Software installation and condition data distribution via CernVM File System in ATLAS

A De Salvo¹, A De Silva², D Benjamin³, J Blomer⁴, P Buncic⁴, A Harutyunyan⁴, A. Undrus⁵, Y Yao⁶
on behalf of the ATLAS collaboration

¹ Istituto Nazionale di Fisica Nucleare (INFN) Sez. Di Roma1 (IT)
² TRIUMF (CA)
³ Duke University (US)
⁴ CERN (CH)
⁵ Brookhaven National Laboratory (US)
⁶ LBNL (US)

Abstract. The ATLAS collaboration is managing one of the largest collections of software among the High Energy Physics experiments. Traditionally, this software has been distributed via rpm or pacman packages, and has been installed in every site and user's machine, using more space than needed since the releases share common files but are installed in their own trees. As soon as the software has grown in size and number of releases, this approach showed its limits, in terms of manageability, used disk space and performance. The adopted solution is based on the CernVM File System, a fuse-based HTTP, read-only filesystem which guarantees file de-duplication, on-demand file transfer with caching, scalability and performance. Here we describe the ATLAS experience in setting up the CVMFS facility and putting it into production, for different type of use-cases, ranging from single users' machines up to large data centers, for both software and conditions data. The performance of CernVM-FS, both with software and condition data access, will be shown, comparing with other filesystems currently in use by the collaboration.

1. Introduction

The ATLAS Experiment at LHC (CERN) [1] is one of the largest collaborative efforts ever undertaken in the physical sciences, with more than 160 millions of readout channels. About 3000 physicists participate, from more than 150 universities and laboratories in more than 30 countries. The detector itself is also one of the largest in High Energy Physics. The software and the conditions data produced by the detector are correspondingly large in size.

Historically the software releases and tools were deployed in the sites in shared-disk facilities, installing the files only once and making them available to all the machines of the clusters. The conditions data were instead distributed to the Grid-enabled storage systems of the sites, in privileged areas, then accessing the files via the Grid client tools.

As the software started growing in size and number of releases, this method of distribution showed its limitations, in terms of manageability, scalability, performance and used disk space in the
sites. In fact, several releases have common files, but it’s not trivial to share the commonalities across them. Moreover, not all the files are regularly used, but they need to be installed in the local software repository of the sites in order to be ready to use them promptly when needed.

The CernVM-FS project is able to solve this problem, providing an HTTP-based distributed filesystem, with file de-duplication facilities and on-demand retrieval of the used files only. The manageability aspect is also improving by using CernVM-FS, since we move from a situation where all the files have to be installed in all the sites to a scenario where all the files are installed only in the central repository, then distributed to the destinations only when accessed by the clients.

This paper explains the ATLAS setup with the CernVM-FS facility for both software and conditions data. The performance of CernVM-FS, both with software and condition data access, will be shown, comparing with other filesystems currently in use by the collaboration.

2. CernVM-FS

The CernVM File System (CernVM-FS) [2] is a file system used by various HEP experiments for the access and on-demand delivery of software stacks for data analysis, reconstruction, and simulation. It consists of web servers and web caches for data distribution to the CernVM-FS clients that provide a POSIX compliant read-only file system on the worker nodes. CernVM-FS leverages the use of content-addressable storage [3].

Traditional file systems store directory trees as directed acyclic graphs of path names and inodes. Systems using content-addressable storage, on the other hand, maintain a set of files having the files’ cryptographic content hash [4] as keys. In CernVM-FS, the directory tree structure is stored on top of such a set of files in the form of "file catalogs". These file catalogs are SQLite database files that map path names to hash keys. The set of files together with the file catalogs is called a "repository". In order to modify a repository, new and updated files have to be pre-processed into content-addressable storage before they are accessible by CernVM-FS clients. This additional effort is worthwhile due to several advantages of content-addressable storage:

- Files with the same content are automatically not duplicated because they have the same content hash. Previous studies have shown that typically less than 20% of files of a new ATLAS software release are unique (cf. Table 1).
- Data integrity is trivial to verify by re-calculating the cryptographic content hash. This is crucial when executable files are transferred via unsecured and possibly unreliable network channels.
- Files that carry the name of their cryptographic content hash are immutable. Hence they are easy to cache as there is no expiry policy required.
- There is an inherent support for versioning and file system snapshots. A new snapshot is defined by a new file catalog that can be atomically switched-to on the clients.

| Repository      | Traditional File System | CernVM-FS |
|-----------------|-------------------------|-----------|
|                 | Volume [GB] | # Objects [Mio.] | Volume [GB] | # Objects [Mio.] |
| Stable releases | 921         | 22              | 115         | 1.8              |
| Nightly builds  | 660         | 13              | 759         | 19               |
| Conditions data | 440         | 0.01            | 420         | 0.008            |

Table 1 The file compression and file de-duplication built into CernVM-FS is most efficient for the stable-releases repository. The other repositories show two types of degradation: conditions data are naturally unique and stored in ROOT [5] files, which are already compressed. Nightly builds are replaced every night but on CernVM-FS, being a versioned storage, they are not physically removed. Not being a critical repository, however, the nightly builds repository can simply be re-created.
CernVM-FS uses HTTP as its transfer protocol. This facilitates the data access from behind the firewalls as typically used on Grid sites and cloud infrastructures. Any standard HTTP proxy can be used to setup a shared data cache for a computer cluster.

CernVM-FS transfers the files and file catalogs only on request, i.e. on an open() or stat() call. Once transferred, files and file catalogs are locally cached. As for any given job only a tiny fraction of all the files is required, the entire history of software and conditions data can stay accessible without overloading web servers and web proxies. Having 10GB of local cache per worker node is sufficient for a cache hit rate of over 99%.

This is crucial for the effective use of virtual machines [6] but it turns out to be useful for physical Grid worker nodes as well [7].

3. CernVM-FS Infrastructure for ATLAS

The ATLAS experiment maintains three CernVM-FS repositories, for stable software releases, for conditions data, and for nightly builds. The nightly builds are the daily builds that reflect the current state of the source code, provided by the ATLAS Nightly Build System [8]. Each of these repositories is a subject to daily changes.

Figure 1 shows the infrastructure components. The infrastructure is built using standard components, namely the Apache web server and the Squid web proxy.

![CernVM-FS infrastructure components for ATLAS](image)

**Figure 1** CernVM-FS infrastructure components for ATLAS. Except for the installation boxes all components of the infrastructure are shared among multiple HEP experiments and replicated for maximal resilience. The Stratum 1 web services and the Squid caches are geographically distributed. The Squid caches can be shared with Frontier caches. For small and non-critical repositories, the installation box, the master storage, and the Stratum 0 web server can be collapsed onto a single host. For non-critical repositories, the replication to Stratum 1 services can be skipped.

Changes to CernVM-FS repository are staged on an "installation box" using a POSIX filesystem interface. There is a dedicated installation box for each repository. CernVM-FS tools installed on these machines are used to commit changes, process new and updated files, and to publish new repository snapshots. The machines act as a pure interface to the master storage and can be safely re-installed if required.
Two NetApp storage appliances in high-availability setup comprise the master storage. The Stratum 0 web service provides the entry point to access data via HTTP. It consists of two load-balanced Apache servers. Hourly replication jobs synchronize the Stratum 0 web service with the Stratum 1 web services. Stratum 1 web services are operated at CERN, BNL, RAL, FermiLab, and ASGC LHC Tier 1 centers. They consist of an Apache web server and two load-balanced Squid frontend caches each. CernVM-FS clients can connect to any of the Stratum 1 servers. They automatically switch to another Stratum 1 service in case of failures. For computer clusters, CernVM-FS clients connect through a cluster-local web proxy. In case of ATLAS, CernVM-FS mostly piggybacks on the already deployed network of Squid caches of the Frontier service [9]. Squid is particular well suited for scalable web proxy services due to its co-operative cache that also exploits the content of neighbour Squids when serving a request [10].

Due to the local Squid proxies and the client caches, Stratum 1 services only have to serve few MB/s each, serving tens of thousands of worker nodes overall. Although originally developed for small files as they appear in repositories containing software stacks, Figure 2 shows that the file size distribution differs significantly for conditions data. Files at the gigabyte range and above possibly impose a threat to the performance because CernVM-FS uses whole file caching and web proxies are optimized for small files. The ATLAS condition database files, however, have not caused a degradation of the service so far. Future work can extend CernVM-FS to store and deliver such large files in smaller chunks.

![Figure 2 Cumulative file size distribution of ATLAS conditions data compared to ATLAS software release files. The file sizes grow more rapidly for conditions data. For conditions data, 50% of all files are smaller than 1 MB, almost 20% of the files are larger than 100 MB.](image)

4. ATLAS software on CernVM-FS

The ATLAS software deployed in CernVM-FS is structured in 4 different sections:

1. stable software releases;
2. nightly software releases;
3. per-site, local settings;
4. software tools.

These items are grouped into 2 different repositories, one for nightly releases and the other for the rest, due to the different usage patterns and refresh cycles. An additional repository is used for conditions data.
4.1. Stable releases

Stable releases of the ATLAS software are stored in a dedicated tree in CernVM-FS. The same tree is used by the software tools, when they need to access part of the main ATLAS software.

The deployment tools are common for cvmfs and shared File System facilities, making the procedure homogeneous and less error prone, gaining from the experience of the past years when installing in shared File Systems in the Grid sites.

The installation agent, for now, is manually activated by the release manager whenever a new release is available, but can be already automatized by running periodically.

The installation parameters are kept in the Installation DB, and dynamically queried by the deployment agent when started. The Installation DB is part of the ATLAS Installation System [11], and updated by the Grid installation team.

Once the releases are installed in CernVM-FS, a validation job is sent to each site, in order to check both the correct CernVM-FS setup at destination and that the configuration of the nodes of the sites is suitable to run the specific software release. Once a release is validated, a tag is added to the site resources, meaning they are suitable to be used for with the given release.

The Installation System can handle both CernVM-FS and non-CernVM-FS sites transparently, while performing different actions:

1. non- CernVM-FS sites: installation, validation and tagging;
2. CernVM-FS sites: validation and tagging only.

Currently, there are more than 600 software releases deployed in CernVM-FS.

4.2. Nightly releases

The ATLAS Nightly Build System maintains about 50 branches of the multi-platform releases based on the recent versions of software packages. Code developers use nightly releases for testing new packages, verification of patches to existing software, and migration to new platforms, compilers, and external software versions. ATLAS nightly CVMFS server provides about 15 widely used nightly branches. Each branch is represented by 7 latest nightly releases. CVMFS installations are fully automatic and tightly synchronized with Nightly Build System processes. The most recent nightly releases are uploaded daily in few hours after a build completion. The status of CVMFS installation is displayed on the Nightly System web pages. The Nightly repository is not connected to the Stratum 0 so that users need to establish a direct connection. By connecting directly, the nightlies branches of the software are available sooner because they avoid the delays associated with hourly replication to the Stratum 1 servers.

4.3. Local settings

Each site needs to have a few parameters set before running jobs. In particular they need to know the correct endpoints of services like Frontier Squids to access the conditions DB, the Data Management site name and any override to system libraries needed to run the ATLAS software.

To set the local parameters via CVMFS we currently use a hybrid system, where the generic values are kept in CernVM-FS, with a possible override at the site level from a location pointed by an environment variable, defined by the site administrators to point to a local shared disk of the cluster.
Periodic installation jobs write the per-site configuration in the shared local area of the sites, if provided, defining the correct environment to run the ATLAS tasks.

Recently we started testing the fully enabled CernVM-FS option, where the local parameters are dynamically retrieved by the ATLAS Grid Information System (AGIS) [12]. In this scenario the sites will not need anymore to provide any shared filesystem, thus reducing the overall complexity of the system.

4.4. Software tools
The availability of Tier3 software tools from CernVM-FS is invaluable for the Physicist end-user to access Grid Resources and for analysis activities on locally managed resources, which can range from a Grid-enabled data center to a user desktop or a CernVM [13] Virtual Machine. The software, comprising multiple versions of Grid Middleware, front-end Grid job submission tools, analysis frameworks, compilers, data management software and a User Interface with diagnostics tools, are first tested in an integrated environment and then installed on the CernVM-FS server by the same mechanism that allows local disk installation on non-privileged Tier3 accounts. The User Interface integrates the Tier3 software with the Stable and Nightly ATLAS releases and the conditions data, the latter two residing on different CernVM-FS repositories. Site-specific overrides are configurable on a local directory but this will be set through AGIS in the future. When the Tier3 software User Interface is setup from CernVM-FS the environment is consistent, tested and homogeneous, together with the available diagnostics tools: this significantly simplifies user support.

5. Conditions data
ATLAS conditions data refers to the time varying ‘non-event’ data needed to operate and debug the ATLAS detector, perform reconstruction of event data and subsequent analysis. The conditions data contains all sorts of slowly evolving data including detector alignment, calibration, monitoring, and slow controls data. Most of the conditions data is stored within an Oracle database at CERN and accessed remotely using the Frontier service [9]. Some of the data is stored in external files due to the complexity and size of the data objects. These files are then stored in ATLAS dataset sets and are made available through the ATLAS distributed data management system or AFS file system.

Since the ATLAS conditions data are read only by design, CernVM-FS provides an ideal solution for distributing the files that would otherwise have to be distributed through the ATLAS distributed data management system. CernVM-FS allows simplifying the large computing sites by reducing the need for specialized storage areas. The smaller computing resources (like local clusters, desktop or laptops) also benefit from the adoption of CernVM-FS.

The conditions data are installed on the dedicated release manager machine from the same datasets that are available through ATLAS distributed data management system. All of these files are then published to the Stratum 1 servers.
6. Performance

Some measurements on CernVM-FS access times against the most used file systems in ATLAS have shown that the performance of CernVM-FS is either comparable or better performing when the system caches are filled with pre-loaded data.

In Figure 3 we report the results of a metadata-intensive test performed at CNAF, Italy, as comparison of performance between CernVM-FS and cNFS over gpfs. The tests are the equivalent of the operations that are performed by each ATLAS job in real life, using a maximum of 24 concurrent jobs in a single machine. The average access times with CernVM-FS are ~16s when the cache is loaded and ~37s when the cache is empty, to be compared with the cNFS values of ~19s and ~49s respectively with page cache on and off. Therefore CernVM-FS is generally even faster than standard distributed file systems.

According to measurements reported in [13], CVMFS also performs consistently better than AFS in wide area networks with the additional bonus that the system can work in disconnected mode.

![Figure 3](image)

**Figure 3.** Comparison of the file access times between CernVM-FS and cNFS over gpfs. The measurements have been performed at CNAF, Italy on metadata-intensive operations. The performance of CernVM-FS is comparable with the one of cNFS. The average access times with CernVM-FS are ~16s when the cache is loaded and ~37s when the cache is empty, to be compared with the cNFS values of ~19s and ~49s respectively with page cache on and off.

---

1 Courtesy of V. Vagnoni, CNAF (Italy)
7. Migration to CernVM-FS in ATLAS

At the time of this writing, more than 60% of the total ATLAS resources have already adopted CernVM-FS as software and condition distribution layer (Figure 4). The switch to the new system did require a short downtime of the sites or no downtime at all, depending on the preferred migration model chosen by the site. In the worst case, a full revalidation of a site using CernVM-FS needs about 24 hours, when sufficient job slots are available for parallel validation tasks, to be compared with the less than 96 hours in case of a non-CernVM-FS equipped site.

![Figure 4.](image-url) Different filesystems used for the ATLAS software in the sites. ARC sites are not included in the plot, however 12 of them already moved to CernVM-FS, while 1 is still using NFS.

8. Conclusions

CernVM-FS has proven to be a robust and effective solution for the ATLAS deployment needs. The testing phase has been extremely fast, less than 6 months, and the switch from testing to production level quick and successful, thanks to the features of the project and the support of the developers.

More than 60% of the whole collaboration resources have already switched to use CernVM-FS and we target to have all the resources migrated by the end of 2012.

9. References

[1] The ATLAS Collaboration, 2008, The ATLAS Experiment at the CERN Large Hadron Collider, JINST 3 S08003

[2] Blomer J, Aguado-Sánchez C, Buncic P and Harutyunyan A, 2011, Distributing LHC application software and conditions databases using the CernVM file system, J. Phys.: Conf. Ser. 331

[3] Tolia N, Kozuc M, Satyanarayana M, Karp B, Bressou T and Perrig A, 2003, Opportunistic Use of Content Addressable Storage for Distributed File Systems, Proc. of the USENIX Annual Technical Conference

[4] Bakhtiari S, Safavi-naini R and Pieprzyk J, 1995, Cryptographic Hash Functions: A Survey, cryptohash95

[5] Naumann A et al., 2009, ROOT: A C++ framework for petabyte data storage, statistical analysis and visualization, Computer Physics Communications, 180(12)
[6] Blomer J, Aguado-Sánchez C, Buncic P, Harutyunyan and Mudrinic M, 2011, A practical approach to virtualization in HEP, *European Physical Journal Plus*, **126**(1)

[7] Lanciotti E, Merino G, Bria A and Blomer J, 2011, An alternative model to distribute VO specific software to WLCG sites: a prototype at PIC based on CernVM file system, *J. Phys.: Conf. Ser.* **331**

[8] Luehring F, Obreshkov E, Quarrie D, Rybkine G and Undrus A, 2010, Organization and management of ATLAS nightly builds, *J. Phys.: Conf. Ser.* **219** 042045 (http://iopscience.iop.org/1742-6596/219/4/042045)

[9] Dykstra D and Lueking L, 2010, Greatly improved cache update times for conditions data with Frontier/Squid, *J. Phys.: Conf. Ser.* **219**

[10] Fan L, Pei C and Almeida J, 2000, Summary Cache: A Scalable Wide-Area Web Cache Sharing Protocol, *IEEE/ACM Transactions on Networking*

[11] De Salvo A, Barchiesi B, Gnanvo K, Gwilliam C, Kennedy J, Krobath G, Olszewski A and Rybkine G, 2008, The ATLAS software installation system for LCG/EGEE, *J. Phys.: Conf. Ser.* **119** 052013

[12] Anisenkov A, Klimentov A, Kuskov R and Wenaus T, 2011, ATLAS Grid Information System, *J. Phys.: Conf. Ser.* **331** 072002

[13] Buncic P, Aguado Sanchez C, Blomer J, Franco L, Harutyunian A, Mato P and Yao Y, 2009, CernVM – a virtual software appliance for LHC applications, 2009, *J. Phys.: Conf. Ser.* **219** 042003