CTA+LHCb Discussion

Minutes of the meeting of 17 August 2018

Present

- **IT:** Michael Davis, Julien Leduc
- **LHCb:** Joël Closier, Christophe Haen

We discussed the LHCb workflows in more detail, in the context of the proposed EOSLHCb/EOSCTALHCb configuration. LHCb understand that they will need to change their workflows and this will require some development effort on their part. For the most common workflows there is no fundamental problem. However, there are some use cases which will require some creative thinking on our part.

Differences between LHCb and ATLAS

CH pointed out some of the differences between ATLAS and LHCb workflows. ATLAS focuses on maximum availablity of data, whereas LHCb focuses on most efficient use of resources. This philosophical difference leads to several differences in practice:

- LHCb keeps fewer disk copies across the grid. ATLAS may copy a file a dozen times to different SEs, while LHCb has one or two. So the impact of transient errors (site down, disk pool unavailable) as well as permanent errors (corrupted disk file) is much higher for LHCb, requiring them to get the file from tape in these cases.
- ATLAS sites will submit many requests for a file to different SEs (including T0), and then when the file arrives they cancel the other requests. LHCb do not do this. If they request a file from T0, it means they really need it.

Latency of writing raw data to tape

Currently data is written directly from the LHCb pit to CASTOR. In our proposed CTA setup there will be one extra hop: first to the big EOS instance, then to the EOSCTA instance. CH is concerned that the additional copy will add latency to the time taken to archive files, perhaps requiring them to increase the size of the disk cache at the pit.

We should measure this in order to determine if this is really a problem. LHCb CASTOR logs will give us the current latencies. We could use the ATLAS CTA test instance to measure how much difference the extra intermediate copy makes.

Data Taking Workflow

The main issues are:

- Concerns about the latency from the extra hop as mentioned above.
- They do not want to open the files up to the grid until they are reported safely on tape. This is because they want the checksum of the disk file and tape file to have been validated before production jobs and users start to process the file, to guard against data corruption.

Data colocation

JC asked if CTA will have the same feature as CASTOR where data is allocated to a tape family. They are concerned that raw data and data from production jobs and user jobs should be kept on physically separate storage and that they should have some control over what goes together. (As I understand, we will offer exactly the same functionality as CASTOR via Storage Classes and Tape Pools, just the terminology used by the experiments is a little different).

Specifically they are concerned about the colocation of data and reducing the number of tape mounts and therefore the latency in order to retrieve a complete dataset. I said we are aware of this problem but do not have a definite solution at the moment. I mentioned that we have a Ph.D. student starting in October to look at how we can optimise access to the tape storage, including data colocation.

Jobs which run on CASTOR

LHCb have some use cases for running jobs on CASTOR which need to be addressed in our CTA setup:

- Calibration team: their workflow requires them to recall a single file from tape and read it once. This is done infrequently, but they need to be able to recall the files quickly (< 1 hour). These jobs are run on CASTOR as it is a waste of time and resources to copy it to the main disk instance in order to read it once then discard it.
- Last option fallback for user jobs: normally users access a disk copy of their files, but in some cases the file may be temporarily unavailable. When this happens they would like to have the option of running the user job on the stager instance, until the disk copies become available again. At the moment, this use case is very rare, but depending on the computing model that will be adopted for Run3, it might increase.

These use cases require read access to tape only. Only the Production role in LHCb VOMS can write to tape.

File Transfer Protocols

They use FTS with `gfal2`/SRM underneath for all staging out of CASTOR. FTS is used everywhere except export from the pit and one-shot jobs which are run on the CASTOR instance as mentioned above.

Ideally they will be able to modify their `gfal2` scripts to change the protocol to XRootD and it will "just work". One issue is that XRootD 3rd Party Copy is not supported in all cases (e.g. transfers to dCache) so this could result in an extra local copy and additional network traffic.

There was a question about grid transfers to RAL/ECHO which only supports `GSIFTP` (Currently done using Sebastien Ponce's XRootD plugin for CASTOR?)

There was a question about how grid certificates will be mapped to privileged users. Will this be under the control of the experiment or will they have to ask us to do it?