Run control and data flow

Matt Kramer

2x2 DAQ/computing meeting
Mar 7 2024

BERKELEY LAB

Run control requirements

- Remotely start/stop runs in all 3 DAQs simultaneously
- Configure each DAQ for each new run
  - Minimize manual access/configuration of each DAQ
- Produce metadata for those systems where the DAQ does not
  - Currently only MINERvA DAQ produces metadata

Run control design

- Two options were discussed:
  1) Modify MINERvA run control to support the CRS/LRS DAQs
  2) Write separate run control software that speaks the MINERvA run control protocol (+ CRS/LRS)
- Decision: Option 2
  - Fewer risks, less ramp-up time
- Implementation: Lightweight Python package
  - Use mnvruncontrol library for full-featured remote control of the MINERvA RC
  - Command line interface (+ simple web GUI?)
  - Configuration dictionary loaded from TOML file

Controlling MINERvA

- MINERvA already has a very “remote control-friendly” design
- Jeremy’s “PostOffice” protocol (in mnvruncontrol) makes it easy to send commands/queries to MINERvA run control
- On my laptop, have dummy instances of MINERvA’s DataAcquisitionManager and RunControl
- Inspection of RunControl code revealed the necessary commands to send to the DataAcquisitionManager
  - Start/stop run, get/release control, query status
- Can successfully tell DataAcquisitionManager to start a run
  - Thanks to PostOffice, RunControl sees the state change without our telling it; we need only talk to DataAcquisitionManager
  - Configuration is sent as part of the start command

Controlling LRS

- LRS DAQ (afi-daq) already includes a remote control server that speaks a very simple protocol
- On my laptop, have afi-daq running in a container
  - Able to send start/stop commands to it
- Remote control also allows sending run config in JSON form
  - Need to explore/test this
- New remote control capabilities can be added by modifying afi-daq/libs/remote-control-server
- Minor annoyance: Listen port is randomized
  - Potential fix: Modify afi-daq/apps/afi-run-control/RcCoreApp.cpp

Controlling CRS

- Current approach:
  - Start run: Run control SSHs into DAQ machine, spawns record_data.py in the background, writes that processes's PID to a tmp file
  - Stop run: SSH in, read PID, send SIGSTOP
- If CRS DAQ adds a persistent remote control server, we can switch to using that

Current status and roadmap

- Preliminary code: https://github.com/mjkramer/mx2x2_run_control/
- Contains classes for starting/stopping runs in the 3 systems
- Immediate-term: Needs CLI, config file loading
  - Where “config file” contains e.g. host/port of each system
- Near-term TODOs
  - Support for run configuration
  - Accounting for latency, startup time, etc. to ensure that run start/stop time is aligned as closely as possible between the 3 systems
  - Failure recovery
  - Metadata generation(?)
- Longer-term: Web GUI with start/stop buttons, common run config params, etc.
  - Retain CLI / config.toml for experts

A word on metadata

- We need to decide whether metadata generation should be the responsibility of the CRS and LRS DAQs
  - Advantage: Direct access to internal details
  - Disadvantages: Risk, complexity
- Worth keeping in mind: Even if metadata generation lives outside the DAQs, we might still need to modify them to export more information
- Need a comprehensive list of metadata fields for each system
  - Will help in choosing an implementation
- Can reference existing metadata support in MINERvA and 2x2_sim

Data flow: The ideal

- Eventual goal:
  1) Data+metadata written to DAQ local storage
  2) Ingest daemon pulls them to scratch dCache via xrootd
  3) Declaration daemon declares them to MetaCat and Rucio
  4) Rucio places FNAL replica in persistent or tape-backed dCache
  5) Rucio asks FTS3 to transfer the data to NERSC via xrootd
  6) File detected at NERSC; prompt processing launched

Data flow: Reaching the ideal

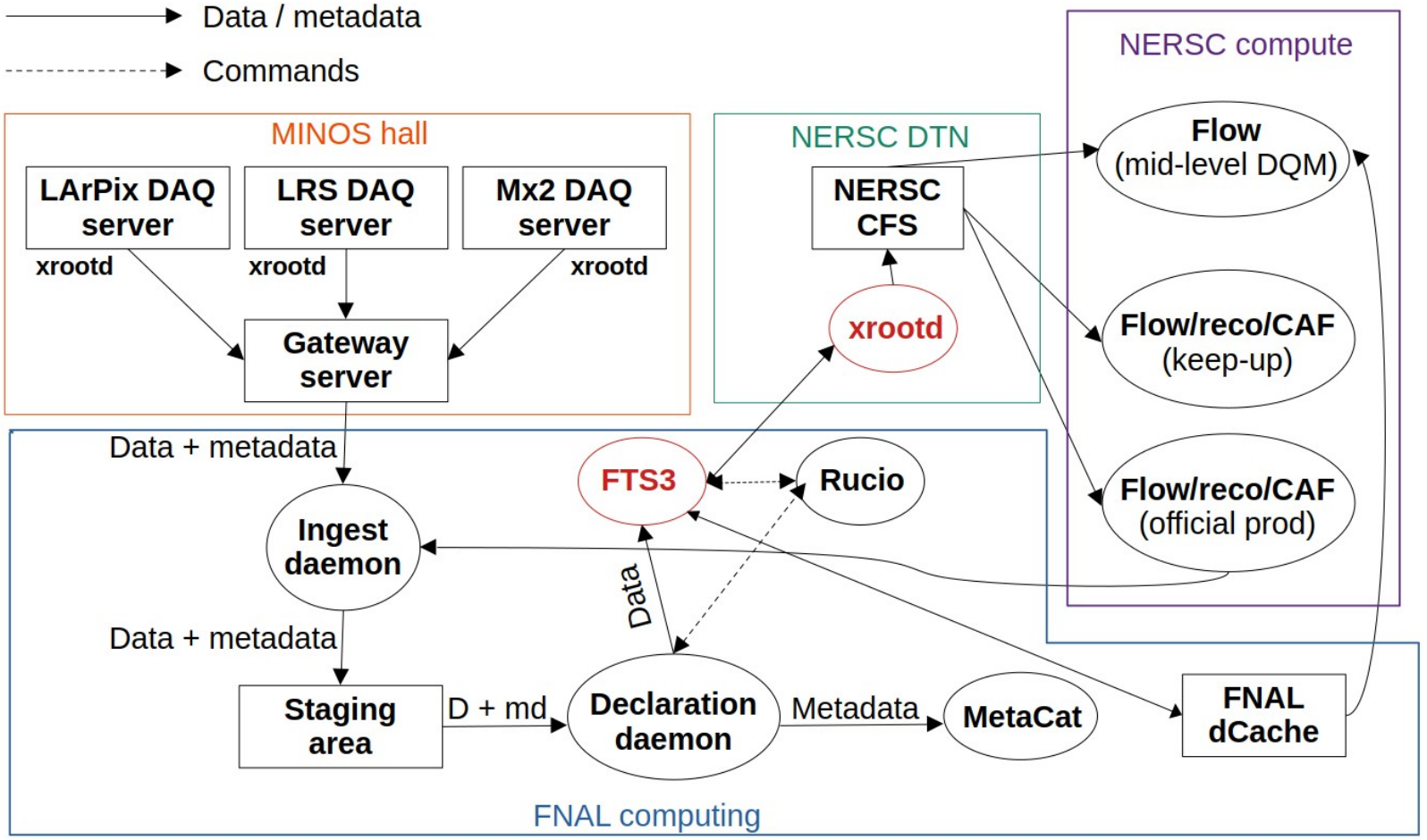
- What do we need to get there?
  - xrootd servers undeground + firewall rule (done?)
  - Full metadata generation (i.e. add CRS and LRS metadata)
  - Configuration of Ingest, DeclaD
  - Support for third-party copy in NERSC xrootd (done?)
  - Rucio rules for NERSC
  - Decision on treatment of “small” files (CRS, MINERvA)
    - Option 1: Tar up CRS files from multiple runs. Separately tar up MINERvA files from multiple runs
    - Option 2: Tar up CRS+LRS+MINERvA for each run

Data flow: In the meantime

- Can now get 5-week SSH keys for both the NERSC “dunepro” and “dunepr” accounts (FNAL-managed and LBNL-managed, resp.), usable from acd-gw01 and acd-gw02
- Central run control (or a separate daemon) can transfer complete data files from acd-daq0X to NFS dropbox
- rsync loop on acd-gw0X can transfer from dropbox to NERSC CFS, FNAL dCache
  - Analogous setup used successfully for transfers from Bern
- If scp is problematic, can use xrootd or gsiftp
  - What’s the max lifetime of a VOMS proxy?

Backup

The big picture



NERSC as a Rucio Storage Element

- XRootD server now running on dtn14. Does not yet support third-party copies using FNAL's "dunepro" credentials
- Need that feature in order to enable Rucio's use of File Transfer Service (FTS3) for transfers between NERSC and FNAL
- Is this the last remaining TODO before NERSC can be activated as an RSE with bidirectional transfers?
- Does the fix need to happen at NERSC or at FNAL?

Rucio rules, FTS3

- Once NERSC is functioning as an RSE, rules need to be added to Rucio for transfer of Minerva (and later 2x2) data
- Can we define rules that will work indefinitely, or will the rules need to be regularly updated?
  - In the latter case, how can we automate these updates?
- Will there be a need for any changes to the File Transfer Service?
- Likewise for ingest and declaration daemons?

In the hall

- Need XRootD servers running in the halls with access to DAQ data
  - Run xrootd on the DAQ machines or on e.g. ops machines?
  - In latter case, how will xrootd access DAQ disks?
  - How many xrootd servers?
- Need to tell gateway to allow ingest daemon to reach xrootd
- Need to write metadata generator for CRS and LRS
  - Post-run scripts called by the central run control?