Xrootd in the Disk Pool Manager (DPM)

Fabrizio Furano (CERN IT-ST)

DPM

- DPM is a multiprotocol and multiVO Storage Element tech, with old roots and modern architecture
  - All the protocols see the same data/metadata and can be used at the same time
  - Multiple VOs are served by the same cluster.
  - The clusters with the disk pools can be local, or distributed across sites, or configured as volatile (caches)

- Historically oriented to giving all the features needed by WLCG, past, present and future
- Particularly oriented to giving standardised setup recipes and a service-level administration CLI (one administers the whole service, not the individual components)
- Historical philosophy: Uses the native daemons of the data protocols it can serve, e.g. Apache for HTTP, Xrootd, globus-gridftp-server

- “Legacy” protocols: SRM, rfio, Cns/dpnsdaemon, dpmdaemon [all on their way out]
- “Modern” protocols: Xrootd, gridftp (ehm, well, “modern”…) , WebDAV/HTTP

DPM for an Xrootd-er

- a normal xrootd cluster that loads plugins to rely on the DPM coordination and namespace. Stuff like:
  - 100% coherent multiprotocol access (gridftp, xrootd, http)
    - SRM/rfio is there too, and will stay indefinitely but we consider it unsupported since June 2019
  - Quotas, POSIX semantics, coherent listings
  - Nice admin interface: dmlite-shell
  - Space reporting
  - Concept of “pool” and “filesystems”, which can be tagged as volatile (cache mode), enabled/disabled and that are monitored for health/space

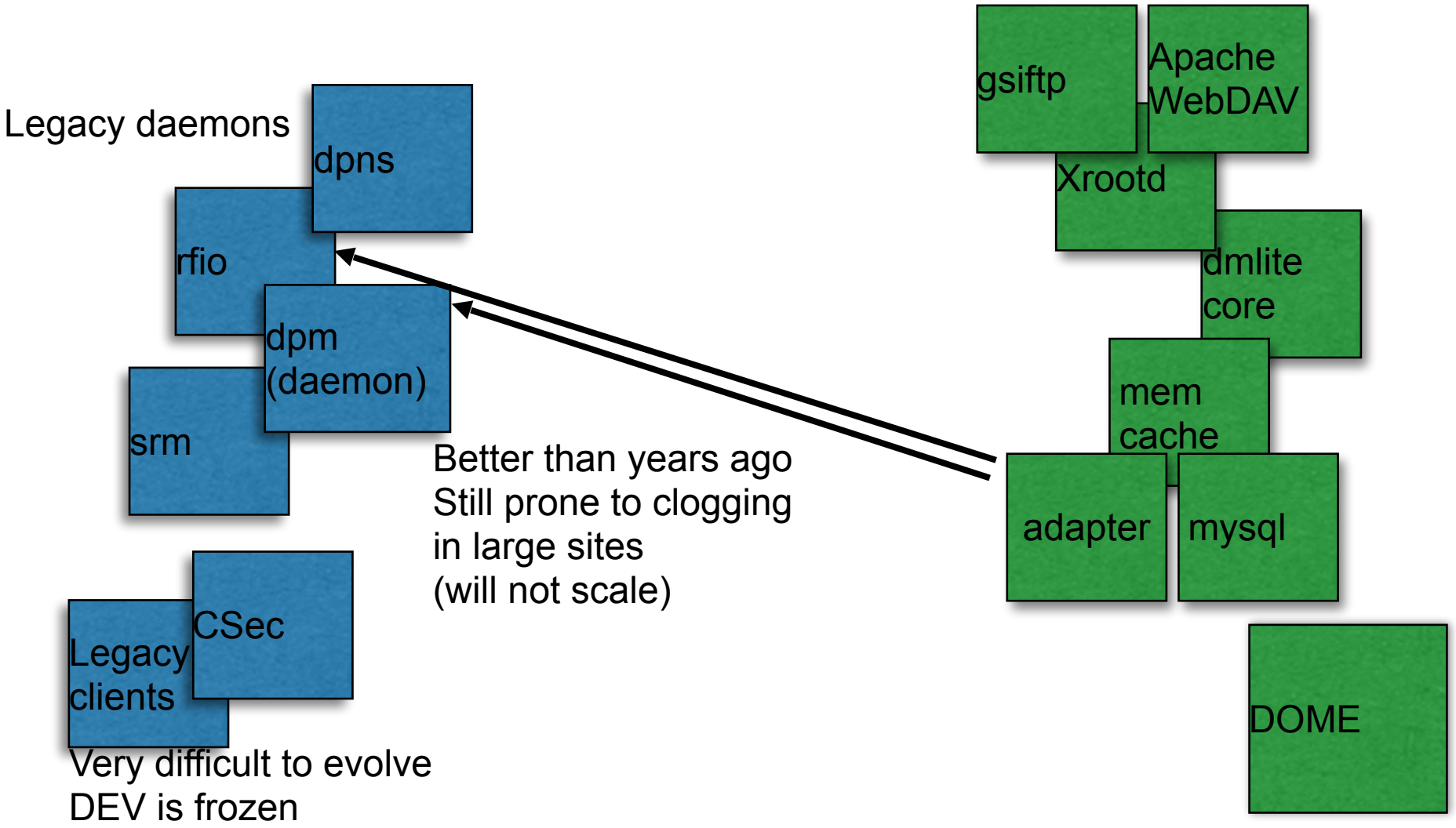
DPM in 2019

- DPM went through several rounds of evolution, with the goal of Long Term Support in mind
  - Goal: a well-working and well-documented open source project where people understand it and can contribute
  - Look around in Linux. Only projects that do this survive longer

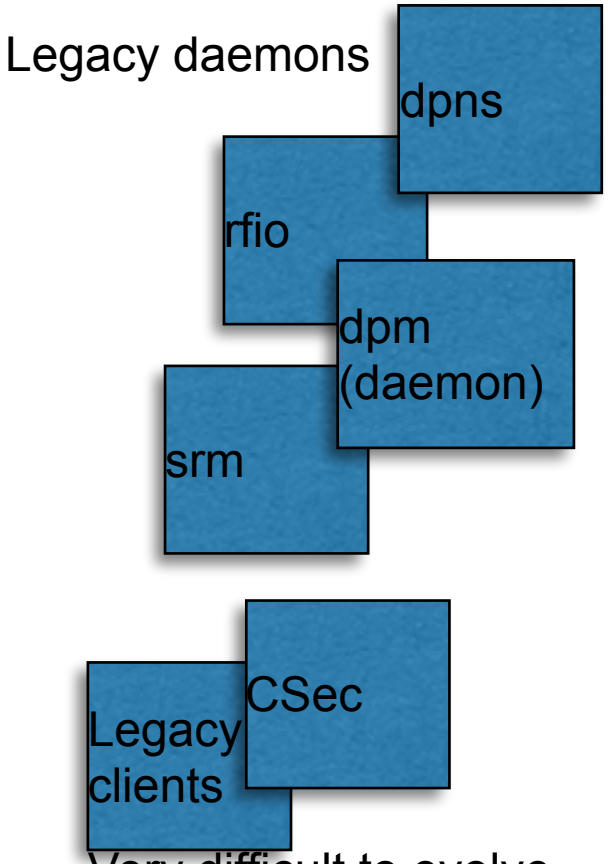
- This was approached in ~2016/17 with the development of DOME (Disk Operations Management Engine): the new DPM core

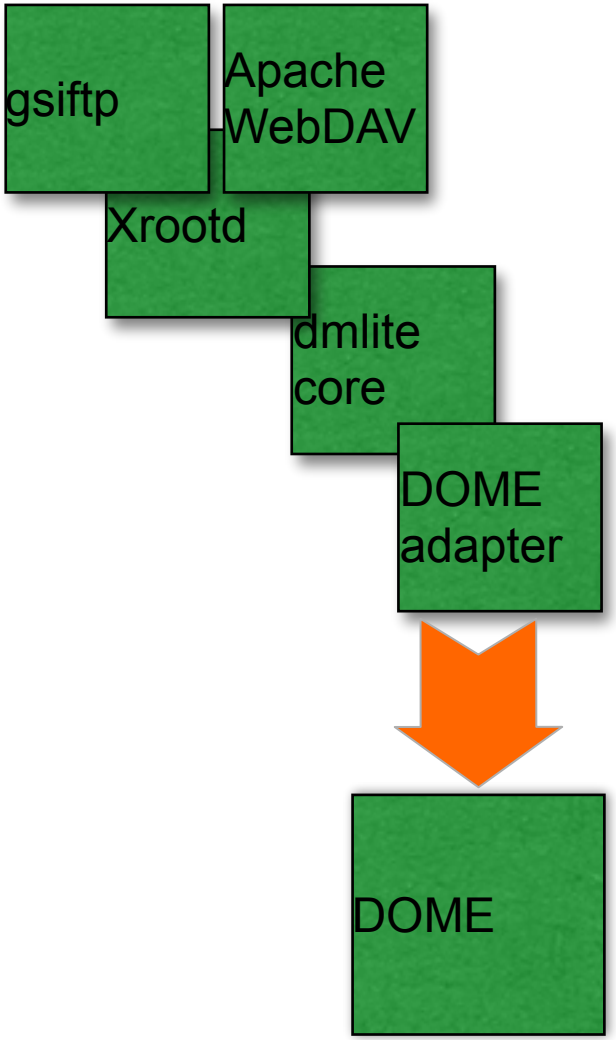
- DOME is a pedantically documented REST service that manages the headnode behaviour, including access to metadata
  - http://lcgdm.web.cern.ch/dome-documentation
- DOME is an internal service. Unlike Cns or dpmdaemon, clients do not see it and are not allowed to use it

DPM components and plugins (2017/2018)

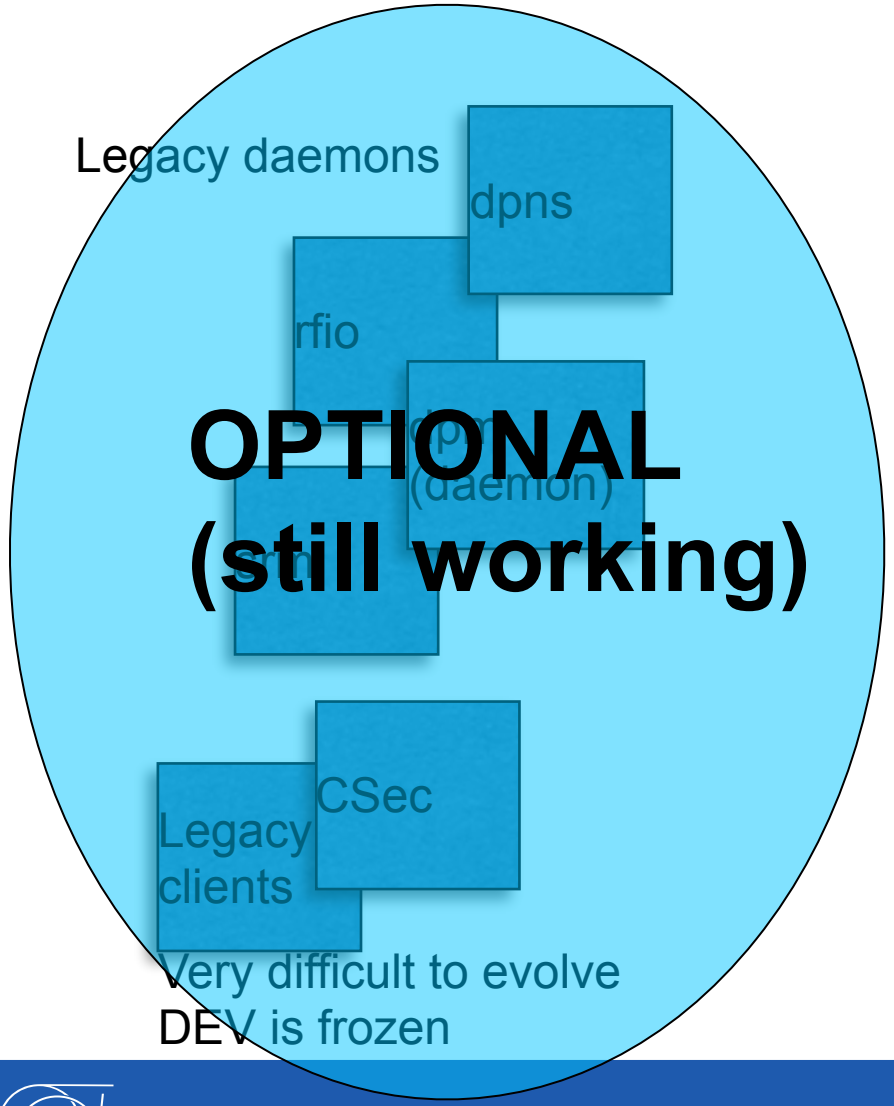


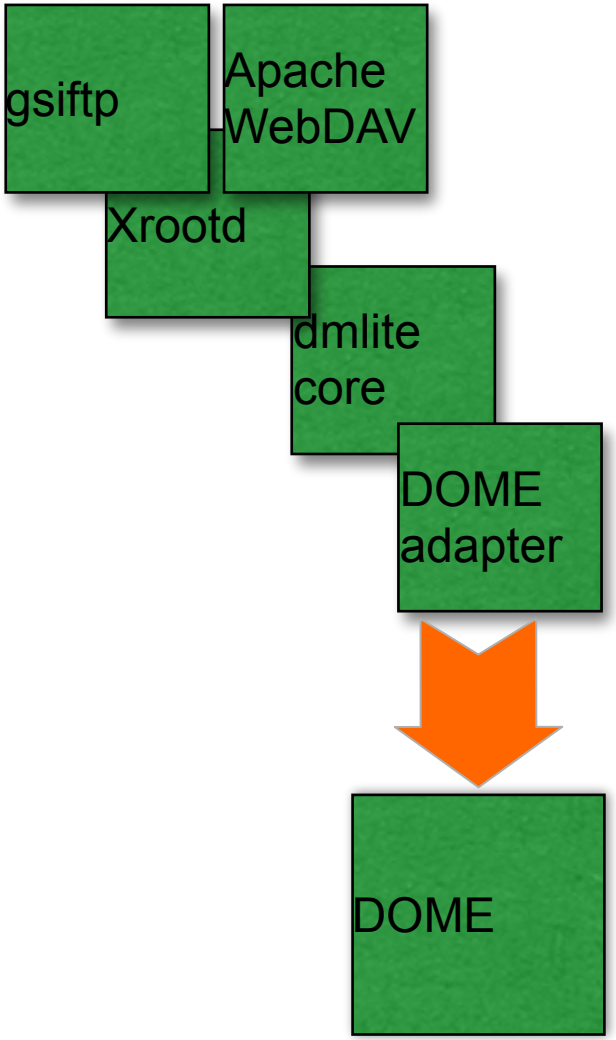
DPM components and plugins (2017/2018)





DPM components and plugins (2017/2018)





An example internal Stat()

- The internal protocol is ONE, fully documented and readable:
  - https://lcgdm.web.cern.ch/dome-documentation

```
GET /domehead/command/dome_getstatinfo
    HTTP/1.1
User-Agent: libdavix/0.6.8 neon/0.0.29
Keep-Alive:
Connection: Keep-Alive
TE: trailers
Host: dpmhead-trunk.cern.ch:1094
Content-Length: 17

> Body block (17 bytes):
{ "lfn": "/dpm" }
```

```
HTTP/1.1 200 OK
Content-Length: 250
{ "fileid": “3",
  "parentfileid": “2",
  "size": “265623786530",
  "mode": “16877",
  "atime": “1523455123",
  "mtime": “1522229608",
  “ctime": “1522229608",
  "uid": “0",
  "gid": “0",
  "nlink": “2",
  "acl": “A70,C50,F50,a70,c70,f50",
  "name": “dpm",
  "xattrs": "{\"type\": 0}”
}
```

Xrootd: a plugin-based data server framework

- Historically DPM uses the native Xrootd daemon to give service to Xrootd clients
- Some plugins (Oss, Acc, Finder, Cks) interface the guts of Xrootd with the DPM metadata subsystems in the head node

- Disk servers are almost vanilla Xrootd
  - A local Acc authorisation plugin validates the signed URLs provided to clients by the head node
  - A Cks plugin provides the checksum features of the DPM core. Supports all the checksum kinds out of the box, checksum caching, checksum recalculations. Seamlessly integrated with the other protocols.

- Xrootd federations work like in a vanilla xrootd cluster. Not much else to say here

Xrootd: a plugin-based data server framework

- By construction the DPM-Xrootd data performance is the same of a vanilla xrootd server cluster, as the Xrootd data path is untouched. It depends hence on the hardware.

- The metadata performance is the one of the DPM core. With decent real hardware one should expect to **comfortably serve** (real peak stat() performance):
  - O(100Hz) transactions for DPM configured in “legacy mode”
  - O(10KHz) transactions for DPM configured in “DOME mode”
  - These are end-to-end values including all the communication, clients and LAN

- With DOME there’s plenty of metadata performance headroom for the next rounds of data/metadata access requirements from the experiments. Existing large systems can scale up at least a factor
- More measurement precision or scaling is not interesting to us now. Will see in a few years.

DOME, fastCGI and XrdHTTP

- The first version of DOME (1.9, 2016) used Apache and fastCGI. It worked
  - Decent transaction performance
  - Somehow tricky configuration, but not too hard (Apache/fastCGI can be rough)

- Then fastCGI got broken beyond repair in CC7. Crappy performance, very high resource usage. No-one expected this. I have wasted many full days understanding the fastCGI code.
  - Long story short, some very important features have been removed in recent fastcgi libs, to accommodate average PHP-like scripts and components (sigh)

- 50 lines of code, no fear and DOME became a plugin of XrdHTTP, which I had contributed before to the xrootd framework
  - Performance got a 5X boost, same REST protocol, same code
  - The config became completely under control, and much simpler

- **Yes, DOME a.k.a the DPM core runs inside Xrootd since end 2017, and the results are great**

One daemon, two services

- The same xrootd daemon in DPM runs two services
  - **the xrootd protocol on port 1094**
    - *This is the xrootd service seen by the users*
  - **the HTTP/HTTPS protocol on port 1094**
    - And XrdHttp loads DOME (REST service) as a plugin
    - *This is the internal REST protocol, not seen by users*

- This port sharing makes things simpler on firewalls. No new ports to open across the cluster

- The same happens in the disk servers, as DOME has to run there too
  - In disk server mode, of course, port 1095

DPM-Xrootd setup

- In DPM we document as much as we can, and DPM-Xrootd is no exception
  - https://twiki.cern.ch/twiki/bin/view/DPM/DpmSetupManualInstallation#xrootd_AN1
- Our preference is to use our command-line setup though.
  - It's based on puppet and reduces by a factor the effort of tinkering with config files, also for the DPM devs in the case of support requests
  - https://twiki.cern.ch/twiki/bin/view/DPM/DpmSetupPuppetInstallation

Back to the xrootd data access service

- By construction, DPM-xrootd supports the official xrootd options, e.g.
  - monitoring
  - TPC
  - X509 with and without proxies
  - ALICE authz (through the usual plugins)
  - macaroons

- And the chances to support the future ones out of the box is very high

Apache or XrdHTTP in DPM ?

- Andy H. had asked: why do you use Apache for HTTP access? Just use the xrootd XrdHTTP that you are already using for the REST commands!
- Technically this is a very legitimate question

- I don't have strong feelings, and doing that would need low tech effort
  - Not necessarily a low setup effort, remember? 100 sites! Puppet stuff around!

- However Apache has its advantages
  - Extremely robust, good performance
  - Supports third-party plugins, e.g. OpenID-Connect natively
  - Its role in DPMs is well consolidated, with TPC, macaroons, X509/VOMS delegation

- It's clunky and memory-hungry however, more or less depending on the versions

- Conclusion: so far there's no need to change a reasonably well-working component. Changing things has a cost for the sysadmins and for the DPM devs