IOPscience iopscience.iop.org

Home Search Collections Journals About Contact us My IOPscience

Using S3 cloud storage with ROOT and CvmFS

This content has been downloaded from IOPscience. Please scroll down to see the full text.

2015 J. Phys.: Conf. Ser. 664 022001

(http://iopscience.iop.org/1742-6596/664/2/022001)

View the table of contents for this issue, or go to the journal homepage for more

Download details:

IP Address: 188.184.3.52

This content was downloaded on 06/01/2016 at 16:05

Please note that terms and conditions apply.

# Using S3 cloud storage with ROOT and CvmFS

**María Arsuaga-Ríos[1], Seppo S Heikkilä[2], Dirk Duellmann[3], René Meusel[4], Jakob Blomer[5], Ben Couturier[6]**

[1−6] CERN (European Organization for Nuclear Research), Geneva, Switzerland

E-mail: [1]`maria.arsuaga.rios@cern.ch`, [2]`seppo.heikkila@cern.ch`, [3]`dirk.duellmann@cern.ch`, [4]`rene.meusel@cern.ch`, [5]`jakob.blomer@cern.ch`, [6]`ben.couturier@cern.ch`

**Abstract.**
Amazon S3 is a widely adopted web API for scalable cloud storage that could also fulfill storage requirements of the high-energy physics community. CERN has been evaluating this option using some key HEP applications such as ROOT and the CernVM filesystem (CvmFS) with S3 back-ends. In this contribution we present an evaluation of two versions of the Huawei UDS storage system stressed with a large number of clients executing HEP software applications. The performance of concurrently storing individual objects is presented alongside with more complex data access patterns as produced by the ROOT data analysis framework. Both Huawei UDS generations show a successful scalability by supporting multiple byte-range requests in contrast with Amazon S3 or Ceph which do not support these commonly used HEP operations. We further report the S3 integration with recent CvmFS versions and summarize the experience with CvmFS/S3 for publishing daily releases of the full LHCb experiment software stack.

## 1. Introduction

The storage and management of the Large Hadron Collider (LHC) data is one of the most crucial and demanding activities in the LHC computing infrastructure at CERN and at the many collaborating sites within the Worldwide LHC Computing Grid (WLCG) [6]. Today, most physics data is still stored with custom storage solutions, which have been developed for this purpose within the High-Energy Physics (HEP) community. Data volume and aggregated speed of data access are increasing demands of the users, therefore CERN and partner institutes are continuously investigating new technological solutions to provide more scalable and performant storage solutions to the user community.

Cloud storage systems have demonstrated good results in terms of scalability and are presented as potentially cost-effective alternatives [9, 11]. They are typically based on a distributed key-value store, and divide the storage namespace up into independent units [5]. In case of Huawei UDS generations, these independent units are known as buckets. The namespace partitioning increases scalability by ensuring that access to one area is unaffected by the activity in other parts of the distributed storage system. In addition, the internal replication and distribution of data replicas over different storage components provides intrinsic fault-tolerance and additional read performance: multiple data copies are available to correct storage media failures and to serve clients. The HTTP-based Amazon Simple Storage Service (S3) API[1]

[1] http://aws.amazon.com/s3

Content from this work may be used under the terms of the Creative Commons Attribution 3.0 licence. Any further distribution of this work must maintain attribution to the author(s) and the title of the work, journal citation and DOI.
Published under licence by IOP Publishing Ltd

has become a de-facto standard among many commercial and open-source storage products for cloud storage access. It may, therefore, become an important integration technology for consistent data access and exchange between science applications and a larger group of sites. One of the advantages of S3 is its open nature that lets the decision between operating a private storage cloud and using commercial cloud services left to the site, based on its size and local cost evaluation. The Universal Distributed Storage (UDS)[2] system developed by Huawei is based on a key-value store with large numbers of inexpensive CPU-disk pairs to form a scalable, redundant storage fabric. Huawei cloud storage systems are also S3 compatible which makes easy to compare with other cloud storages systems. The new UDS generation *Huawei OceanStor V100R002C00* is is recently installed in the CERN data center and it is being evaluated in comparison with the previous generation already installed at CERN and other cloud storage systems such as Amazon S3 or Ceph [3]. The scalability of the first UDS generation with a multi-client S3 benchmark has been evaluated in a previous work [10]. In this paper, a comparison between both UDS generations is carried out in order to verify the scalability and performance obtained by a newer UDS generation: *Huawei OceanStor V100R002C00*. Moreover, the UDS generations performance is evaluated using some key HEP applications such as ROOT [2] and CvmFS [7] with S3 back-ends. A multi-client S3 compatible benchmark is executed in different configurations to measure the aggregated throughput of byte-range operations. The aim of this is to verify the scalability of both UDS generations in order to deal with multi-range/single-range HTTP downloads operations widely used in the HEP community.

This paper is structured as follows. Section 2 describes design and features of both UDS generations. Section 3 studies the scalability of both UDS generations by considering raw upload and download operations. Section 4 analyses the scalability of different data access patterns by performing vector reads with the ROOT framework. Section 5 presents a real CvmFS/S3 application for LHCb software uploads. Finally, Section 6 draws conclusions from the study results.

## 2. Huawei Universal Distributed Storage (UDS) generations

This section gives an overview of the Huawei UDS massive data storage systems. The focus is on the hardware and features of the cloud storage setups that are used at CERN. Two Huawei cloud storage generations are deployed in the CERN data center with a storage capacity of 768 terabytes (TB) and 1.2 petabytes (PB). Both UDS generations are designed for handling large amounts of data with two main functional components:

- Control Nodes (OSC). The OSCs are user-facing front-end nodes which implement the S3 access protocol and delegate the storage functions to storage nodes. OSC nodes are in charge of scheduling, distributing and retrieving the data of the storage nodes.
- Storage Nodes (SOD). The SODs are independent storage nodes, which manage data and metadata on individual hard disks. All stored objects are divided into one megabyte (MB) chunks that are spread and stored on the storage nodes.

The connectivity between these components is provided with three switches. The system is accessed via two 10Gb network connections from a group of CERN-based client nodes. The main hardware features for both Huawei UDS generations are shown in Table 1. The first UDS generation consists of seven control nodes and 384 storage nodes, each storage node being a disk-processor pair comprised of a 2 TB disk coupled to a dedicated ARM processor and memory. In contrast, the second UDS generation presents four control nodes and 300 storage nodes with 4TB disk each.

[2] http://enterprise.huawei.com/ilink/cnenterprise/download/HW 259595

**Table 1.** Hardware specifications for the UDSs deployed at CERN data center.

| Huawei UDS | Storage capacity | OSCs | SODs | Disk capacity per SODs |
|---|---|---|---|---|
| `First generation` | 768 TB | 7 | 384 | 2 TB |
| `Second generation` | 1.2 PB | 4 | 300 | 4 TB |

The S3 API of the second UDS generation supports latest S3 features such as multipart uploads, which is useful when operating with large HEP data files. The evaluated Huawei UDS generations use three replicas to ensure the data availability and reliability. Data replicas are distributed to different storage nodes such that a loss of one or multiple disks will not have impact on data availability. In case of a disk failure, an automated self-healing mechanism ensures that the other storage nodes handle the data on the faulty disk. Corrupted or unavailable data is replaced using the remaining replicas.

## 3. Raw data performance comparison

In this section, a raw data performance comparison between both UDS generations is carried out in order to study the throughput and metadata scalability. These tests are based on the benchmark software from Zotes' et. al. work [10], hosted and distributed among all involved machines via the distributed file system AFS [8].

### *3.1. Throughput performance*

The cloud storage throughput performance is measured using 100 MB files and 2100 buckets in total. The upload and download throughput results show that the available network bandwidth of 20 gigabits could be filled for both UDS generations. Each additional front-end node is able to download around 1000 MB per second (see Figure 1 and Figure 2) or upload roughly 500 MB per second as is shown in Figure 3 and Figure 4. Figure 4 shows the throughput performance for 100 MB uploads; where the new Huawei generation is able to provide slight increase to the maximum upload speed.

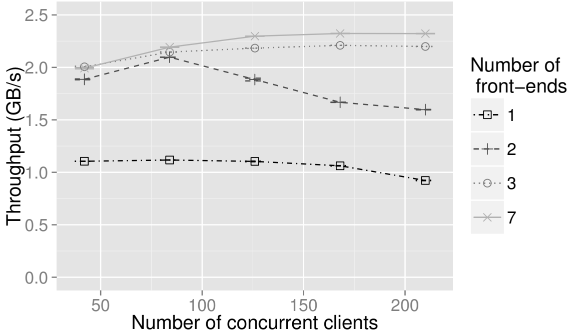

**Figure 1.** Throughput performance for downloads in the first UDS generation.

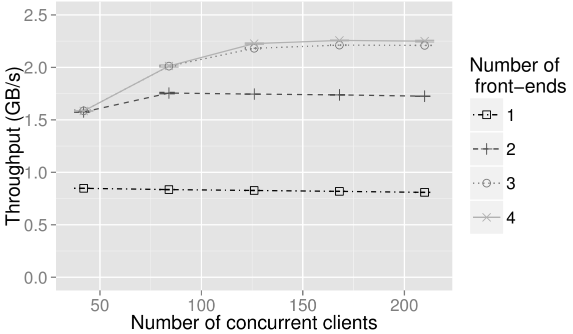

**Figure 2.** Throughput performance for downloads in the second UDS generation.

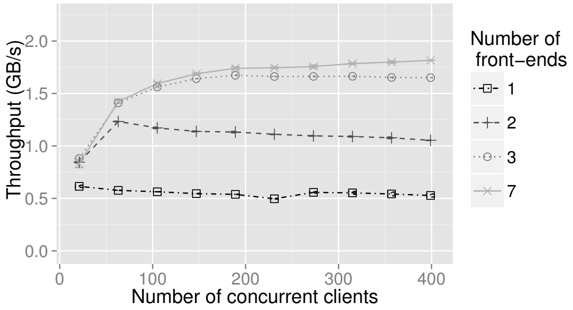

**Figure 3.** Throughput performance for uploads in the first UDS generation.

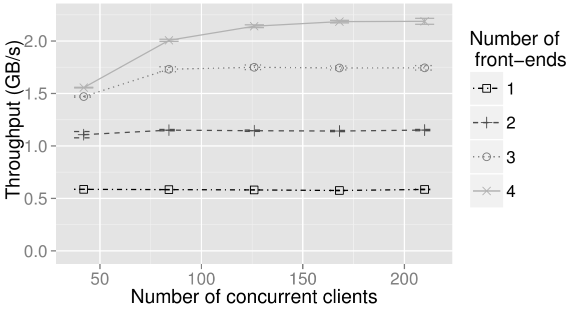

**Figure 4.** Throughput performance for uploads in the second UDS generation.

*3.2. Metadata performance*

The metadata performance is evaluated using 4 kB files. Each front-end node adds linearly around 3500 files per second to the total download rate for both UDS generations, see Figure 5 and Figure 6. Simply adding more front-end nodes could likely further increase the achieved maximum 4 kB download performance. In contrast, the front-ends do not present a limiting factor with 4 kB uploads as it is shown in Figure 7 and Figure 8.

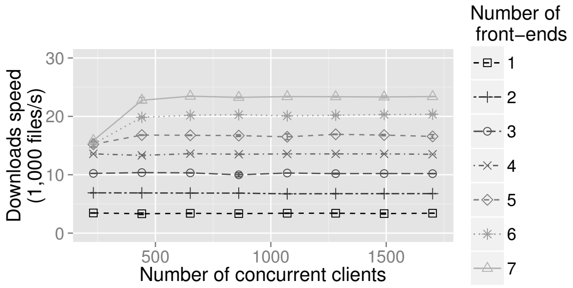

**Figure 5.** Metadata download speed in the first UDS generation.

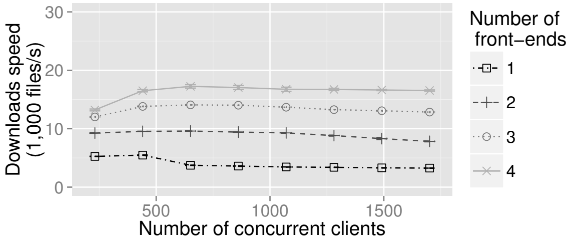

**Figure 6.** Metadata download speed in the second UDS generation.

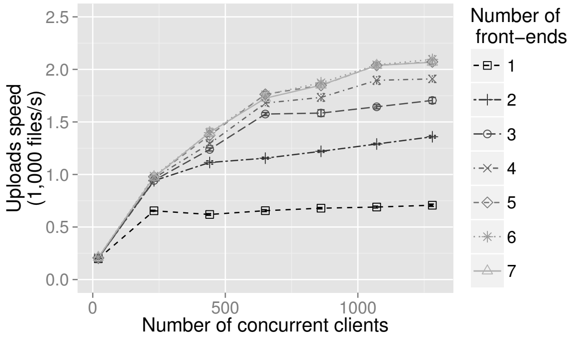

**Figure 7.** Metadata upload speed in the first UDS generation.

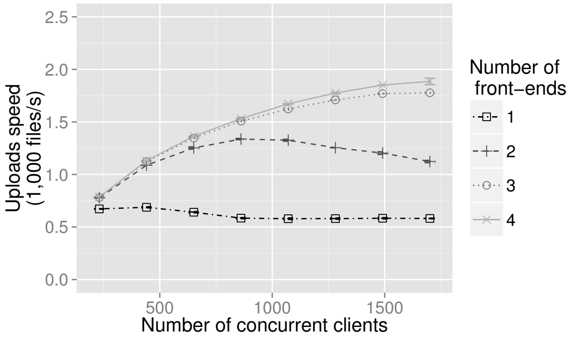

**Figure 8.** Metadata upload speed in the second UDS generation.

## 4. S3 Data access patterns with ROOT data analysis framework

ROOT [2] is the principal framework for HEP data processing. Every day, thousands of physicists use ROOT applications to analyze their data or to perform simulations. ROOT files follow a tree structure and it can be arbitrary complex, because it could contain variables (column headers) or branches, which are more complex objects, including other trees. A variable is always the end point of a branch. Usually, trees are split into branches to benefit from the caching mechanisms, because the tree structure is more complex than a table and the splitting level can be adjusted to match the user's needs. For this paper, a multi-client ROOT benchmark is implemented to offer a full parametric configuration by allowing different access patterns per volume size or entries number to ROOT files. This benchmark allows the use of heterogeneous clients, i.e. physical and/or virtual machines with different features, because it measures the aggregated throughput until a given deadline. Moreover, it offers the configuration of different ROOT parameters such as the number of entries to read, the selection of branches, and the access protocol to the cloud storage such as xroot [1] or S3, by using the davix plugin [4] for the last one.

A ROOT file, which analyses the W and Z bosons with 11918 entries and 5860 branches, is selected from the ATLAS experiment in order to execute multiple vector reads. Two use cases are considered for each access pattern test. In the first use case, only one bucket and one file are accessed by many concurrent clients. The results of this use case are only shown for the first UDS generation, because the second UDS generation was not expected to scale with only one bucket. A second use case considers 64 buckets, which is the same number of machines that builds up the deployed client cluster, and each bucket contains one copy of the ATLAS ROOT file aforementioned. Thus, each machine has assigned one bucket to access. The aim of these tests is to study the scalability of data reads with different access patterns used in a typical data analysis execution with the ROOT framework. Thanks to these tests, it is demonstrated that only UDS generations support multi-range get requests operations, which are commonly used in HEP analysis, when other storage systems such as Amazon S3 and Ceph do not support them.

### *4.1. Vector reads by volume size*

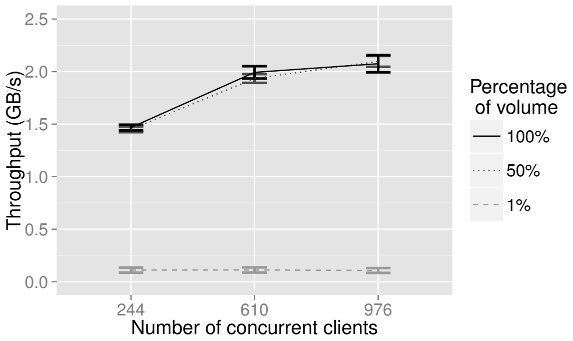

**Figure 9.** Scalability of different volume reads through 1 bucket in first UDS generation.

This subsection presents the scalability study of both UDS generations by reading different percentages of volume of the ATLAS file. In order to consider different percentages of volume, a branch selection process is performed. This branch selection process consists of selecting the branches from an ordered list from the smallest branch to the biggest one, until the percentage of volume to test is achieved. In this evaluation, all the entries are read for each percentage of volume. In the first use case, when only one bucket is accessed from all the concurrent clients, results show that the full bandwidth is reached when reading the 100% and the 50% of volume of the ATLAS file, see Figure 9. The throughput obtained when reading 1% of volume is decreasing because the cloud storage becomes over-stressed. In the case of the 64 buckets, Figure 10 and Figure 11, both UDS generations achieve the full network bandwidth for the same percentage of volume as in the previous case (100% and 50%). As expected, the 1% percent of volume reads scale up until too many clients access to the same small amount of data.

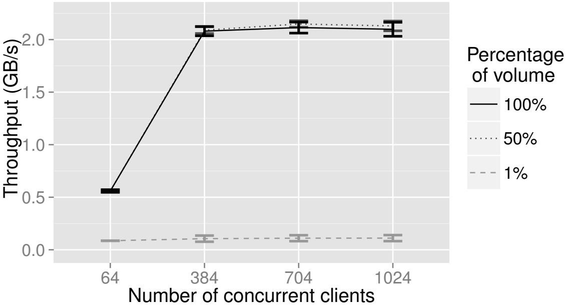

**Figure 10.** Scalability of different volume reads through 64 buckets in first UDS generation.

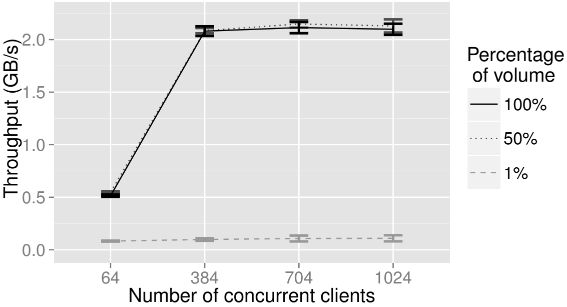

**Figure 11.** Scalability of different volume reads through 64 buckets in second UDS generation.

### *4.2. Vector reads by entries number*

In this subsection, all branches are selected in order to focus on different percentage of entries read for both use cases: 1 bucket and 64 buckets. The entries selection process is random until the number of entries accomplishes the percentage of entries to test. In addition, this random selection process allows the performance evaluation of the UDS generations with sparse accesses. Table 2 shows the different data volume read for each sparse access and the number or read calls without counting the cache calls.

**Table 2.** Access details for different sparse entry access.

| Percentage of entries | Total entries | Read calls | Volume (MB) |
|---|---|---|---|
| 100% | 11918 | 23 | 758.42 MB |
| 0.1% | 12 | 12 | 320.92 MB |
| 0.001% | 1 | 5 | 42.08 MB |

Figure 12 shows the results for the fist use case, when one bucket is accessed from all clients. The first UDS generation successfully scales until it reaches the full network bandwidth when reading 100% and 0.1% of entries. In case of 0.001% of entries, the bucket is over-stressed due to too many clients are accessing to the same small amount of data. Figure 13 and Figure 14 show the throughput obtained by accessing 64 buckets. Both UDS generations successfully scale until they reach the full network bandwidth. These two use cases consider the accesses to the 100% of data volume by selecting the 100% of branches from the ATLAS file. The same evaluation has been carried out by selecting set of branches that compose 50% and 1% of the file volume with the same range of entry accesses and the results present again that both UDS generations scale in the same way as Figure 13 and Figure 14.

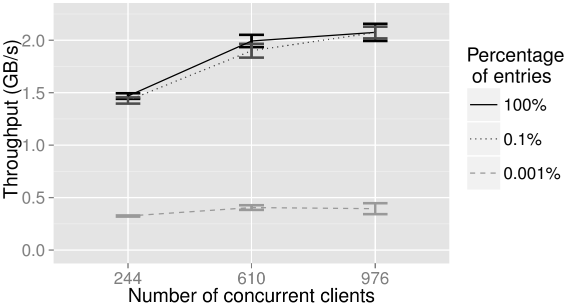

**Figure 12.** Scalability of sparse entry accesses through one bucket in first UDS generation.

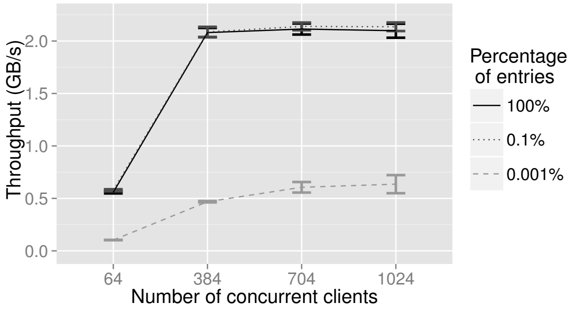

**Figure 13.** Scalability of sparse entry accesses through 64 buckets in first UDS generation.

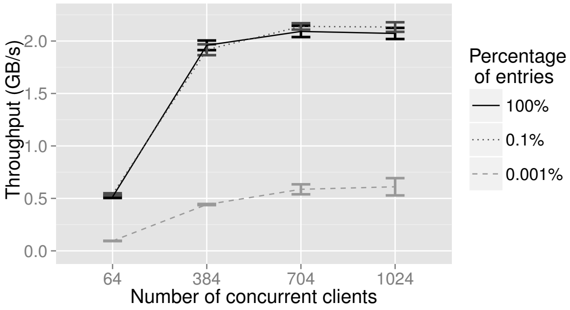

**Figure 14.** Scalability of sparse entry accesses through 64 buckets in second UDS generation.

## 5. S3 Integration and test deployment in CvmFS

In this section, we evaluate the ability of cloud storage to work as a back-end for a CvmFS (CernVM File System) while it is storing real LCHb experiment software. The CvmFS is a read-only cached file system optimised to deliver experiment software in a fast, scalable and reliable way [7]. It has been widely used in WLCG since 2012, but is also now getting users from outside of the HEP community. In a typical use case the users mount CvmFS repositories and files get downloaded transparently when needed. One of the CvmFS challenges is to be able to publish new software as fast as possible and to enable these files to be accessed through HTTP protocol. The HTTP-based S3 API is for this reason inherently compatible back-end storage interface for the CvmFS. The CvmFS with S3 storage back-end was tested together with the LHCb experiment by publishing daily the latest version of the LHCb experiment's software stack. The test was run 110 days and in average 162 ± 28 thousand new files occupying 7.45 ± 3.78 GB of data was uploaded daily to the Huawei cloud storage through the S3 API. The difference in the deviations is because some of the binary files do not always change. The repository size growth is shown in Figure 15. The CvmFS data was stored first in first generation UDS (stratum 0) and replicated based on availability to the second generation UDS (stratum 1). The S3 storage back-end operated without any anomalies. The flat part between 22th day and 25th day was due to expired repository signature, which is not related to the S3 functionality. In the end of the test, the CvmFS repository contained over 18 million unique files occupying in total over 880 GB of storage space. The CvmFS software has also garbage collection feature, which can be used to delete old data in order to free disk space. The S3 back-end ability was evaluated to

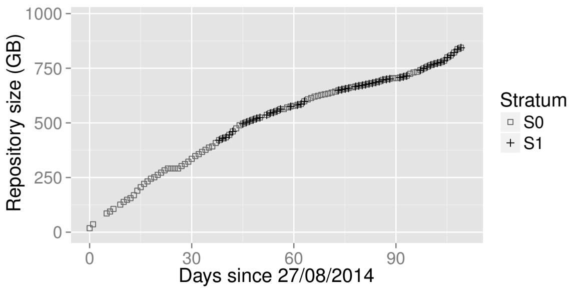

**Figure 15.** The CvmFS repository size growth with the first UDS generation (S0) and the second UDS generation (S1).

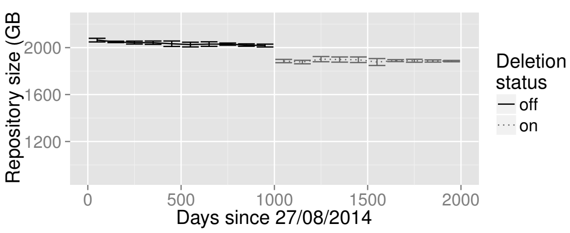

**Figure 16.** Maximum upload performance before and during a deletion process.

support this feature by deleting files with 400 parallel requests while simultaneously uploading files with the maximum speed. The achieved upload performance during deletion process is shown in Figure 16. The test was done only with the second UDS generation, because the first generation does not support large scale deletes properly. The delete requests decrease the maximum upload speed only slightly, which is expected because the cloud storage uses part of its resources to serve the delete requests.

## 6. Conclusions

In this paper, we evaluate the recent UDS version V100R002C00 focusing on scalability in realistic HEP applications such as ROOT analysis and software distribution via CvmFS/S3. Results show that both Huawei storage systems fill the 20 Gigabit network bandwidth by obtaining a successful scalability regarding throughput and metadata performance measurements. Moreover, we use the ROOT framework to simulate end user analysis access which is often characterised by sparse, random access. During this evaluation, we discover that Amazon S3 and Ceph do not support the multi-range HTTP requests that are commonly used in HEP analysis, when both UDS generations support them and even reach the full network bandwidth in all cases. Both Huawei cloud storage systems have been demonstrated to function as expected as back-end for a large scale software repository hosting nightly builds of the LHCb experiment software. The deployment has been stable over a period of more than 15 weeks, showed the expected performance and has allowed to reach a repository size, which is one order larger in volume than all traditional production deployments together. This activity has allowed us to confirm both the successful integration of S3-support into the CvmFS software stack and the stable real-life deployment of a cloud storage system in HEP context.

## Acknowledgments

The authors would like to thank the team of engineers from Huawei, who helped to setup and maintain the UDS systems at CERN.

## References

[1] Dorigo A., Elmer P., Furano F., and Hanushevsky A. XROOTD - a highly scalable architecture for data access. *WSEAS Transactions on Computers*, 4(4):348–353, April 2005.

[2] Naumann A. and Brun R. ROOT - a C++ framework for petabyte data storage, statistical analysis and visualization. *Computer Physics Communications*, 180(12):2499–2512, 2009.

[3] Weil S. A. *Ceph: reliable, scalable, and high-performance distributed storage.* PhD thesis, University of California Santa Cruz, 2007.

[4] Furano F., Devresse A., O. Keeble, Hellmich M., and Ayllón A. Towards an http ecosystem for hep data access. In *Journal of Physics: Conference Series*, volume 513, pages 032–034. IOP Publishing, 2014.

[5] DeCandia G., Hastorun D., Jampani M., Kakulapati G., Lakshman A., Pilchin A., Sivasubramanian S., Vosshall P., and Vogels W. Dynamo: amazons highly available key-value store. *ACM SIGOPS Operating Systems Review*, 41(6):205–220, 2007.

[6] Bird I., K. Bos, N. Brook, et al. LHC computing grid: Technical design report. Technical Report LCG-TDR-001, CERN, 2005.

[7] Blomer J, Aguado-Sánchez C, Buncic P, and Harutyunyan A. Distributing LHC application software and conditions databases using the CernVM file system. *Journal of Physics: Conference Series*, 331(4), 2011.

[8] Morris J., Satyanarayanan M., Conner M., Howard J., Rosenthal D., and Smith F. Andrew: A distributed personal computing environment. *Communications of the ACM*, 29(3):184–201, 1986.

[9] Wu J., Ping L., Ge X., Wang Y., and Fu J. Cloud storage as the infrastructure of cloud computing. In *Intelligent Computing and Cognitive Informatics (ICICCI), 2010 International Conference on*, pages 380–383. IEEE, 2010.

[10] Zotes Resines M., Heikkila S. S., Duellmann D., Adde G., Toebbicke R., Hughes J., and Wang L. Evaluation of the huawei uds cloud storage system for cern specific data. *Journal of Physics: Conference Series*, 513(4):042024, 2014.

[11] Benedict S. Performance issues and performance analysis tools for HPC cloud applications: a survey. *Computing*, 95(2), 2013.