Using iRODS as a presentation layer for Research Data Storage at UCL

iRODS User Group meeting 2017, Utrecht. 2017-06-14

Daniel Hanlon (d.hanlon@ucl.ac.uk) - University College London

---

Contents

Research Data Services

- Central services for all *researchers*
    - all schools, faculties and departments
    - >5000 active researchers at UCL
- Drivers
    - Burden of data management
    - USB HDDs
    - Re-use of old data
    - Sharing of new data
    - Compliance

Research Data Storage

- Project-centric storage for live data
    - PI retains control
    - Shared working storage
    - 10TB allocation
    - Time limited

- Current state in iRODS
    - 215 projects, 635 users
    - 30 million objects
    - 334 TB
    - (up to)100K objects, 2TB/day

---

iRODS infrastructure

Hardware:

- 2PB storage
    - DDN WOS
- WOS GOA
    - 3 WOS zones
- ICAT server
    - irods.rd.ucl.ac.uk
    - cache resource
- ICAT database

    - postgres pair
    - pgpool-II

---

iRODS infrastructure

Middleware:

- One zone
    - /rdZone
- $ ilsresc
  wos:compound
  ├──── wosArchive:wos
  └──── wosCache
- No home directories
    - Project-centric
    - /rdZone/live/
- Interfaces
    - iCommands

    - Cyberduck
    - DAVRODS

- Configuration

    - irods_environment.json
    - RDS.cyberduckprofile

- "PAM" authentication

    - ...mostly (see next slide)

Authentication

- replace ~irods/iRODS/server/bin/PamAuthCheck
- echo <*password*> | PamAuthCheck

#!/usr/bin/env node var fs = require('fs');

```
var checkAccount = function(loginRequest){
    if *<username fits UCL format?>*
      //bind to LDAP and authenticate
    else
      //do something else to authenticate
    fi
```

```
    if ( *fails to authenticate by whichever method* ){
      console.log('Not Authenticated');
      *process.exit(1);*
    } else {
      console.log('Authenticated');
      *process.exit(0);*
    }
}

var credentials={};
credentials.password=fs.readFileSync('/dev/stdin').toString();
credentials.username=process.argv[2];
checkAccount(credentials);
```

Cache management

- Goals:
  - move data in to WOS as quickly as possible
  - clear cache
- SHA256 checksums
  - acPostProcForPut
  - compare cache and archive

DirectTrimThis.r

```
for a file in cache...
  checksum if not already done
  find archive replicas
  if archive replica exists
    if cache checksum == archive checksum
      trim cache replica
    else
      trim archive replica
  else
    copy to archive
```

Cache management

- Bash script wrapper calling irule -F

DirectTrimThisWrapper

```
foreach file in cache
  is the file open?
    show its state
  else
    call irule DirectTrimThis.r
wait for 300s if the cache is empty
```

- Log to wherever or tmux + stdout

Utilities

- C++
    - woid2isha
    - isha256sum
    - GUI: QT iRODS checksum utility
- bash
    - ivi, ilocate, icat, izcat, izput

(demo)

In progress + future work

- In-line cache management
    - Client write complete = policy compliant in WOS

- msiWoid2isha
- Patch to msiSysMetaModify to allow datestamp and more
    - itouch
- Windows support
    - iCommands
    - VM/docker

- Patch to WOS driver to change delete process
    - infinite versions
- Checksum utility -> reliable upload tool

Thankyou

https://github.com/danielhanlon/iwos