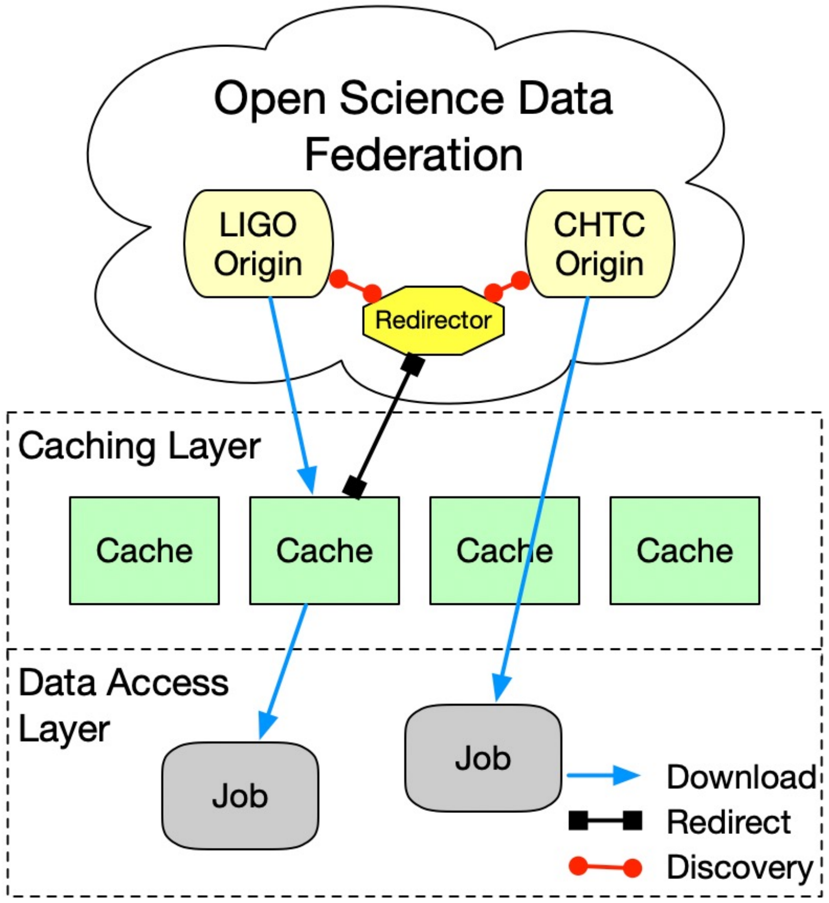
To the OSPool and Beyond: The guts of the OSDF client

Brian Bockelman

XRootD Workshop, March 2023

The Open Science Data Federation (OSDF)

- The OSDF aims to help researchers move objects to and from computation in support of open science.
- The OSDF provides:
  - The *origin service*, which integrates an existing object store or filesystem into the OSDF.
  - A *redirector*, helping clients to find the objects.
  - A set of distributed *caches* for scalable data distributions.
  - A client for accessing objects from jobs.



Why OSDF?

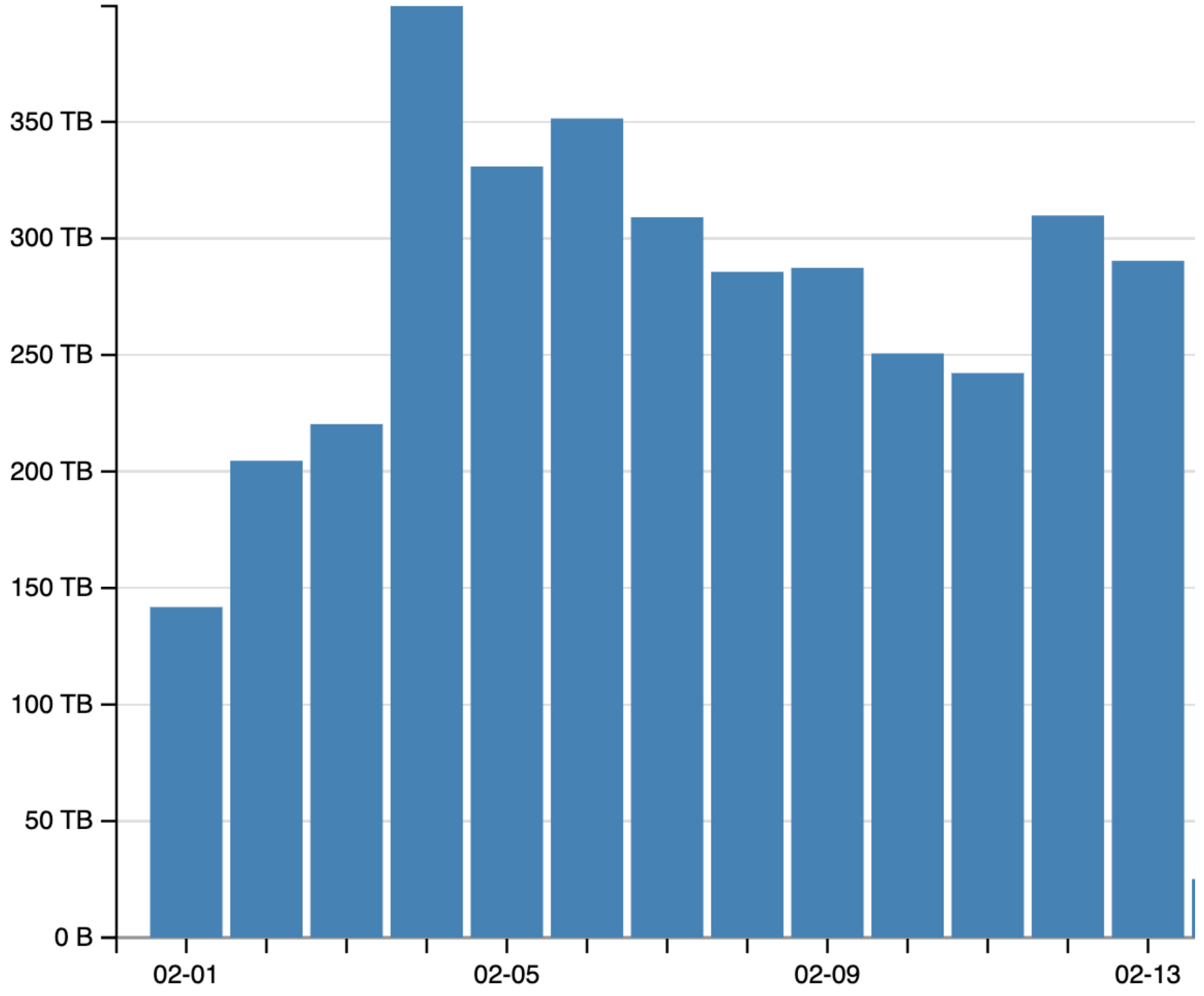
By connecting to the OSDF, the filesystem / object store owner can distribute their objects in a scalable, reliable manner with a consistent authorization scheme.

OSDF "at a Glance"

During February,

PROJECTS	CACHES	TRANSFERRED
42	27	5.4 PB

Boise
Sunnyvale
Pasadena
San Diego
Honolulu
Houston
Kansas City
Lincoln
Cincinnati
Atlanta
Jacksonville
New York
Newport News

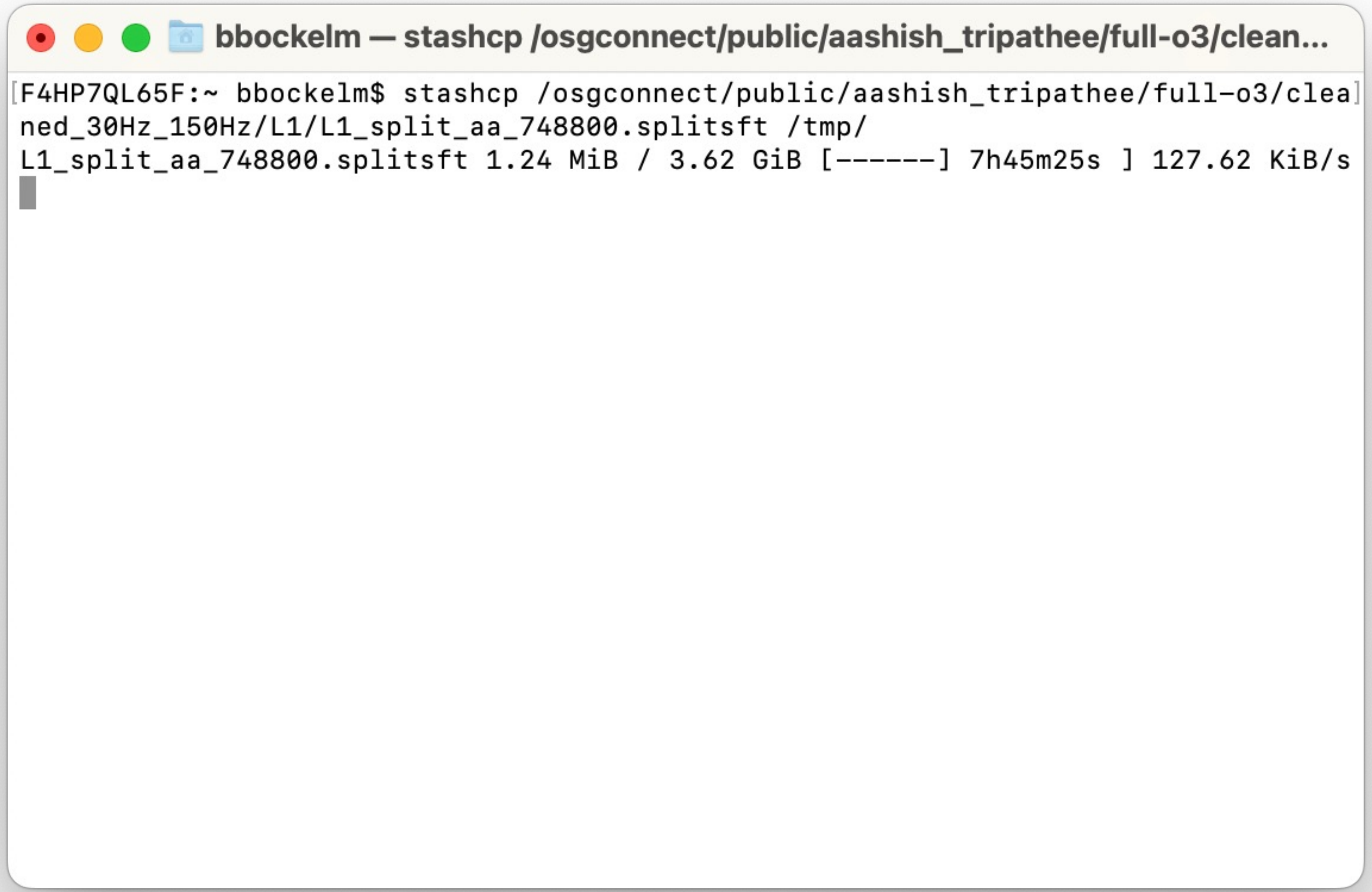


What’s needed in a client?

1. **Invoke**: The client determines work to be done.
2. **Authorize**: Acquire the credentials necessary for each transfer.
3. **Discover**: For each transfer, find an appropriate endpoint to do the work.
   - Not all caches are willing to serve the entire namespace.
   - Some transfers are directly with origins (uploads).
4. **Transfer**: Actually attempt the data upload / download.
5. Repeat! (If needed on errors)

The OSDF Client: **Invoke**

- The client has two binaries,
  - **stashcp**: A `cp` like interface. Intended to be invoked by users at the CLI.
  - **stash_plugin**: A HTCondor file transfer plugin.
- `stashcp` has user-friendly features like progress bar, transfer resumption, recursive downloads.
- `stash_plugin` provides structured output and error messages about transfer results.
- Implementation is in go; everything is in a self-contained, statically-linked binary.


bbockelm — stashcp /osgconnect/public/aashish_tripathee/full-o3/clean...

```
[F4HP7QL65F:~ bbockelm$ stashcp /osgconnect/public/aashish_tripathee/full-o3/clea]
ned_30Hz_150Hz/L1/L1_split_aa_748800.splitsft /tmp/
L1_split_aa_748800.splitsft 1.24 MiB / 3.62 GiB [------] 7h45m25s ] 127.62 KiB/s
```

Works on Windows, Mac, and Linux!

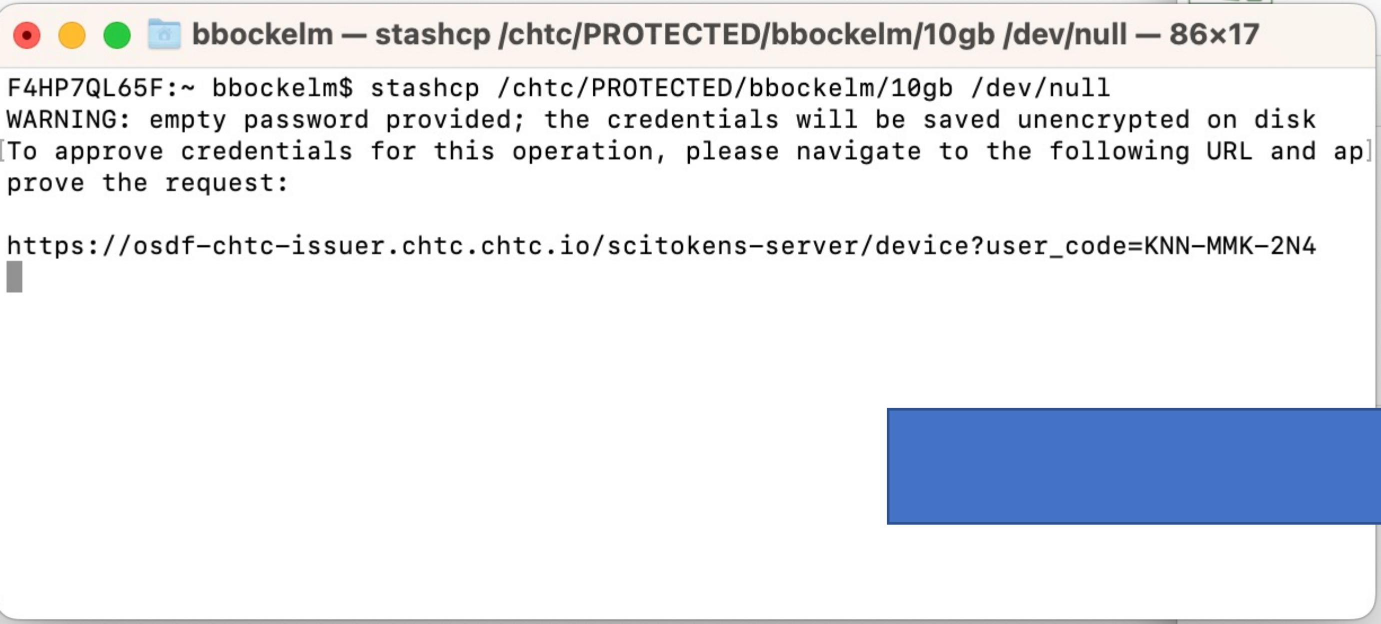
The OSDF Client: Authorize

- The client will download the list of namespaces from the OSG topology service and determine if any transfers need a token for authorization.
- If a token is needed, it will:
    - Look first in the environment ($BEARER_TOKEN, $BEARER_TOKEN_FILE)
    - Look in the encrypted client configuration for a usable token.
    - Try to generate a token if none are found.

The OSDF Client: Authorize

- If no token is found:
  - Invoke an API to determine if there’s a token issuer associated with the URL.
  - Refresh the token if a refresh token is present.
  - Do an automated OAuth2 client registration with the issuer if no client available.
  - Perform device flow code to get a new access token and refresh token.
- All is saved in an encrypted file on disk. Decryption password is saved in the kernel keyring for the duration of the session.

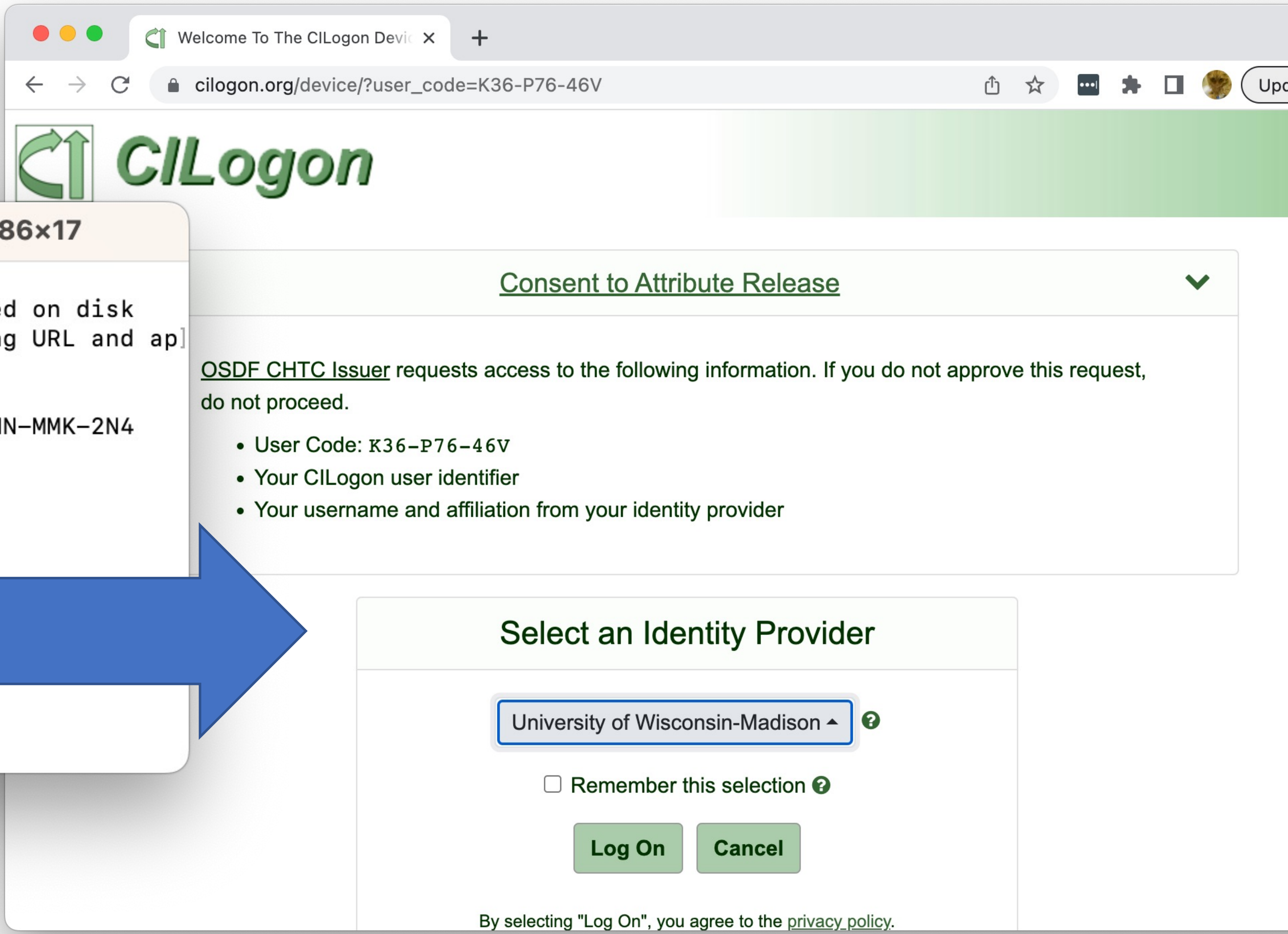
The OSDF Client: Authorize



bbockelm — stashcp /chtc/PROTECTED/bbockelm/10gb /dev/null — 86×17

```
F4HP7QL65F:~ bbockelm$ stashcp /chtc/PROTECTED/bbockelm/10gb /dev/null
WARNING: empty password provided; the credentials will be saved unencrypted on disk
To approve credentials for this operation, please navigate to the following URL and ap
prove the request:

https://osdf-chtc-issuer.chtc.chtc.io/scitokens-server/device?user_code=KNN-MMK-2N4
```



Welcome To The CILogon Devi

cilogon.org/device/?user_code=K36-P76-46V

CILogon

Consent to Attribute Release

OSDF CHTC Issuer requests access to the following information. If you do not approve this request, do not proceed.

- User Code: K36-P76-46V
- Your CILogon user identifier
- Your username and affiliation from your identity provider

Select an Identity Provider

University of Wisconsin-Madison

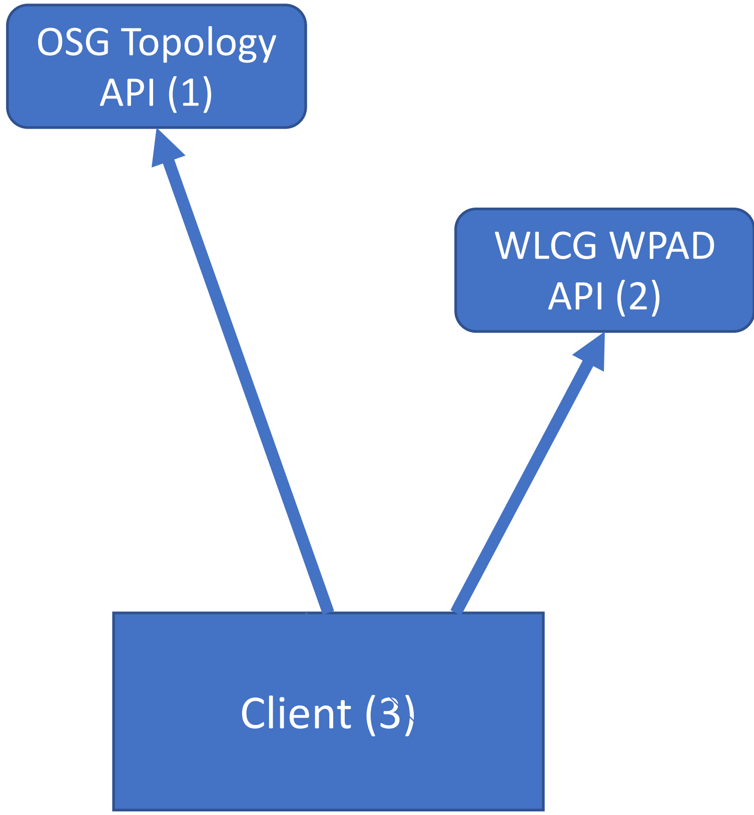
☐ Remember this selection

Log On Cancel

By selecting "Log On", you agree to the privacy policy.

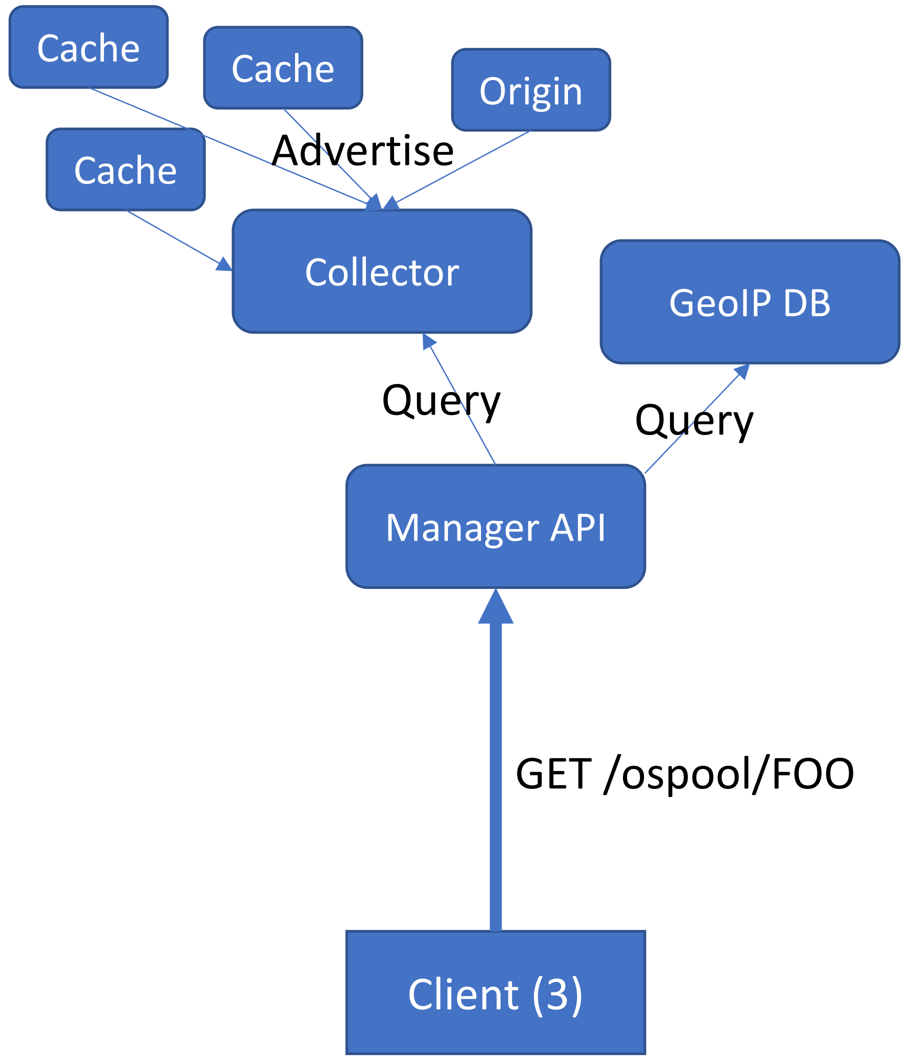
Cache Discovery - Today

- This is where the amalgam of functionality is quite apparent:
    1. Invoke an API to get the (unordered) set of caches willing to serve the namespace.
    2. Invoke a second API – based on the WLCG WPAD / GeoIP service to get the list of closest caches.
    3. Compute intersection of the set (1) and the list (2) to get the final list.
- Problems:
    - Steps (1) and (2) are run by different teams and the cache list gets out of sync.
    - Logic is all client-side – any updates to the cache discovery algorithm requires a new client release!
    - At no point does this consider actual cache status. A cache that is down or overloaded may be first on the list.



Cache Discovery – "Tomorrow"

- Services in OSDF advertise to a central collector.
- Manager webapp periodically queries the collector to build the topology of the OSDF namespace.
- Client sends a GET to the manager, is redirected to the nearest cache (HTTP 307).
  - Redirect headers also contain alternate sources.
  - Will also contain the information on the token issuer for this base path.



Cache Discovery – “Tomorrow”

```
> GET /osgconnect/public/osg/testfile.txt HTTP/2
> Host: osdf-cache-manager.osgdev.chtc.io
> user-agent: curl/7.86.0
> accept: */*
>

< HTTP/2 307
< content-type: text/html; charset=utf-8
< date: Mon, 27 Mar 2023 13:06:14 GMT
< link: <stash.farm.particle.cz:8000>; rel="duplicate"; pri=1,
        <ds-914.cr.cnaf.infn.it:8000>; rel="duplicate"; pri=2,
        <fiona-r-uva.vlan7.uvalight.net:8000>; rel="duplicate"; pri=3,
        <stashcache.edi.scotgrid.ac.uk:8000>; rel="duplicate"; pri=4,
        <xcachevirgo.pic.es:8000>; rel="duplicate"; pri=5, ...
< location: http://stash.farm.particle.cz:8000/osgconnect/public/osg/testfile.txt
```

Transfer (and Repeat)

- This is the easy part – after the “cache discovery”, one has a HTTP URL.
- The clients simply invoke $YOUR_FAVORITE_HTTP_CLIENT.
  - The one we’ve chosen has a few niceties – download resumption.
- On failure, we walk down the list of sources and retry up to 3 times.
- Clients will pipeline up to 5 transfers in parallel.

Active threads & Future Activities

- We want the clients to be usable by <u>anyone</u>. Particularly, this means error messages must be “human optimized” not “developer optimized”.
    - This is not just ‘write better error messages’ in the client but also ‘change XRootD to provide better error messages’.
- Clients discussed have all been command line. To really capture “normal users”, we need <u>browser-based clients</u>.
    - Goal is to allow uploads/downloads from laptop through the browser; no standalone software download needed.

Thanks!

This project is supported by National Science Foundation under Cooperative Agreement OAC-2030508. Any opinions, findings, conclusions or recommendations expressed in this material are those of the authors and do not necessarily reflect the views of the National Science Foundation.