dCache - HowTo

- Deploying system test dCache
- Admin Interface, scripting
- Pool decommissioning
- Setup replication
- Cleanup after lost pool
- Migrate from one tape system to another
- Using ACLs
- Debug FTS transfer

Admin Interface Scripting

- **Scripting vi ssh (just pipe commands into the interface)**
- **Scripting via Jython in combination with pcells**
- **Scripting via Jython**

Pool Decommissioning

Set the pool read-only

```
[vm-dcache-001] (chrisPool@dCacheDomain) admin > pool disable -rdonly
```

Move data off the pool (migration move -target=pool | pgroup | link)

```
[vm-dcache-001] (chrisPool@dCacheDomain) admin > migration move -target=pool pool_r3

or

[vm-dcache-001] (chrisPool@resilientChrisPoolsDomain) admin > migration move -target=pgroup
ResilientPools
```

Make sure no file is left on the pool

```
[vm-dcache-001] (chrisPool@dCacheDomain) admin > rep ls
```

Stop the pool

```
[root@vm-dcache-001 resilient]# dcache stop resilientChrisPoolsDomain
```

Remove pool/domain from layoutfile

Remove pool from PoolGroup

```
[vm-dcache-001] (PoolManager@dCacheDomain) admin > psu removefrom pgroup ResilientPools chrisPool
```

Remove pool from PoolManager

```
[vm-dcache-001] (PoolManager@dCacheDomain) admin > psu remove pool chrisPool
```

Remove directory structure from actual backend storage

Setup Replication

Setup Replication

- Create Domain and replica cell

  ```
  [replicaDomain]
  [replicaDomain/replica]
      replica.poolgroup = ResilientPools
      replica.enable.same-host-replica=true
      replica.limits.replicas.min=2
      replica.limits.replicas.max=3
  ```

- Enable replication in dcache.conf

  ```
  dcache.enable.replica = true
  ```

- Put pools into the replica pool group
- Just recently there was a command added to refetch the resilient pools when adding new pools to the PoolGroup: update poolgroup

Lost Pool

Three cases:

1.) copy on tape
2.) resilient pools
3.) non-resilient disk only pools

how to solve 3:

* Check which files were on lost pool (list of pnfsIDs)

* Check which of the lost files might be on tape --> t_locationinfo itype=0 (subtract these from the above pnfsIDs)

* Which might be cached on some other pool (hot spot reps) and are cached or as primary copy on some other pool(if primary, subtract them from the list, if cached put them precious there or migration move them to a pool that is save and put them precious there. After that remove them from the list)

* Remove the resulting list from the t_locationinfo with the ilocation of the lost pool_name

* Also clean up t_locationinfo_trash

* The resulting list is the list of lost files. (Ask chimera for the paths and appologise to your users that you lost the files, send them the list and let them delete them from the namespace, catalog)

Migrate Tape systems

Preparation:

- Each file in a pool has one of the 4 primary states: “cached” (<C---), “precious” (<-P--), “from client” (<--C-), and “from store” (<---S).

Procedure:

Remove old hsm instance entry from pools that are connected to the old tape system

```
  [vm-dcache-001] (pool_write@writePoolDomain) admin > hsm remove osm
  [vm-dcache-001] (pool_write@writePoolDomain) admin > hsm create osm osmNew -command=/usr/share/dcache/
    lib/hsmcp.rb -hsmBase=/hsmTape_new/data -hsmInstance=osmNew
```

Configure one read pool and one write pool

```
[root@vm-dcache-001 ~]# dcache pool create --size=419430400  --meta=db --lfs=precious /var/pools/
tapeMigrationWritePool tapeMigrationWritePool tapeMigrationPoolDomain
[root@vm-dcache-001 ~]# dcache pool create --size=419430400  --meta=db --lfs=precious /var/pools/
tapeMigrationReadPool tapeMigrationReadPool tapeMigrationPoolDomain
```

Create read only link and write only links, connect them to the pools

```
psu create link tape-write-link tape-store world-net any-protocol
psu set link tape-write-link -readpref=0 -writepref=10 -cachepref=0 -p2ppref=0
psu create link tape-read-link tape-store world-net any-protocol
psu set link tape-read-link -readpref=10 -writepref=0 -cachepref=10 -p2ppref=0
psu add link tape-write-link tapeMigrationWritePool
psu add link tape-read-link tapeMigrationReadPool
```

Create hsm entries on the pools, the old instance connected to the read pool and the new instance connected to the write pool

```
[vm-dcache-001] (tapeMigrationReadPool@tapeMigrationPoolDomain) admin > hsm create osm osm -
hsmInstance=osm -command=/usr/share/dcache/lib/hsmcp.rb -hsmBase=/hsmTape/data
[vm-dcache-001] (tapeMigrationWritePool@tapeMigrationPoolDomain) admin > hsm create osm osmNew -
hsmInstance=osmNew -hsmBase=/hsmTape_new/data -command=/usr/share/dcache/lib/hsmcp.rb -c:gets=1 -
c:puts=1 -c:removes=1
```

Check of there are files on the tapeMigrationReadPool

```
[vm-dcache-001] (tapeMigrationReadPool@tapeMigrationPoolDomain) admin > rep ls
```

Migrate Tape systems

Get a list of files per tape from the tape administration

Add the command to the pnfsID list:

```
[root@vm-dcache-001 ~]# vi pnfsIdList, :%s/^/\\s tapeMigrationReadPool rh restore /g
```

Send the commands to the dCache admin insterface:

```
[root@vm-dcache-001 ~]# ssh -l admin -p 22224 localhost < allPnfsIDs_oldHSM_rhRestore
```

Migration move the files from the read pool to the write pool

```
[vm-dcache-001] (tapeMigrationReadPool@tapeMigrationPoolDomain) admin > migration move -concurrency=8 -
smode=delete -tmode=precious -target=pool tapeMigrationWritePool
```

Flush files from write pool to the new tape system (flush pnfsid)

```
[root@vm-dcache-001 ~]# ssh -l admin -p 22224 localhost < allPnfsIDs_oldHSM_flush
```

Check if all files have an entry on the new tape system and in the t_locationinfo table

Create tmp table in database with all pnfsIDs

```
chimera=# create TEMP table tmp_ids (id character varying(36));
chimera=# copy tmp_ids from '/tmp/allPnfsIDs_oldHSM'; (this files can not be in /root, put it to
where the postgres user can read it)
```

Check against t_locationinfo table

```
 chimera=# select ipnfsid,ilocation from t_locationinfo where ipnfsid in (Select id from tmp_ids)
AND ilocation like 'hsm://osm/%';
```

Check number of entries above against the row count in the CSV file

```
wc -l /tmp/allPnfsIDs_oldHSM
```

Delete entries from t_locationinfo, with the old tape (allthough this step can be ommitted as it does not hurt)

```
    chimera=# delete from t_locationinfo where ipnfsid in (Select id from tmp_ids) AND ilocation
```

Using ACLs

Enable ACLs in layout file:

```
[dCacheDomain/pnfsmanager]
 pnfsmanager.enable.acl = true
```

Export file should contain:

```
/ localhost(rw,root_squash,acl)
```

Mount nfs with vers=4.1

Client side altering ACLs:

```
nfs4_setfacl -e <directory path | file>
```

```
Client side getting ACLs:
    nfs4_getfacl <directory path | file>
```

```
Admin Interface:
```

- `getfacl`
- `setfacl`

How to Debug FTS, SRM transfers

- Have srm.persistence.enable.history=true —> This allows you better debugging

- Example:
  - Billing logs: this shows is the pnfsID and the TURL

```
02.20 18:40:42 [pool:dcache-atlas72-03@dcache-atlas72-03Domain:remove]
[00003EBBDE28AA514ABB890769FD718A4D27,6398872127] [Unknown]
atlas:atlasproddisk@osm {0:""}

02.20 18:40:44 [pool:dcache-atlas72-03:transfer]
[00003EBBDE28AA514ABB890769FD718A4D27,6398872127] [/upload/b4285977-33c8-4904-a7fc-
c4104dd4bac6/AOD.01534542._000072.pool.root.1] atlas:atlasproddisk@osm 6398872127 6171563 true
{GFtp-2.0 145.100.32.111 60089} [door:GFTP-dcache-door-atlas14-499890@gridftp-dcache-door-
atlas14Domain:1424447822672-1424447823185] {10001:"No such file or directory:
00003EBBDE28AA514ABB890769FD718A4D27"}
```

How to Debug FTS, SRM transfers

- Get the SRM transfer ID from the database: —> the unique part of the TURL is needed for the query
  - dcache=# select * from putfilerequests where turl ilike '%b4285977-33c8-4904-a7fc-c4104dd4bac6%';

```
id      | nextjobid | creationtime  | lifetime | state |                          errormessage                          | schedulerid |
schedulertimestamp | numofretr | maxnumofretr | laststatetransitiontime |  requestid  | credentialid | status
code |                                                        surl                                                    |
turl
           |                              fileid                              | parentfileid | spacereservationid |    size    | retentionpolicy |
accesslatency
-------------+-----------+---------------+----------+-------+-------------------------------------------------------------------+-------------
+--------------------+-----------+--------------+------------------------+-------------+-------------+-------
-----+--------------------------------------------------------------------------------------------------------------------------
+------------------------------------------------------------------------------------
---------------+-----------------------------------------------------------------+-------------+------------------+-----------+----------------
+---------------
 -1193254804 |           | 1424447822548 |  3600000 |     11 |  at Fri Feb 20 17:57:12 CET 2015 state Failed : Request lifetime expired.
|           |      1424430720535 |         0 |           10 |          1424451432775 | -1193254805 |  -2034723623 |
    | srm://dcache-se-atlas.desy.de:8443/srm/managerv2?SFN=/pnfs/desy.de/atlas/dq2/atlasproddisk/rucio/data12_8TeV/78/9b/
AOD.01534542._000072.pool.root.1 | gsiftp://dcache-door-atlas14.desy.de:2811//upload/b4285977-33c8-4904-a7fc-c4104dd4bac6/
AOD.01534542._000
072.pool.root.1 | /upload/b4285977-33c8-4904-a7fc-c4104dd4bac6/AOD.01534542._000072.pool.root.1 |             | 310019
| 6398872127 |                |
(1 row)
```

How to Debug FTS, SRM transfers

```
[dcache-core-atlas03.desy.de] (SRM-dcache-se-atlas03) admin > ls -l "-1193254804"
Put file id:-1193254804 state:Failed
   SURL: srm://dcache-se-atlas.desy.de:8443/srm/managerv2?SFN=/pnfs/desy.de/atlas/dq2/atlasproddisk/rucio/data12_8TeV/78/9b/AOD.
01534542._000072.pool.root.1
   TURL: gsiftp://dcache-door-atlas14.desy.de:2811//upload/b4285977-33c8-4904-a7fc-c4104dd4bac6/AOD.01534542._000072.pool.root.1
   Size: 6398872127
   Access latency: null
   Retention policy: null
   Space reservation: 310019
   History:
      2015-02-20 16:57:02.548 Pending: Request created (0 ms)
      2015-02-20 16:57:02.548 TQueued: Request enqueued. (1 ms)
      2015-02-20 16:57:02.549 PriorityTQueued: Waiting for thread. (0 ms)
      2015-02-20 16:57:02.549 Running: Processing request (0 ms)
      2015-02-20 16:57:02.549 Running: run method is executed (0 ms)
      2015-02-20 16:57:02.549 AsyncWait: Doing name space lookup. (35 ms)
      2015-02-20 16:57:02.584 PriorityTQueued: Waiting for thread. (0 ms)
      2015-02-20 16:57:02.584 Running: Processing request (0 ms)
      2015-02-20 16:57:02.584 Running: run method is executed (0 ms)
      2015-02-20 16:57:02.584 RQueued: Putting on a "Ready" Queue. (0 ms)
      2015-02-20 16:57:02.584 Ready: Execution succeeded. (60 min)
      2015-02-20 17:57:12.775 Failed: Request lifetime expired.
```