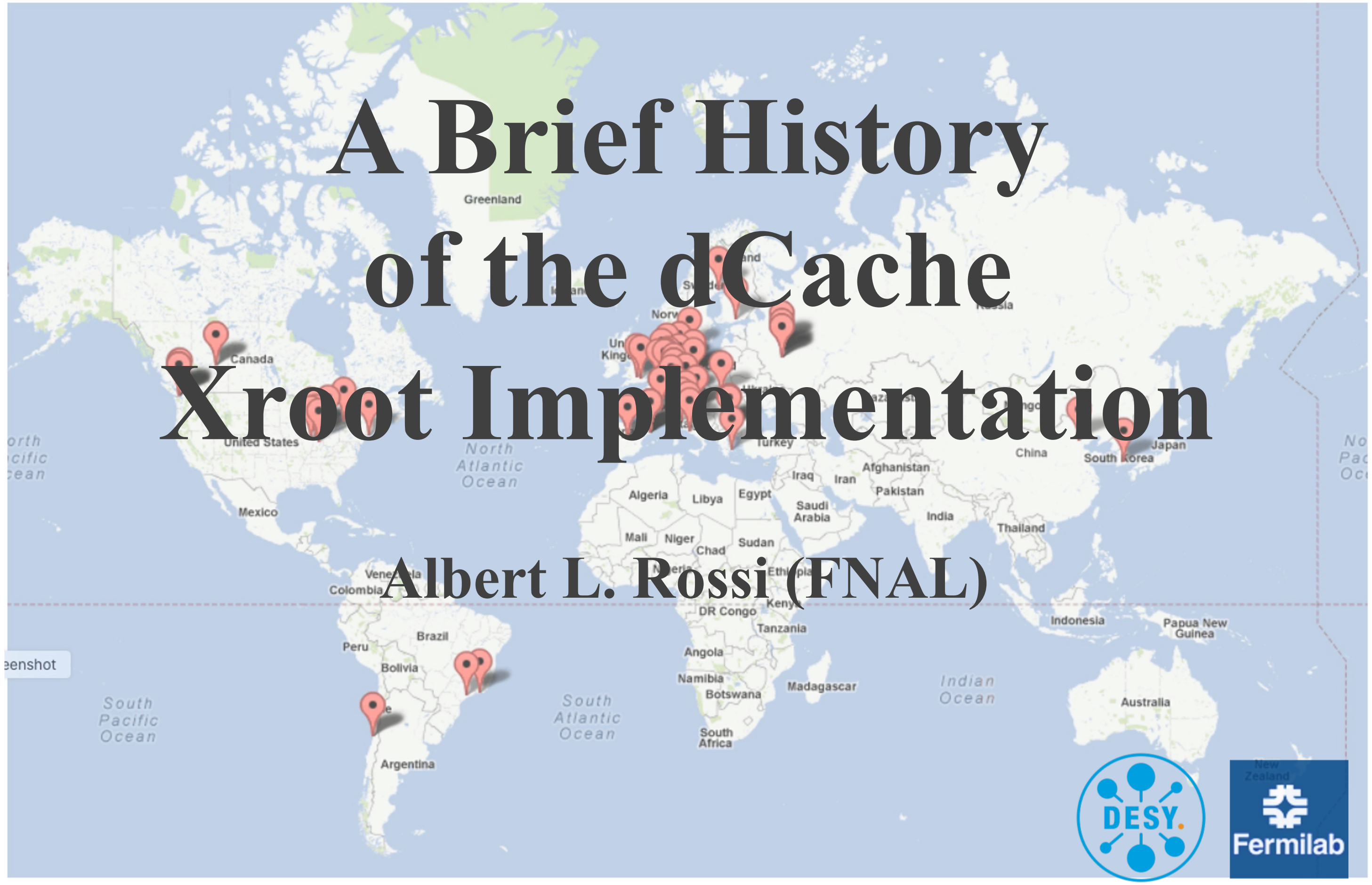


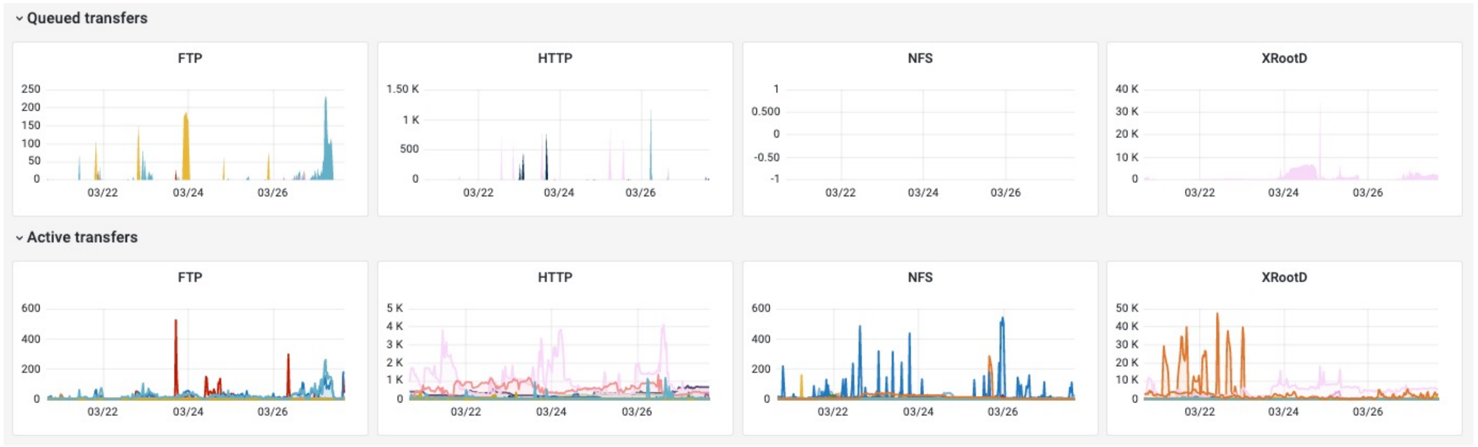
A Brief History of the dCache Xroot Implementation

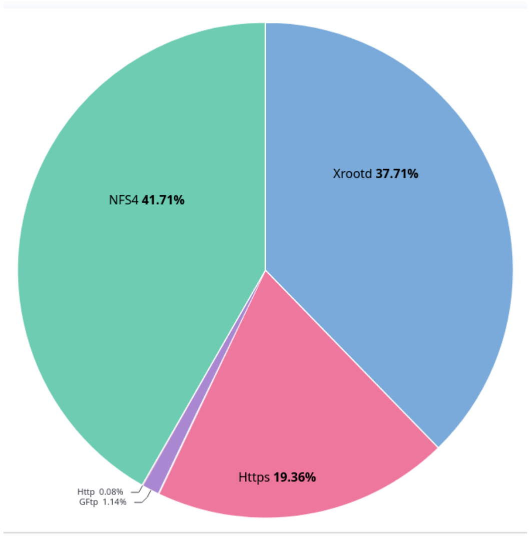
Albert L. Rossi (FNAL)

RESEARCH FOR GRAND CHALLENGES

dCache Xroot protocol usage

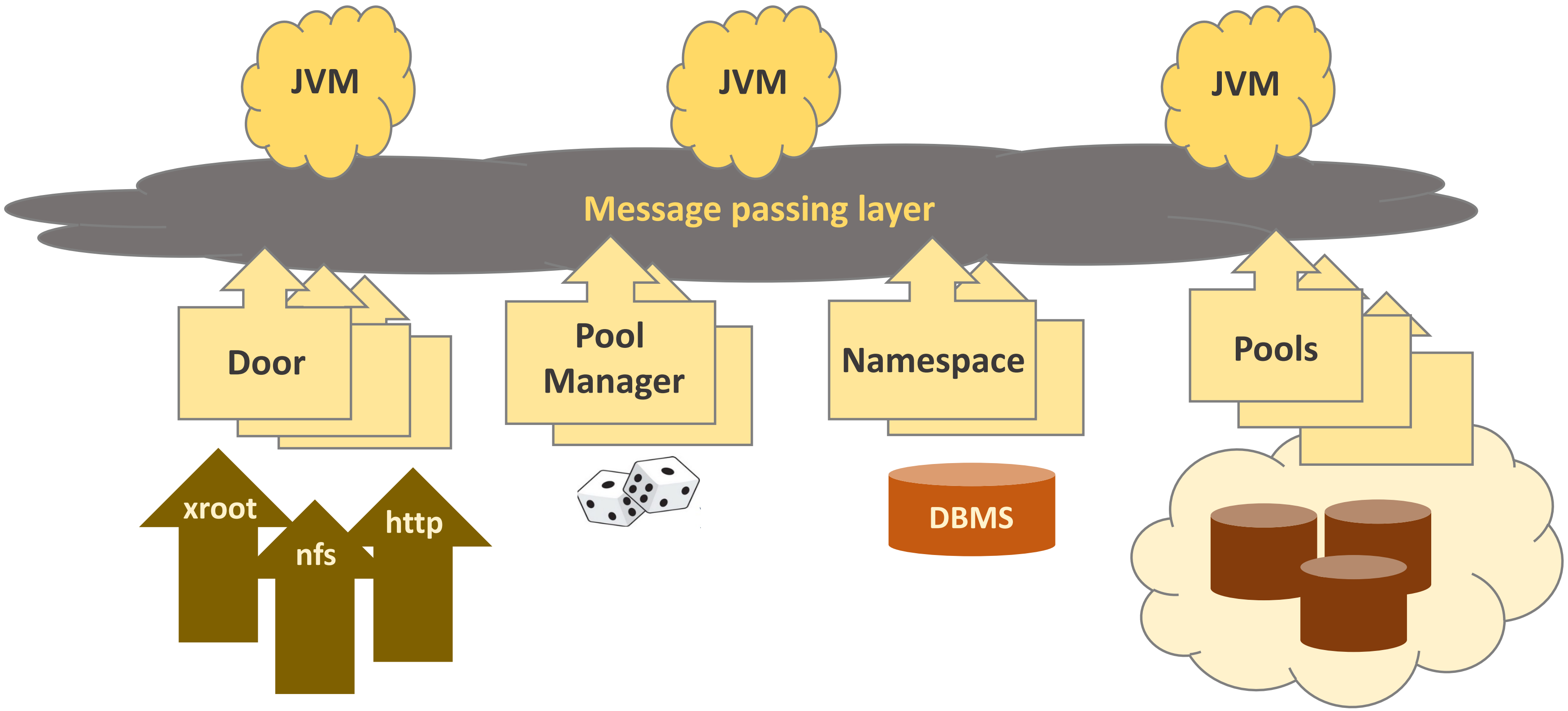
FNAL Public dCache all experiments, 7 days ending 28/03/2023





DESY all experiments, 7 days ending 28/03/2023

dCache on one slide



dCache Xroot as XRootD server

Door = server with manager and redirector roles

- contacts Pool Manager and receives a pool endpoint for either a read or a write; redirects the client to the pool
- requires authentication/authorization (done via communication with a special dCache security service, ***gPlazma***)

Pool = data server

- uses a unique id (opaque) generated at the door to identify authorized connections
- rejects a connection if the id is invalid or expired

dCache Xroot library dependencies

Both the door and pool service are written on top of a **Netty** layer:

https://netty.io/

- an asynchronous event-driven network application framework for rapid development of maintainable high performance protocol servers & clients; uses Java NIO (Non-Blocking IO) (https://en.wikipedia.org/wiki/Non-blocking_I/O_(Java))

The door and pool service, along with CTA integration, use the *xrootd4j* library:

https://github.com/dCache/xrootd4j

- maintained by dCache (viz., me for the moment); constitutes all of the Xroot protocol implementation in Java which is not specific to dCache (so it has no dCache dependencies)

dCache Xroot development 2018-2023

Interoperability

1. Added Third-Party Copy (TPC)
2. Added GSI TPC proxy management
3. Expanded Signed Hash Verification
4. Implemented `unix` authentication
5. Regularized error codes/messages
6. Added checksum CGI handling to door
7. Added support for `tried/rc` CGI
8. Added TLS support to door and pool
9. Added token authorization support
10. Added token authentication support (ZTN)
11. Added query support for TPC on pools
12. Added support for `kXR_fattr`, `kXR_prefname`

dCache Enhancements

1. Allowed client to reattempt open on pool when I/O stalls
2. Fixed `-P` handling: moved upload commit for persist-on-successful-close to the pool
3. Added `authn` protocol chaining and multiple (security) protocol door
4. Changed chunk size to conform with standard (8 MiB) for both server and TPC client
5. Broke up write into max frame-size chunks
6. Added ability to proxy transfers through door
7. Added support for relative paths in URL

Third-Party Copy

Implementation challenges

- no dCache Xroot stand-alone client
- no corresponding objects for request/response types on client-side
- had to adjust code to recognize via CGI elements the origin of the connection at the door
- had to adjust how file size is determined
- had to provide for authentication (next slide)

Changes to GSI

Implementation challenges

- GSI package required significant refactoring
- Initially, needed to support both local proxies (generated from host certs or distributed as robocerts) and subsequently proxy delegation
- Had to rework the implementation of the Diffie-Hellman handshake (as agreed upon)

Signed Hash Verification (`sigver`)

Implementation challenges

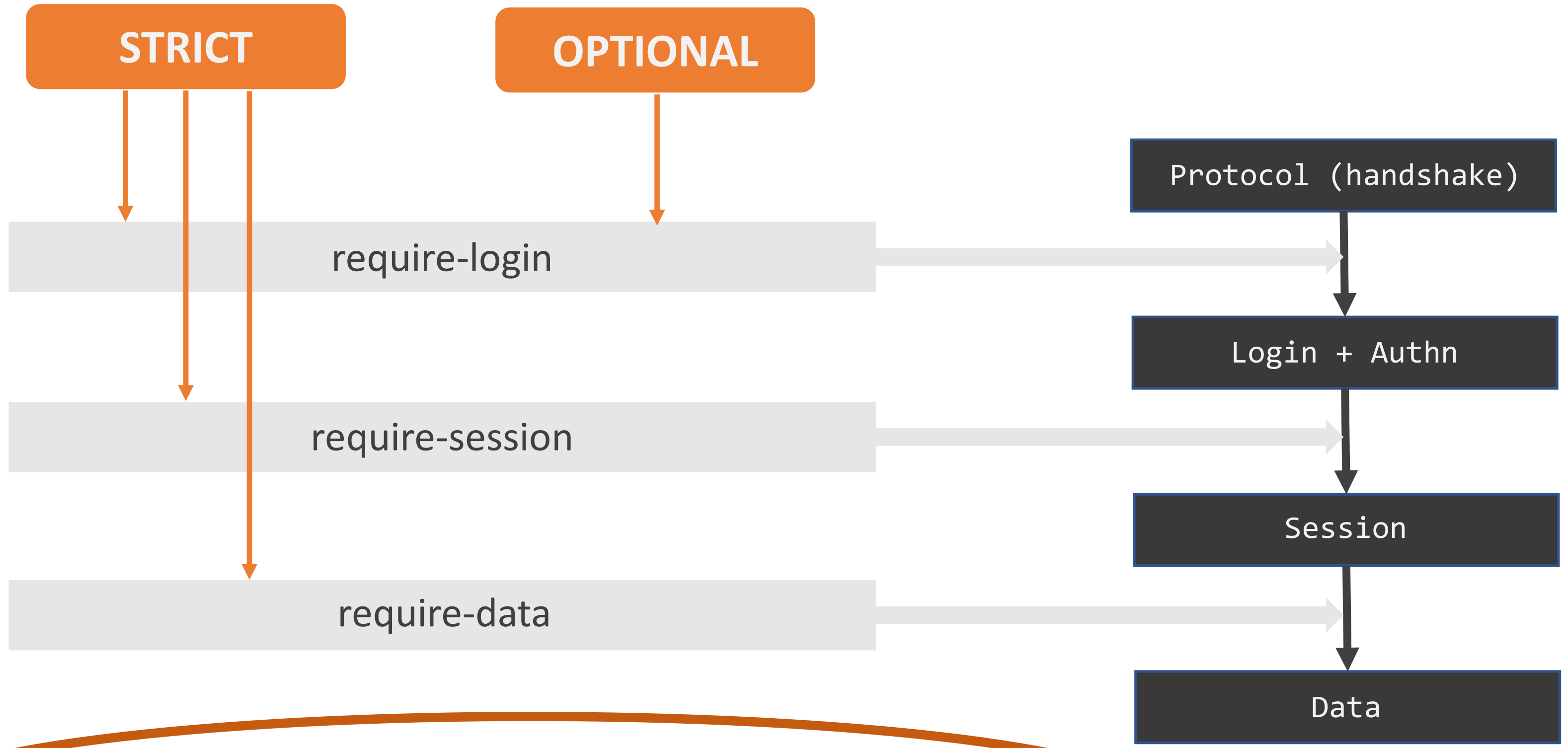
- Was not fully supported by server and there was (obviously) no client-side implementation
- An additional problem was getting the XRootD client to turn on `sigver` to the pool
- Required us to implement the Xroot "Unix" security protocol

TLS (protocol version 5)

Implementation challenges

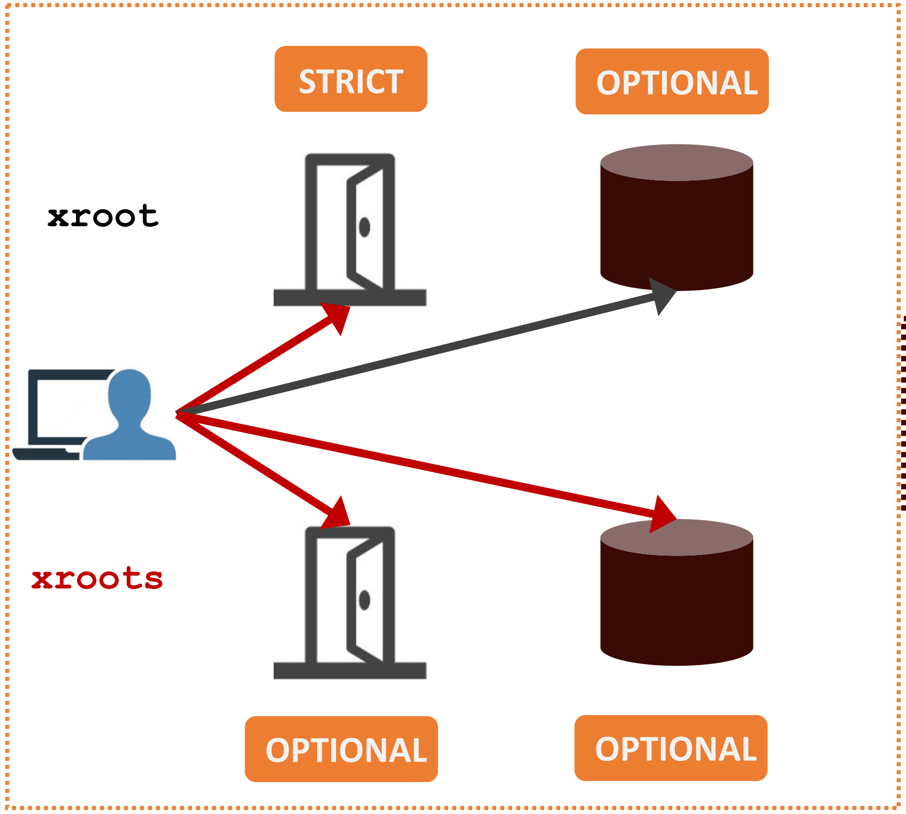
- Several iterations were necessary to adapt to an evolving specification
- Also implemented support in the TPC client for the `expect` flag
- Switched from native Java to Netty implementation of SSL
- Changed returned URL to use hostname instead of IP on redirect to pool (TLS requirement)

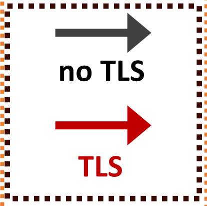
dCache TLS settings (OPTIONAL, STRICT)

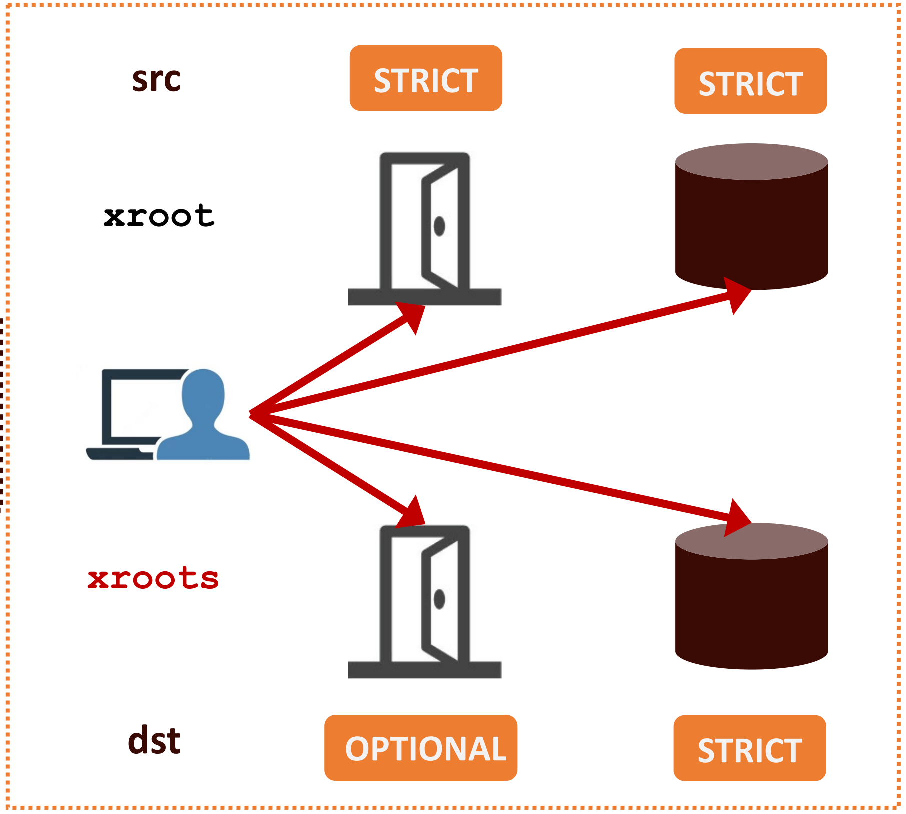


RECENT CHANGE: => OFF if client does not support/request TLS
ZTN is not included as possible protocol

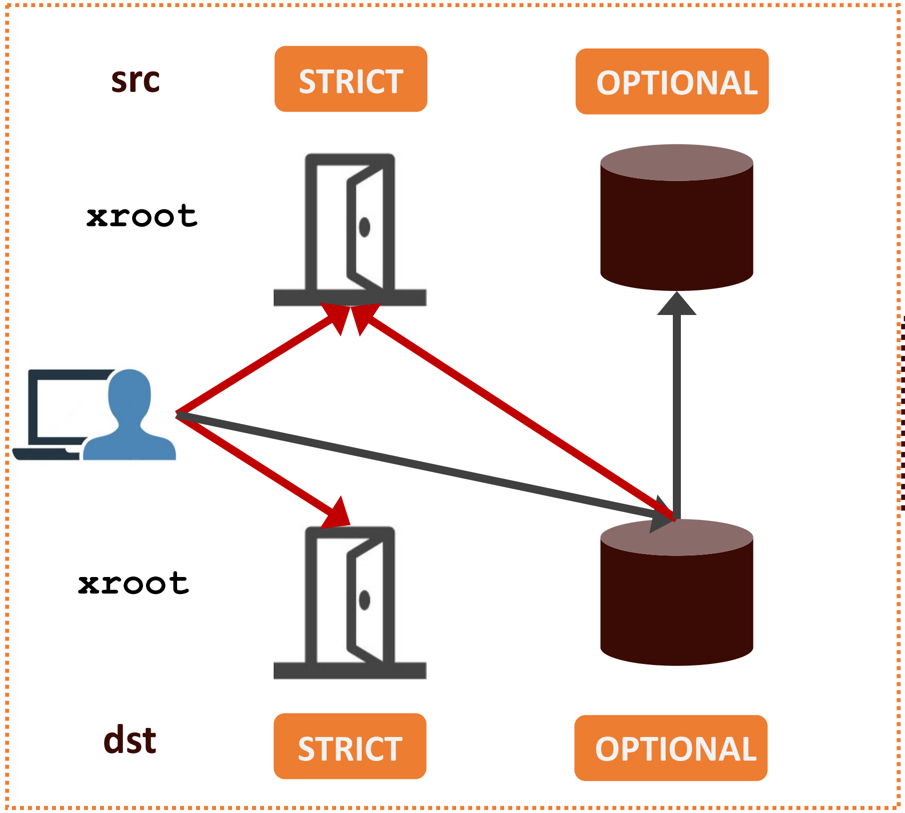
dCache TLS: data encryption

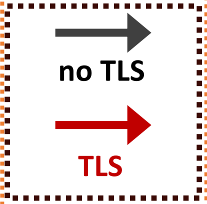


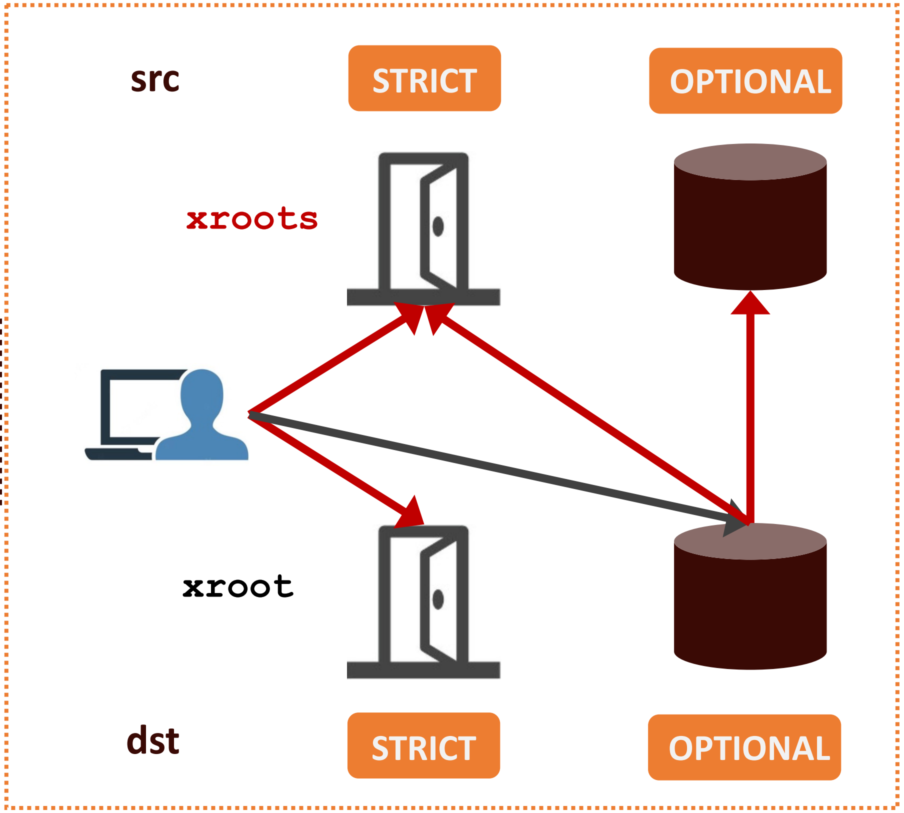




dCache TLS with TPC: (x)root vs (x)roots (src)







no data encryption

data encryption

Tokens

Implementation challenges

- SciToken/JWT/OIDC modules already existed in *gPlazma*
- More refactoring of GSI package necessary
- Question of what to do in the case of TPC
- Example of ALICE plugin
- Discussion finally converged on `--tpc only` with rendezvous token

ZTN token verification

Implementation challenges

- Not strictly required in dCache
- Originally defaulted to this token if bearer token for authorization was missing from the URL path query
- Removed because SLAC was not (initially) doing this
- Further discussion by the WLCG group, however, led to a decision to follow that model, so it was restored

Previously missing support

- Added checksum info to support CGI element and checksum CGI handling to door query
- Added missing support for TPC query on pools
- Added tried/rc
- Added support for kXR_prefname in the kXR_locate request (required by TLS)
- Added kXR_fattr request and response to library (outside contributor)
- Regularized the error codes and messages returned

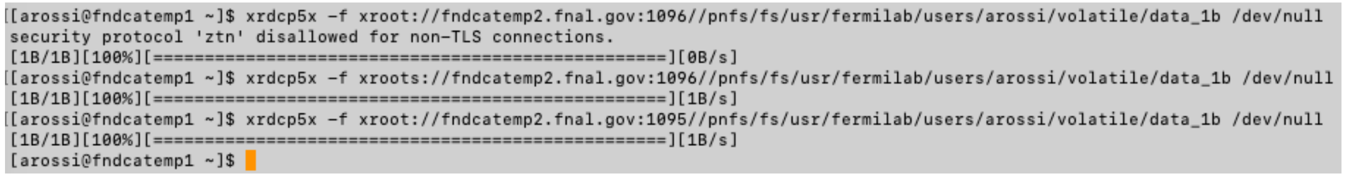
dCache multiprotocol door (issues)

1) Suppressing the TLS warning

1096: `ztn,gsi,unix` TLS is optional

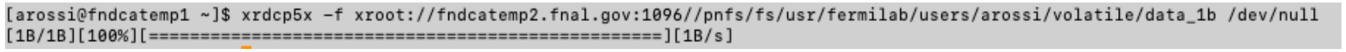
1095: `gsi,ztn,unix` TLS is optional

using voms proxy



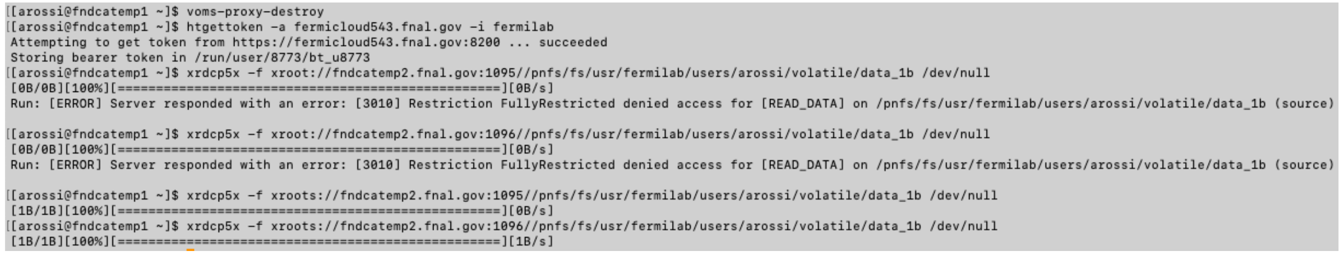
```
[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroot://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 security protocol 'ztn' disallowed for non-TLS connections.
 [1B/1B][100%][==================================================][0B/s]
[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroots://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [1B/1B][100%][==================================================][1B/s]
[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroot://fndcatemp2.fnal.gov:1095//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [1B/1B][100%][==================================================][1B/s]
 [arossi@fndcatemp1 ~]$
```

NOW: checks whether client has activated or is capable of TLS (see slide 11) and omits `ztn` from the list:



```
[arossi@fndcatemp1 ~]$ xrdcp5x -f xroot://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
[1B/1B][100%][==================================================][1B/s]
```

Of course, without `xroots`, token authentication still fails on the TLS optional doors:



```
[[arossi@fndcatemp1 ~]$ voms-proxy-destroy
[[arossi@fndcatemp1 ~]$ htgettoken -a fermicloud543.fnal.gov -i fermilab
 Attempting to get token from https://fermicloud543.fnal.gov:8200 ... succeeded
 Storing bearer token in /run/user/8773/bt_u8773
[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroot://fndcatemp2.fnal.gov:1095//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [0B/0B][100%][==================================================][0B/s]
 Run: [ERROR] Server responded with an error: [3010] Restriction FullyRestricted denied access for [READ_DATA] on /pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b (source)

[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroot://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [0B/0B][100%][==================================================][0B/s]
 Run: [ERROR] Server responded with an error: [3010] Restriction FullyRestricted denied access for [READ_DATA] on /pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b (source)

[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroots://fndcatemp2.fnal.gov:1095//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [1B/1B][100%][==================================================][0B/s]
[[arossi@fndcatemp1 ~]$ xrdcp5x -f xroots://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
 [1B/1B][100%][==================================================][1B/s]
```

dCache multiprotocol door (issues)

2) Client holds multiple credentials (both x509 and token)

- Depending on the order of the protocols, client will always log in with either x509 (for 1095, as above) or token (for 1096, as above)
- Can be forced by having the client set the env variable `XrdSecPROTOCOL` to the comma-separated list of protocols the client actually wants to use

Proxying transfers through door

Use case: all external connections blocked from internal network on which the pools reside

1. **Door proxy requests a pool source accessible from its address, not the client's**

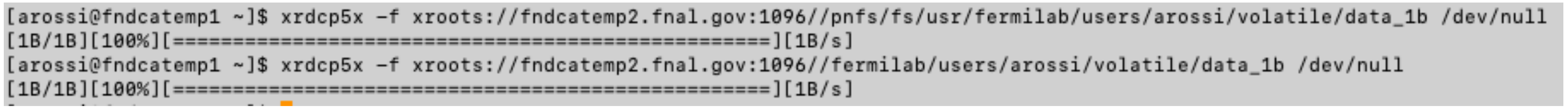
2. **Door proxy must return address to client on basis of its own reachability (i.e., IPv4 vs IPv6)**

3. **Internal address for the proxy to use is configurable (like our https implementation)**

Use of relative paths

dCache establishes a user root or home

- This depends on login authentication used (mapping files, token configurations)
- Other protocols express the target in terms of what the user thinks is its root, but dCache Xroot has always required the full path from the namespace root
- No longer the case (here, `/pnfs/fs/usr` is the root for `arossi`)



```
[arossi@fndcatemp1 ~]$ xrdcp5x -f xroots://fndcatemp2.fnal.gov:1096//pnfs/fs/usr/fermilab/users/arossi/volatile/data_1b /dev/null
[1B/1B][100%][==================================================][1B/s]
[arossi@fndcatemp1 ~]$ xrdcp5x -f xroots://fndcatemp2.fnal.gov:1096//fermilab/users/arossi/volatile/data_1b /dev/null
[1B/1B][100%][==================================================][1B/s]
```

Currently not supported by dCache

CLIENT REQUEST TYPES

`kXR_gpfile`
`kXR_prepare`
`kXR_bind`
`kXR_pgwrite`
`kXR_truncate`
`kXR_pgread`
`kXR_writev`

QUERY REQUEST TYPES

`kXR_QStats`
`kXR_QPrep`
`kXR_Qspace`
`kXR_Qckscan`
`kXR_Qvisa`
`kXR_Qopaque`
`kXR_Qopaquf`
`kXR_Qopaqug`

OPEN REQUEST OPTIONS

most of them are ignored and fall over to default behavior

Upcoming Event

(© HTW Berlin/Alexander Rentsch)

Workshop details

17th International dCache Workshop

2023-02-04 workshop

We are happy to announce the 17th International dCache Workshop in Berlin, Germany from 2023-05-31 to 2023-06-01. The workshop will take place in person and hosted by HTW Berlin - University of Applied Sciences. https://indico.desy.de/e/dcache-ws17 As usual we will present forthcoming developments and technologies, both provided by and influencing dCache. However, we are always happy to include presentations from you, especially if you have a special configuration, user community or use case, or experience with new hardware.

Thank you for your attention.

arossi@fnal.gov

support@dcache.org

https://www.dcache.org

https://github.com/dCache