Enabling Object Storage via shims for Grid Middleware

Sam SKIPSEY, Alastair Dewhurst, Gareth Douglas Roy, David Britton,
Wahid Bhimji, David Crooks,
Shaun de-Witt

Rationale

- Existing Axiom: Grid Storage needs to be POSIXish.
- Assertion: this is only true at the highest levels (File catalogues). Specific files at site reached via Experiment specific catalogues/mappings.
- Can we simplify T2(T1?) storage by just exporting direct object store interfaces?
- (Additionally, Rucio, for example, seems optimised for non-POSIX systems, given its tendency to leave trails of empty directory entries behind it.)

Ceph

- Popular parallel distributed object store based on CRUSH algorithm.
- Provides various layers over the object store
  - CephFS for POSIX-like fs etc
- Most Grid interest has been in using CephFS under DPM/ dCache/StoRM etc.
  - Adds Yet Another Indirection Layer
  - Re-invents the wheel.

Rados, Rados-striper

- Why not just use rados (low level object API)?
  - rados layer is dumb (does not stripe large objects into chunks)
- Sebastien Ponce - rados-striper (object chunks)
  - in Giant release.
  - Used in Sebastien's Xrootd Ceph interface.
- (Open question: how does this differ from an ec pool with "zero parity"?)

RAL - ATLAS

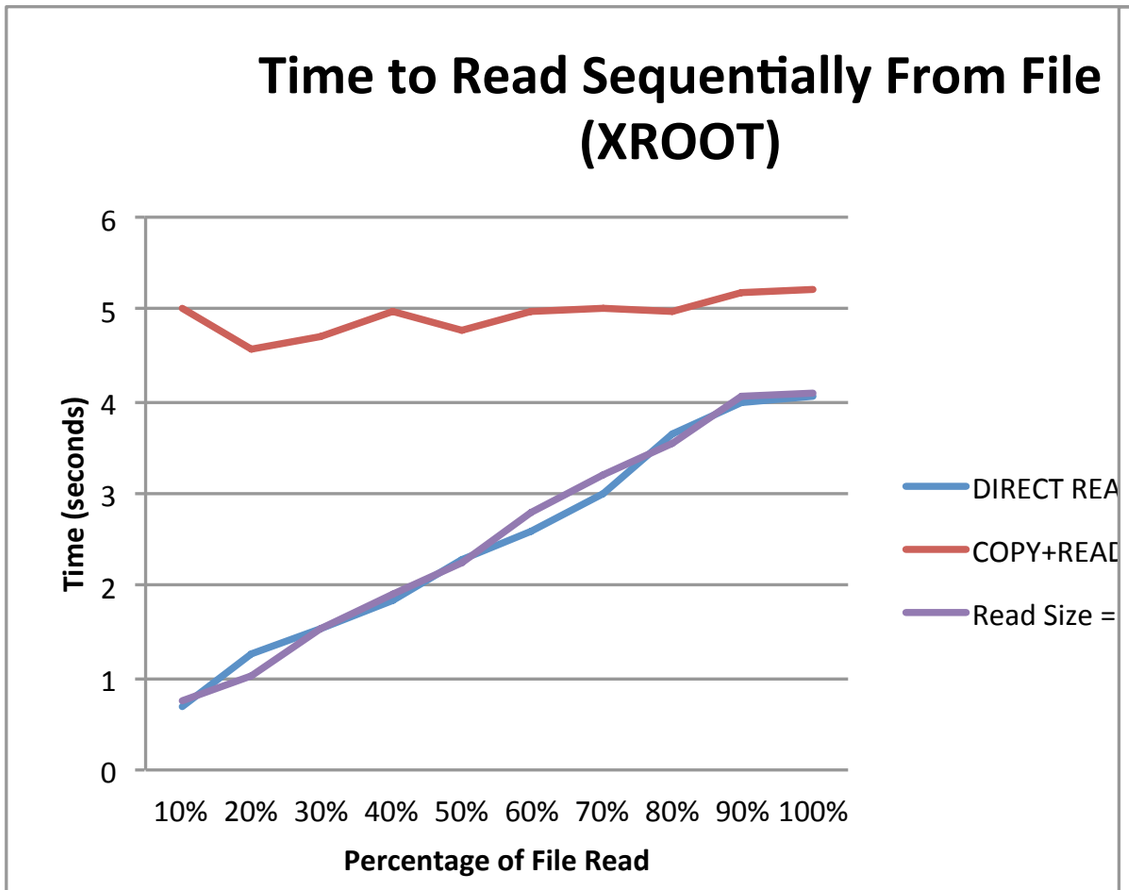
- UK hub of activity for CEPH work.
- Installation and usage tests (ATLAS, other nonWLCG work)
- Monitoring and configuration
- ceph-talk@cern.ch mailing list
- ***Gridftp (direct rados, rados-striper) plugin development.***
- **Performance tests.**

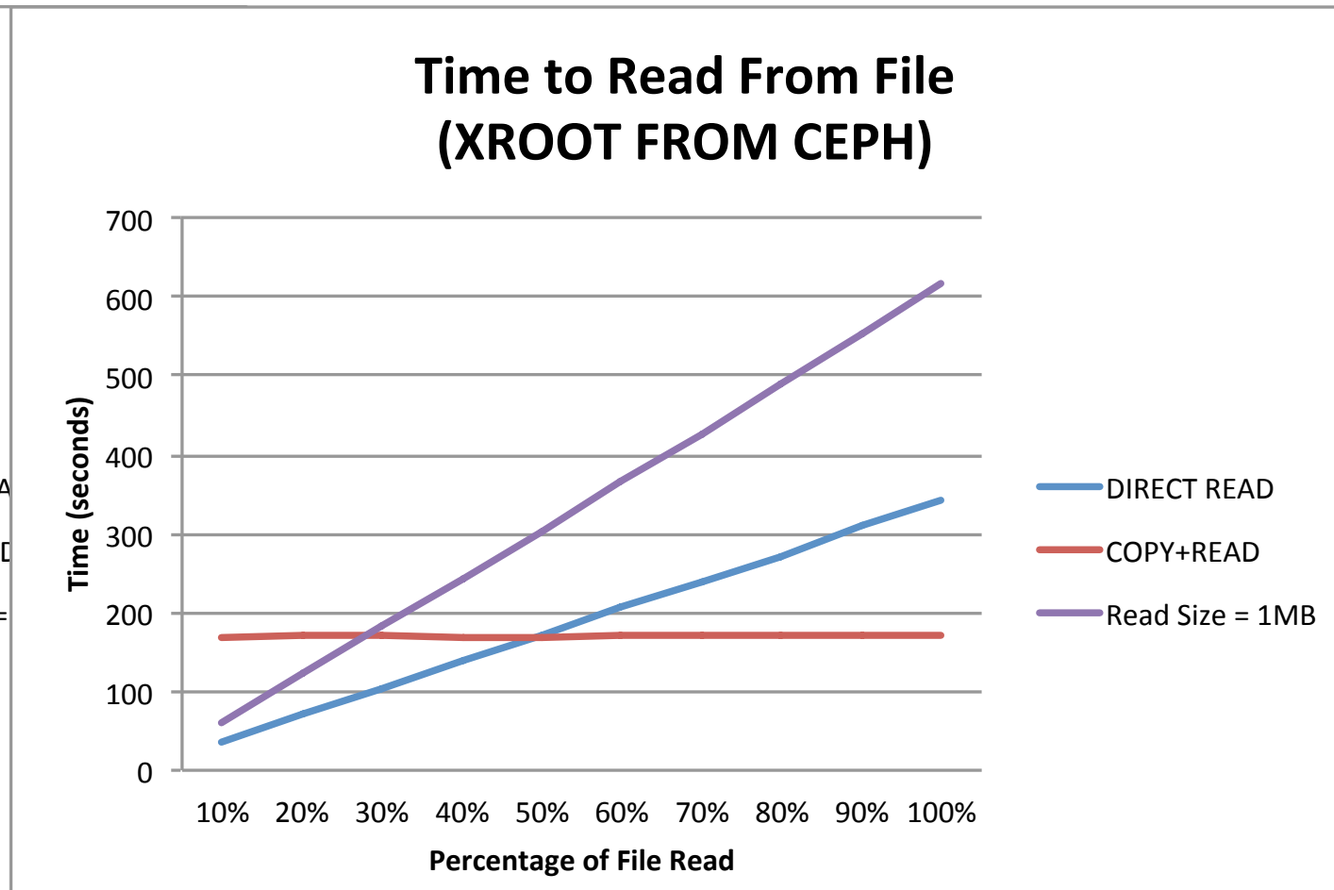
RAL Ceph Pool

- striped, ec coded pools
  - (EC parameters to be optimised)
- Also provides an S3 interface, via dynafed.
- Cluster is purely test - configuration and hardware is not optimal
  - determining good config is also a research goal!

RAL transfer Tests

- (Thanks to Shaun deWitt)

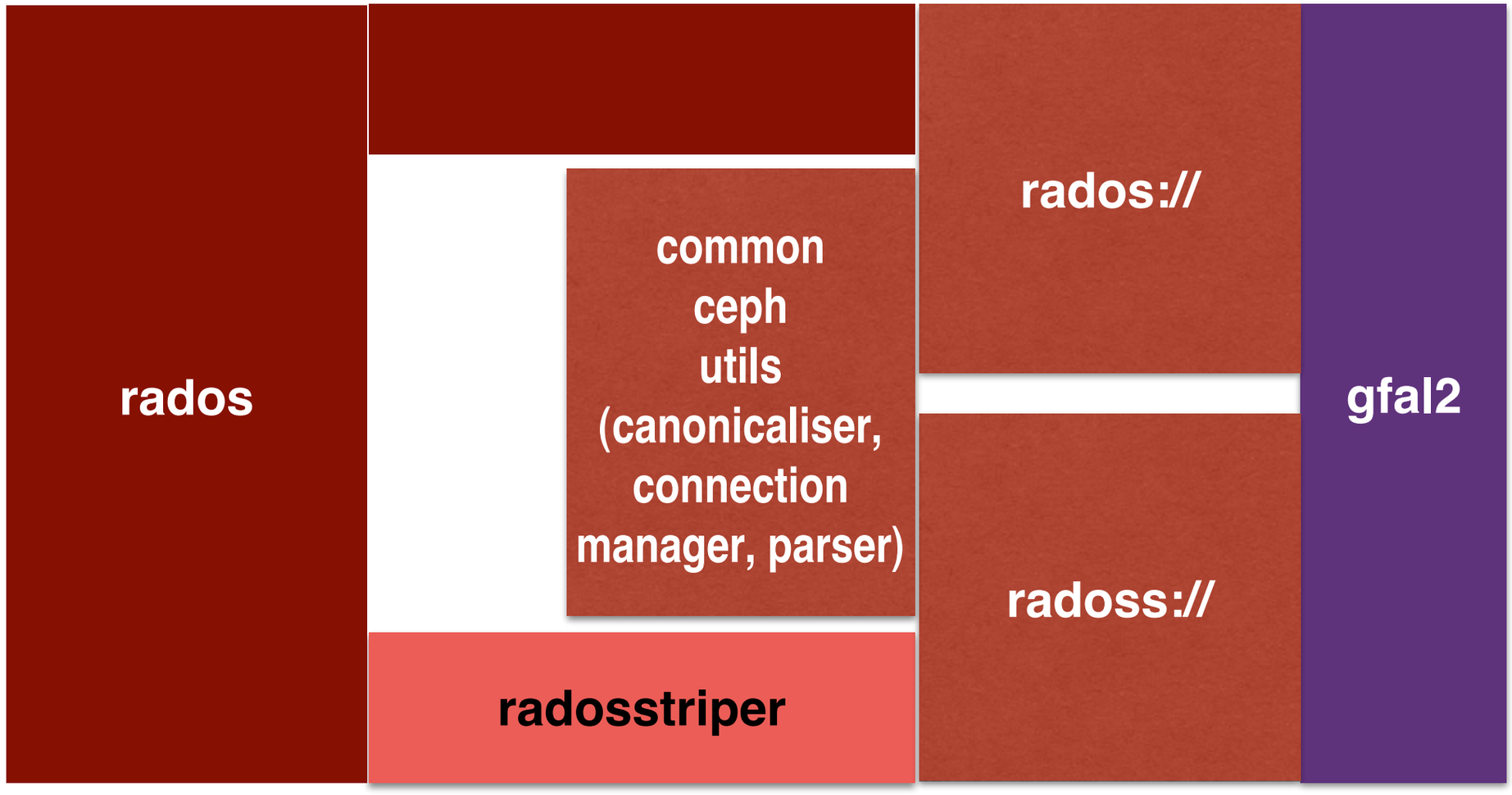




(Test uses Sebastien Ponce's xrootd via radosstriper, using random data - performance gap is not inherent to ceph or xrootd
(directly getting component chunks is v fast)

gfal2 - ceph

- Main Glasgow contribution to UK work
- Based on gfal_file_plugin code
- C rados, gfal2 APIs (for pedagogic reasons)
- 2 plugins
  - RADOS (rados://…)
  - RADOSSTRIPER (radoss://…)
- duplicates code, but less messy than single


rados
common
ceph
utils
(canonicaliser,
connection
manager, parser)
radosstriper
rados://
radoss://
gfal2

URL schemes

- Sebastien's URI scheme (implicit cluster and host)
- rados[s]://user@pool,stripes,stripesize,totalsize:/OBJECTNAME
- Potential URL scheme (explicit cluster and host, RFC compliant)
- rados[s]o://user@mon/clustername/pool/OBJECTNAME?stripes=X&stripesize=Y&totalsize=Z

Canonicalisation

- OBJECTNAME is not a path (but we allow ‘/‘ characters).
  - We must canonicalise OBJECTNAMEs before passing to rados
  - (Principle of least surprise - “weak filesystem”)
- We can/should provide percent-encoding of / chars in URLs

Remote tests at RAL

- Using the explicit URL syntax, we can more easily express remote transfers (copies to or from a remote instance of ceph).

- Work already done on characterising remote transfer scaling with protocols backed with “conventional” filesystems.

Examples using all URL|I schemes

```
[ceph@node017 cluster]$ gfal-copy file:///etc/passwd radosso://admin@10.141.101.17/5a3db366-4dc1-45f0-b304-7333995b1391/diamond-data/BA
NANA
In ceph plugin interface init
Copying 1     0s  File size: 1KB
In open
Copying 1   [DONE]  after 0s
[ceph@node017 cluster]$ gfal-ls radoso://admin@10.141.101.17/5a3db366-4dc1-45f0-b304-7333995b1391/diamond-data/
In ceph plugin interface init
In opendir
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000007
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000005
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3
b6087be2-e2dc-4d91-a116-df01eeb81b84
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000006
BANANA.0000000000000000
1d33e460-7e93-46b7-90a1-eb66cec3af71
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000004
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000002
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3.0000000000000000
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000003
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000001
[ceph@node017 cluster]$
```

```
[ceph@node017 cluster]$ gfal-cat radoss://admin@diamond-data:/BANANA | head
In ceph plugin interface init
In open
root:x:0:0:root:/root:/bin/bash
bin:x:1:1:bin:/bin:/sbin/nologin
daemon:x:2:2:daemon:/sbin:/sbin/nologin
adm:x:3:4:adm:/var/adm:/sbin/nologin
lp:x:4:7:lp:/var/spool/lpd:/sbin/nologin
sync:x:5:0:sync:/sbin:/bin/sync
shutdown:x:6:0:shutdown:/sbin:/sbin/shutdown
halt:x:7:0:halt:/sbin:/sbin/halt
[ceph@node017 cluster]$ gfal-rm rados://admin@diamond-data:/BANANA.0000000000000000
In ceph plugin interface init
rados://admin@diamond-data:/BANANA.0000000000000000    DELETED
```

Transparent striping with radosstriper backend

```
8290304 bytes (8.3 MB) copied, 0.0440881 s, 188 MB/s
[ceph@node017 cluster]$ gfal-copy file://home/ceph/cluster/example radoss://admin@diamond-data,2,4096,4096:/BANANAsss
In ceph plugin interface init
Copying 1     0s  File size: 8MB
In open
Copying 1   [DONE]  after 1s
[ceph@node017 cluster]$ rados ls -p diamond-data
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000007
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000005
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3
b6087be2-e2dc-4d91-a116-df01eeb81b84
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000006
BANANA.0000000000000000
1d33e460-7e93-46b7-90a1-eb66cec3af71
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000004
BANANAsss.0000000000000001
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000002
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3.0000000000000000
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000003
BANANAsss.0000000000000000
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000001
[ceph@node017 cluster]$ gfal-rm radoss://admin@diamond-data:/BANANAsss
In ceph plugin interface init
radoss://admin@diamond-data:/BANANAsss  DELETED
[ceph@node017 cluster]$ rados ls -p diamond-data
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000007
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000005
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3
b6087be2-e2dc-4d91-a116-df01eeb81b84
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000006
BANANA.0000000000000000
1d33e460-7e93-46b7-90a1-eb66cec3af71
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000004
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000002
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3.0000000000000000
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000003
b07b7c7d-7fbc-42f1-ab2f-8a0eb9ed6dd3//00000001
```

8 MB file

4MB stripe size

Two chunks created

Both chunks deleted

Issues

- At present, there is no real intersection between the CEPH security layer and Grid security. (dynafed?)

- gfal-sum (checksum) not implemented
  - would like to use the internal ceph checksums, but this has been a goal for ceph itself for >1year!

- Weird bug with multibyte object names.

```
[ceph@node017 cluster]$ gfal-rm radoss://admin@diamond-data:/沖縄用例
In ceph plugin interface init
radoss://admin@diamond-data:/沖縄用例    FAILED
gfal-rm: error: Is a directory
[ceph@node017 cluster]$ gfal-ls radoss://admin@diamond-data:/沖縄用例
In ceph plugin interface init
radoss://admin@diamond-data:/沖縄用例
```

Conclusions/Further work

- Direct Object store interface are a practical alternative to “heavy” POSIX layers.
- Still an emerging area, with lots to explore for feasibility.
- (e.g. Advanced use would need post-root file formats, and segment data into k,v tagged objects per event (say))