Contribution ID: **255** Type: **Poster**

Towards redundant services in dCache

Thursday, 13 October 2016 16:30 (15 minutes)

As a robust and scalable storage system, dCache has always allowed the number of storage nodes and user accessible endpoints to be scaled horizontally, providing several levels of fault tolerance and high throughput. Core management services like the POSIX name space and central load balancing components however are merely vertically scalable. This greatly limits the scalability of the core services as well as provides single points of failures. Such single points of failures are not just a concern for fault tolerance, but also prevent zero downtime rolling upgrades.

For large sites, redundant and horizontally scalable services translate to higher uptime, easier upgrades, and higher maximum request rates. In an effort to move towards redundant services in dCache, we are reporting on attacking this problem at three levels. At the lowest level dCache needs a service to locate the various dCache nodes. In the past a simple non-redundant UDP service was used for this, but in the latest release this functionality has been ported to Apache ZooKeeper. ZooKeeper is originally part of Hadoop and is a redundant, persistent, hierarchical directory service with strong ordering guarantees. In particular, the strong ordering guarantees make ZooKeeper perfect for coordinating higher level services. On top of the location service, dCache uses a common message passing system to communicate between various services. In the past this relied on a simple star topology, with all messages going through a central broker. This broker forms a single point of failure and is possibly a bottleneck under extreme load conditions. This problem is addressed with a multi-rooted multi-path topology consistent of a set of core brokers forming a fully connected mesh and all other services connecting to all brokers. Finally, each of the central services is made scalable and redundant. For some services this is trivial, as they maintain minimal internal state. For others, the ability of Apache ZooKeeper to act as a coordination service is central. In some cases a leader election procedure ensures that various background tasks are only executed on a single node. In other cases shared state can be stored in the ZooKeeper, e.g. to ensure that a file is only staged from tape once. Further changes to the internal message routing logic will allow load balancing over multiple instances of a service.

The first two steps outlined above will have been deployed in production by the time this paper has been published. Redundancy and scalability of higher level services is currently only available for the trivial services, while other services will be extended over the following releases.

Primary Keyword (Mandatory)

Storage systems

Secondary Keyword (Optional)

Tertiary Keyword (Optional)

Primary authors: Dr ROSSI, Albert (Fermilab); Mr ASHISH, Anupam (DESY); BERNARDT, Christian (Deutsches Elektronen-Synchrotron (DE)); LITVINTSEV, Dmitry (FNAL); BEHRMANN, Gerd (NEIC); Mr STAREK, Jürgen (DESY); Dr SAHAKYAN, Marina (DESY); FUHRMANN, Patrick (DESY); MILLAR, Paul; Mr MKRTCHYAN, Tigran (DESY)

Presenter: MILLAR, Paul

Session Classification: Posters B / Break

Track Classification: Track 4: Data Handling