Sharing Access to iRODS Files

Chris Smith (chris@distributedbio.com)
Distributed Bio
iRODS User Group Meeting 2012
Pharma iRODS Implementation Goals

• Find projects through metadata
  – Collate metadata from disparate databases and files in single location
  – Owner, sample, organism, experiment etc..
  – Simple interface to search and find the right file

• Pro-actively manage high performance, but limited, cluster storage
  – ~50 million files, 250TB
  – Which projects should be off cluster storage

• Allow users to drive life cycle
  – Previously admins did all the heavy lifting moving data around

• Provide an archive tier
  – But still allow for querying and easy retrieval
Pharma iRODS Implementation Requirements

• iRODS should not get between users and their normal data processing activities
  – or degrade performance ... FUSE filesystem on cluster is not feasible

• Should not have to move data from the cluster storage in order to allow for querying
  – need to register it into iRODS “in place”

• The cluster storage represents “the truth”, and iRODS needs to reflect this truth
  – that includes when moving data from the second tier back to the cluster storage
Pharma Data Lifecycle Management
The Interesting Part…

iRODS

ICAT

Oracle DB

Compute Cluster

NFS

WAN

NFS

active

Resource

inactive

Resource

MSS

Cache

archive

Resource

MSS

Tape

/inactive1

Storage

/inactive2

Storage

/iRODS

Server

/iRODS

Server

Copyright Distributed Bio LLC 2012
The “Direct Access File Driver”

- Retains original user/group ownership and file mode for files in vaults of this type
- Basically wraps “unix file system” driver calls with changes of user/group context
  - performs actions as the “client user”
  - does require using the RUN_SERVER_AS_ROOT mode
  - iRODS and UNIX/Linux need to see the same namespace of usernames and groupnames
- Modified the function signature of some file driver routines (create, open, mkdir) to pass a structure containing filesystem meta-data information
Capturing Filesystem Meta-data

• iput, irsync, ireg need to capture meta-data for files/directories
  – passes this using new key/value pairs: fileUid, fileOwner, fileGid, fileGroup, fileMode, fileCtime, fileMtime, fileSourcePath
  – new R_OBJT_FILESYSTEM_META table, indexed by object_id, that is parallel to R_DATA_MAIN and R_COLL_MAIN

• when file driver create/open/mkdir calls are made, this information is passed along to the drivers if available (although only the direct access driver will actually use it)
Current Issues and Limitations

• iget/irsync doesn’t use this information to restore permissions
  – would only make sense when running iget/irsync as root (much like the -p option to tar)

• Can’t set meta-data when calling imkdir, and can’t ireg or iput an empty directory (to force meta-data collection relative to a reference)

• When doing irepl to a “direct access” resource, there are sometimes issues creating sub-directories and files if the containing directory permissions are too restrictive
  – consistent semantics, but not always the desired behaviour
  – adopted this behaviour in lieu of performing operations as root, and then fixing up permissions
Future Directions

• Get the code into iRODS SVN (maybe in time for 3.2 release?)
• Address the limitations and harden the code
  – especially important when doing things with elevated privilege
• Needs some “best practice” documentation to help guide on it’s use (e.g. keep usernames in sync, understanding limitations and their impact)
• Perhaps add a “read-only” mode to the driver
  – can manipulate ICAT information, but can’t make any changes to the underlying filesystem