Performance evaluation of distributed file systems for the phase-II upgrade of the ATLAS experiment at CERN

Adam Abed Abud[1, 2], Fabrice Le Goff[2] and Giuseppe Avolio[2]

[1] University of Pavia (Dipartimento di Fisica, via Bassi 6, I-27100 Pavia, Italy)
[2] CERN (CH-1211 Geneva 23, Switzerland)

E-mail: adam.abed.abud@cern.ch

Abstract.
Over the next few years, the LHC will prepare for the upcoming High-Luminosity upgrade in which it is expected to deliver ten times more pp collisions. This will create a harsher radiation environment and higher detector occupancy. In this context, the ATLAS experiment, one of the general purpose experiments at the LHC, plans substantial upgrades to the detectors and to the trigger system in order to efficiently select events. Similarly, the Data Acquisition System (DAQ) will have to redesign the data-flow architecture to accommodate for the large increase in event and data rates. The Phase-II DAQ design involves a large distributed storage system that buffers data read out from the detector, while a computing farm (Event Filter) analyzes and selects the most interesting events. This system will have to handle 5.2 TB/s of input data for an event rate of 1 MHz and provide access to 3 TB/s of these data to the filtering farm. A possible implementation for such a design is based on distributed file systems (DFS) which are becoming ubiquitous among the big data industry. Features of DFS such as replication strategies and smart placement policies match the distributed nature and the requirements of the new data-flow system. This paper presents an up-to-date performance evaluation of some of the DFS currently available: GlusterFS, HadoopFS and CephFS. After characterization of the future data-flow systems workload, we report on small-scale raw performance and scalability studies. Finally, we conclude on the suitability of such systems to the tight constraints expected for the ATLAS experiment in phase-II and, in general, what benefits the HEP community can take from these storage technologies.

KEYWORDS
Distributed file systems · storage technologies · software-defined storage · ATLAS Data Acquisition · Dataflow

1. Introduction

The Large Hadron Collider (LHC) is a particle accelerator that collides proton bunches at a design center-of-mass energy of 14 TeV and at a rate of 40 MHz. The ATLAS experiment at CERN is one of the four major detectors at the LHC. Its goal is to record and analyze the interactions produced by the pp collisions. The current ATLAS data acquisition (DAQ) system receives data at approximately 150 GB/s and saves only a fraction to permanent storage at

a rate of 1.5 GB/s. Over the next few years, the LHC will prepare for the upcoming High-Luminosity upgrade (HL-LHC) in which it is expected to deliver ten times more pp collisions. Therefore, as a consequence of the harsher environment and the higher detector occupancy the ATLAS experiment is going to upgrade the detectors and redesign completely the trigger and data acquisition system in order to efficiently select events.

2. Phase-II TDAQ Architecture

The baseline architecture for the Phase-II upgrade includes a hardware trigger (L0 trigger) selecting events with a rate up to 1 MHz and within a maximum latency of 10 μs. The Readout system will receive Front-End electronics data at the L0 trigger rate before sending them to the Dataflow system for an estimated data traffic of 5.2 TB/s. The Dataflow system is responsible for buffering, transporting and formatting the event data, acting as an interface between the Readout and the trigger systems. A more refined selection is then applied by the Event Filter (EF) which takes as input the L0 events stored in the Dataflow system. The EF will access 10% of the data at 1 MHz and full event access at 400 kHz. This will lead to a total read bandwidth of 2.6 TB/s. Finally, accepted events are then formatted and sent to permanent storage at a rate of 10 kHz. Figure 1 shows a schematic representation of the main functional blocks of the DAQ architecture for Phase-II [1].

2.1. Dataflow system

The Dataflow system is a key element of the DAQ architecture. It is divided into three components: Event Builder, Storage Handler and Event Aggregator. The Event Builder is the logical interface between the Readout and the Dataflow system. The Storage Handler is a large buffering system which will need to sustain an aggregated I/O throughput of 7.8 TB/s (5.2 TB/s of data input and 2.6 TB/s of data output). A large storage buffer will decouple the Readout from the Event Filter, allowing a trade-off between processing power and storage resources. In fact, the interfill time of the LHC can be can be exploited in order to process some of the recorded events. Moreover, the buffering system can also provide robustness in the Event Filter farm in case of varying operational parameters, such as changes in the prescale factors during data taking that might lead to an increase of data throughput. The last component of the Dataflow system is the Event Aggregator. This system formats and groups the selected events from the EF before sending them to permanent storage.

3. Distributed file systems

Distributed file systems (DFS) represent a class of technologies whose features map the data production and access patterns needed for the Dataflow system. A DFS is a system where files are shared and distributed between multiple nodes in a *hierarchical* and *unified* view [2]. Clients can then access the data as if it was stored on a local machine. In this paper, three different DFSs have been tested: Gluster [3], Hadoop [4] and Ceph [5]. The following sections shortly describe the architecture of the three distributed file systems.

3.1. Gluster

Gluster is an open-source software-defined file storage. It can be deployed on commodity hardware and runs on top of a kernel file-system that supports extended attributes such as xfs, ext4, btrfs. There is no metadata server in the Gluster architecture as the metadata information is stored within the data. The intelligence of the file system resides in the way the data is accessed and retrieved. In fact, Gluster locates files algorithmically using the Elastic Hash Algorithm (EHA). The EHA takes as input the file-name and the file-path and it converts them into a unique value of fixed length. File objects are then distributed across the system

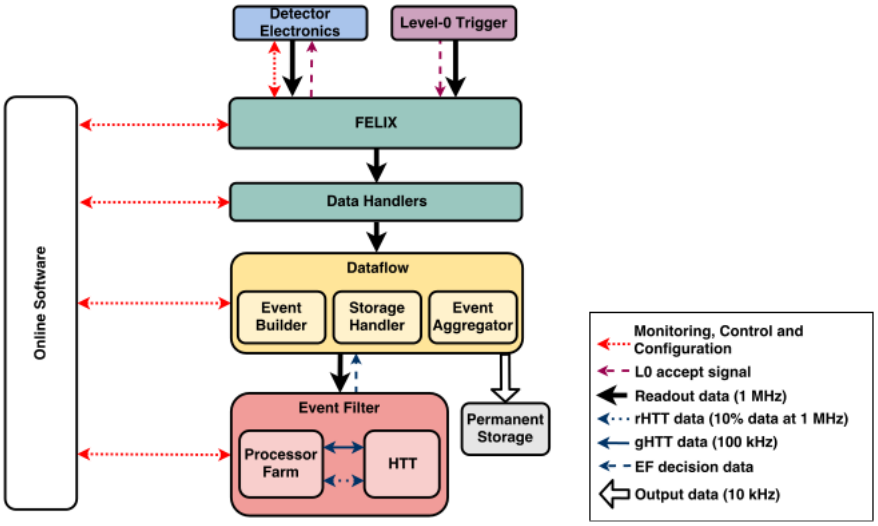

Figure 1. Representation of the DAQ architecture for Phase-II [1].

based on the result of the EHA. Corner cases determine the complexity of Gluster. Events such as addition of new storage devices, file replication or hardware failures are efficiently handled by the system.

3.2. Hadoop

Apache Hadoop (HDFS) is a open-source centralized and distributed file system. This means that metadata is managed by a single server (*namenode)* which is also the master node of the cluster. Data is split into blocks with a configurable block size and distributed across several storage nodes (*datanodes*). The master node is also responsible for managing the namespace of the cluster and for mapping file-names with their corresponding file-blocks. However, a centralized architecture is the limiting factor for file system growth. This is because the metadata needs to be stored in memory and, therefore, the size of the RAM of the master node constitutes a bottleneck. Finally, read and write file access are easily managed by HDFS. The *namenode*, in fact, provides the location of the files to be read or to be written and clients perform I/O operations directly with the responsible storage devices.

3.3. Ceph

Ceph is a distributed file system capable of providing an interface for object, block and file storage [6]. The system is built on top of Rados (*reliable autonomic distributed object store*) which is an object store designed to be reliable and scalable to thousands of nodes. Ceph is also capable of saving block device images as objects within Rados which makes it particularly useful for cloud-based computing systems. In addition, by adding a metadata server it is possible to configure a Ceph cluster to act as a file system. The basic architecture consists of an object storage device (OSD) and a monitoring node. The OSD is the node storing the data while the monitoring node is responsible for handling the status of each OSD and to manage the overall cluster health. Compared to the architecture of Gluster and Hadoop, Ceph sits in the middle. In fact, it makes use of a metadata server which can be distributed across multiple nodes, each responsible for a subset of the namespace. Finally, the placement strategy of Ceph is based on the CRUSH (*controlled replication under scalable hashing*) algorithm [7]. CRUSH uses a combination of hashing functions and metadata operations to distribute objects among storage

devices. It also takes into account the replication factor in order to provide reliability to the system.

4. Performance evaluation

The goals of the testing is to investigate how a DFS performs against a set of controllable parameters such as block-size or access pattern. The size of the cluster for each of the DFS consisted of three nodes each with one 7200 RPM hard drive, while a 1 Gb/s network is used to connect all the nodes together. Replication factors have not been introduced and tests were done bypassing the disk and memory caches as the objective is to understand the raw performance of the systems. The main tool used for testing is the flexible-I/O software [8].

4.1. Results

In this section, the results of the testing are summarized. Figure 2 (left) shows the sequential reading performance as a function of the I/O block-size for the three systems under investigation. The DFSs have the same response for block-sizes ranging from 2 KB up to hundreds of MB. The maximum throughput is constrained by the network which limits the performance to approximately 120 MB/s. Finally, it is worth mentioning that Hadoop was not tested for block-sizes smaller than 256 KB as the system was unstable.

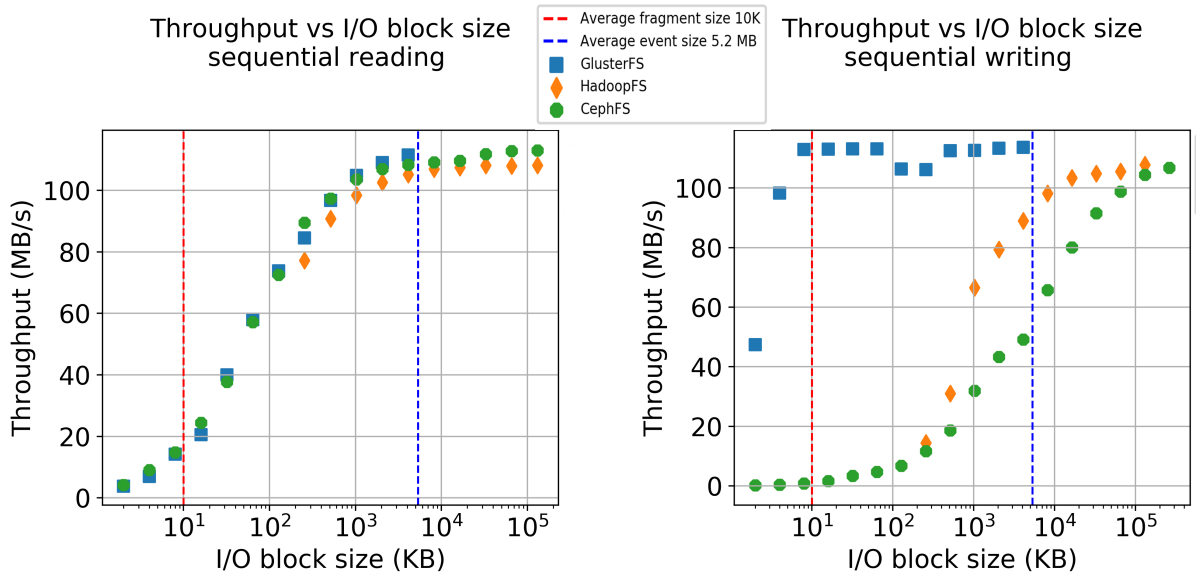

Figure 2. Throughput as a function of I/O block-size for Gluster, Hadoop and Ceph. Sequential reading (left) and sequential writing (right).

Figure 2 (right) shows the performance throughput as a function of the I/O block-size for sequential writing. Gluster reaches the maximum performance for block-sizes in $\mathcal{O}$(10 KB) while Hadoop in $\mathcal{O}$(10 MB) and ceph in $\mathcal{O}$(100 MB). This suggests an advantage in implementing an architecture which does not use any metadata server such as Gluster over architectures which either have a centralized or a distributed metadata system. The impact of the metadata is, in fact, a very important parameter to consider when designing a high-throughput storage system.

In the case of sequential reading the throughput performance is constrained by the network. For both sequential reading and sequential writing there is no limitation due to the performance of the storage devices. This is because the testing is run on a distributed system where the

network is limited at 120 MB/s while the maximum performance of the storage devices are respectively 160 MB/s for sequential reading and 180 MB/s for sequential writing (figure 3).

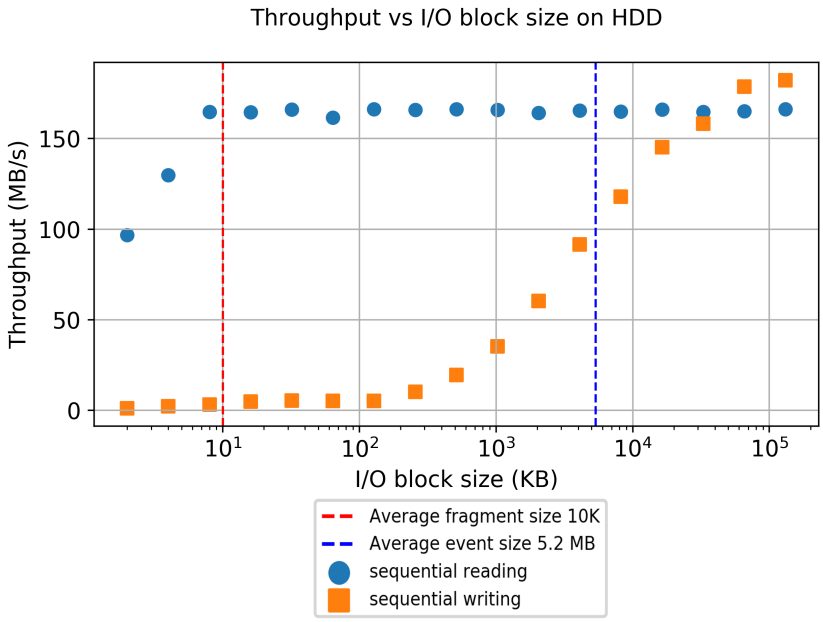

Figure 3. Sequential reading and sequential writing performance of 7200 rpm HDD.

5. Conclusions

In this paper we have shown an initial performance evaluation of three distributed file systems: Gluster, Hadoop and Ceph. Even if the testing was carried on a small setup, the results have shown that the throughput for sequential reading is the same among the three systems. On the other hand, the Gluster file system shows an exceptionally good performance for sequential writing as compared to both Hadoop and Ceph. This suggests that one could profit from an architecture which does not employ metadata servers. In fact, the challenging workload expected for future HEP experiments is in the order of $\mathcal{O}$(10 PB) in size and $\mathcal{O}$(10 TB/s) in throughput. Therefore, it is worth selecting technologies that do not use dedicated metadata servers such as for example key-value stores. Finally, testing with a larger cluster will be needed to further assess the performance, scalability and limitations of DFSs.

References

[1] ATLAS Collaboration, "Technical Design Report for the Phase-II Upgrade of the ATLAS TDAQ System," Tech. Rep. CERN-LHCC-2017-020. ATLAS-TDR-029, CERN, Geneva, Sep 2017.
[2] B. Depardon, G. L. Mahec, and C. Séguin, "Analysis of six distributed file systems," 2013.
[3] Gluster, "https://www.gluster.org/."
[4] Hadoop, "http://hadoop.apache.org."
[5] Ceph, "http://ceph.com."
[6] RedHat, "https://www.redhat.com/en/topics/data-storage."
[7] S. A. Weil, S. A. Brandt, E. L. Miller, and C. Maltzahn, "Crush: Controlled, scalable, decentralized placement of replicated data," in *Proceedings of the 2006 ACM/IEEE Conference on Supercomputing*, SC '06, (New York, NY, USA), ACM, 2006.
[8] FIO, *flexible I/O, [software]*. Available from https://github.com/axboe/fio.