virtio-fs

Present and Future

Hanna Czenczek <hreitz@redhat.com>
German Maglione <gmaglione@redhat.com>

KVM Forum 2023

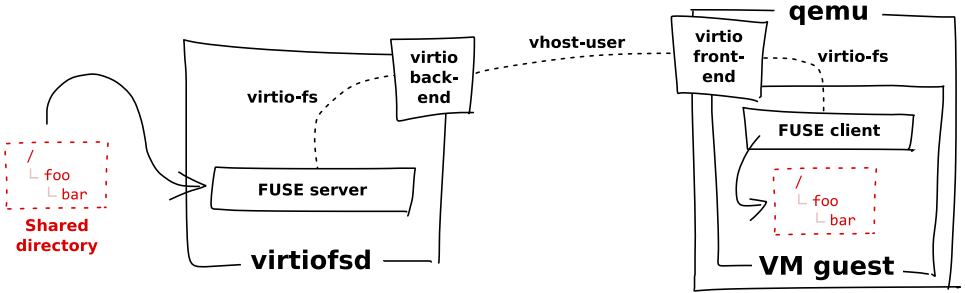
Red Hat

Content

1. virtio-fs and virtiofsd today (overview, options)
2. Live migration
3. Integration with other projects
4. virtio-fs from a technical perspective
5. Future plans

Overview

virtio-fs: sharing a directory tree between host and VMs
virtiofsd: vhost-user device daemon written in Rust



https://virtio-fs.gitlab.io/

https://gitlab.com/virtio-fs/virtiofsd

Section 1

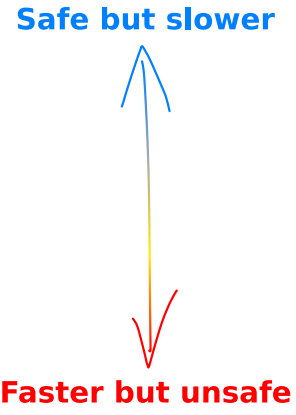
Options You Should Know About

Red Hat

Cache Modes

Tell guest how to cache:

- `--cache=never`:
  - Disable page and dentry caches
- `--cache=auto`:
  - Use defaults (read cache, writethrough)
  - Cache dentries for 1 s
- `--cache=always`:
  - Keep page cache when (re-)opened
  - Cache whole directories (indefinitely)
- `--writeback`:
  - Writeback cache



Red Hat

Context for `--inode-file-handles`

```
/          = 1
└ foo      = 42
  └ bar    = 120
```

```
open("/foo/bar"):
  1.lookup("foo") → 42
  42.lookup("bar") → 120
  120.open()
```

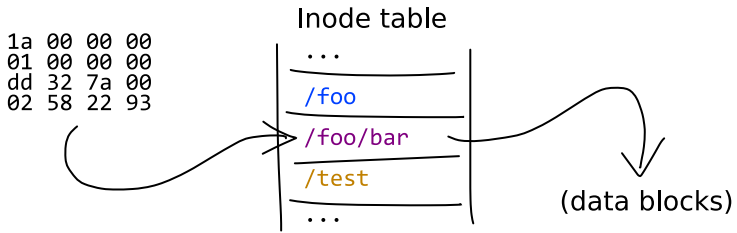
→ virtiofsd must open inodes from ID

Red Hat

O_PATH vs. File Handles

Mapping inode ID → inode:

1. O_PATH FDs
   - FD count limit
   - Keeps files open: Problems with unlinking (NFS silly rename)
2. File handles: Pure data, per-FS unique ID locating an inode
   - Needs CAP_DAC_READ_SEARCH (i.e., root), FS support



Red Hat

Configuration

`--inode-file-handles`:

- `never` (default): Always use `O_PATH` FDs
- `prefer`: Use file handles if supported
- `mandatory`: Never use `O_PATH` FDs

Section 2

Live Migration

Red Hat

Problem: Internal State

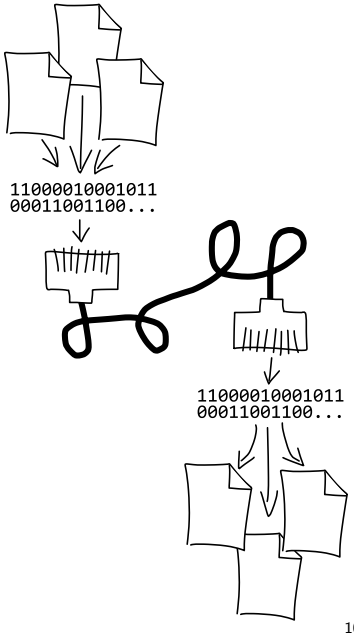
State:

- Mappings of inode IDs to inodes
- FDs for open files

→ Transfer and restore on destination

Problems:

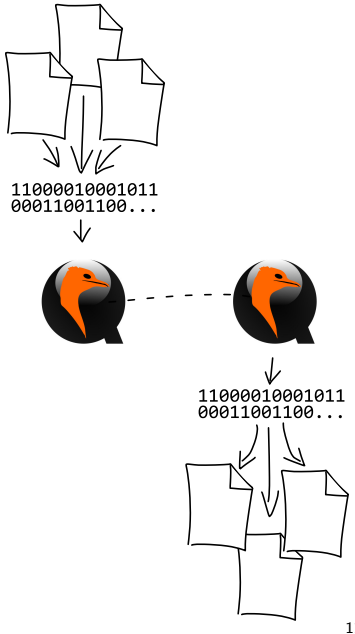
1. Serializing/deserializing
2. Transfer channel



Red Hat

Transfer Channel: QEMU's Migration Stream

- No additional configuration/privileges
- Make use of QEMU's features (e.g. migration through file)
- Requires extending the vhost(-user) protocol
  - Blocking blob transfer during downtime
  - Proposal/discussion: https://lists.nongnu.org/archive/html/qemu-devel/2023-04/msg01575.html



Red Hat

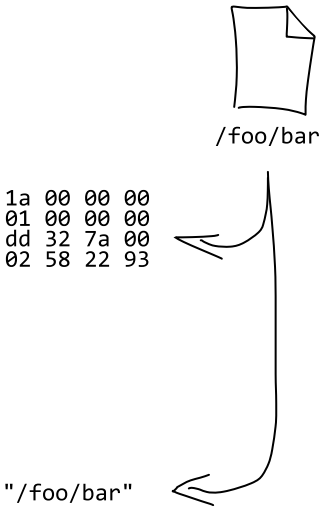
Serialization: Indexed Inodes

File handles:

- Either immediately, or by converting FDs
- Requires `CAP_DAC_READ_SEARCH`
- Only valid on the very same FS

File paths:

- Need to be reconstructed (costly)
- Allows migration to different FS
- Cannot handle external renames



Red Hat

Serialization: Open Files

- Transfer associated inode ID + `openat()` flags → reopen
  - Problem: File deleted
  - Problem: Permissions changed
- Silly renaming? Delaying `unlink`? Tricky.
- Cannot transfer FDs without same-time same-host same-FS channel
  - Add same-virtiofsd-process restriction: Skip state transfer altogether
    - See “external migration” work from Anton Kuchin https://lists.nongnu.org/archive/html/qemu-devel/2023-02/msg05295.html

Section 3

Integration with other projects

Red Hat

Integration with other projects: dracut & systemd fstab-generator

Running a VM from a virtiofs share and/or defining a virtiofs mount unit via kernel commandline, for instance using dracut: `root=virtiofs:<mount-tag>`

(or `rootfstype=virtiofs root=<mount-tag>`)

```
qemu \
    ...
    -object memory-backend-file,id=mem,size=4G,mem-path=/dev/shm,share=on \
    -machine memory-backend=mem \
    -chardev socket,id=vfsdsock,path=${socket_path} \
    -device vhost-user-fs-pci,queue-size=1024,chardev=vfsdsock,tag=host \
    -kernel ${kernel} \
    -initrd ${initrd} \
    -append 'root=virtiofs:host rootflags=rw,defaults'
```

Red Hat

Integration with other projects: kubevirt

Sharing ConfigMaps, Secrets, DownwardAPI, ServiceAccounts, PVCs and node directories[0] dynamically propagating the changes to the VM.

[0] https://kubevirt.io/user-guide/virtual_machines/disks_and_volumes/

Red Hat

Integration with other projects: kubevirt

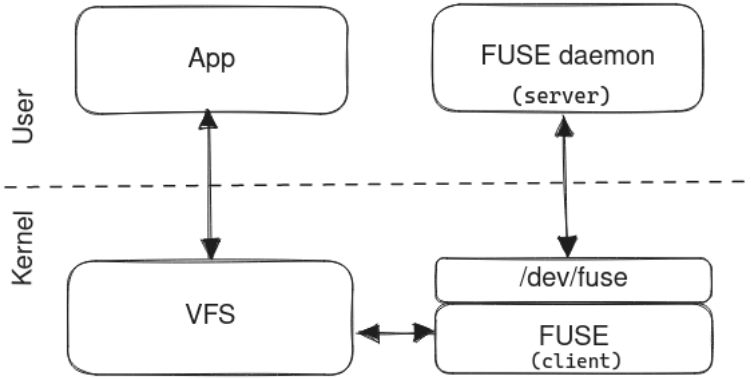
```
apiVersion: kubevirt.io/v1
kind: VirtualMachineInstance
spec:
  domain:
    devices:
      filesystems:
        - name: serviceaccount-fs
          virtiofs: {}
volumes:
    - name: serviceaccount-fs
      serviceAccount:
        serviceAccountName: default
```

Section 4

How virtiofsd works

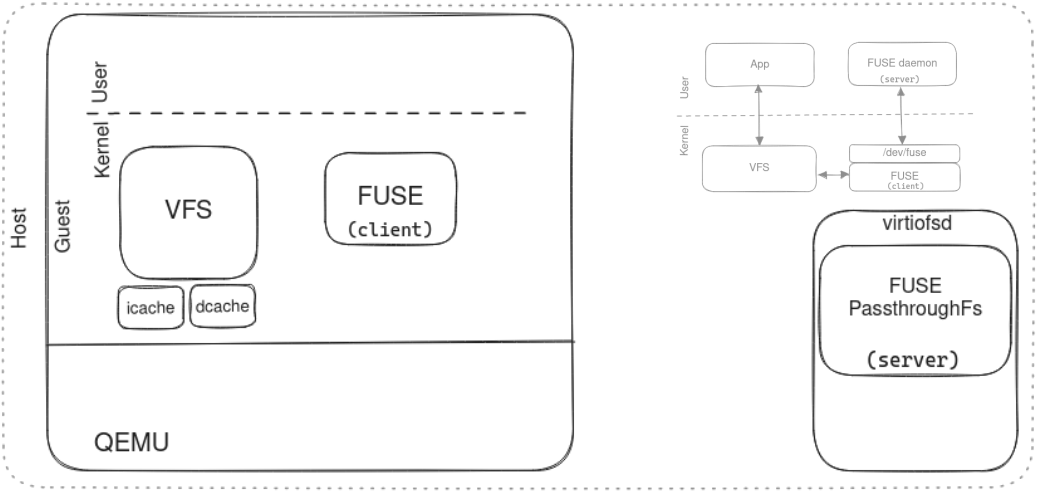
Red Hat

How virtiofsd works: FUSE



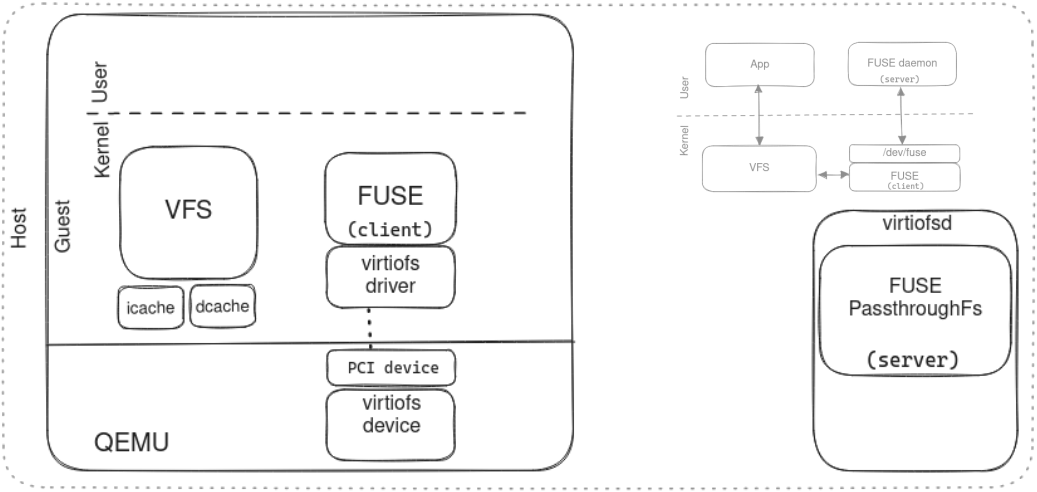
Red Hat

How virtiofsd works: FUSE



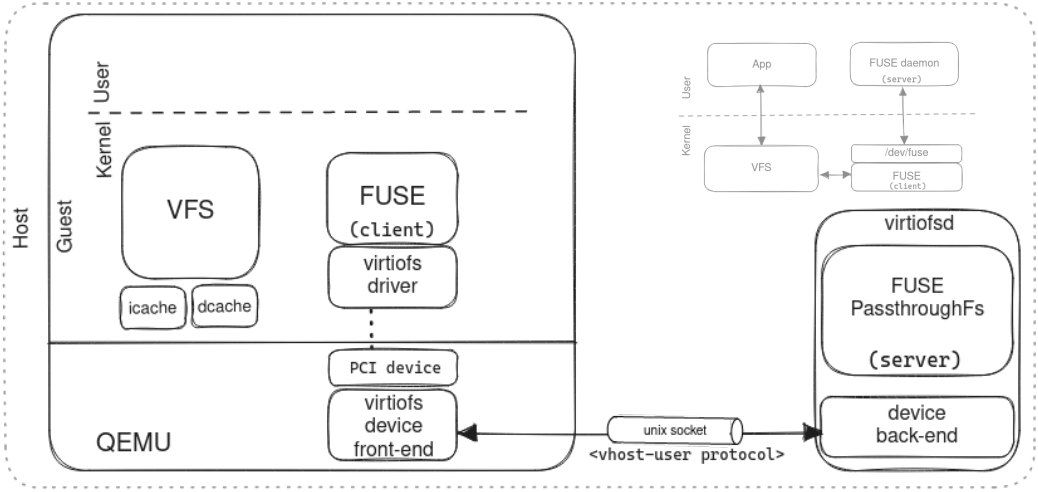
Red Hat

How virtiofsd works: virtio



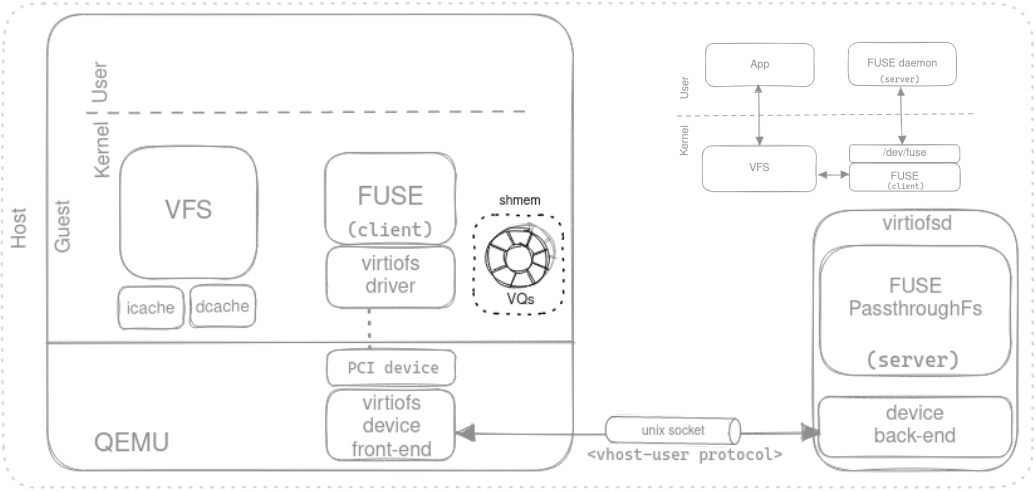
Red Hat

How virtiofsd works: vhost-user



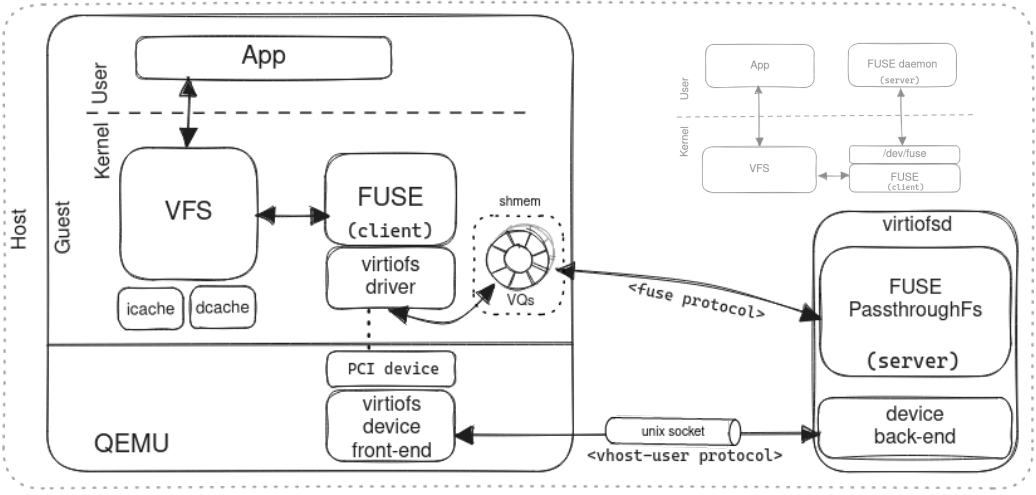
Red Hat

How virtiofsd works: vhost-user



Red Hat

How virtiofsd works



Section 5

Future plans

Red Hat

Future plans

- Live migration support.
- Extract virtiofsd functionality in its own lib crate. So that it can be more easily embedded in other projects, such as libkrun and Cloud-Hypervisor.
- Move the sandboxing code to an external tool: more flexibility to add features, such as LandLock isolation, keep listening after the client disconnects, I/O throttling using cgroups, etc.
- virtio-vhost-user support: VM to VM sharing.
- read-only sharing, io_uring support, operation coalescing, inotify, etc.

The end.

Thanks for listening.

Part I

Appendix

Extended Attributes

- `--xattr`: Enable support
- `--posix-acl`: Allow guest to use `posix_acl` xattrs
  - Needs host support: Do not remap with `--xattrmap`, problematic with NFSv4
- `--security-label`: Set created nodes' SELinux labels
  - Separate for (potential) atomicity

`--xattrmap`: Add prefixes, allow/deny matching keys

- Complex, see *xattr-mapping* documentation