

# Towards a serverless CernVM-FS

Jakob Blomer<sup>1,\*</sup>, Gerardo Ganis<sup>1</sup>, Simone Mosciatti<sup>1</sup>, and Radu Popescu<sup>1</sup>

<sup>1</sup>CERN, Geneva, Switzerland

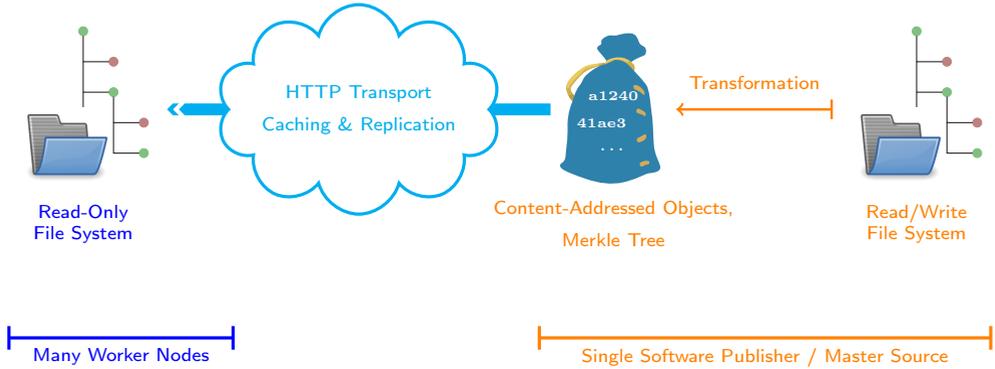
**Abstract.** The CernVM File System (CernVM-FS) provides a scalable and reliable software distribution and—to some extent—a data distribution service. It gives POSIX access to more than a billion binary files of experiment application software stacks and operating system containers to end user devices, grids, clouds, and supercomputers. Increasingly, CernVM-FS also provides access to certain classes of data, such as detector conditions data, genomics reference sets, or gravitational wave detector experiment data. For most of the high-energy physics experiments, an underlying HTTP content distribution infrastructure is jointly provided by universities and research institutes around the world. In this contribution, we will present recent developments and future plans. For future developments, we put a focus on evolving the content distribution infrastructure and at lowering the barrier for publishing into CernVM-FS. Through so-called serverless computing, we envision cloud hosted CernVM-FS repositories without the need to operate dedicated servers or virtual machines. An S3 compatible service in conjunction with a content delivery network takes on data provisioning, replication, and caching. A chain of time-limited and resource-limited functions (so called “lambda function” or “function-as-a-service”) operate on the repository and stage the updates. As a result, any CernVM-FS client should be able to turn into a writer, possession of suitable keys provided. For repository owners, we aim at providing cost transparency and seamless scalability from very small to very large CernVM-FS installations.

## 1 Introduction

The CernVM File System (CernVM-FS), a global read-only POSIX file system, provides the software distribution backbone for the experiments at the LHC and numerous other scientific collaborations within and beyond High Energy and Nuclear Physics (HENP). [1–3] The CernVM-FS client provides the virtual `/cvmfs` directory tree on worker nodes in the grid as well as on containers and virtual machines in the cloud, on compute nodes of supercomputers, and on end-user devices. At the top directory level, CernVM-FS mounts *repositories* into the `/cvmfs` tree. Repositories are independent file system instances, and similar to Internet web servers there is no central directory of available CernVM-FS repositories. At the time of writing, the authors are aware of more than 100 active repositories with a total of more than 1

---

\*e-mail: [jblomer@cern.ch](mailto:jblomer@cern.ch)



**Figure 1.** Content in CernVM-FS is edited on special *release manager machines* that provide an exclusive, writable access to the repository. New and modified files are cryptographically hashed, as well as the resulting file system tree. Clients always show a consistent file system tree of a specific revision.

billion files under management. Within WLCG, content distribution is handled by five public data mirror servers (*Stratum 1s*) located in Europe, the U.S., and Asia, as well as some 400 site-local cache servers that are shared with the Frontier service [4].

CernVM-FS is asymmetric by construction. In order to facilitate scalability for reading with many clients from many locations, writing into the repository is a centralized operation called *publishing*. During publishing, new and modified files are transformed into a content-addressed format and a new, consistent directory tree (a *snapshot*) is fixed (c.f. Figure 1). While costly at the time of writing, the content-addressed format provides de-duplication, versioning, an always consistent view of the repositories, and it decouples the file system semantics from the content distribution. The data transport and storage layers simply operate on immutable and mutually independent blobs.

In this contribution, we outline improvements to the CernVM-FS publishing process. Based on support for S3 storage [5] and distributed writing where multiple nodes can publish concurrently into separate directory sub trees [6], we describe how to evolve the CernVM-FS server side for “serverless” operations. We envision a scenario in which any CernVM-FS client with suitable keys can acquire a short-term lease for a directory sub tree and publish new content directly into cloud storage. For the simple case where the file system modifications can be described as a tarball of new files, we envision S3 endpoints that allow direct publishing for such payloads.

## 2 Strategic Use Cases

The architecture of CernVM-FS is optimized for hosting a large number of small files that exhibit a small cluster-wide hot working set of the order of Gigabytes. The following four use cases are CernVM-FS core application areas.

1. **Experiment production software.** Application software releases that are meant for widespread distribution, such as the software in `/cvmfs/cms.cern.ch`. This use case is considered stable except for HPC de-

ployment, which despite several successful installations is still an active area of development [7].

2. **Integration builds.** Software artifacts in preparation for production software, such as the software in `/cvmfs/lhcbdev.cern.ch`. The contents are delivered to CI pipelines and developers. In contrast to application software, integration builds have a high turn-over and require regular CernVM-FS' garbage collection runs. Current efforts are focused on reducing the content propagation from the around 10 minutes default to less than one minute. [8]
3. **Unpacked Container Images.** Extracted root file system directories or layer directories of container images, such as the contents of `/cvmfs/singularity.opensciencegrid.org`. Container engines that can start an image from a root file system in a directory, such as Singularity [9], are supported out of the box. For Docker, a CernVM-FS graph driver plugin provides equivalent functionality [10] and avoids the need to preload entire images (layers) in order to start a container. Current efforts focus on containerd integration and simplified publishing of extracted images.
4. **Auxiliary Data.** Due to its general availability, CernVM-FS is a good match for low-volume data such as the detector conditions data in `/cvmfs/alice-ocdb.cern.ch`. CernVM-FS breaks files larger than a few megabytes into smaller chunks and uses variable chunk sizes based on rolling checksums, so that common sub parts in different files are de-duplicated with high probability.

Besides these core application areas, CernVM-FS can be used with custom plug-ins for the client cache and for authentication. This has enabled its use, for instance, with a cache plug-in tailored to an HPC environment on the Piz Daint supercomputer at the Swiss CSCS HPC center. In the Open Science Grid, CernVM-FS is successfully used to provide a namespace for multiple petabytes of data (e.g. LIGO experiment data). [11] This installation uses CernVM-FS authentication plug-ins as well as a dedicated distribution infrastructure independent of the regular CernVM-FS CDN.

### 3 Serverless Computing

The so-called *serverless* cloud computing paradigm is an evolution of the Infrastructure-as-a-Service paradigm. Instead of hosting entire applications on virtual hardware, applications are deconstructed into their core constituents, which are then individually hosted and orchestrated in the cloud. Although the specifics of application constituents slightly differ depending on the commercial cloud provider or open source serverless framework, they usually comprise

- stateless functions (ephemeral containers) that represent the application logic,
- triggers and events (such as REST API calls) that invoke functions,
- API frontend description,
- persistent storage in the form of object stores and databases,
- persistent storage of secrets such as storage credentials.

Ideally, application developers do not need to take care of infrastructure components such as containers but only push their code. The infrastructure can scale the individual components according to the demands, e. g. it can run a larger number of functions when more API calls are issued during peak hours. Disadvantages of serverless computing are the entry barrier of decomposing applications and, lacking a serverless computing industry standard, a soft vendor lock-in.

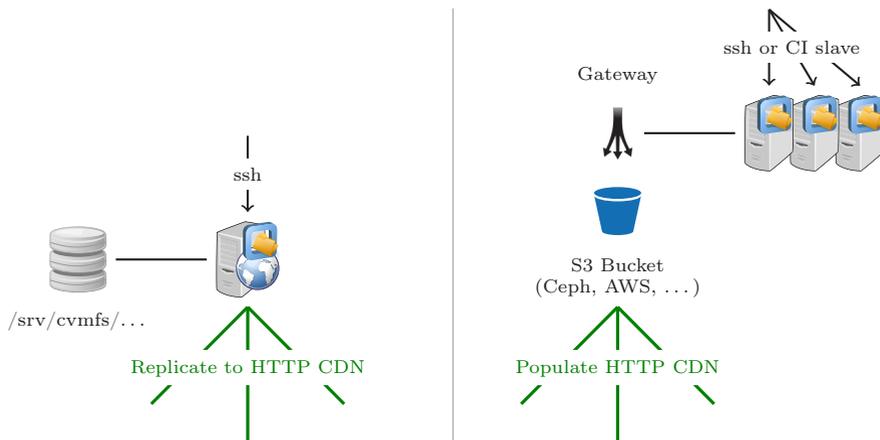
## 4 CernVM-FS Serverless Publishing

The classic CernVM-FS publishing process works by logging in to a dedicated release manager machine with an attached storage volume that provides the editable, authoritative repository copy. File system changes are written into a staging area and only upon commit published as a consistent change set. Technically, a union file system such as `overlayfs` provides the writable access and the scratch area on top of a standard read-only CernVM-FS client. Towards a serverless model, the release manager machine has to become a set of stateless components that can be spawned on demand. Serverless publishing aims at removing dedicated CernVM-FS server infrastructure components and replace them by standard, sharable cloud services.

The CernVM-FS release manager machine can be configured to directly publish to an S3 compatible storage system. [5] The S3 interface has been successfully tested with Amazon Web Services as well as with the open source S3 gateway to Ceph [12] at CERN and RAL. The first S3 backed production repositories are currently being commissioned at CERN. Since S3 is based on standard HTTP, CernVM-FS clients can directly connect to the S3 host. Moreover, S3 cloud storage is often provided across multiple regions and integrated with geographically distributed caching edge servers. This can replace the CernVM-FS application-level Stratum 0 to Stratum 1 replication and thus make published content instantaneously available to clients.

As of CernVM-FS 2.5, a *gateway* component has been added to the CernVM-FS publishing tools [8]. The gateway is used in production, for instance, for the `cernvm-prod.cern.ch` repository. The gateway is a web service in front of the authoritative storage. It provides an HTTP API to upload signed change sets from possibly multiple release manager machines that can operate concurrently on the repository. In order to prevent write conflicts, the gateway maintains a database of secret keys that are used to acquire exclusive short-term leases (locks) to directory sub trees. In this deployment, release manager machines do not directly modify the authoritative storage but they send their change sets to a gateway from which a lease has been acquired. Figure 2 shows the classic publishing and the publishing components using S3 storage and gateway services.

Towards a fully serverless publishing, the gateway services need to be encapsulated in a container (a *cloud function*) that is started on demand when change sets are available. The release manager machine and the associated handling of union file system details shall be moved from a server-side component to a CernVM-FS client add-on. Linux user namespaces facilitate the on-demand switch from a read-only client to a writable client. A sub shell can spawn a new user namespace and mount namespace and, using a union file system, provide a writable `/cvmfs` mountpoint that is valid and visible only within the current sub shell. Any client with a suitable secret key can thus publish to the gateway in the cloud. The development efforts to fully implement the additional functionalities are expected to span the next two to three years.



**Figure 2.** Publishing content into CernVM-FS. Left hand side a classic installation with a single release manager machine and attached authoritative storage. Right hand side an installation using S3 storage and the CernVM-FS gateway.

## 5 Portals

As an alternative path to publish into CernVM-FS, we developed a prototype add-on to release manager machines that provides *portals* in the form of S3 endpoints. These portals are attached to a directory in the repository. They can receive file packs (e.g. tarballs, RPMs, ...) which are directly extracted into the CernVM-FS target path. This allows users in possession of a suitable S3 key pair to push their personal analysis software artifacts into CernVM-FS without any CernVM-FS specific software. Direct ingestion of tarballs circumventing the usual union file system mechanics is also beneficial for publishing container images.

While initially attached to a release manager machine, portals can naturally be mapped to serverless environments. Reacting on a newly uploaded object into an S3 bucket is a standard use case for a *trigger*. The publishing of a file pack can be encapsulated in a cloud function.

## 6 Outlook

The CernVM File System provides a stable and highly scalable read-only file system for production software, integration builds, unpacked container images, and auxiliary data to grids, clouds, supercomputers, and end-user devices. This core will continue receiving attention, while new developments focus higher-level services and on scaling up publishing workloads. We leverage the existing support for S3 storage and multiple, stateless release manager machines, and aim at removing any dedicated CernVM-FS server infrastructure components. Doing so, we foresee to be able to scale the number of writers (human users or computer agents) from tens to hundreds per repository.

## References

- [1] J. Blomer, C. Aguado-Sanchez, P. Buncic, A. Harutyunyan, Journal of Physics: Conference Series **331** (2011)

- [2] J. Blomer, P. Buncic, R. Meusel, G. Ganis, I. Sfiligoi, D. Thain, *Computing in Science and Engineering* **17**, 61 (2015)
- [3] The CernVM-FS developers, *The cernvm file system: v2.5.1* (2018), <https://doi.org/10.5281/zenodo.1472994>
- [4] D. Dykstra, L. Lueking, *Journal of Physics: Conference Series* **219** (2010)
- [5] M. Arsuaga-Ríos, S. Heikkilä, D. Düllmann, R. Meusel, J. Blomer, B. Couturier, *Journal of Physics: Conference Series* **664** (2015)
- [6] J. Blomer, P. Buncic, G. Ganis, N. Hardi, R. Meusel, R. Popescu, *Journal of Physics: Conference Series* **898** (2017)
- [7] J. Blomer, G. Ganis, N. Hardi, R. Popescu, *Delivering LHC Software to HPC Compute Elements with CernVM-FS* (Springer, 2017), p. 724–730, Number 10524 in *Lecture Notes in Computer Science*
- [8] R. Popescu et al., *Towards a responsive CernVM-FS architecture*, Poster at this conference (2018)
- [9] The Singularity developers, *Singularity 3.0.0* (2018), <https://doi.org/10.5281/zenodo.1451202>
- [10] N. Hardi, J. Blomer, G. Ganis, R. Popescu, *Journal of Physics: Conference Series* **1085** (2018)
- [11] D. Weitzel, B. Bockelman, D. Dykstra, J. Blomer, R. Meusel, *Journal of Physics: Conference Series* **898** (2017)
- [12] S.A. Weil, Ph.D. thesis, University of California Santa Cruz (2007)