HTTP TPC & token-based AuthN/Z

Andrea Ceccanti
andrea.ceccanti@cnaf.infn.it

on behalf of the DOMA TPC WG
December 11th 2018

DOMA Third-party Copy (TPC) WG

DOMA WG dedicated to “improving bulk transfers between WLCG sites… finding viable replacements to the GridFTP protocol”

https://twiki.cern.ch/twiki/bin/view/LCG/ThirdPartyCopy

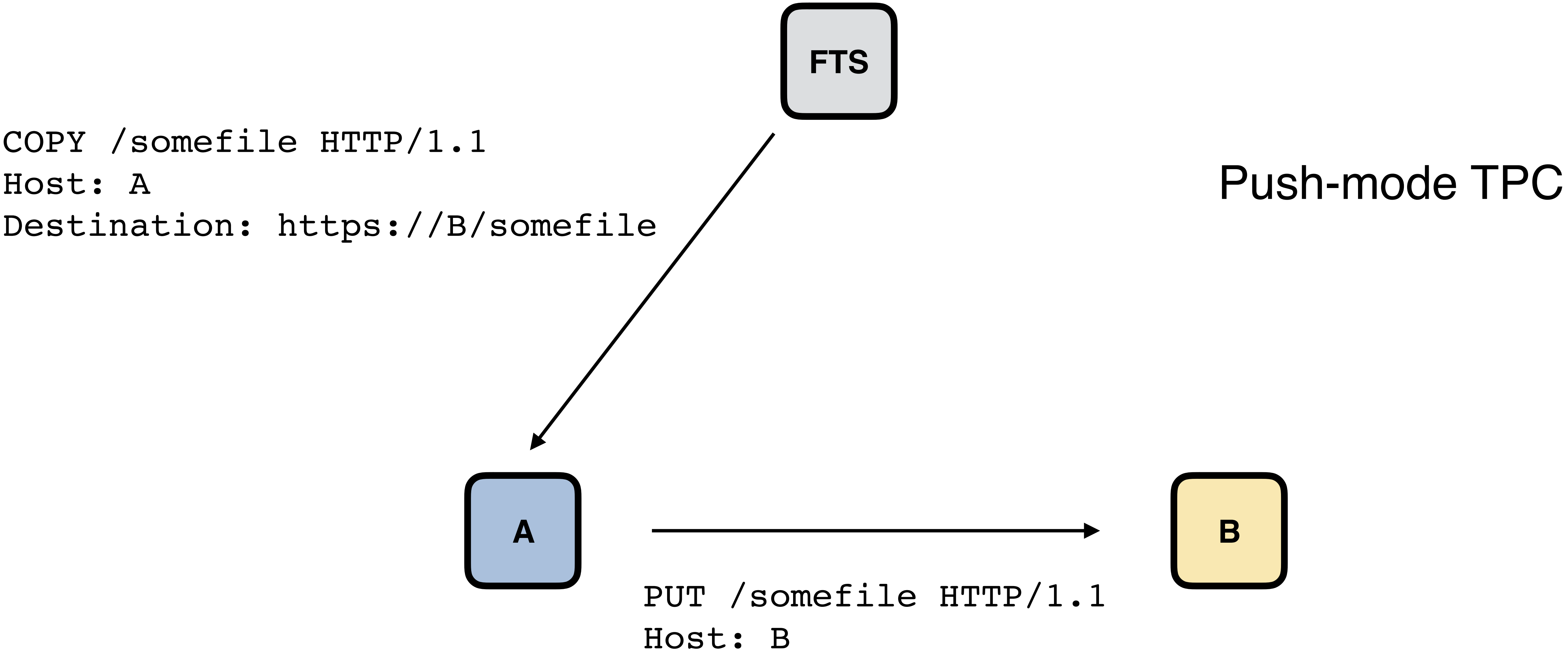
TPC has two sub-activities:

- HTTP/WebDAV TPC
- XRoot TPC

Today I only talk about HTTP/WebDAV TPC

HTTP/WebDAV Third Party Copy

Extend the WebDAV COPY verb semantics to trigger a third-party copy to/from a remote endpoint



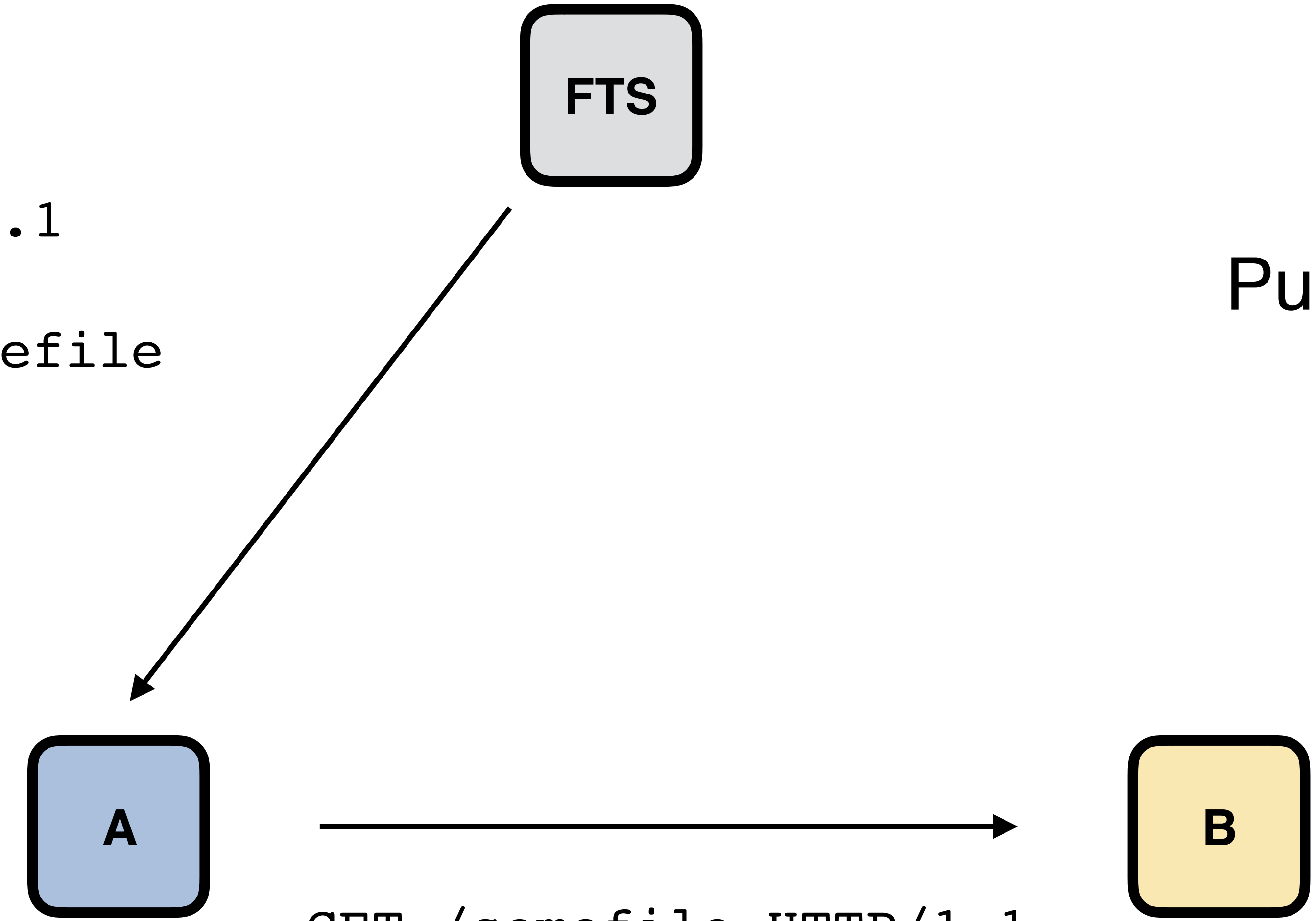
```
COPY /somefile HTTP/1.1
Host: A
Destination: https://B/somefile
```

Push-mode TPC

```
PUT /somefile HTTP/1.1
Host: B
```

HTTP/WebDAV Third Party Copy

Extend the WebDAV COPY verb semantics to trigger a third-party copy to/from a remote endpoint



```
COPY /somefile HTTP/1.1
Host: A
Source: https://B/somefile
```

```
GET /somefile HTTP/1.1
Host: B
```

Pull-mode TPC

TPC and delegated authorization

TPC can work

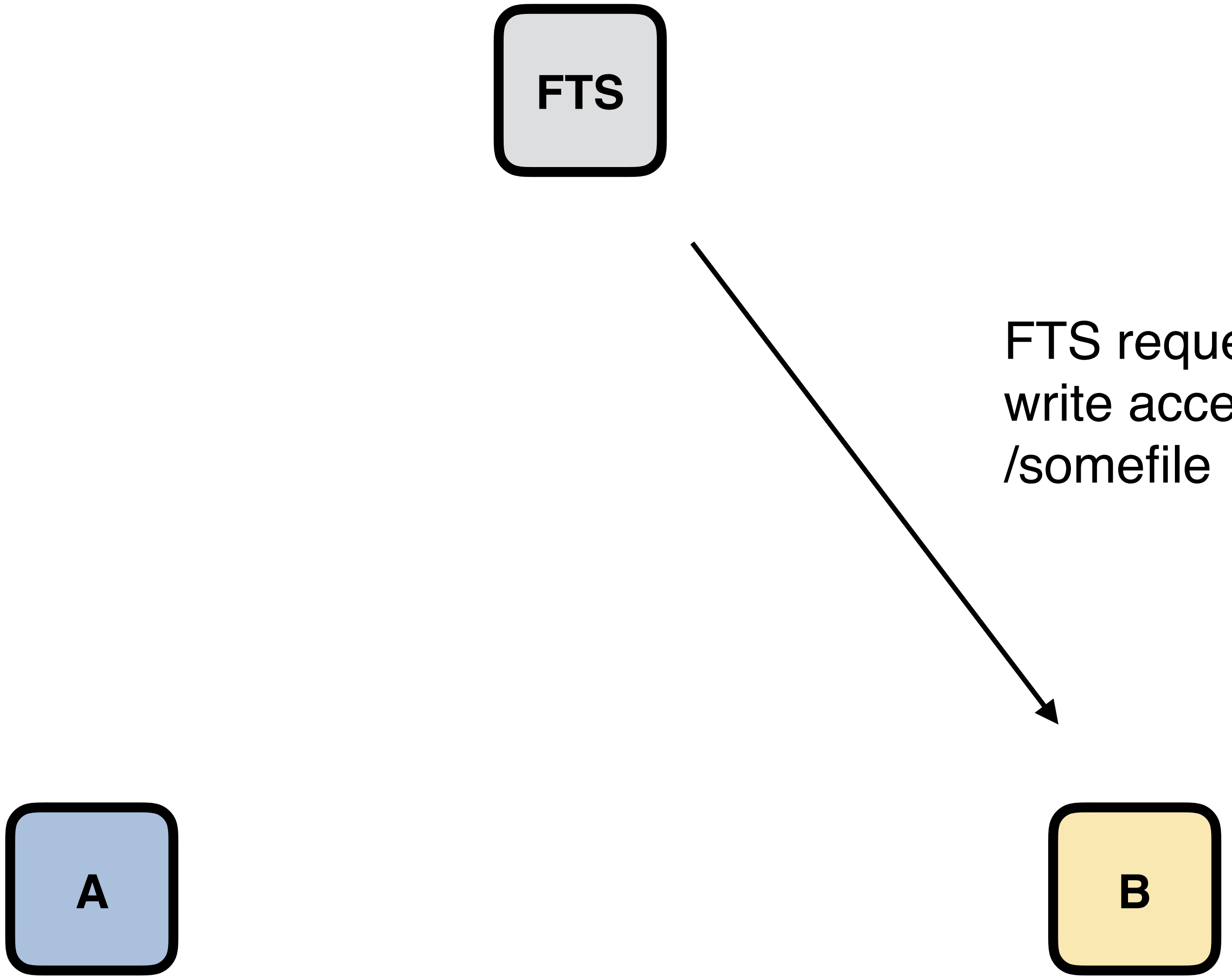
- without delegation, by building direct trust across SEs (via dedicated accounts, service certificates, VO-registered robot certificates…)
- with Gridsite or GSI delegation, for SEs that implement it

but the WG agrees on the need to avoid technologies that

- only work with X.509 credentials
- are proprietary
- not universally supported by WLCG storage

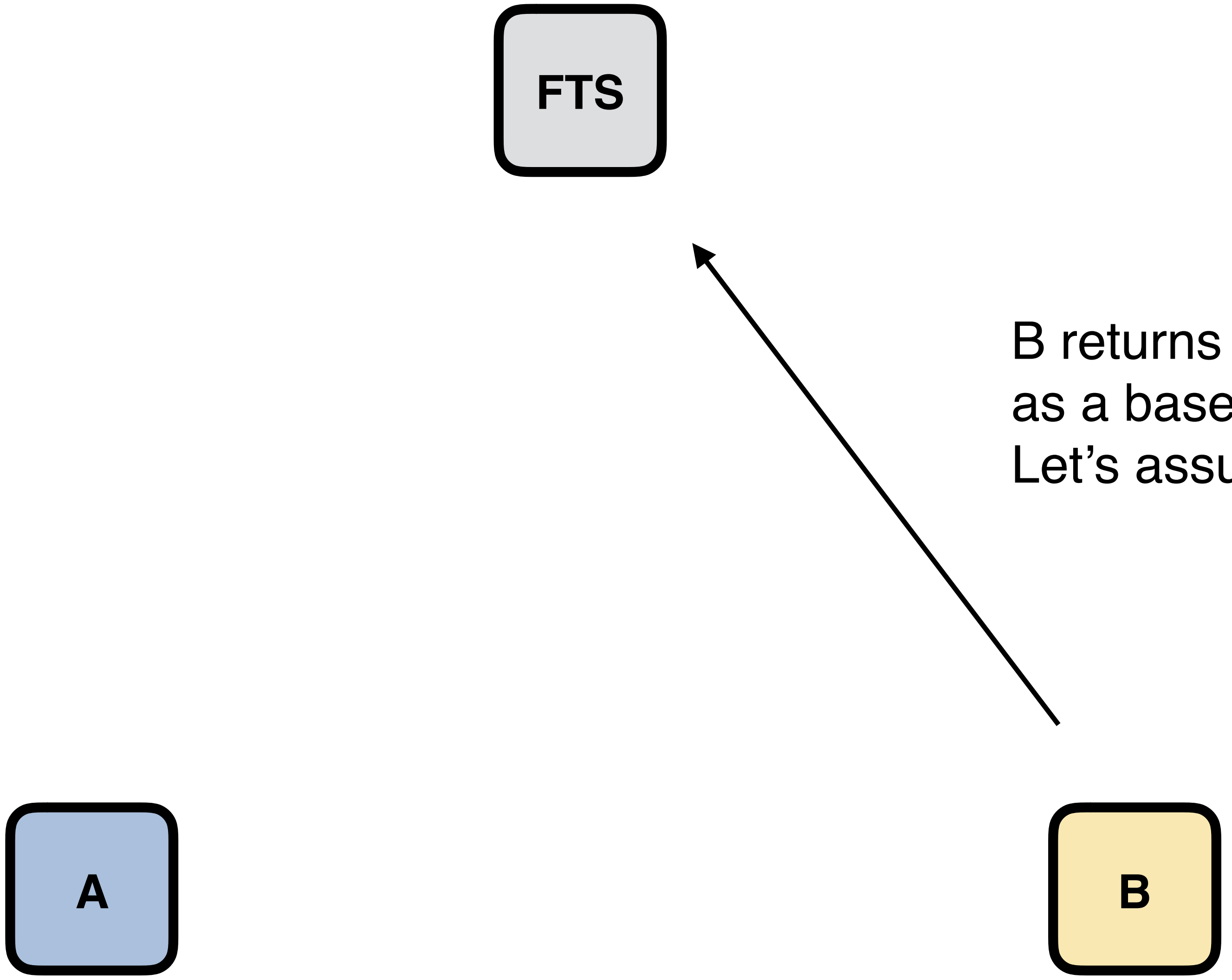
Token-based delegated authorization seems the natural solution for supporting TPC

Token-based delegated AuthZ example



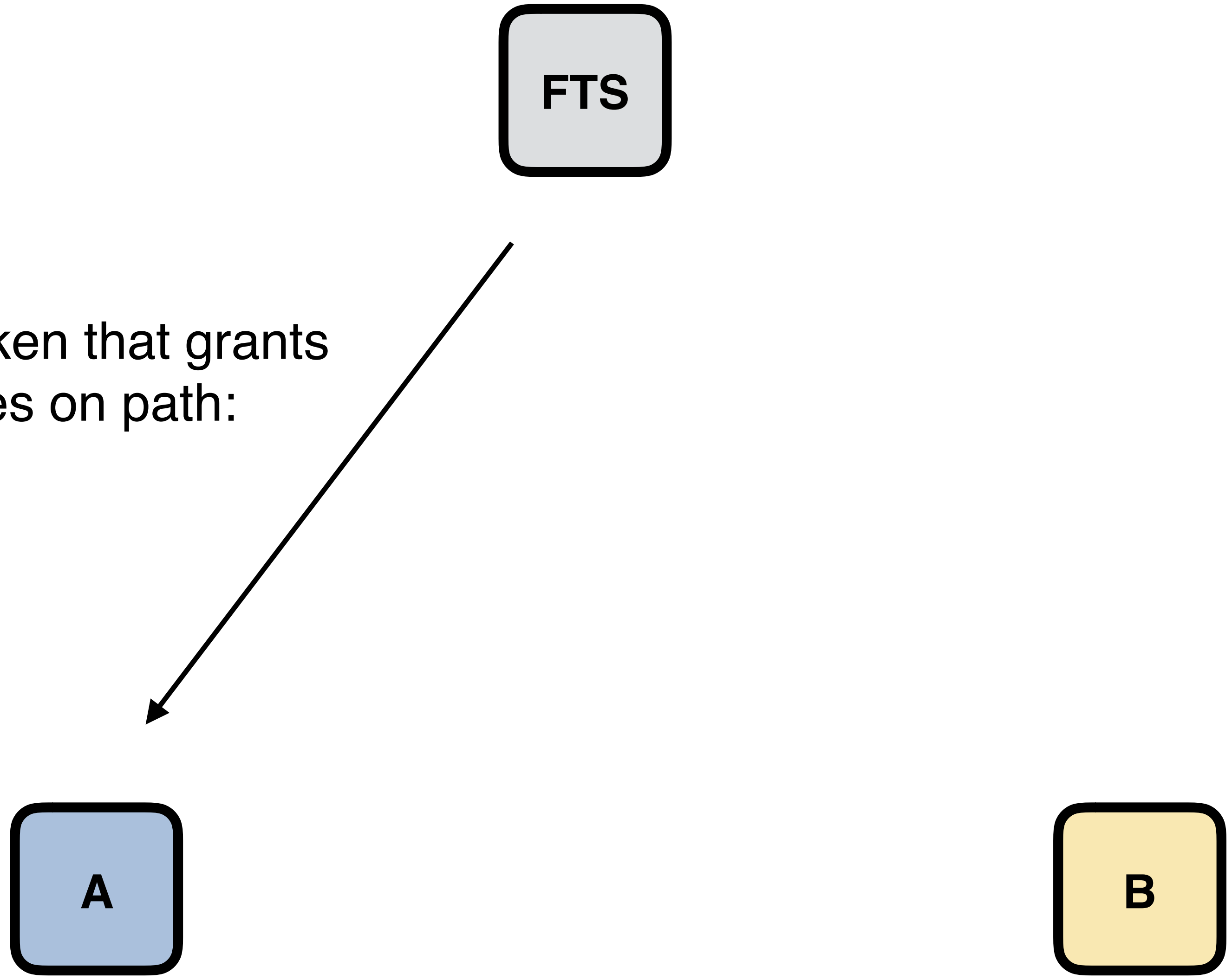
FTS requests a token that grants write access to path: /somefile

Token-based delegated AuthZ example



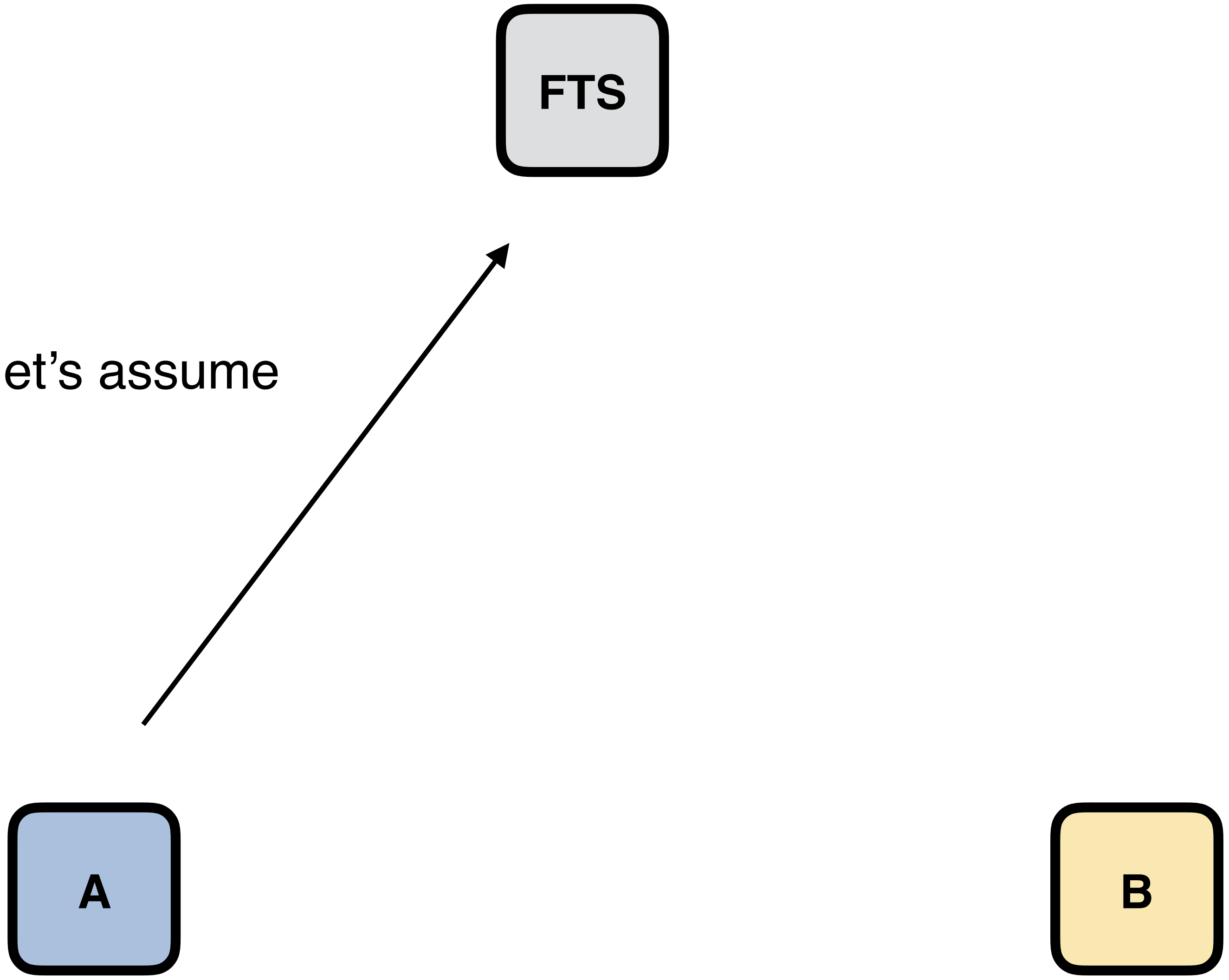
B returns a token, typically
as a base64-encoded string
Let’s assume this string is **T_B**

Token-based delegated AuthZ example



FTS requests a token that grants read/copy privileges on path: /somefile

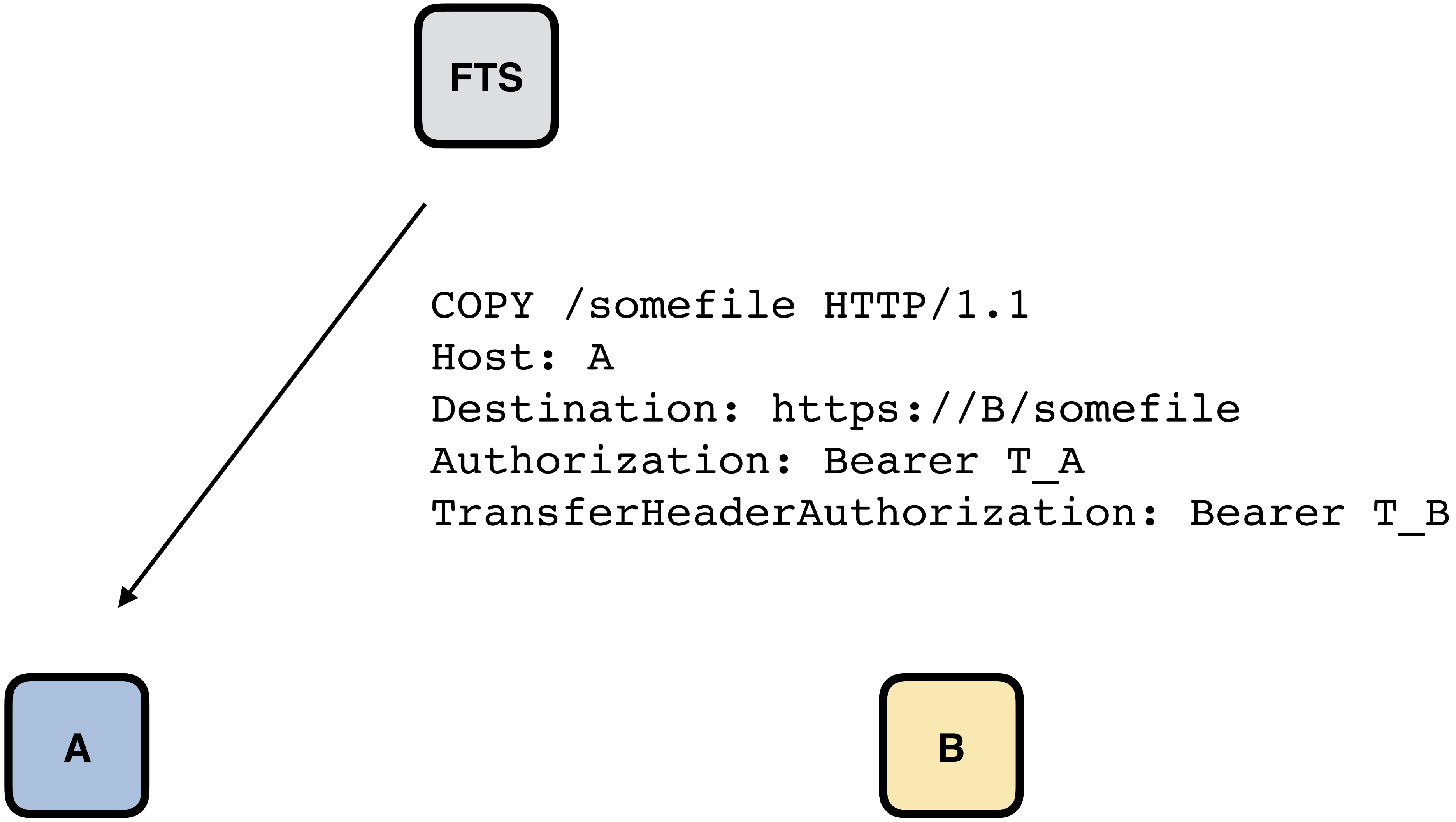
Token-based delegated AuthZ example



A returns a token, let's assume this token is **T_A**

Token-based delegated AuthZ example

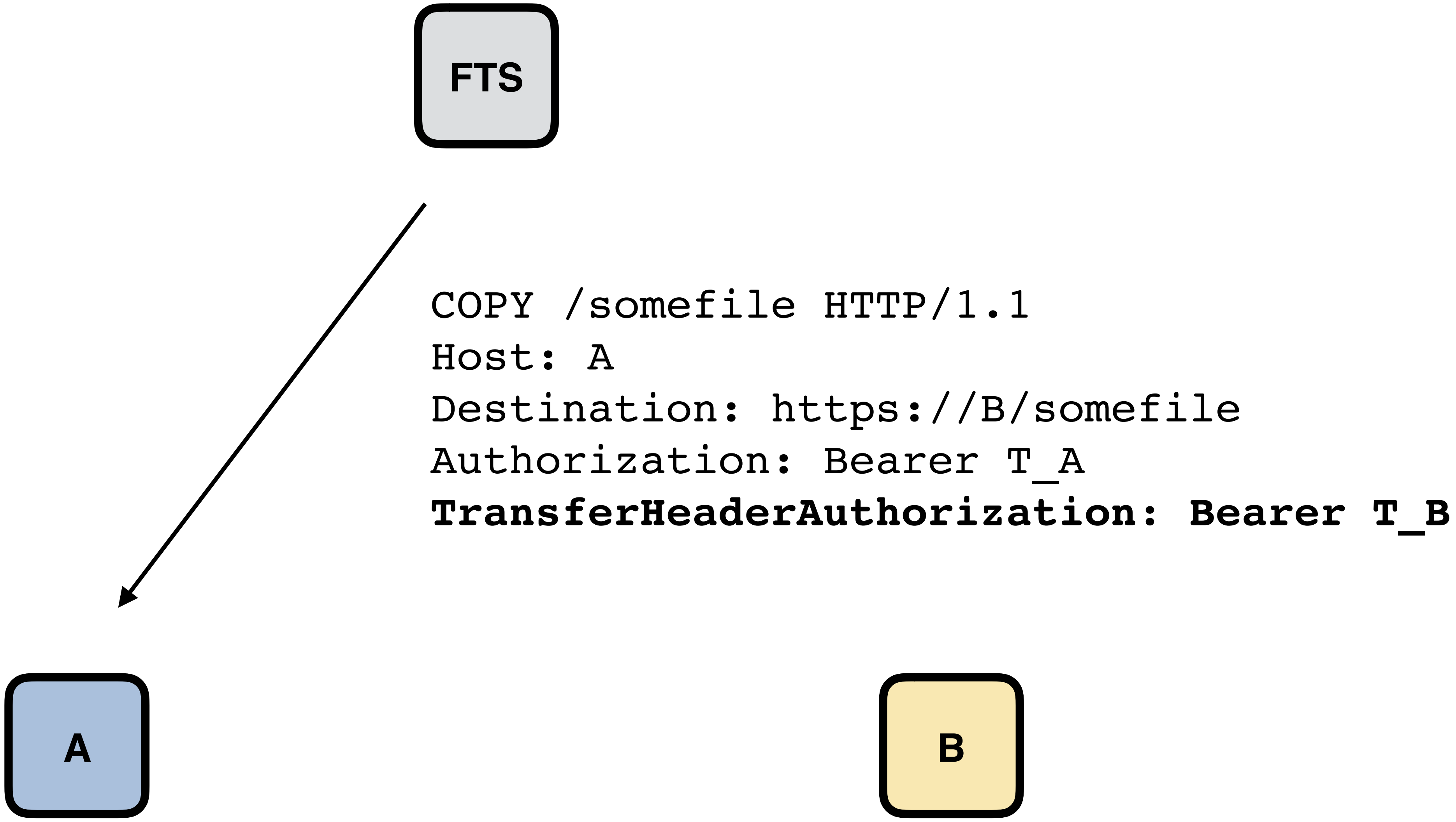
FTS can now request a TPC
from A/somefile to B/somefile



```
COPY /somefile HTTP/1.1
Host: A
Destination: https://B/somefile
Authorization: Bearer T_A
TransferHeaderAuthorization: Bearer T_B
```

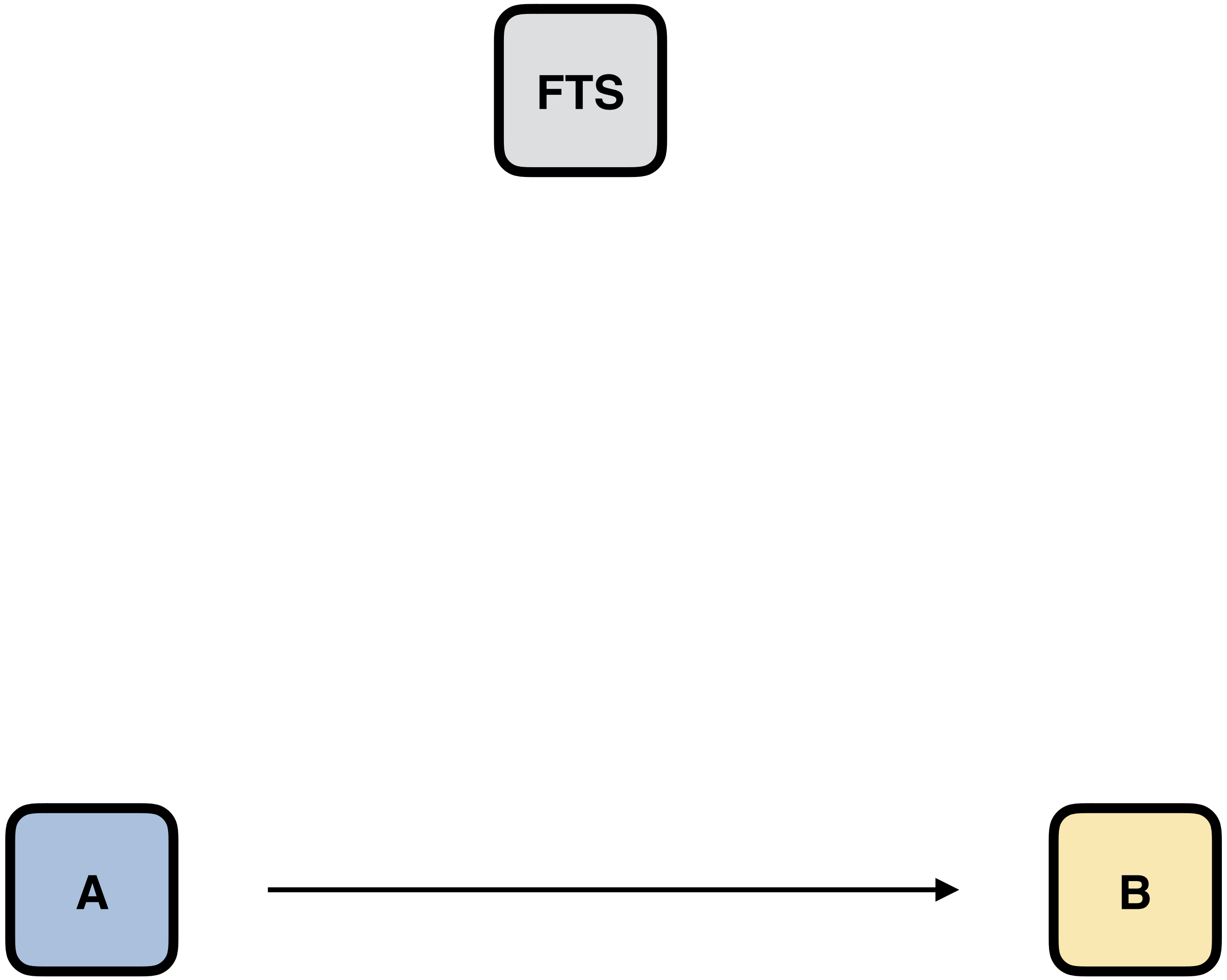
Token-based delegated AuthZ example

The protocol provides a way to request that certain headers in the COPY request are included in related transfer requests: all headers in the copy request starting with **TransferHeader** will be copied in the transfer request without such prefix.



```
COPY /somefile HTTP/1.1
Host: A
Destination: https://B/somefile
Authorization: Bearer T_A
TransferHeaderAuthorization: Bearer T_B
```

Token-based delegated AuthZ example



```
PUT /somefile HTTP/1.1
Host: B
Authorization: Bearer T_B
```

SE-issued authorization tokens

A token is issued by the SE and understood **only** at such SE

Different token formats can coexist

- Macaroons, JWTs, …

Different protocols for requesting tokens can coexist

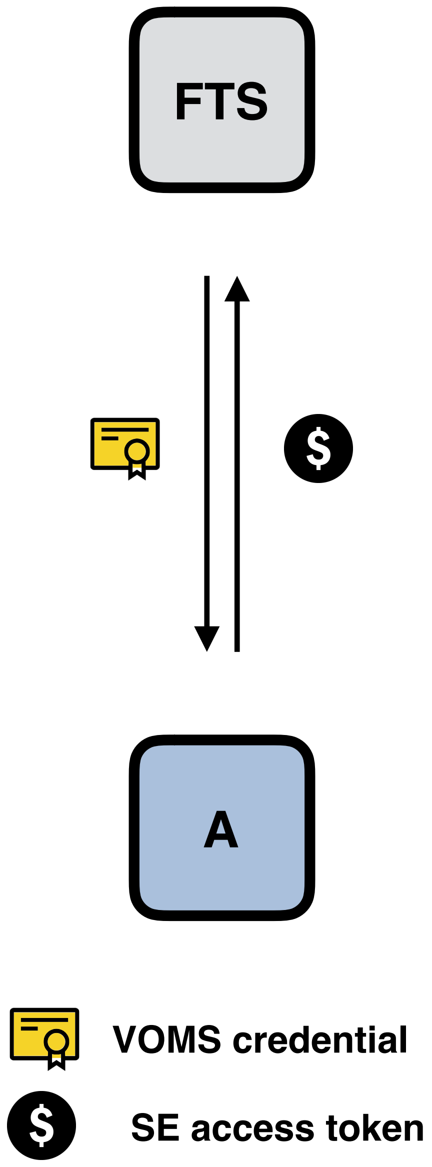
- Macaroon requests, OAuth, …
- but the client (i.e., FTS) needs to support all the different protocols to make endpoints talk to each other

Obtaining access tokens

A VOMS proxy is exchanged with a bearer token, possibly limiting the privileges associated with the token and its scope

- with caveats, for Macaroons
- with OAuth scopes and audience, for JWTs

Macaroons can also be limited after token issue time, without further calls to the SE, but this requires macaroon handling capabilities at the client



Work started to converge on a common, OAuth-based interface to request tokens

- independent of the actual token technology used

DOMA TPC and WLCG AuthZ WG

How can the WLCG AuthZ WG work be integrated?

OAuth/OpenID Connect is being increasingly supported by WLCG data services

- FTS, dCache, XRootD, StoRM,…

Possible integration points:

- VO-issued tokens supported for AuthZ at the SEs, following the rules of the WLCG common JWT profile
- VO-issued tokens understood at the SE token request endpoint, and exchanged for the SE-issued authorization token