Quantifying XRootD Scalability and Overheads

2014 J. Phys.: Conf. Ser. 513 032025

(http://iopscience.iop.org/1742-6596/513/3/032025)

Journal of Physics: Conference Series **513** (2014) 032025 doi:10.1088/1742-6596/513/3/032025

Quantifying XRootD Scalability and Overheads

S de Witt and A Lahiff

[1]Science and Technology Facilities Council, Rutherford Appleton Laboratory, Harwell, Didcot, UK, OX12 0QQ

shaun.de-witt@stfc.ac.uk

Abstract. Both ATLAS and CMS experiments are making increasing use of the XRootD architecture to provide access to data not held locally to where a job is running. The anticipation is that this will lead to fewer job failures, although the efficiency of jobs that would otherwise have failed may be reduced. In this paper we look at the overhead and scalability of the XRootD software system, and the overhead of the infrastructure needed to support remote access to data.

1. Introduction

Both the ATLAS and CMS experiments at LHC are making significant changes to their data model and, in particular, the way data is accessed. Up to now, jobs have been steered to the site which hosts the data and, for the most part, the data sets required are streamed from the local storage to the worker nodes and then read locally. While this method of analysis has proved quite effective in analysing the large volumes of data generated as part of the LHC data analysis framework it does require very careful data management by the experiments. Even within this, there about 2% of jobs that fail on the grid because some of the data required for the job in unavailable due to hardware failures or failed transfers.

To overcome these issues both of these experiments have chosen to use XRootD [1] to provide a means of job recovery. If data is not found at a site, this technology allows jobs to access the data at other sites (job fallback) via redirection, as described in some detail in the next section. The ATLAS FAX [2] project is minimising changes to the data model; jobs are still steered towards where data is expected to be, but in the event of failure XRootD will be used to try and find an *on-line* copy of the data at another site. The CMS AAA [3] project is more radical in that jobs will be sent anywhere on the grid without reference to data placement and use XRootD to locate the data. Since the data may be quite distant, streaming access to the data may be slow due to bandwidth limitations. To overcome this, it is envisaged that data will be accessed directly using remote opens on data sets.

This paper does not look at the network latencies, but looks at the scalability (in terms of number of concurrent requests which can be processed) and overheads associated with XRootD itself and compares it to existing technologies used in LHC data processing, particularly comparing it to the SRM [4].

2. XRootD Redirection

To understand the work in this paper, it is necessary to have some understanding of XRootD. This technology provides a means for federated *data access* through redirection. It is based on a

hierarchical model similar to DNS. A client contacts a *local* redirector to ask for some data and if the data is held locally then the client will get redirected directly to the data. If the data is not available locally, then the redirector will query another redirector higher up in hierarchy, say a *National Redirector* that will then contact other redirectors that it knows about. If one of these has the data locally, then a connection is made to between the client and the (national) storage. Again, if the data is not found nationally, queries are redirected to the *continental redirector* to seek for data in a continental domain and, if not found, the query is redirected to a *global redirector*. Once data is found, a direct connection from the remote storage to the client is established.

It is important to note that the XRootD infrastructure does not require a common global namespace. Each site within the federation maintains its own top level namespace, but can have some element in common. The XrootD redirector provides a facility for namespace mapping, so for example an experiment may have a file stored in */data/directoryA/subDirX/foo*, but two sites may store this file under */siteA/StorageSystemRoot/experiment/data/directoryA/subDirX/foo* or */root/exp/data/directoryA/subDirX/foo*. The local redirector associated with the site allows for mapping of the directory */data* to the correct name for the local site. This is necessary for WLCG, but is not necessary to use with the XRootD architecture. This helps make the system extremely scalable and deployable at a large number of sites. However, currently there is no direct integration with object storage systems like HDFS [5] or CEPH [6], other than using the FUSE layer.

An important point to note that once the data is found in this model, a direct connection to the client is established regardless of which redirector has been contacted initially. Thus if a client at some site contacts the global redirector to access data at their local site, then the data is still accessed from the local storage without reference to other parts of the hierarchy. It is possible to set up redirectors as proxy servers, which means data will flow through them, but this is not considered in this work.

3. Testing Method

As stated in the introduction, this paper looks at two different aspects of XRootD; the scalability and overhead. At STFC, these tests were performed on two different systems and using different testing methods. This was necessary since we cannot perform scalability tests on production systems without impacting users, but the test system is not part of large XRootD federation. Both systems used CASTOR [7] as the underlying storage system, accessing data that was on disk, ignoring the tape system.

3.1. Scalability Testing

The goal of the scalability tests was to determine how the software scales with increasing numbers of concurrent requests for data. We explicitly wanted to ensure that the throughput to clients scaled with the storage system and not the XRootD redirector, and to look at resource usage (memory and CPU) as a function of the number of concurrent transactions.

The test system used for scalability tests consisted of entirely local resources, with seven disk servers of varying capacity between 20 and 40 TB. All servers were equipped with 10 Gb Ethernet cards. There was a single XRootD redirector hosted on a physical machine with 4GB RAM and 500GB system drive. The XRootD server itself was configured to use UNIX only authentication. The CASTOR storage system was used for archiving the data. Tests clients were from the RAL batch farm and were submitted to the TORQUE/MAUI queue.

Two tests were performed using the xrootd protocol; reading and writing. For the write tests, jobs were submitted to the batch farm, each of which would initially generate a 2GB file of random binary data. At a pre-set time, the job would then trigger a write to CASTOR using xrdcp (the xrootd copy command). All jobs were scheduled to start this transfer at the same time; if any jobs were not scheduled or took too long to generate the test file, the job was aborted.

The same jobs were submitted with increasing numbers of jobs. Since each job went to a unique client, the client itself did not cause any effects on the scalability. Tests were repeated between 5 and

ten times (time constraints and batch farm usage prevented more repeat tests), with the number of concurrent jobs increasing from 1 to 500.

3.2. Overhead Testing

As previously stated, XRootD is proposed as the infrastructure to support data access at remote sites. However, in order to maintain job efficiencies, it is important that both the network throughput is acceptable and that the architecture itself does not impose significant overheads. If each redirection takes several seconds, after four or five 'hops', this overhead will become a limiting factor in these efficiencies. Unlike the scalability tests above, these tests were based purely on read tests, with data being read by a dedicated client at STFC.

Testing of the overhead was done using the CMS production infrastructure at STFC as the source site of the data. The CMS production infrastructure at RAL as used for this test consisted of 31 disk servers all with 10GB NIC cards, with total capacities varying from 40 to 92GB. Of course the RAL production system is connected to the CMS AAA infrastructure as is shown in Figure 1. As noted previously, regardless of which redirector is contacted, the actual transfer between the storage and the client is performed over the local network. While this largely takes out the effect of the network from the results, the fact that tests were performed on the production infrastructure means real jobs were also accessing data on the disk.

In this case, CMS already provide 256 random files of consistent size for their own load testing purposes and these file were used as the source for each test. The tests performed were simply 100 sequential reads, each of one test file chosen at random. These reads were repeated contacting the local redirector at STFC, the European redirector at Bari, and final the global redirector at CERN. For each test, the total time to perform the transfer was recorded, from the request being made to the completion of the transfer.

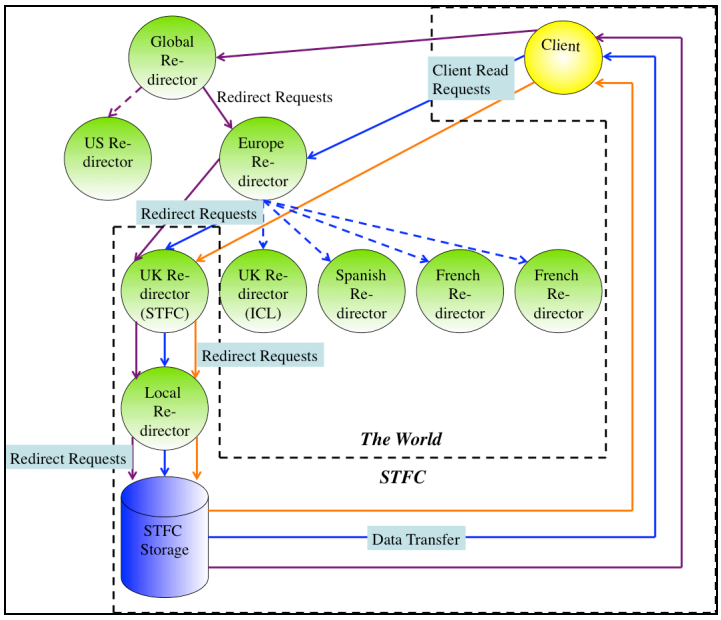

Figure 1: CMS Production Infrastructure used for Testing

4. Results

4.1. Scalability Tests

As stated previously, two aspects of scalability were investigated; verification that throughput to and from the storage system is not affected by use of the *xrdcp* protocol, and that the resource usage by the XRootD redirector scales at best linearly with the number of connections. Resources were monitored

using Ganglia, and monitoring as the number of concurrent transfers increased from 1 to 500 (in irregular steps). During these tests, no increase in CPU or memory usage was observed, and the increase in network packets was consistent with the number of concurrent requests.

Figure 2 shows the aggregate throughput for varying numbers of clients. These results are consistent with rfio [8], the current default protocol used by the CASTOR storage system. Reads rates increase at about 30 MB/sec/client and writes at 26 MB/s/client. The dip in both read and write shows the impact of multiple transfers arriving at the same disk server; in this case the disk server itself becomes the bottleneck to performance.

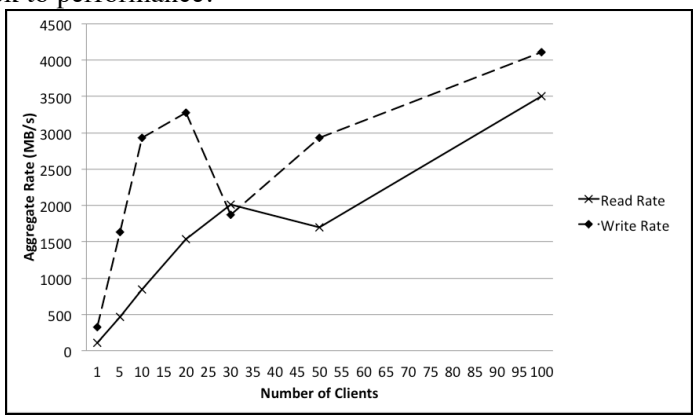

Figure 2: Throughput to Storage System Using XRootD

In terms of resource consumption (memory and CPU), the XRootD redirector did not consume any observable additional resources as the number of clients increased. The additional load was observed on the underlying disk servers. Each connection slightly increased CPU load (<2% per connection), but did increase memory usage by about 500MB per connection. This memory usage clearly limits the number of concurrent connections that should be made if used in this mode. No analysis has yet been made for resource usage when using ROOT analysis framework, into which XRootD has been integrated [9].

4.2. Overhead Tests

A summary of the results obtained in the scalability test in shown in Table 1. Looking at the minimum transfer times, which are results least likely to be affected by network performance, it is clear that each redirector adds an overhead of about 300ms. Even the mean transfer time does not show huge overheads with distance in the hierarchy. However, the maximum times do show a significant increase. While these are outliers in the data, they could become important for sites. The cause for this increase is currently not understood.

Table 1: XRootD Transfer Times Contacting Different Redirectors

	Min. Transfer Time	Mean Transfer Time	Max. Transfer Time
RAL	23.27	26.19	45.92
BARI	23.62	27.16	75.16
CERN	23.97	30.35	84.36

Binning the transfer times into 2-second bins is shown below in Figure 3. As can immediately be seen, for most transfers it is much better to talk to the local XRootD redirector than to start at some other point in the hierarchy. While this is not a great surprise, of more interest is how little overhead redirection puts on the transfer; for the most cases it is 2-3 seconds per redirect. Also of interest in the apparent 'ringing' seen in the transfer times when using the global redirector. The cause of this is also

unclear, but it may be related to the redirector at CERN being composed of multiple instances in a cluster.

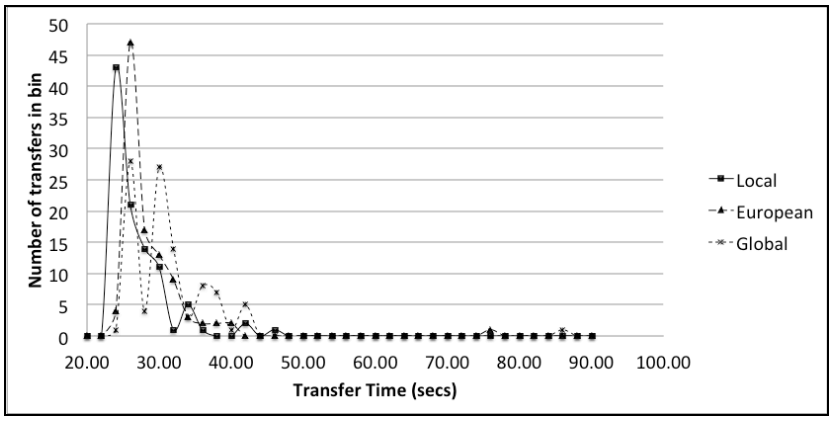

Figure 3: XRootD Transfer Times with Distance

5. Conclusion

Based on the analysis performed, it would appear the XRootD does indeed offer a massively scalable and high performance technology for data access in a distributed environment. It has been demonstrated that, at least using NIS authentication locally, the load put on the redirector is insignificant, making it an ideal candidate to run on virtual machines. If other authentication methods are needed then additional analysis would be required – although this authentication can be pushed down to the storage level itself with each disk server being responsible for its own authentication and this spreading the security load.

Finally, the low overhead is highly attractive. For the SRM, the overhead in accessing an on-line file can be between 5 and 30 seconds. Thus the 2-3 second overhead put on by XRootD is a significant improvement. In addition, the addition of job fallback should see fewer grid jobs failing and thus reducing the overhead of monitoring and resubmission.

References

[1] Boeheim C, Hanushevsky A, Leith D, Melen R, Mount R, Pulliam T and Weeks B 2006 *Scalla: Scalable ClusterArchitecture for Low Latency Access Using xrootd and olbd Servers*. Technical report, Stanford Linear Accelerator Center.
[2] Bauerdick L et al 2012 *J. Phys.: Conf. Ser.* **396** 042009
[3] Grandi C *et al* 2012 *J. Phys.: Conf. Ser.* **396** 032053
[4] Domenici A and Donno F 2009 *Grid Computing* 99-105 (Springer US)
[5] Borthakur D 2008 *HDFS architecture guide*. Hadoop Apache Project. http://hadoop. apache. org/common/docs/current/hdfs_design. pdf.
[6] Weil S A, Brandt S A, Miller E L, Long D and Maltzahn C 2006 *Proceedings of the 7th Symposium on Operating systems design and implementation* 307-320 (USENIX Association)
[7] Baud J P, Couturier B, Curran C, Durand J D, Knezo E, Occhetti S, Barring O 2003 *arXiv preprint cs/0305047*
[8] Foster I, Kohr Jr D, Krishnaiyer R and Mogill J 1997 *Proceedings of the fifth workshop on I/O in parallel and distributed systems* 14-25 (ACM)
[9] Furano F, Hanushevsky A 2010 *J. Phys.: Conf. Ser.* **219** 072005 doi:10.1088/1742-6596/219/7/072005