Using the Dynafed dynamic data federator as a Rucio storage element

Frank Berghaus

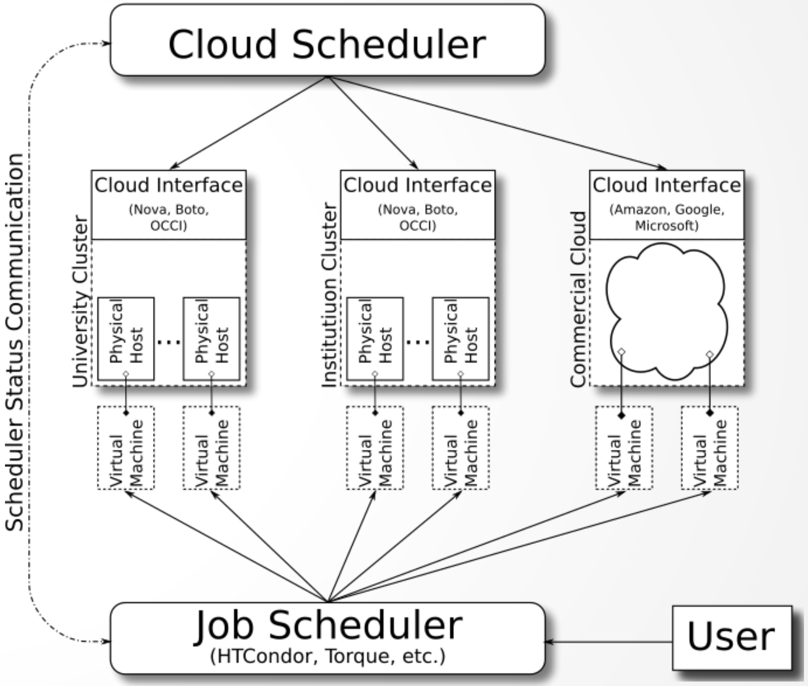
University of Victoria

much help from
UVic, TRIUMF, CERN-IT, ATLAS, Belle-II

Introduction & Motivation

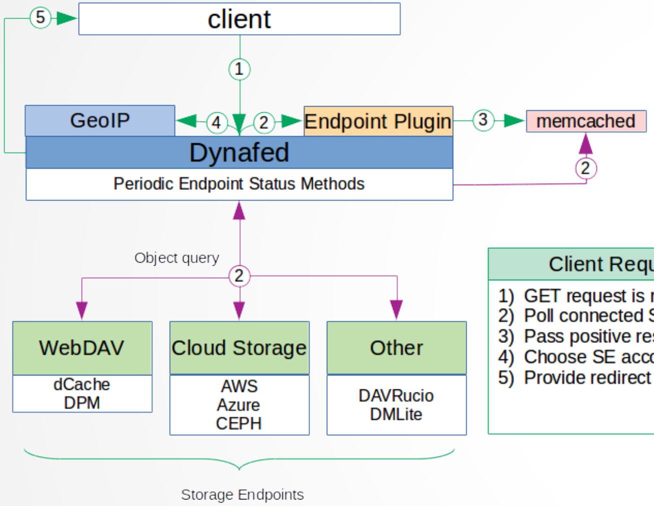
P Armstrong et al, arXiv:1007.0050

- Distributed cloud system
  - cloudscheduler
  - In production for >8 years
  - User: DIRAC (Belle-II) or PanDA (ATLAS)
- Cloud Scheduler at UVic and CERN
- Cloud Resources:
  - In Canada, US, UK, Germany, Austria and at CERN
  - O(10^3) cores – easy to add more
- CE: HTCondor & Cloudscheduler
- SE: dCache (UVic), EOS (CERN)
- Limited by remote access to storage



Dynafed: Redirect To Nearby Storage

Reading from Dynafed



Client Request Processing

1) GET request is received
2) Poll connected SEs
3) Pass positive results to memcached
4) Choose SE according to GeoIP
5) Provide redirect link to client

Object Query

1) Dynafed checks cache for existing entry
2) Query all connected endpoints
3) Wait for returns until timeout
4) Cache all positive results
5) Provide list of file replicas

- Dynafed redirects to close storage
- Operating three configurations:
  - Belle-II at UVic:
    - R/O access (production)
  - ATLAS at CERN:
    - R/W to cloud storage (dev)
    - R/W to grid storage (dev)
- Instances operated by others:
  - data-bridge at CERN for *@home
  - Belle-II Dynafed at INFN
  - RAL ECHO
- Part of a WLCG Demonstrator

M Ebert et al, CHEP2018 presentation #105

S Pradi et al, CHEP2018 presentation #479

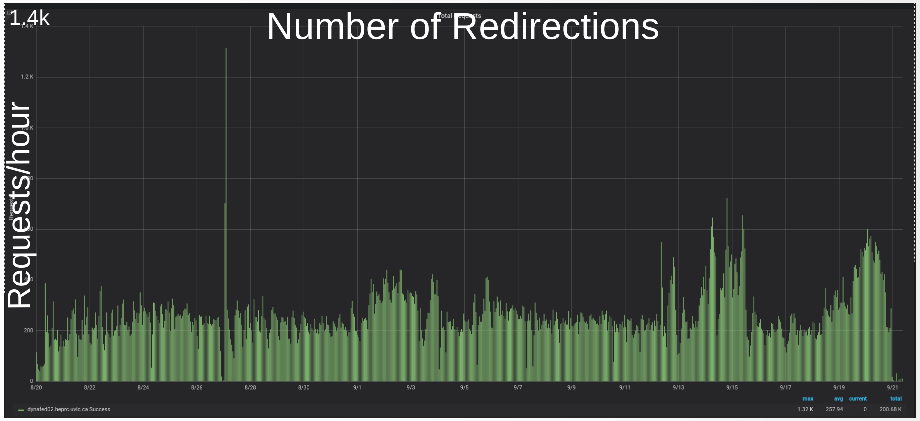
See Alastair’s presentation

Victoria Dynafed for Belle-II

See Paul Laycock's presentation

- With gfal2 support Belle-II will be able to use Dynafed as SE
- Workaround for Belle-II DIRAC:
  - gfalFS provides fuse mount within Linux directory tree:

    ```
    gfalFS -s ${HOME}/b2data/belle davs://dynafed02.heprc.uvic.ca:8443/belle
    ```
  - Jobs access Belle-II data from “local” directory ~/b2data/belle
- In production for the last two MC campaigns



- Easy addition of new endpoints
  - Added traditional Belle-II SEs while transferring new input data sets to own Endpoints:
    - Instant access to new files without configuration change on jobs/workers
- gfalFS and Dynafed work well for reading input data
  - Output is still written to UVic dCache using SRM
  - Waiting on gfal2 to be added to Belle-II offline computing

- Load is balanced across co-located storage endpoints
  - MC campaign: longer running jobs request at least one file
  - User analysis: short jobs request one file
  - Skimming & merging: shorter josb request multiple files
  - ~3000 job slots → 35TB per day
- Easy and effective network usage
  - Same configuration for all workers (6 separate clouds are used for Belle-II)
  - With same files used by many jobs network transfers stay local

Dynafed as ATLAS Storage Element

- Grid Rucio Storage Element:

  dynafed-atlas.cern.ch/data/grid

  – CERN (EOS), LRZ (dCache), ECDF (DPM)
  – CERN-EXTENSION_GRIDDISK

- Cloud Rucio Storage Element:

  dynafed-atlas.cern.ch/data/cloud

  – CERN (CephS3)
  – CERN-EXTSION_CLOUDDISK

- Authenticate with X.509+VOMS:

```
glb.allowgroups[]: "/atlas/*" /data rwl
glb.allowgroups[]: "/atlas/Role=production/*" /data rlwd
```

- Allow ATLAS Users to browse Dynafed by harvesting DNs from VOMS:

```
glb.allowusers[]: "/DC=ch/DC=cern/OU=Organic..." /data rl
…
```

- Rucio supports and SEs expose `HTTP+WebDAV`

See Paul's XDC presentation on OpenID

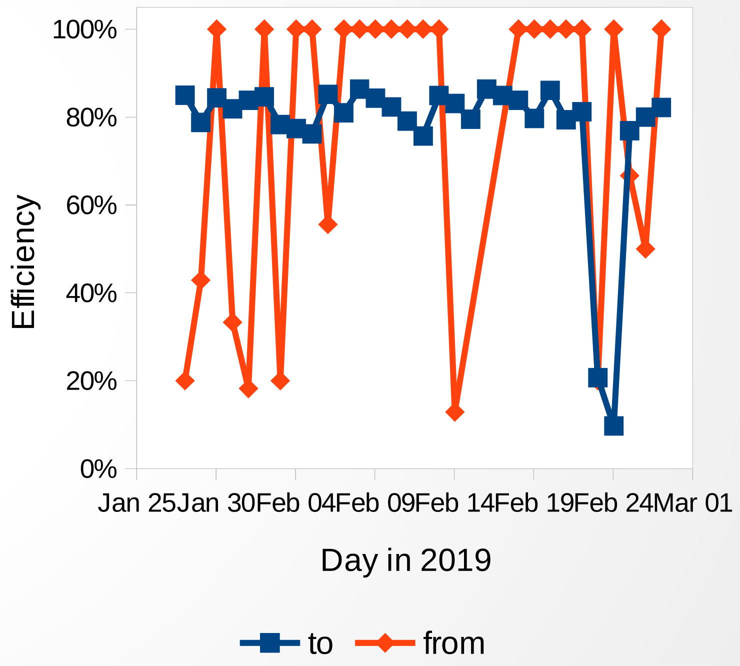
Experience With ATLAS and Dynafed

- Workload management:
  - Functional tests run against Dynafed
- Data management:
  - Works:
    - Reading, writing, deleting
    - Redirection
  - Work in progress
    - Checksums
    - Third party copy
    - Space reporting/accounting

Functional ✓

Production ✗

Transfers to and from Dynafed



Rucio, Dynafed, and Checksums

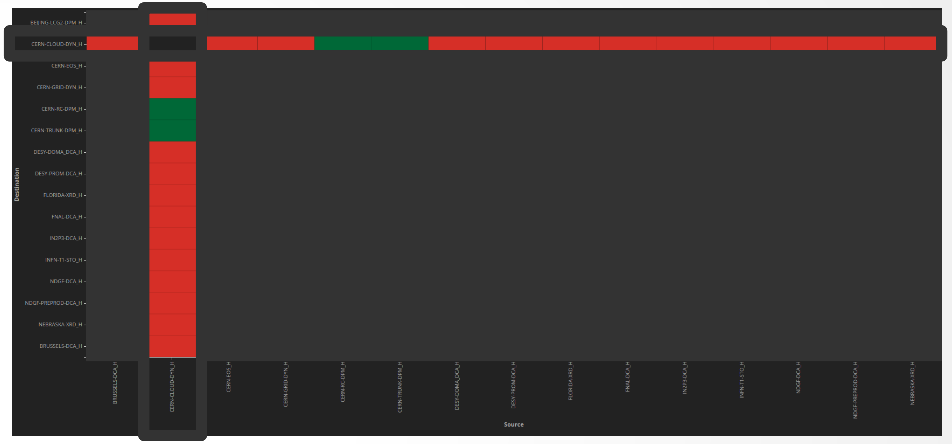
- Mechanism:
  - Grid: User is responsible, `Want-Digest [RFC3230]`
  - Cloud: Provider is responsible, `Content-MD5 [RFC1544]`
- Algorithm
  - Grid: `ADLER32 [RFC1950]`, `MD5 [RFC1321] (Rucio uses both/either)`
  - Cloud: `MD5 [RFC1321]` only
- Rucio expects the grid *mechanism*
  - *Workaround*: Flag for Rucio not to request checksum from Dynafed
- Dyanfed ongoing development:
  - On `Want-Digest`: call out to get checksum (if not in cache)
  - Cache checksum
  - *note*: hide implementation details
- In the pipeline – sometime this year

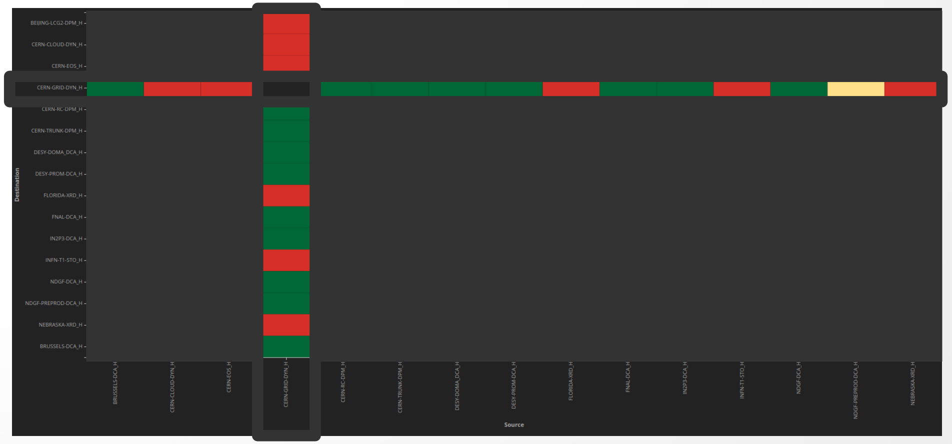
Third Party Copy – Cloud Storage

- Functionality released in December 2018
- On a copy COPY:
  - Redirect copy request, if supported
  - Else local call:
    - Default: gfal-copy
    - Note: if non-dynafed endpoint supports TPC it will push/pull
- Evaluating with DOMA-TPC

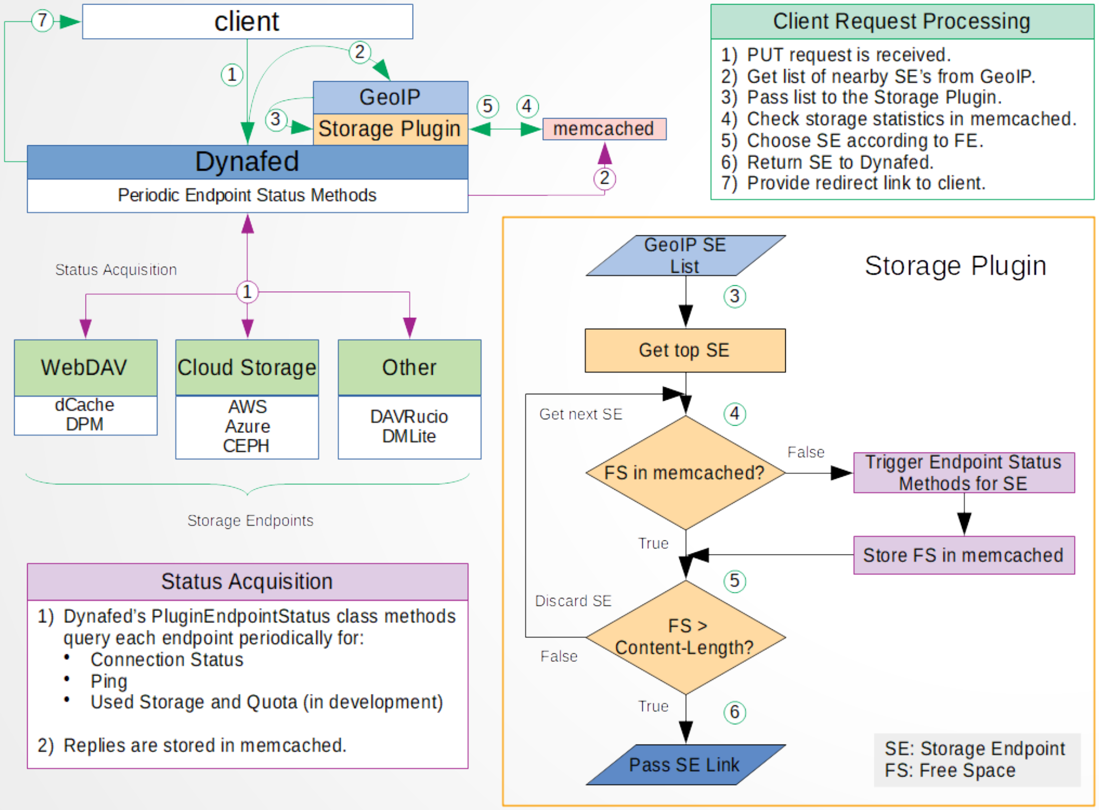
See Robert presentation on DvNE

See Alastair's presentation on RAL ECHO





Dynafed Storage Plugin



Client Request Processing

1) PUT request is received.
2) Get list of nearby SE's from GeoIP.
3) Pass list to the Storage Plugin.
4) Check storage statistics in memcached.
5) Choose SE according to FE.
6) Return SE to Dynafed.
7) Provide redirect link to client.

Status Acquisition

1) Dynafed's PluginEndpointStatus class methods query each endpoint periodically for:
   - Connection Status
   - Ping
   - Used Storage and Quota (in development)
2) Replies are stored in memcached.

SE: Storage Endpoint
FS: Free Space

- Issue with writing to Dynafed
  - Free space on endpoints unknown
- Query usage and quota from endpoints using script
  - Add results to cache
  - Generate `JSON` to inform Rucio
- Use:
  - WebDAV [RFC4331]
  - CephS3 r/o admin interface when possible
- Commercial providers don't provide quota
  - Query usage form billing
  - Manually set quota
- Work in progress: file list

Summary

- HPC workloads on distributed clouds works
- Dynafed shown to provide data access for O(10^3) workers
- Dynafed as a Storage Element is work in progress
  - Not be the design purpose of Dynafed
  - Work done will be interesting for others, hopefully :-)
- The code-camp and the contribution work flow are great!

Thank You!