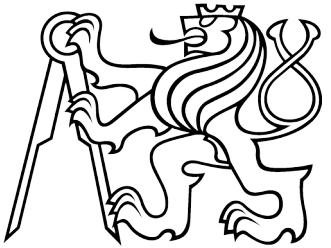


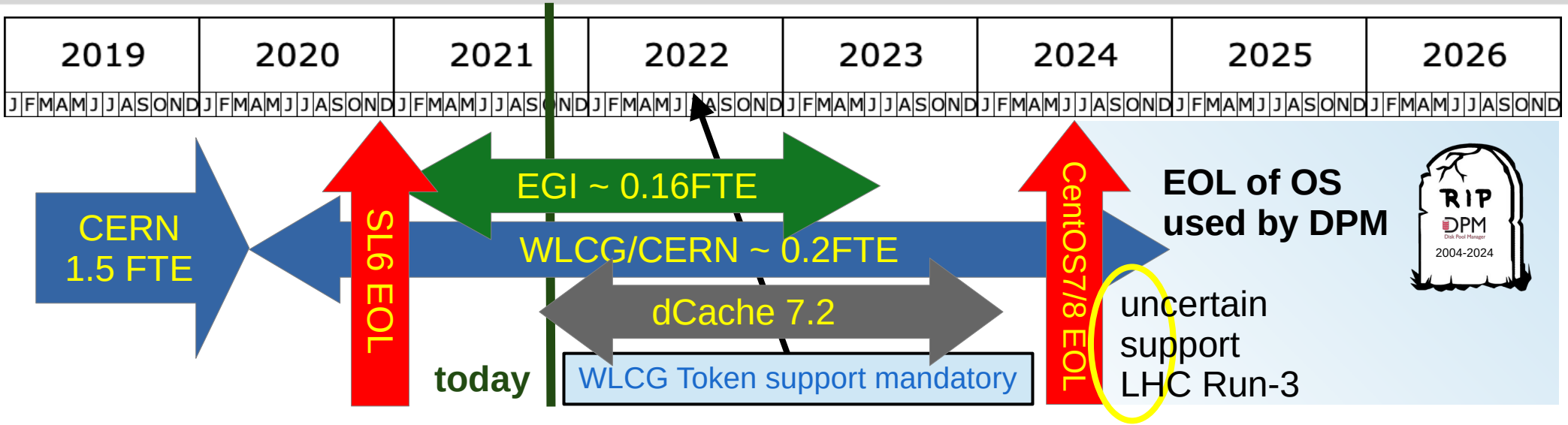
DPM status and migration tools

Petr Vokáč (CTU, EGI)

EGI Conference 2021

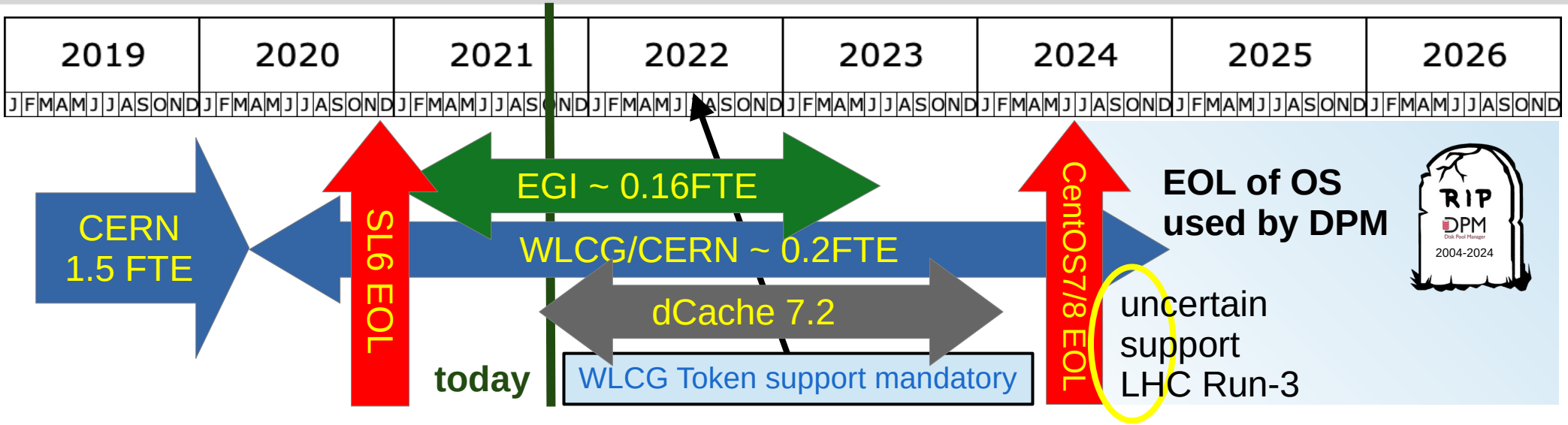
20th October 2021

DOME DPM support



- March 2020 GDB – WLCG / CERN The DPM Collaboration
  - ***provide support*** / bugfixes for DPM sites *during Run-3*
    - CentOS release changes => uncertain support at the end of Run-3
  - ***no new features*** (besides WLCG JWT), no CentOS8 release
  - consider EOS as successor / replacement for HL-LHC era
- **EGI – 5PM over next 30 months starting in 2021**
  - **support for EPEL / UMD releases,** *CentOS8 release + python3*
  - **tools for *migration to dCache* + assisting non-HEP communities**

DOME DPM support



- No plans with tokens in prod.
  - not compliant, only one VO
  - no interest to fix, test, deploy
- ATLAS HL-LHC DRAFT
  - replace DPM during Run-3
- WLCG plan to organize GDB
  - sites plans to replace DPM
- Changes in CentOS release
  - only 5 years support
  - makes DPM support beyond CentOS7 less attractive
- Migration to dCache 7.2
  - preferably by summer 2023
  - end of EGI migration support

ATLAS HL-HLC schedule

		31.3	Recommended transition plan from DPM completed	Q4 2021
		31.4	Transition plan from all DPM sites	Q4 2022
		31.5	All sites moved away from DPM	Q2 2024

dmlite release 1.15

- ***Bugfix*** release with improved transfer reliability and performance
  - 10 months since 1.14 (167 commits, 9250 insertions, 4799 deletions)
  - ***~ 3.5%*** failed HTTP-TPC caused by calling close twice LCGDM-2953
  - ***~ 0.5%*** failed HTTP-TPC IP based security & IPv4/v6 LCGDM-2961
    - fix requires dmlite 1.15.0 on all machines and **`TokenId`** set to **`none`** or **`id`** (to be tested)
  - CRL support for HTTP-TPC
  - same TLS version and cipher list configured for all protocols
  - ***dmlite-shell – all management tools integrated***, python3 support
    - storage dumps, dbck, lost&dark checks, APEL AMS, DOME API, …
  - dmlite 1.15.1 for compatibility with 1.14.x, gsiftp EL8, stuck HTTP
- EPEL / UMD dmlite 1.15.x releases (release summary, notes)
  - SLC6 EOL – no longer supported
  - ***CentOS 8*** EPEL release available
    - no legacy / SRM, no BDII

only small fraction of DPM sites use latest dmlite 1.15

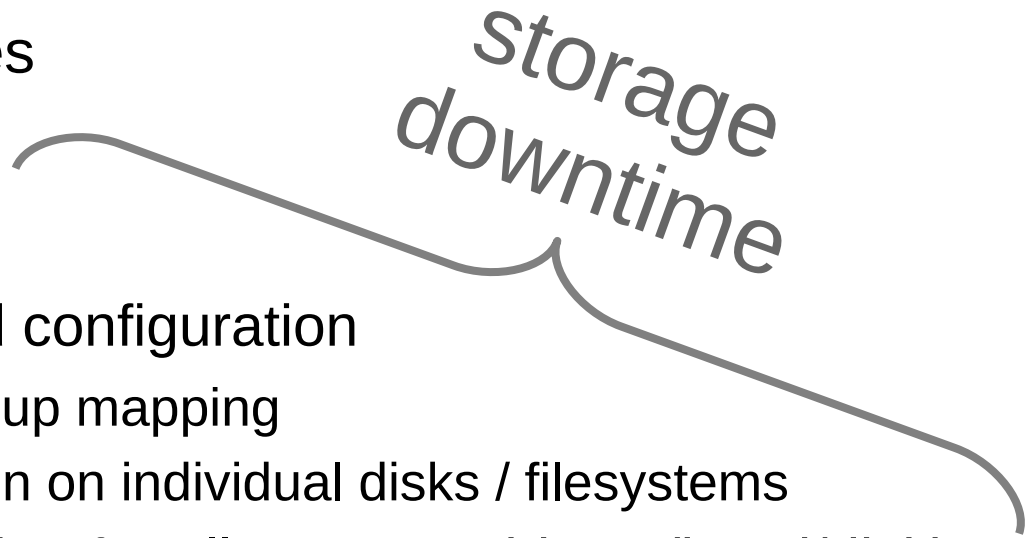
DPM to dCache migration

- In-place dCache migration is just **one of available options**
  - sites should consider their future plans first
  - provide easy migration path to the compatible storage
- Transparent migration
  - ***Migrate*** just ***catalog*** (database) and keep ***files untouched***
    - both SE store files on ***posix filesystem***
- No visible difference for clients (Rucio, FTS, gfal2, …)
  - dCache can easily match DPM features & provide more
    - sometimes with slightly more complex configuration
  - same protocols (HTTPS, xroots, gsiftp, SRM)
  - same hosts:ports / firewall configuration (almost)
  - same authentication (X.509, +tokens)
- ***dCache DPM replacement transparent for end users / VOs***
  - **Goal: 1 day downtime DPM → dCache migration**

make ***migration*** as ***simple*** as legacy to DOME DPM transition

dCache migration plan

- Migration tools will be integrated in dmlite 1.15.2 (not yet released)
  - integrated in `dmlite-shell -e 'help migrate'` (python3 only)
  - script `migrate.py` can be used also directly (no dependencies)
- Migration steps (documentation – WIP, based on migr. experience)
  - Install dCache 7.2 RPM on all DPM machines and initialize pgsql
  - DPM consistency checks / fixes
  - Stop DPM
  - Migration tools
    - Export DPM namespace and configuration
      - manually update user & group mapping
      - review spacetoken allocation on individual disks / filesystems
    - Generate dCache configuration for all DPM machines (head/disk)
    - Import DPM namespace in dCache generate hardlink scripts
  - Create hardlinks, distribute dCache configs, start dCache services, pushtag / WriteToken, ACL
  - Remove DPM



DPM storage consistency

- ***Before migration DPM*** storage must be in ***consistent state***
- Over years each DPM accumulate problems
  - not always visible in case files not accessed (e.g. invisible to Rucio)
  - software issues
    - crashed dmlite services / issues in threading
      - appeared with high production load (DOME migration, protocol migration)
    - forced service restarts
    - disk draining to did not honor DPM pool definition
    - admin mistakes
    - ...
  - Hardware issues / lost files not always correctly cleaned

validate protection for special characters

- All tools now integrated in dmlite-shell, online update
  - `dmlite-shell -e 'help dbck'`
  - `dmlite-shell -e 'dbck lost-and-dark-cleanup'` since 1.15.2
  - `dmlite-shell -e 'dbck dpm-dbck [update]'`

consistency checks and updates may take hours

Namespace migration test

- PRAGUELCG2 DPM – ***2.5PB, 23M files, 21M directories***
- ***DPM*** MySQL database ***data export ~ 2 hours***
  - normal dumps used by Rucio takes 5 minutes
  - more data necessary in full namespace dump
  - different dump data organization for simple import
  - db server have all DPM data in memory (db buffer size tuning)

site can test how much time export and import takes before declaring downtime for migration

- **data import in dCache** PostgreSQL database **~ 6 hours**
  - import rate same all the time – no slowdown with increasing db size
  - data stored on NVMe (SHM?)
  - synchronization off (`fsync = off`, `synchronous_commit = off`)
    - ***~ 50 hours*** with synchronization enabled!

import/export directly on db machine

dCache database size ¾ compared to DPM

DPM → dCache namespace migration (ex. 1h → im. 3h) = 4 hours per each 1PB

Config – user / group maps

- DPM automatically create identity based on VOMS X509 proxy
  - dpm_user → certificate subject, dpm_group → VOMS FQAN
  - some user / group can be merged
  - DPM users / groups without files not in the DPM topology export
- User mapping in DPM export
  - format: 'user',DN[,dcache_user[,dcache_uid]]
  - automatically / manually add dcache_user, dcache_uid
  - remove user line from mapping => use default user for given VO
- Group mapping in DPM export
  - format: 'user',FQAN[,dcache_group[,dcache_gid]]
  - automatically / manually add dcache_group, dcache_gid
- Generate corresponding gplazma2 and vogroup & multimap files
  - primary group from first FQAN + all other mapped FQAN groups
  - specific mapped username + uid or default VO username

~ 100 unique DNs
~ 30 unique FQAN
at prague DPM

overwrite dump with
'path',user,group,mode
configuration for all
subdirectories

Config migration – space

- Limit dCache available space per-VO / spacetoken
  - allow files to be distributed to all disknodes
  - aggregated performance
- DPM pools, available storage:/fs and spacetokens
  - space reservation / enforcing storage size limits
    - spacetokens associated with dCache LinkGroups
    - LinkGroups associated with directories using WriteToken tags
    - simple online DPM “quotatokenset” configuration
      - ~ 20 config lines / admin shell commands for dCache
      - ~ 10 lines for each individual disknode
  - can be simplified for sites supporting just one VO
    - no space reservation necessary
  - dCache 7.2 comes with “offline” quota (updated daily)
    - a bit tricky to use for our use-case (need to gain more experience)

100 filesystems
4 VOs => few thousands config lines

Summary

- There are ~ 60 DPM sites with ~ 90 DPM and ~ 100PB
- Changes in (Cent)OS releases made 06/2024+ support less certain
  - no interest in tokens also means sites may have to migrate sooner
  - support for DPM will go down in next years
- Moving data to different storage is long / ops intensive process
- Migration tools available soon for in-place update to dCache 7.2
  - works with simple testbed (headnode + 2x disknode with 2 VOs)
  - documentation & DPM release with migration tools this month
  - first production DPM site migration in progress
    - just following documentation – validate migration process
  - technically possible to migrate same way to any storage with posix fs
    - no plans to develop such tools / no volunteers
    - dCache provides sufficient features for our use-cases

BACKUP

`TokenId none` configuration

- DPM headnode authenticate and redirects transfers to disknodes
- WebDAV transfers can be redirected using plain HTTP protocol
- Hash generated by headnode can be verified by disknode
  - TokenId + PFN + TokenPassword + expiration + operation (r/w)
  - TokenId by default set to client IP address – issues
    - IP based security is not bulletproof
    - Client can use different IP while talking to headnode / disknode
      - IPv4 vs. IPv6 – happy-eyeball mechanism RFC8305
    - Disknode can't validate hash for different client IPv4 vs. IPv6 address
- Since dmlite 1.14.x redirection use HTTPS (default / puppet config)
  - Including client IP in auth. hash no longer useful and causing troubles
  - dmlite 1.15.x adds `TokenId none` configuration option
    - `/etc/dmlite.conf.d/domeadapter.conf`
    - `dpm::params::token_password: 'none'`
    - `class dpm::headnode or disknode { … token_id => 'none' … }`

Migration strategy

- Site stop providing “grid storage”
  - consolidation of storage sites
  - disk-less / cache only site (small sites)
    - Both options require really good network for data intensive sciences
- Copy all data to the new storage
  - straightforward, but require significant additional (local) space
  - slow (months) and require significant effort also from central Ops
    - e.g. migrating 30 ATLAS DPM sites would be nightmare
  - gives full flexibility choosing new storage technology
    - a lot of sites expressed interest e.g. in erasure encoding
    - HEPiX Erasure Encoding Working Group – currently a bit empty
  - operate multiple sites as one storage – NDGF style
    - all complexity handled by central team, reduced expertise at small site
- Migration without moving data transparent to the clients / VOs

HEPiX survey

EOS migration(?)

- EOS can use simple posix filesystem as storage backend
- Option 1:
  - same namespace migration possible
    - use DPM export generated by “dCache migration tools”
    - import data in EOS namespace (not implemented / planned)

with no data movement storage can’t magically start to provide EC data

- Option 2:
  - EOS can be configured with transparent redirection (e.g. for ENOENT)
  - used for CASTOR to ***EOS online migration*** in 2012
  - copy data from DPM to EOS in chunks / by individual directories
    - can be also done without too much additional space
      - gradually delete data migrated data from DPM
    - needs non-negligible effort from storage admins
    - must be done very carefully not to loose data – verify migrated files
- Other migrations also technically possible to storage implementations that supports posix filesystem as storage backend

DPM → dCache (technical details)

- Dump DPM namespace (~ hour for 2PB storage)
  - improved tools for consistency checks & updates
  - export file/replica data in simple csv
    - path, user, group, mode, {c,a,m}time + [spctk, target, size, csum, pfn]
    - no support to migrate full ACLs (DPM admin, permission incomp.)
  - export storage topology data
    - user/group mapfiles – manual updates for dCache mapping
    - filesystems, spacetokens, dpm pools
    - path – overwrite exported file/dir mode + user + group (cleanup)
- Import in dCache database
  - based on FsSqlDriver + JdbcFs code – directories, files, links
  - t_inodes, t_dirs, t_locationinfo (t_inodes_checksum, t_inodes_data)
  - export pfn → 36byte pnfsid mapping
- Import output – script for each disknode that hardlink files in dCache pool directory

gplazma.conf

```
#
# https://dcache.org/old/manuals/Book-7.2/config-gplazma.shtml
# https://dcache.org/old/manuals/2014/presentations/20140324-PM-ISGC-gplazma.pdf
#
auth     optional   x509
auth     optional   voms
auth     optional   scitoken


map      optional   multimap gplazma.multimap.file=/etc/dcache/multi-mapfile.group
map      optional   vogroup vo-group-path=/etc/dcache/vo-group.json
map      sufficient multimap gplazma.multimap.file=/etc/dcache/multi-mapfile.user
map      optional   multimap gplazma.multimap.file=/etc/dcache/multi-mapfile.vo
#map      requisite  nsswitch


account  requisite  banfile
account  requisite  argus


session  requisite  roles
session  sufficient omnisession
#session  requisite  nsswitch


#identity optional   nsswitch
```

Example multi-mapfile.group

```
# MultiMap plugin (multimap) configuration
# ========================================
# default location: gplazma.multimap.file=${dcache.paths.etc}/multi-mapfile
# documentation: https://www.dcache.org/old/manuals/Book-7.2/config-gplazma.shtml#multimap-plugin-multimap
#
# mapping sources:
#   group:vorole_* comes from vorolemap
#   group:oidc_* comes from gplazma.scitoken.issuer configuration

# DTEAM
group:vorole_dteam            gid:1000 group:writer
username:oidc_dteam           gid:1001,true group:writer
oidcgrp:/dteam                gid:1000

# WLCG
group:vorole_wlcg             gid:1100 group:writer
username:oidc_wlcg            gid:1101,true group:writer
oidcgrp:/wlcg                 gid:1100

# ATLAS identity mappings
fqan:/atlas                   gid:2000 group:writer
fqan:/atlas/Role=pilot        gid:2010
fqan:/atlas/Role=production   gid:2011
fqan:/atlas/Role=lcgadmin     gid:2012
fqan:/atlas/cz                gid:2100
fqan:/atlas/de                gid:2110

username:oidc_atlas           gid:2001,true group:writer

oidcgrp:/atlas                gid:2000
oidcgrp:/atlas/pilot          gid:2010
oidcgrp:/atlas/production     gid:2011
oidcgrp:/atlas/lcgadmin       gid:2012
oidcgrp:/atlas/cz             gid:2100
oidcgrp:/atlas/de             gid:2110
```

Example multi-mapfile.user

```
# MultiMap plugin (multimap) configuration
# ========================================
# default location: gplazma.multimap.file=${dcache.paths.etc}/multi-mapfile
# documentation: https://www.dcache.org/old/manuals/Book-7.2/config-gplazma.shtml#multimap-plugin-multimap
#
# mapping sources:
#   group:vorole_* comes from vorolemap
#   group:oidc_* comes from gplazma.scitoken.issuer configuration

# Individual local users
oidc:b0096aea-4eda-4d58-89bc-4c1880e78cb8@atlas uid:10000
oidc:58280cfd-ed7f-4954-90c7-cfde610cb963@wlcg uid:10000
"dn:/DC=ch/DC=cern/OU=Organic Units/OU=Users/CN=vokac/CN=610071/CN=Petr Vokac" username:vokac uid:10000
"dn:/DC=org/DC=terena/DC=tcs/C=CZ/O=Czech Technical University in Prague/CN=Petr Vokac 252509" username:vokac
uid:10000
```

Example multi-mapfile.vo

```
# MultiMap plugin (multimap) configuration
# ========================================
# default location: gplazma.multimap.file=${dcache.paths.etc}/multi-mapfile
# documentation: https://www.dcache.org/old/manuals/Book-7.2/config-gplazma.shtml#multimap-plugin-multimap
#
# mapping sources:
#   group:vorole_* comes from vorolemap
#   group:oidc_* comes from gplazma.scitoken.issuer configuration

# DTEAM
group:vorole_dteam            uid:1000 username:dteam
username:oidc_dteam           uid:1001


# WLCG
group:vorole_wlcg             uid:1100 username:wlcg
username:oidc_wlcg            uid:1101


# ATLAS identity mappings
fqan:/atlas                   uid:2000 username:atlas
username:oidc_atlas           uid:2001
```

Example vo-group.json

```
[
  {
    "fqan": "/dteam",
    "mapped_gid": 1000,
    "mapped_uname": ""
  },
  {
    "fqan": "/wlcg",
    "mapped_gid": 1100,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas",
    "mapped_gid": 2000,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas/Role=pilot",
    "mapped_gid": 2010,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas/Role=production",
    "mapped_gid": 2011,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas/Role=lcgadmin",
    "mapped_gid": 2012,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas/cz",
    "mapped_gid": 2100,
    "mapped_uname": ""
  },
  {
    "fqan": "/atlas/de",
    "mapped_gid": 2110,
    "mapped_uname": ""
  }
]
```

Example psu

```
[storage1_example_org_pool_projectA_Domain]
[storage1_example_org_pool_projectA_Domain/pool]
pool.name=storage1_example_org_pool_projectA
pool.tags=projectA
pool.path=/data/${pool.tags}
pool.size=5G
pool.wait-for-files=${pool.path}/data

[storage1_example_org_pool_projectB_Domain]
[storage1_example_org_pool_projectB_Domain/pool]
pool.name=storage1_example_org_pool_projectB
pool.tags=projectB
pool.path=/data/${pool.tags}
pool.size=15G
pool.wait-for-files=${pool.path}/data

[storage2_example_org_pool_projectA_Domain]
[storage2_example_org_pool_projectA_Domain/pool]
pool.name=storage2_example_org_pool_projectA
pool.tags=projectA
pool.path=/data/${pool.tags}
pool.size=5G
pool.wait-for-files=${pool.path}/data

[storage2_example_org_pool_projectB_Domain]
[storage2_example_org_pool_projectB_Domain/pool]
pool.name=storage2_example_org_pool_projectB
pool.tags=projectB
pool.path=/data/${pool.tags}
pool.size=15G
pool.wait-for-files=${pool.path}/data
```

Example link config

- Link pgroups

```
(local) admin > \c PoolManager
(PoolManager) admin > psu create pgroup -dynamic -tags=projectA spacemanager_poolGroup_ProjectA
(PoolManager) admin > psu create link spacemanager_WriteLink_ProjectA any-store world-net any-protocol
(PoolManager) admin > psu set link spacemanager_WriteLink_ProjectA -readpref=10 -writepref=10 -cachepref=0 -
p2ppref=-1
(PoolManager) admin > psu addto link spacemanager_WriteLink_ProjectA spacemanager_poolGroup_ProjectA
(PoolManager) admin > psu create linkGroup spacemanager_WriteLinkGroup_ProjectA
(PoolManager) admin > psu set linkGroup replicaAllowed spacemanager_WriteLinkGroup_ProjectA true
(PoolManager) admin > psu set linkGroup onlineAllowed spacemanager_WriteLinkGroup_ProjectA true
(PoolManager) admin > psu addto linkGroup spacemanager_WriteLinkGroup_ProjectA spacemanager_WriteLink_ProjectA
(PoolManager) admin > # ... same for projectB ...
(PoolManager) admin > save
```

- Configuration LinkGroupAuthorization.conf

```
LinkGroup spacemanager_WriteLinkGroup_ProjectA
/projectA/Role=*

# ... same for projectB ...
```