# Data Transfer to UK-RDF

Archiving and Copying from ARCHER






# Introduction

- Archer like many HPC systems has a complex structure
- Multiple types of file system
  - Home
  - Work
  - Archive
- Multiple node types
  - Login
  - Compute
  - Serial batch/post-processing
  - Data transfer

# Home

- Four file-systems unified view as /home
  - /home1
  - /home2
  - /home3
  - /home4
- Approx 60TB each
- Standard Network-Attached-Storage (NAS)
- Backup supported
- Compilation and interactive tools
  - Not intended for high-performance or large data-sets.

# Directories in /home

- Every project has an allocation on one home file-system
- Your home directory will live here
  - /home/*<project>*/*<group>*/*<username>*
  - e.g. guest accounts in /home/y14/y14/guestXX/

# Work

- Three file systems, unified view as /work
  - /fs2 1.5 PB
  - /fs3 1.5 PB
  - /fs4 1.8 PB
- High performance parallel file-system build using **lustre**
- Main working disks for parallel jobs
  - Only file-systems available on the compute nodes so binaries and data files should live here

# Directories in /work

- Every project has a directory-tree in one of the /work file-systems
  - **/work/**<*project*>/<*group*>/<*username*>
- Projects can create different groups to manage space allocation if they want.
- If you need to share data within a group use
  - **/work/**<*project*>/<*group*>/**shared**
- If you want data to be readable outside the project use
  - **/work/**<*project*>/**shared**

# Quotas

- Allocations are set using group quotas.
- Group quota limits total amount of space taken by files with a particular group-id.
  - Each file can only be in one group.
- Default group for files.
  - Follows group of directory if “**s**” flag set on directory.
  - Otherwise effective group of user.
- The default group permissions are set so that group-id should default correct branch of the directory tree.
  - Unless changes using **chgrp, rsync tar** etc.

# Archive (The RDF)

- Home file system for Data Analytic Cluster
- Three distinct file-systems
  - /epsrc (EPSRC projects)
  - /nerc (NERC projects)
  - /general 235TB (others including guest accounts)
- Parallel file-systems built using **gpfs**
- Independent of ARCHER (part of the Research Data Facility)
  - General storage infrastructure. File-systems can grown.
- Tape Backup to second site.
- Primarily intended as a safe-haven for important data.
  - Though performance is not bad either
- Similar directory structure to /work
  - Allocation by file-set

# Mount locations

| | ARCHER home | ARCHER work | RDF archive |
|---|---|---|---|
| login | ☑ | ☑ | ☑ |
| compute | ☒ | ☑ | ☒ |
| Serial batch/pp | ☑ | ☑ | ☑ |
| Data transfer | ☒ | ☒ | ☑ |
| DAC | ☒ | ☒ | ☑ |

# Accessing the RDF

Directly mounted on ARCHER at:

```
/epsrc
/nerc
/general
```

depending on your funding body.

RDF additionally has its own Data Transfer Nodes (DTNs): **dtn01.rdf.ac.uk**, **dtn02.rdf.ac.uk**. Should be used when transferring between the RDF and a remote machine.

# Getting data onto the RDF

You should copy the data in some way.

Do not use **mv**!

Danger of corruption of data when going between different filesystems

Simplest solution is copying with **cp**

# Archiving – Motivation

More efficient use of the filesystem – single file requires fewer metadata operations to move/copy/access.

Can dramatically improve performance, especially with a large number of small files.

Example, 23GB of data = ~13000 32KB-5MB files:

```
$> time cp -r mydata /general/z01/z01/dsloanm/

real    59m47.096s
user    0m0.148s
sys     0m37.358s
```

# Archiving – Motivation

Same files in an archive:

```
$> time cp mydata.tar /general/z01/z01/dsloanm/

real    3m3.698s
user    0m0.008s
sys     0m33.958s
```

Some initial overhead required for archive creation but time saved on subsequent accesses.

# Archiving – Utilities

Common archiving utilities on ARCHER:

- tar
- cpio
- zip

Some technical differences but choice mostly personal preference.

Generally recommend forgoing compression to speed up process but there is a compression/transfer time trade-off.

# Archiving – tar creation

Ubiquitous “tape archive” format.

Common options:

-c create a new archive
-v verbosely list files processed
-W verify the archive after writing
-l confirm all file hard links are included in the archive
-f use an archive file

Example command:

```
tar -cvWlf mydata.tar mydata
```


# Archiving – tar extraction and verification

-x extract from an archive

```
tar -xf mydata.tar
```

-d "diff" archive file against a set of data

```
$> tar -df mydata.tar mydata
mydata/damaged_file: Mod time differs
mydata/damaged_file: Size differs
```

Note: tar archives do not store file checksums

Original data must be present during verification.

# Creating an archive

- Can take substantial CPU resources
  - submit a batch job to the post-processing nodes (serial queue)

```
#!/bin/bash --login
#
#PBS -l select=serial=true:ncpus=1
#PBS -l walltime=00:20:00
#PBS -A y14
#
# Change to the directory that the job was submitted from
cd $PBS_O_WORKDIR
tar -cvWlf mydata.tar datadirectory
cp mydata.tar /general/y14/y14/guestXX/
```

  - user@archer:~> qsub archive.pbs

# Archiving – cpio creation

Archiving utility provided by most Linux distributions.

Common options:

-o create a new archive (copy-out mode)

-v verbose

-H use the given archive format (crc recommended)

No recursive flag – combine with “find” for directories

Example command:

```
find mydata/ | cpio -ovH crc > mydata.cpio
```

# Archiving – cpio extraction and verification

-i extract from archive (copy-in mode)

-d create directories as necessary

```
cpio -id < mydata.cpio
```

--only-verify-crc verifies file checksums (skips extraction)

```
$> cpio -i --only-verify-crc < mydata.cpio
cpio: mydata/file: checksum error (0x1cd3cee8,
should be 0x1cd3cf8f)
204801 blocks
```

# Archiving – zip creation

Widely used and supported by most major systems, including current versions of Windows.

Common options:

-r recursively archive files and directories

-0-9 compression level (-0 recommended on ARCHER)

Example command:

```
zip -0r mydata.zip mydata
```

Note: zip files **do not preserve hard links** (data is copied).


# Archiving – zip extraction and verification

Uses a separate utility for extraction.

```
unzip mydata.zip
```

-t test archive (zip file stores CRC values by default)

```
$> unzip -t mydata.zip
Archive: mydata.zip
      testing: mydata/  OK
      testing: mydata/file    OK
No errors detected in compressed data of mydata.zip.
```

# Copying – Utilities

Local copy from ARCHER

- cp
- rsync

Via SSH

- scp
- rsync

For very large transfers

- Grid-FTP
- bbcp


# Copying – Local Copy

```
cp -r source /epsrc/gid/gid/destination
```

Copying to the mounted RDF filesystem exactly the same as a normal copy between directories.

```
rsync -r source /epsrc/gid/gid/destination
```

Pro: rsync will not attempt to transfer files that already exist.

Con: this “mirroring” requires a large number of metadata operations, slowing performance.

Recommend rsync over cp when resynchronising a previously copied directory containing large files.

# Copying – Remote Copy

For remote transfers DTNs should be used.

- Possible mechanisms:
  - scp – encrypted
  - GridFTP – for large files, certificate-based
  - bbcp – same authentication as scp but unencrypted
  - Globus online – large files, browser-based, simple GUI
- See later talk for full details

# Typical workflow

- Run large parallel job on ARCHER (1000's of cores)
  - produces many files on /work
- Transfer
  - store in an archive, possibly compress, copy data to RDF
- Not appropriate to do transfer by hand on login nodes
  - shared resource with other users and could overload them
- Not appropriate to do transfer in ARCHER job
  - you pay for all the cores even though you're not using them
  - MOM nodes are shared resource and could overload them
- Solution
  - Do this in a single PBS job
  - ARCHER script finishes by doing a "qsub archive.pbs" to serial queue

# Summary

RDF mounted directly on ARCHER login nodes. DTNs available for remote transfers

Archiving improves performance. Be aware of metadata operation bottleneck. Use serial queue for large transfers.

More than one way to copy data. Additional security-performance trade off.

For advice contact: support@archer.ac.uk


# Links and References

- Data Management Guide:
  - http://www.archer.ac.uk/documentation/data-management/
- UK-RDF Guide:
  - http://www.archer.ac.uk/documentation/rdf-guide/
- User Guide – ARCHER Filesystems:
  - http://www.archer.ac.uk/documentation/user-guide/resource_management.php#sec-3.3
- GridFTP tools:
  - http://toolkit.globus.org/toolkit/downloads/



# Reusing this material




archer

epcc

THE UNIVERSITY OF EDINBURGH