ICAT Deployment at HZB

Rolf Krahl

ICAT F2F Meeting, RAL, November 2017

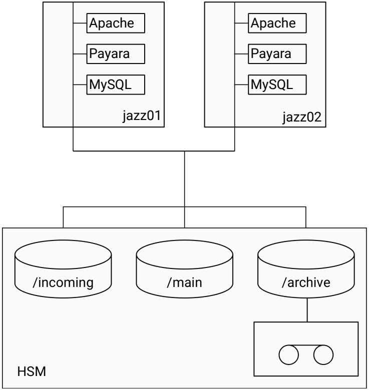
Outline

1. ICAT Deployment with Docker

2. Data Ingestion into IDS

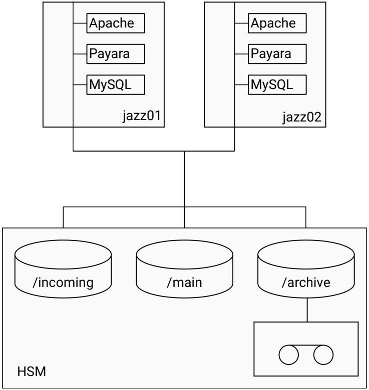
Overview

- Storage system: HSM providing file systems `/incoming`, `/main`, and `/archive`.
- `/archive` comprises disks and tapes, `/incoming` and `/main` only disks.
- Two dedicated server for ICAT to ensure availability.
- Use Docker inside the server.
- Use separate containers for MySQL database, Payara with ICAT compononts, and Apache as frontend, respectively.



Deployment Alternatives

- Active/passive: all services on one server, the second server is either in idle standby or in maintenance.
- Distributed services: in normal operation use both servers to separate services. One server for user operation (TopCAT), one server for IDS and ingest.
- Use network to route traffic to the appropriate server.



Deployment Configurations

Need to provide three different configurations:

1. All ICAT components in one server
2. One server with ICAT, Lucene, and IDS, no TopCAT
3. One server with TopCAT and ICAT, no Lucene, no IDS

ICAT Docker Image

- Have one generic ICAT docker image with all components installed.
- In the image, Payara is installed, but not set up. The Payara domain is deleted.
- A setup script creating and configuring the Payara domain is run during startup of the container if the domain is not present.
- If the domain is present, the setup script only starts Payara with the given domain. This allows to stop and to start the container without creating the domain again each time.
- A configuration directory is linked into the container by a bind mount. This external configuration controls which ICAT components to install in Payara and contains all configuration files for the components.
- This way, one single image will work for all possible deployment configurations. Furthermore, the image does not contain any secrets, such as database passwords.

Data Ingestion Preconditions

- Some 40+ experimental stations at BESSY II
- Very heterogenous, different workflows, different control systems, different operating systems
- Users bring their own experimental station
- Only a small number of high data volume producers
- Consequence: setup a generic workflow with very low prerequisites at the instrument side. Treat the few high volume producers individually.

Generic Workflow

- Setup an incoming area in central storage for ingestion.
- Instruments communicate with central systems by simple web service calls.
- One call to start ingestion, taking investigation identifier as parameter. It creates a dedicated SMB share in the incoming area and returns path and credentials.
- Instrument mounts the share and stores the data.
- Second call to finish, take name of the share as parameter. It transfers the data to IDS and removes the SMB share.
- Only requirement at instrument side: ability to make web service calls and to mount SMB shares.
- Python scripts to make the calls are provided as an option.

Ingestion Variant I: Write to Archive Storage

Variant I to transfer the data from the incoming area to IDS:

- Workflow:
  1. Write a ZIP file per dataset in IDS archive storage.
  2. Create the dataset object along with the datafile objects in ICAT.
- Notes:
  - Need to protect against concurrent file access by IDS: see my talk from March 2015 F2F meeting for technical details.
  - Need to adhere to the IDS internal structure when creating the ZIP file in archive.
  - Works with the current version of ids.server and some modifications in the plugin.

Ingestion Variant II: Storage Plugin Reads Incoming

Variant II to transfer the data from the incoming area to IDS:

- Modified storage plugin: MainStorage looks for files in two places, main storage and incoming area.
- Plugin should access incoming area read only to protect against concurrent file access.
- Workflow:
  1. arrange the files in incoming such that the plugin will find them.
  2. Create the dataset object along with the datafile objects in ICAT.
  3. Trigger a `WRITE` operation to create the archive file.
  4. Remove the files from incoming.
- Note: there is currently no `WRITE` API call in IDS (but a pending pull request to add it).

Ingestion Variant III: Write to Main Storage

Variant III to transfer the data from the incoming area to IDS:

- Workflow:
  1. Copy the datafiles to IDS main storage.
  2. Create the dataset object along with the datafile objects in ICAT.
  3. Trigger a `WRITE` operation to create the archive file.
- Notes:
  - Must protect against concurrent file access by IDS in main storage! This is currently not possible, but see my proposal on file system locking in the IDS storage plugin.
  - There is currently no `WRITE` API call in IDS.