EPJ Web of Conferences **214**, 04047 (2019) https://doi.org/10.1051/epjconf/201921404047
CHEP 2018

A data caching model for Tier 2 WLCG computing centres using XCache

Teng Li[1,*], *Robert* Currie[1] and *Andrew* Washbrook[1]

[1]University of Edinburgh, School of Physics and Astronomy, UK

Abstract. As the WLCG continues to consolidate the data storage resources to reduce maintaining costs, more storage elements will be decommissioned or federated in the future. In both cases data streaming would become a common method for data accessing in many LHC Tier 2 and Tier 3 computing centres which makes it vital to optimise the network usage (i.e. latency reduction and bandwidth utilisation). Consequently, the data cache service sits closely to the computing resources becomes a desirable tool. In this paper we discuss how to setup a XCache server at an example WLCG Tier 2 site (Edinburgh) and use the ATLAS analysis queue as a testbed to study the data caching performance. The results could be used to guide the future development to make the caching server more effective.

1 Motivation

During LHC Run 1 and Run 2, the WLCG [1] mostly applied the computing model that co-locates the computing and storage resources. Typically, the central data catalogue distributes data replicas across all sites under a certain set of rules, while the job broker sends the job payload to the sites that possesses the required replicas as input (e. g. the way PANDA [2] works with RUCIO [3]). The mechanism scaled well during the last decade, however the situation is changing with the evolution of the WLCG:

- The WLCG is consolidating data storage resources to reduce maintenance costs. In the future, more storage elements will be decommissioned or turned into volatile pools in smaller sites which would then have to stream data from remote data storage.
- More data storage will be federated and managed under common namespaces. For instance, FAX [4] and AAA [5] have already been in service for years, while GridPP is also investigating VO-agnostic data lake solutions.

As a result of both facts, streaming data option becomes more popular than static data placement, which makes the network the third resource apart from processors and data storage. Therefore, optimisations of the network usage become vital, including latency (RTT) reduction and better use of the bandwidth, which usually has huge impacts on the job performance.

Among various technologies, data caching is usually a cheap and effective way to optimise the network utilization. There are many available data caching solutions, and in this paper, we focus on the XRootD proxy cache (XCache) [6, 7] as it has the mostly desired

* Corresponding author: teng.li@ed.ac.uk

features that are suitable for a Tier 2 site which usually supports various VOs and has limited manpower. We discuss the deployment procedures of XCache, and use the ATLAS [8] analysis queue as a testbed to evaluate the caching performance, which could be referred to optimise the data caching service. The results could be used for other data caching solutions as well.

2 XRootD proxy cache

XCache is originally developed as an extended layer of AAA project, the federated data storage for CMS [9] where XCache serves as a proxy server with the disk-based data caching functionality. In a typical workflow that involves XCache, the downloading or reading requests of input data files sent by the worker nodes are redirected to the XCache server. The latter then tries to download the data from an external storage (e.g. a remote SE or a data lake) and serves it back to the client, during which the input data is written to the cache disk for future fast access.

The reasons XCache is suitable for high energy physics workflow are listed as below:

- It supports data caching at both whole file and sub-file level. The latter is typically profitable for high energy physics analysis jobs as usually only small fragments of input data files are needed.
- As one of the basic features of the XRootD server, XCache is clusterable. Proxy managers could be built to handle the load-balance for multiple XCache servers to scale up.
- It could be deployed as a POSIX serverless cache, which is typically useful to minimize access latency for worker nodes that have a shared filesystem.
- It is based on a plugin structure which could be easily utilized to exploit caching performance (e. g. implementing decision library to filter cold files).

3 Deployment and test

3.1 Deployment

In order to measure the caching performance metrics, an XCache server instance is deployed at an example WLCG Tier 2 site (Edinburgh). The ATLAS analysis queue is used as the testbed. The overview of the testbed is shown in Figure 1.

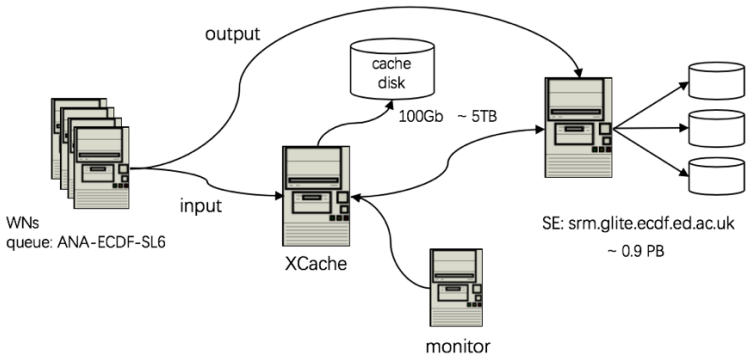

Figure 1. Overview of the XCache testbed

The frontend of the system is 40 worker nodes (80 CPU cores) running on the Scientific Linux 6 system, which exclusively run ATLAS analysis jobs. Due to the fact that XCache is a read-only system, an XRootD client plugin is implemented and installed on the worker nodes to separate the input/output data flow by redirecting the input stream to the XCache server.

The XCache server is deployed with a resize-able cache disk under the same local area network with the worker nodes to minimise the access latency. The concerned configurations are listed in Table 1.

Table 1. Configurations of the test XCache server.

	Value	Implication
pfc.ram	2 Gb	Limit of XCache ram usage
pfc.diskusage	0.85/0.95	Disk cleaning up watermark (low/ high)
pfc.blocksize	512 Kb	I/O block size
pfc.prefetch	8	Number of pre-fetch file blocks

The storage backend is a DPM-based storage pool which has a total capacity of 0.9 PB, configured with the XRootD protocol. The system simulates the scenario that a computing element is directly attached to a remote storage element, which would be one of the common cases for lightweight sites in the future.

3.1 Performance measurement

Roughly four months of data is taken to measure the caching metrics. To evaluate the cache hit rate sensibility to the cache disk capacity, the averaged naive caching hit rate (bytes read from the cache divided by bytes read by the worker nodes) is calculated with different sizes of cache disks, as shown in Figure 2. The cache hit rate varies with different period of time and capacities of cache disk, and reaches 48.9% with a 500Gb cache disk at this scale.

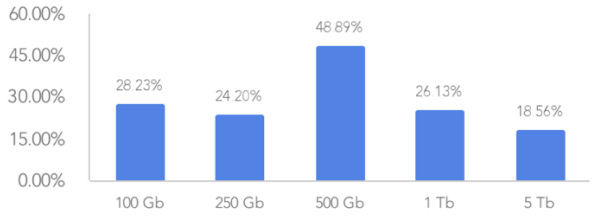

Figure 2. Cache hit rate under different cache disk capacities

For the further study of the caching efficiency, the popularity of the cached files is analysed. Table 2 shows the I/O traffic of different types of files, and their contributions to the cache hit during the four months runtime of the XCache server.

Most cache hit comes from the (D)AOD files (the ATLAS reconstructed data file) and the libraries files (user libraries distributed by PANDA). Figure 3 and Figure 4 show the popularity distribution and popularity tendency with the lifetime of the cached files, respectively. It is obvious that the library files and data files behave fairly differently. The former is extremely hot which contributes ~29% of the outbound traffic with only 1.5% of the inbound traffic, while a very large proportion of the latter is accessed only once after

written to the cache disk. Both the reconstructed data files and library files are obviously hotter during the first 12 hours of their lifetime.

Table 2. Information of cached files

File type	Written into cache (GB)	Read from cache (GB)	Cache hit rate	Contribution to overall cache hit
(D)AOD	26560.3	36874.5	28.0%	70.6%
library	441.8	4693.9	90.6%	29.1%
output	1304.2	1344.2	3.0%	0.3%
log	9.1	9.1	0	0

In view of the above-mentioned facts, the following optimisation methods are expected to provide much better caching performance:

- Since a large proportion of the cache disk is filled by cold data, a decision library could be implemented to filter them out.
- The popularity of the cached files becomes obviously colder after a short time, so the clean-ups should be frequently performed and focused on aged files.
- Different kinds of files show different access patterns. We recommend to cache the small hot files (i. e. library files) with SSDs or memory, or directly on the worker nodes.

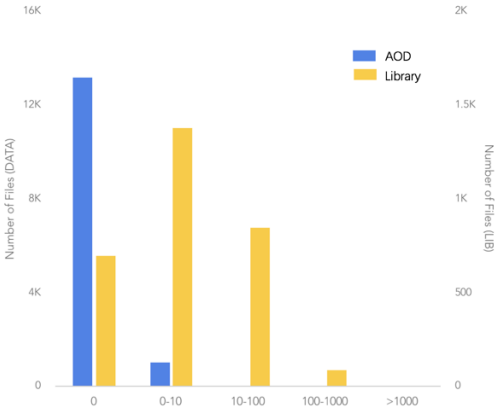

Figure 3. Hotness distribution of cached files

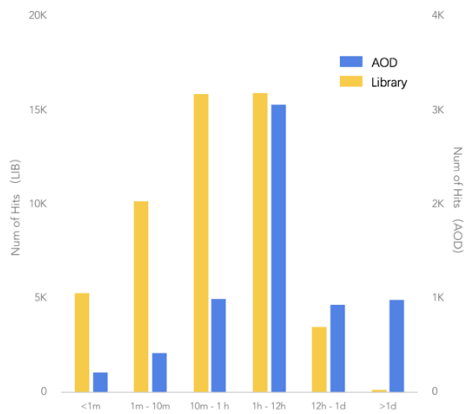

Figure 4. Popularity tendency of cached files

4 Summary

We deployed an XCache server to test its feasibility as an opportunistic data caching service for the WLCG Tier 2 sites, and used the ATLAS analysis queue to study the caching performance. During the four months run time, the XCache server performed fairly stably, showing its advantages as a volatile storage element comparing to a full-stack storage element. We analysed the data caching metrics, and the results show a great potential for further optimisation. The future research and development could be focused on the following items:

- Since WLCG sites usually support multiple experiments, multiple-VO support should be provided, and caching performances need to be studied independently for different workflows.
- The performance optimisation for data caching, as listed in the previous chapter.

- Different use cases of XCache are yet to be studied for certain workflows, e.g. partial file cache for analysis jobs, POSIX server-less cache for worker nodes with shared filesystems and the integration with distributed data management systems.

References

1. I. Bird, *Computing for the Large Hadron Collider*, Annu. Rev. Nucl. Part. S. **61** 99-118 (2011)
2. T. Maeno, *PanDA: distributed production and distributed analysis system for ATLAS*, J. Phys. Conf. Ser. **006** 062036 (2008)
3. C. Serfon, et al., *Rucio, the next-generation Data Management system in ATLAS*, Nucl. Part. Phys. Proc. **273-275** 969-975 (2016)
4. R. Gardner et al., *Data federation strategies for ATLAS using XRootD*, J. Phys. Conf. Ser. **513** 042049 (2013)
5. L. Bauerdick et al., *Using Xrootd to Federate Regional Storage*, J. Phys. Conf. Ser. **396** 042009 (2012)
6. XRootD project page: http://www.xrootd.org/ [Accessed: 25-2-2019]
7. L. Bauerdick et al., *XRootd, disk-based, caching proxy for optimization of data access, data placement and data replication*, J. Phys. Conf. Ser. **513** 042044 (2014)
8. ATLAS Collaboration, *The ATLAS Experiment at the CERN Large Hadron Collider*, JINST **3** S08003 (2008)
9. CMS Collaboration, *The CMS experiment at the CERN LHC*, JINST **3** S08004 (2008)