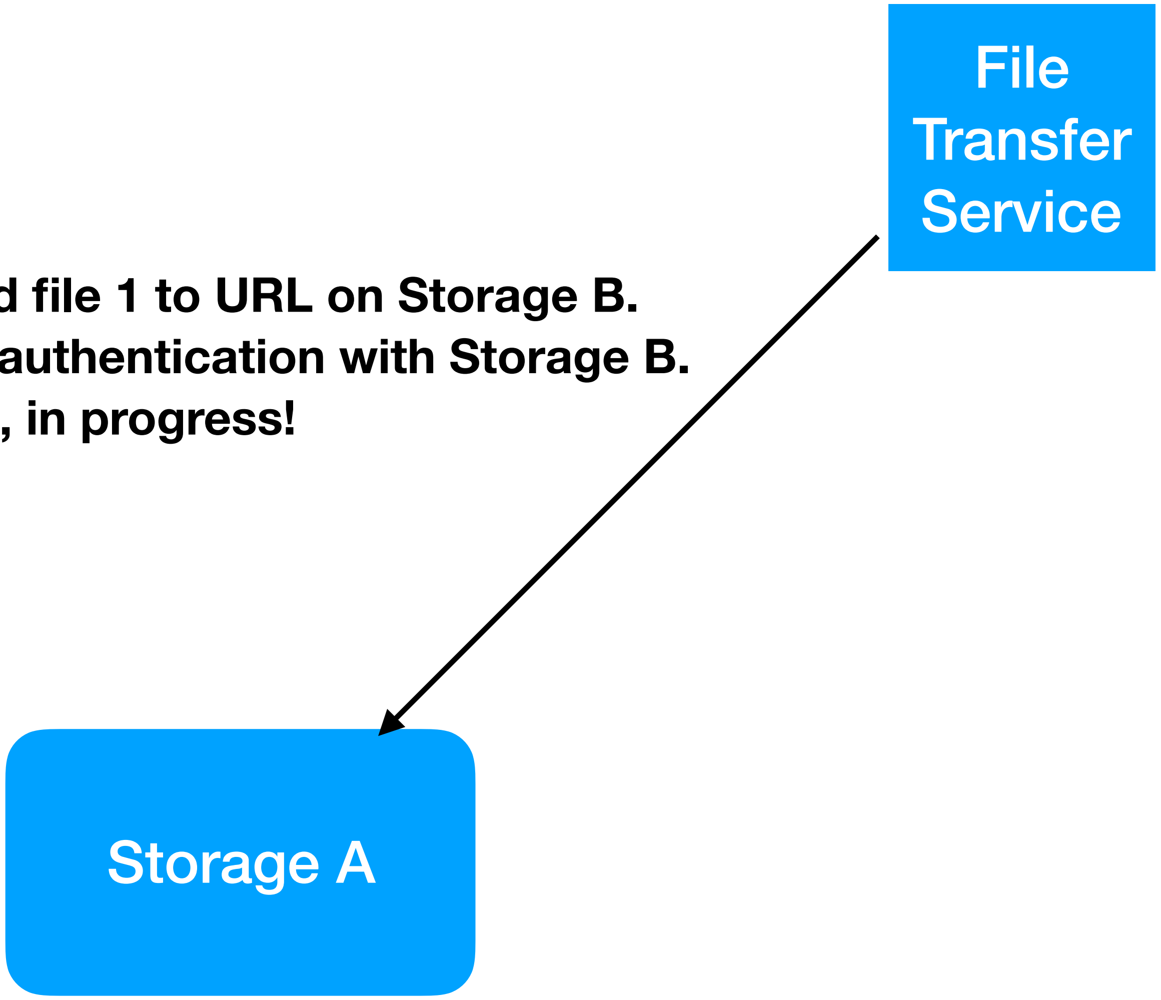
Transfers Under GridFTP - Where we are today!


File Transfer Service
Request: Send file 1 to port 1234 on Storage B.
Response: OK, in progress!
Request: Start receiving file 1.
Response: OK, listening on port 1234
Storage A
Send bytestream over TCP
Storage B

- **FTS must be authorized to talk to both endpoints.**
- **Endpoints support the same protocol (GridFTP).**
- **Queueing (in implementation) is in FTS layer.**

Alternate TPC Model - Where we might go!

Alternate TPC Model - Where we might go!

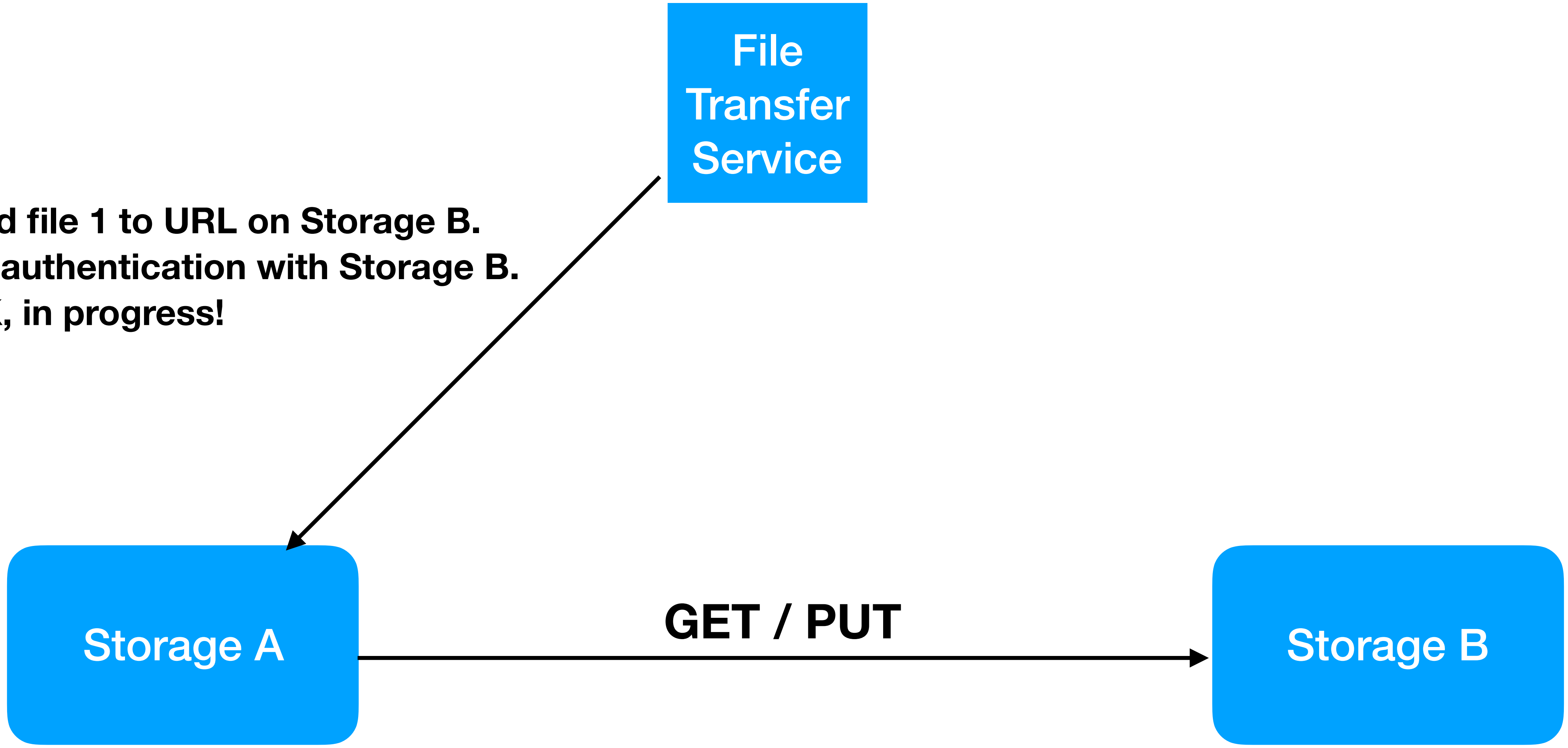


Request: Send file 1 to URL on Storage B.

- **Use given authentication with Storage B.**

Response: OK, in progress!

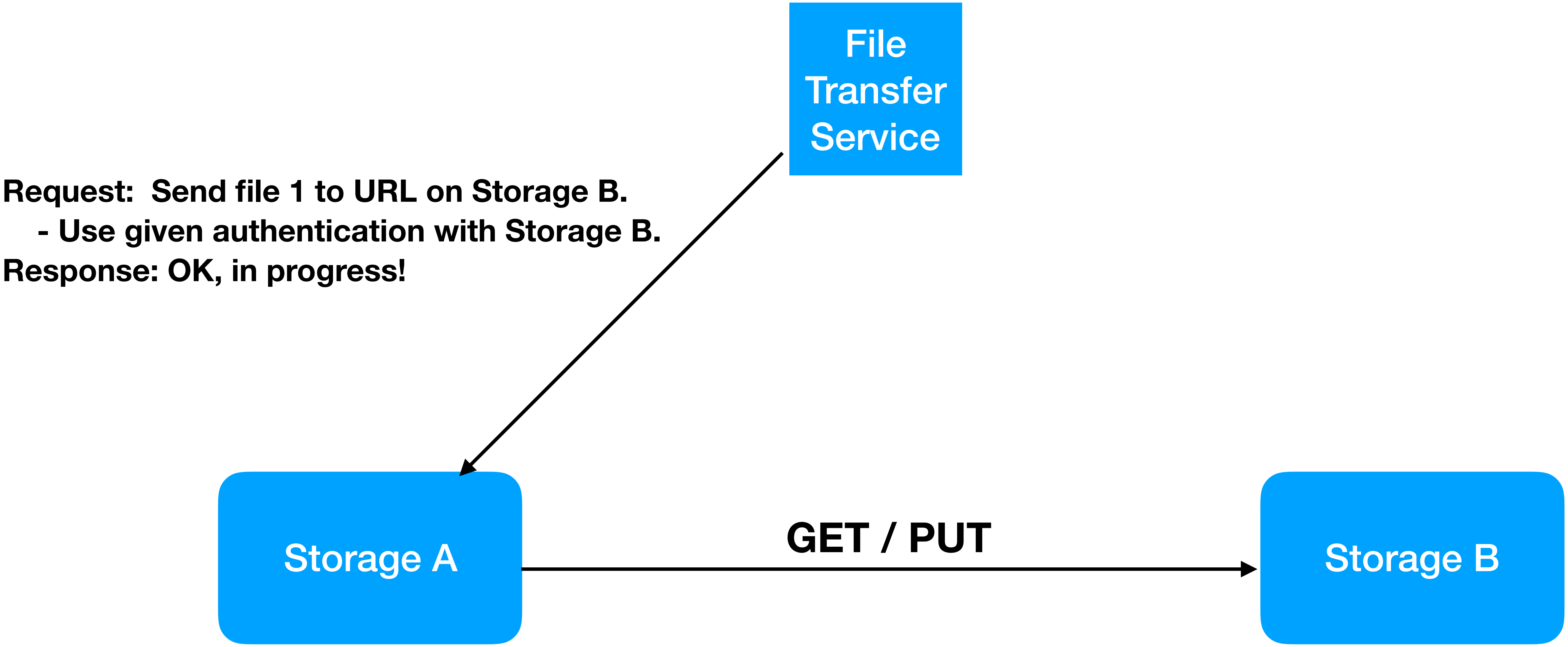
Alternate TPC Model - Where we might go!



Request: Send file 1 to URL on Storage B.
- Use given authentication with Storage B.

Response: OK, in progress!

Alternate TPC Model - Where we might go!



Request: Send file 1 to URL on Storage B.
- Use given authentication with Storage B.

Response: OK, in progress!

- FTS only communicates with the active storage (A).
  - FTS provides URL for B and authz token.
- Transfer from A->B may occur on any mutual protocol.
- FTS relies on storage A for heavy lifting.

HTTPS / WebDAV

- WebDAV is a set of HTTP extensions that provide a more standardized, file-like API with minimal HTTP changes.
  - Example: “`MKCOL`” (make collection) is mostly equivalent to a POSIX `mkdir()`.
- Another WebDAV extension is `COPY`, which instructs the WebDAV server to copy to/from a given URL.
  - Precisely what is needed for the alternate TPC model!
  - The URL is given in the `Source` header; not necessarily HTTPS!

```
COPY /store/path HTTP/1.1
Host: storage.site1.com
Source: https://storage.site2.com/store/path.src
```

HTTPS / WebDAV - Authorization Step

- It’s clear FTS can use its favorite existing mechanism when communicating with the “active” SE (Storage A).
    - How does it transfer a credential to the active SE for use with Storage B?
- In X509-land, we have the concept of delegating a credential for this movement.
    - Unfortunately, the delegation procedure is only “standardized” at the transport layer (TCP).
    - The WLCG community has a somewhat ad-hoc* standard for this based on SOAP, as defined by gridsite.
        - It appears complex and perhaps a touch backwards to start new implementations here.

*** https://egee-jra1-data.web.cern.ch/egee-jra1-data/GridSiteDelegation/HEAD/doc/glite-security-delegation-interface/DelegationInterface.html**

Generation Leap - Bearer Tokens

- Outside our community, in HTTPS, authorization is expressed by a string in a specific header.
  - Referred to as bearer tokens: whoever has access to the token (“the bearer”) has its authorizations.
  - Assumes we have a private / secure communication channel (such as HTTPS).
- Often, this is *capability based* not *identity based*. The token authorizes the bearer to do a certain action (“write to file /store/foo inside the CMS area”); X509 provides an identity that the site must figure out how to map (“what is Brian Bockelman allowed to do at my site?”).
  - For more in-depth discussion, see https://indico.cern.ch/event/658060/contributions/2890286/

Bearer Tokens

- Two approaches to bearer tokens:
    - Completely opaque: must coordinate with an external agent to determine token validity and corresponding authorization.
    - Standardized schema: 3rd party can parse, validate, and authorize from the token itself.
- For this group, we have utilize the “JSON Web Token” format with mutually agreed-upon:
    - Approach to verification.
    - Interpretation of authorizations.

Sample token, decoded:

```
{
“iss":"https://scitokens.org/cms",    # To
“scp":["write:/store/user/clundst","re
“sub":"clundst",   # Subject name, for tr
“jti”:"b8d54a62-cd33-4b4b-bb64-11b8
“exp":1521561382,   # Expiration and v
"iat":1521557782,
"nbf":1521557782
}
```

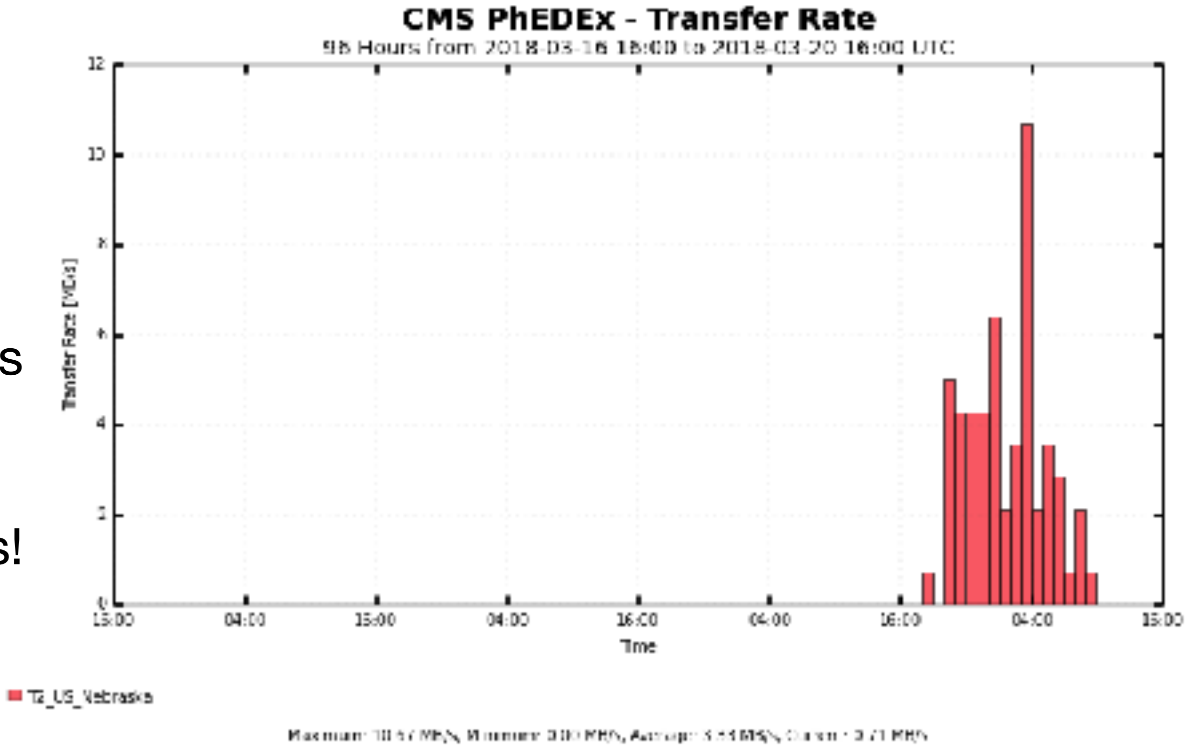
Working up the Stack

- Within the WLCG Authorization Working group, we are working on a consensus on the token profile.
    - Minor changes from the existing SciToken format, but compatible in the broad brush.
- We have an initial prototype functioning as XRootD plugins.
    - Stable enough to put at production servers at three different sites.
    - We have handshake-level agreement from all the other “WLCG storage” elements to implement this approach. Except for dCache, get this *somewhat* for free as the XRootD layer is shared.
    - dCache implementation is not from-scratch as they already utilize OIDC tokens.
- GFAL2, DAVIX, and FTS have patches in release (or testing) supporting the end-to-end.
- PhEDEx changes available as patch and Rucio changes are in a testing branch.

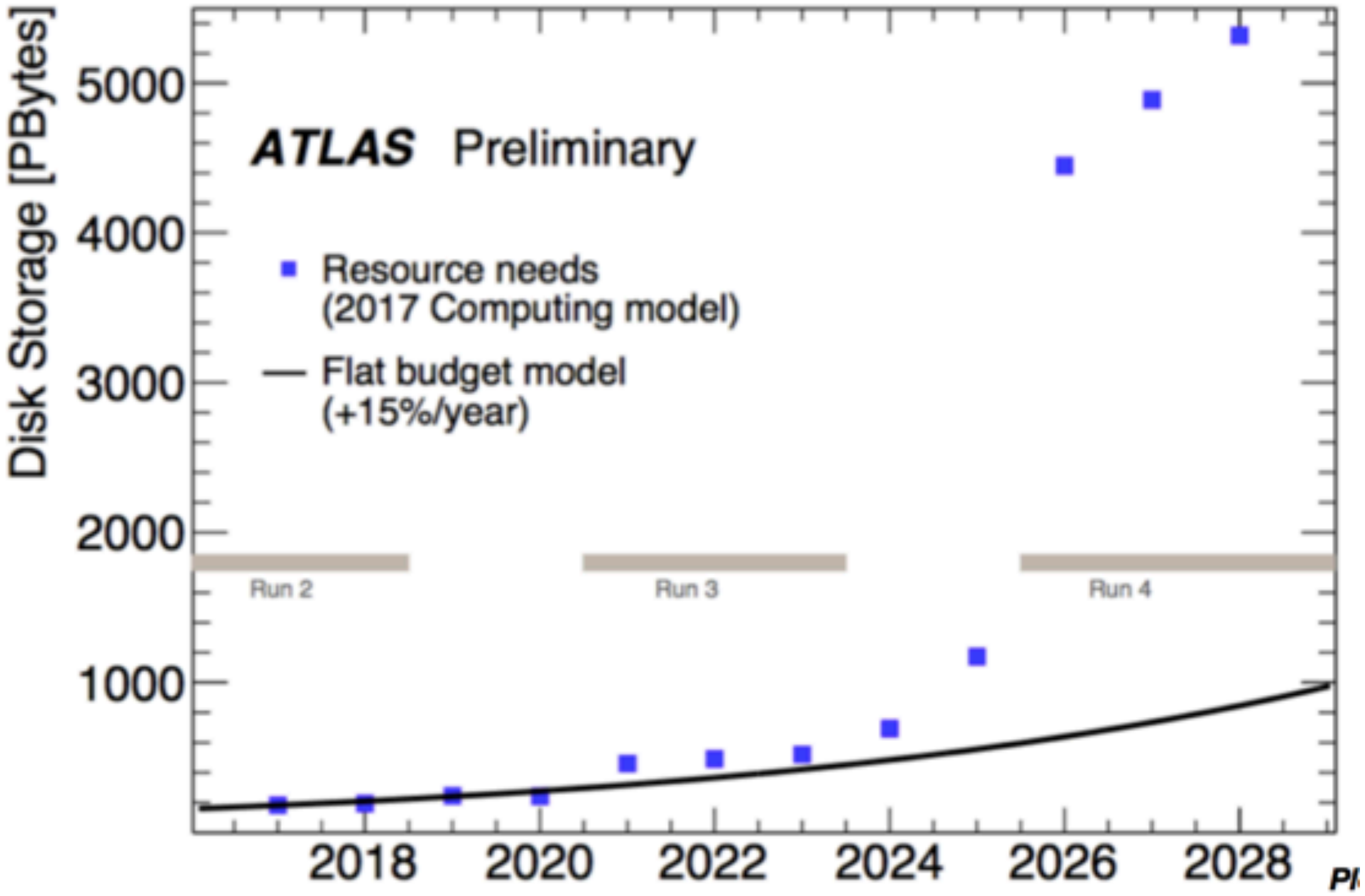
Working the vertical: patches across about a dozen software packages.

A Sunny Outlook (for a work in progress)

- Want to see the nitty gritty? See the parallel presentation this afternoon:
    - https://indico.cern.ch/event/658060/contributions/2886775/
- We are just now verifying functionality of the vertical stack.
- Looking for souls interested in doing performance studies -
    - Potentially also studying different transport protocols!
    - Want to scale up to the “1 PB moved” level.
- Increasing the number of sites participating - and the types of sites.
- **This is the opening act: visit with us again at CHEP to see how far we get!**



DRINK!


Disk Storage [PBytes]
5000
4000
3000
2000
1000
ATLAS Preliminary
■ Resource needs
(2017 Computing model)
— Flat budget model
(+15%/year)
Run 2
Run 3
Run 4
2018
2020
2022
2024
2026
2028
Pl