EPJ Web of Conferences **245**, 07012 (2020) https://doi.org/10.1051/epjconf/202024507012
CHEP 2019

A fully unprivileged CernVM-FS

Jakob Blomer[1,*], *Dave* Dykstra[2], *Gerardo* Ganis[1], *Simone* Mosciatti[1], and *Jan* Priessnitz[1]

[1]CERN, Geneva, Switzerland
[2]Fermilab, Chicago, U.S.

Abstract. The CernVM File System provides the software and container distribution backbone for most High Energy and Nuclear Physics experiments. It is implemented as a file system in user-space (Fuse) module, which permits its execution without any elevated privileges. Yet, mounting the file system in the first place is handled by a privileged suid helper program that is installed by the Fuse package on most systems. The privileged nature of the mount system call is a serious hindrance to running CernVM-FS on opportunistic resource and supercomputers. Fortunately, recent developments in the Linux kernel and in the Fuse user-space libraries enabled fully unprivileged mounting for Fuse file systems (as of RHEL 8), or at least outsourcing the privileged mount system call to a custom, external process. This opens the door to several, very appealing new ways to use CernVM-FS, such as a generally usable "super pilot" consisting of the pilot code bundled with Singularity and CernVM-FS, or the on-demand instantiation of unprivileged, ephemeral containers to publish new CernVM-FS content from anywhere. In this contribution, we discuss the integration of these new Linux features with CernVM-FS and show some of its most promising, new applications.

1 Introduction

The CernVM File System (CernVM-FS) provides the software and container distribution backbone for most High Energy and Nuclear Physics experiments [1, 2]. In order to access the contents of CernVM-FS repositories, worker nodes and end-user laptops mount the CernVM-FS client under the `/cvmfs` name space. In contrast to software distribution techniques that use pre-built bundles (such RPM packages or container tarballs), the mounted file system allows on-demand loading of the usually small subset of binaries that is really accessed at any given moment.

The CernVM-FS client uses the File System in User Space (Fuse) framework [3]. Fuse is a standard component of all major Linux distributions and available as an extra package for macOS. Fuse implements a minimal, forwarding kernel-level file system that issues upcalls to a user space *Fuse module* that in turn implements the actual file system logic. This architecture permits execution of the file system client without any elevated privileges. Yet, mounting the file system in the first place requires administrative privileges, as the `mount()` system call is restricted.

*e-mail: jblomer@cern.ch

On fully opportunistic resources, such as supercomputers or containers in a commercial cloud, this has led to various approaches to circumvent the restriction. The CernVM-FS *parrot connector* [4] wraps applications in a gdb-like sandbox that emulates a mounted volume. On some HPC systems, users prepare a copy of the file system tree on the shared storage, often comprising millions of small files. Other systems deploy "fat containers" that can grow to hundreds of gigabytes and contain a substantial part of all the available experiment software. These approaches tend to be fragile and cumbersome.

In this contribution, we review the current and future state of mounting Fuse file systems. We show how the latest Linux and Fuse developments lift restrictions on mounting Fuse file systems and the implications of this development for opportunistic CernVM-FS deployments.

2 Privileges for file systems in user space

As mounting a file system is a restricted operation on Linux systems (except for recent kernels), even Fuse user-space modules require some help from a system administrator in order to be used. In the case of the CernVM-FS client, a system administrator needs to

- Either install the CernVM-FS distribution-specific package. This will setup the client such that it starts as privileged user and drops the privileges immediately after the mounting.
- Or enable a privileged helper program that comes with the Linux distribution's Fuse package and that takes care of the mounting on the users behalf (see Figure 1).

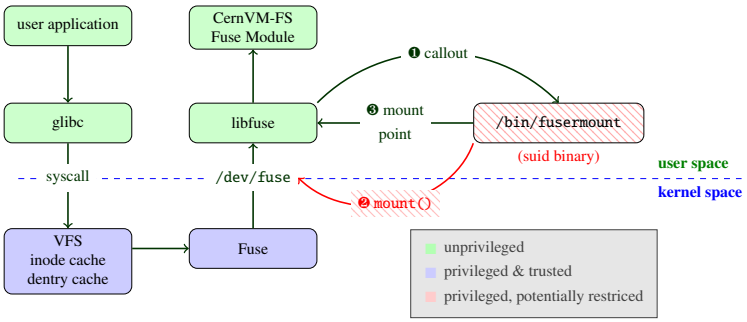

Figure 1. A successful Fuse mount returns a file descriptor to /dev/fuse, which is subsequently used by the fully unprivileged *Fuse module*.

On fully opportunistic resources, no such system-level installation or configuration change can be assumed.

3 Custom mount helpers

As of the Fuse version 3 user space libaries, the task of mounting /dev/fuse can be handed off to a trusted, external helper (see Figure 2). Fuse 3 support has been added to CernVM-FS version 2.7 and later. We furthermore backported the Fuse 3 libraries to Red Hat Enterprise Linux (RHEL) version 6 and 7 compatible platforms and made them available in the Extra Packages for Enterprise Linux (EPEL) package repository.

While a custom mount helper is still a privileged program, it can be a utility that fits better into the environment at hand[1]. For supercomputers that already use and trust Singularity [5],

[1]For detailed information, search for "pre-mounting" in the CernVM-FS technical documentation at https://cvmfs.readthedocs.io

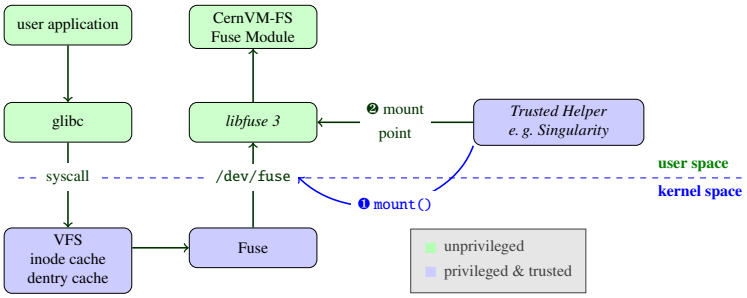

Figure 2. As in Fuse vanilla deployments, a privileged helper program takes care of mounting the file system. But in contrast to Fuse version 2, in Fuse version 3 this program can be a custom one different from /bin/fusermount.

for instance, we added support to Singularity version 3.4 and later to act as a Fuse mount helper. In this way, Singularity can start a container and hand over the mount point to an unprivileged CernVM-FS file system client within the container.

4 Unprivileged mounting using namespaces

Namespaces are a Linux kernel mechanism to provide kernel resource isolation between process groups. Namespaces are an enabling technology for Linux containers, giving each container the illusion of running exclusively on the system. As of Linux kernel version 4.18 (available for instance in RHEL version 8), Fuse file systems can be mounted within namespaces (see Figure 3).

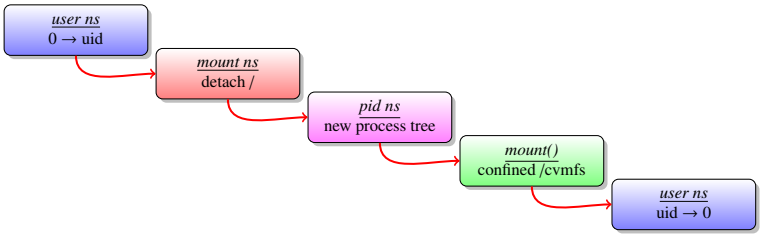

Figure 3. In order to enable Fuse mount points to be private to a namespace, a certain set of namespaces has to be created in order. In the process, mounting takes place as "fake root user", i.e. as a privileged user whose actions are confined to a namespace. Consequently, mount points created within a namespace are invisible to processes outside the namespace.

Effectively, Fuse file systems within namespaces allow for unprivileged containers that can mount the CernVM-FS client without any further host support by making the /dev/fuse device available to the container. At the timing of this writing, popular container tools such as Docker and kubernetes do not yet start unprivileged containers with Fuse namespace mounts enabled by default but they can be enabled optionally.

5 Applications

The means to mount the CernVM-FS client without administrator assistance enables new, promising applications.

5.1 Universal pilot

The experiment pilot job is a placeholder grid job that, upon landing on a worker node, connects to the experiment task queue in order to fetch and execute actual payloads. Experiment pilot jobs typically expect to find a working CenVM-FS client in order to spawn the environment for the payload. Given the possibility to mount CernVM-FS as a pilot user, however, the CernVM-FS client can be bundled to provide a "universal pilot". This self-extracting bundle would

1. create a new user namespace
2. mount the experiment CernVM-FS repository
3. run the experiment pilot jobs from the CernVM-FS mountpoint
4. the pilot job can optionally mount additional CernVM-FS repositories
5. the pilot job runs Singularity from the CernVM-FS mountpoint
6. the payload job runs inside the Singularity container

We have written a package called `cvmfsexec` [6] that makes it easy to do steps 1, 2, and 4. The package even works with older kernels (such as on RHEL 7) if unprivileged user namespaces and fusermount are available, but then it has a limitation that the Fuse mountpoints go into the system-wide mount namespace and do not get unmounted if the program is sent a hard kill signal.

5.2 CernVM-FS on-demand publisher node

The CernVM-FS publisher node is a machine that is used by repository owners to maintain the repository content. On the publisher node, a union file system [7] (e. g. OverlayFS) is used to create a writable CernVM-FS mountpoint by unifying the read-only virtual CernVM-FS directory with a locally writable scratch directory. The publisher nodes are typically carefully maintained and dedicated to a certain repository. As a result, software build products are typically copied from build nodes to the publisher node, instead of publishing directly from the build nodes. While the CernVM-FS storage gateway services [8] synchronizes multiple publisher nodes that operate on the same repository, the required effort to maintain each of the publisher nodes is not reduced.

Unprivileged Fuse namespace mounts in concert with *fuse-overlayfs* [9] can provide a simpler way to publish to a CernVM-FS repository. Instead of a set of dedicated publisher nodes, *any* node can temporarily become a publisher. Such an on-demand publisher spawns an ephemeral container that provides a writable CernVM-FS mountpoint connected to the repository gateway services (see Figure 4). In this way, software builder nodes could directly publish their build products provided repository access keys are available.

Unprivileged, on-demand publishing has been shown as proof-of-concept and is expected to be released in CernVM-FS version 2.8.

6 Conclusion

Being a Fuse file system, using the CernVM-FS client has always been relatively non-intrusive. The inherently privileged nature of mounting file systems, however, turned out to be a burden for certain client deployments, in particular in opportunistic environments such

```
$> cvmfs enter hsf.cvmfs.io /users/joe
  ... opens a shell in an ephemeral container
  with write access to the repository
$> cvmfs publish
  ... back to read-only mode
```

Figure 4. The `cvmfs enter` command creates an isolated environment through namespaces as described in Section 4. The writable mountpoint inside this mini container is provided by the CernVM-FS client and the Fuse implementation of the OverlayFS union file system.

as HPC centers and commercial container clouds. Fortunately, recent developments in the Linux kernel and the Fuse user space libraries have largely lifted the restrictions for mounting Fuse file systems. The CernVM-FS client as of version 2.7 is fully integrated with these developments. As these changes find widespread use in Linux distributions and container management tools, we expect major simplifications to use CernVM-FS on opportunistic resources as well as simplifications in the management of repository publisher nodes.

References

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

[2] J. Blomer, B. Bockelman, P. Buncic, B. Couturier, D.F. Dosaru, D. Dykstra, G. Ganis, M. Giffels, H. Nikola, N. Hazekamp et al., *The cernvm file system: v2.7.0* (2019), `https://doi.org/10.5281/zenodo.3608672`

[3] N. Rath et al., *"libfuse" [software], version 3.9.0* (2019), `https://github.com/libfuse/libfuse`

[4] 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

[5] G.M. Kurtzer, cclerget, M. Bauer, I. Kaneshiro, D. Godlove, Vanessasaurus, WestleyK, Y. Cote, E. Arango, G. Vallee et al., *sylabs/singularity: Singularity 3.4.0 release* (2019), `https://doi.org/10.5281/zenodo.3382358`

[6] D. Dykstra, *"cvmfsexec" [software], version 2.3* (2020), `https://github.com/cvmfs-contrib/cvmfsexec`

[7] C.P. Wright, J. Dave, P. Gupta, H. Krishnan, E. Zadok, M.N. Zubair, Tech. Rep. FSL-04-01b, Stony Brook University (2004)

[8] A. Forti, L. Betev, M. Litmaath, O. Smirnova, P. Hristov, eds., *Towards a serverless CernVM-FS*, Vol. 214 (2019)

[9] G. Scrivano et al., *"fuse-overlayfs" [software], version 0.7.5* (2020), `https://github.com/containers/fuse-overlayfs`