CephFS: Architecture Introduction & New Features

Greg Farnum, IBM

RADOS SOFTWARE COMPONENTS

ceph-mon

ceph-mgr

ceph-osd

Monitor

- Central authority for authentication, data placement, policy
- Coordination point for all other cluster components
- Protect critical cluster state with Paxos
- 3-7 per cluster

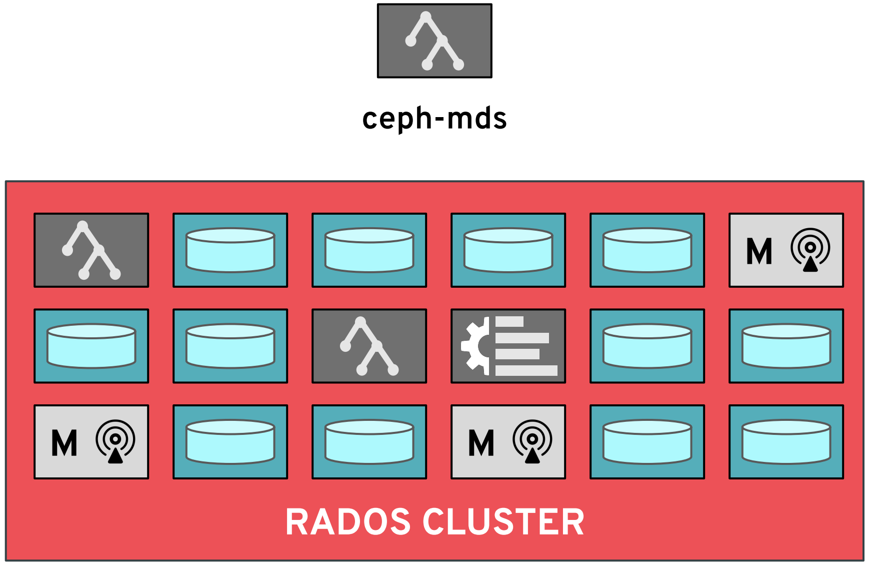
Manager

- Aggregates real-time metrics (throughput, disk usage, etc.)
- Host for pluggable management functions
- 1 active, 1+ standby per cluster

OSD (Object Storage Daemon)

- Stores data on an HDD or SSD
- Services client IO requests
- Cooperatively peers, replicates, rebalances data
- 10s-1000s per cluster

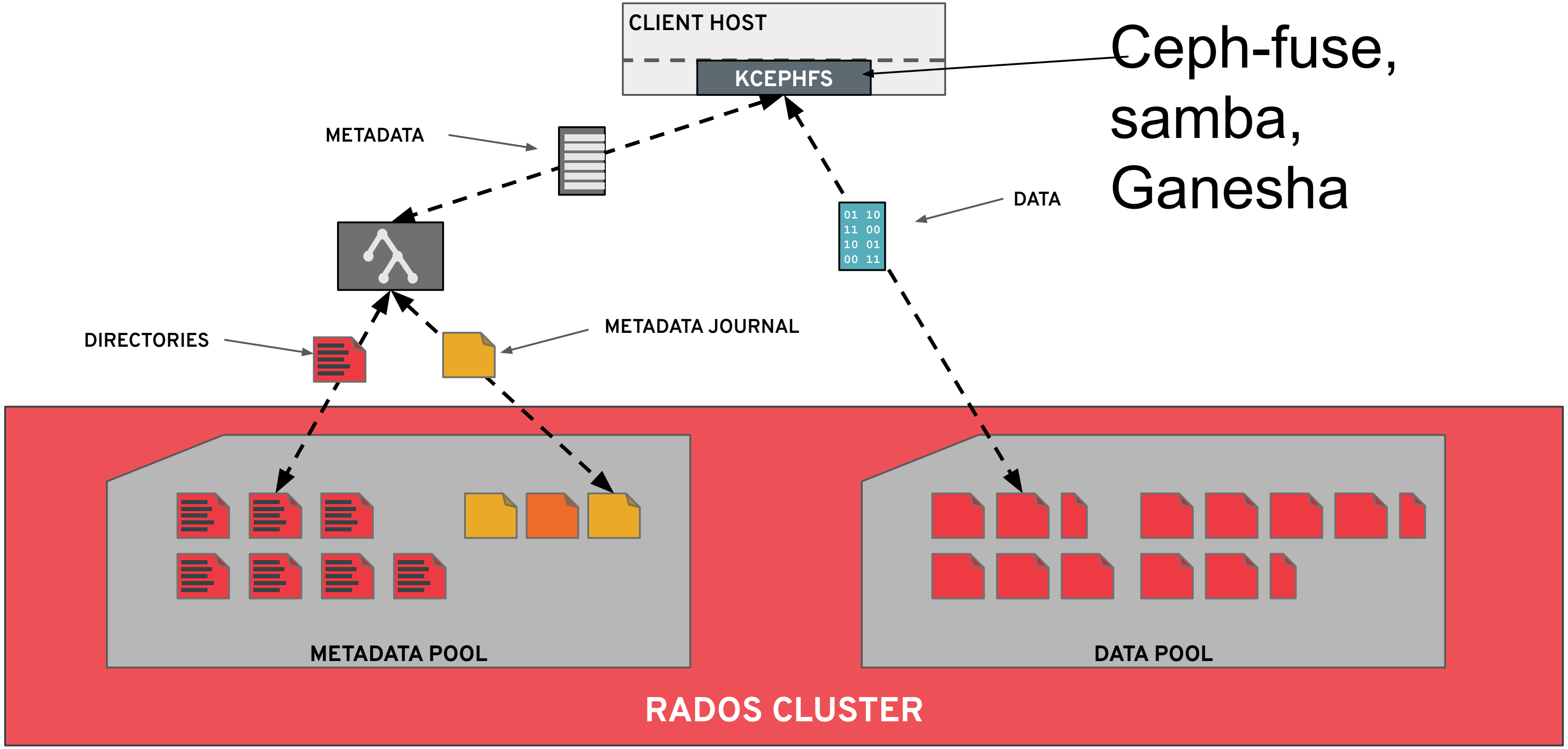
CEPH-MDS: CEPH METADATA SERVER



MDS (Metadata Server)

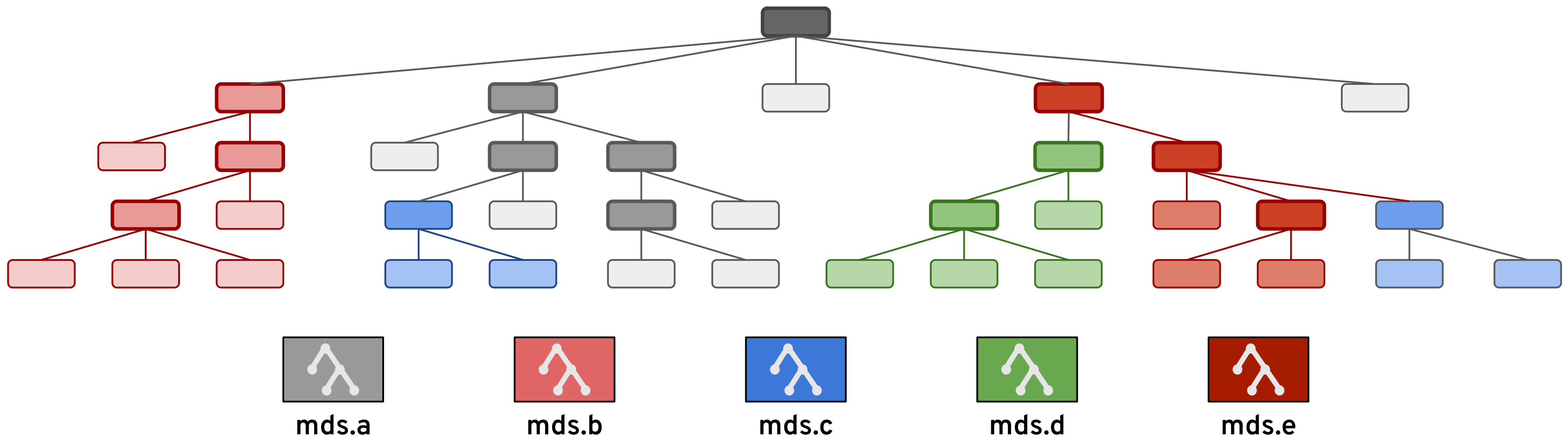
- Manage file system namespace
- Store file system metadata in RADOS objects
    - File and directory metadata (names, inodes)
- Coordinate file access between clients
- Manage client cache consistency, locks, leases
- Not part of the data path
- 1s - 10s active, plus standbys

METADATA IS STORED IN RADOS



Ceph-fuse, samba, Ganesha

SCALABLE NAMESPACE



- Partition hierarchy across MDSs based on workload
- Fragment huge directories across MDSs
- Clients learn overall partition as they navigate the namespace
- Subtree partition can maintain directory locality
- Arbitrarily scalable by adding more MDSs

CONSISTENT ACROSS CLIENTS

- Clients and MDS cooperatively maintain a distributed cache of metadata including inodes and directories
- MDS hands out capabilities (aka “caps”) to clients to delegate access to parts of inode metadata/data

CEPHFS CAPS

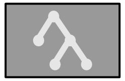
CLIENT HOST
KCEPHFS

pAsLsXsFcrwb

Pin – p
the inode exists
Auth – A
authentication metadata: mode, uid, gid
Link – L
the inode's link count (number of dentries linked to an inode)
Xattr – X
the inode's xattrs; their presence and values
File – F
file data, file size, mtime et al

Shared – s
The client has shared access to this state; one of many
Exclusive – x
The client is the only one with access to this state
Read – r
The client can read state
Write – w
The client can write the state
Cache – c
The client can cache the state locally
Buffer – b
The client can buffer changes to the state locally

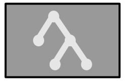
CEPHFS CAP BREAKDOWN – COMBINATIONS

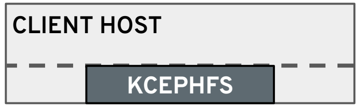


pAsLsXsFcrwb

- Not every cap uses every permission
- pin: binary; the client can remember an inode's existence
- Auth, Link, Xattr: Shared or eXclusive
- [ALX]s – can save the state locally, reference it
    - Hurray, we can do permission checking on the client
- [ALX]x – Nobody else can look at this state; we "own" it
    - We can *change* the metadata locally and tell the MDS later on!

CEPHFS CAP BREAKDOWN – COMBINATIONS





pAsLsXsFcrwb

- Fs: can cache and read mtime, size locally
- Fx: can write mtime, size locally
- Fr: can read the file data (...synchronously from OSD)
- Fc: can cache file data for local reads
- Fw: can write the file data (synchronously to OSD)
- Fb: can buffer data writes; flush in the background

CEPHFS SNAPSHOTS

- Snapshot any directory
  - Applies to all nested files and directories
  - Granular: avoid “volume” and “subvolume” restrictions in other file systems
- Point-in-time consistent for solo client
  - from perspective of POSIX API at *client*
  - *not* client/server boundary
- Easy user interface via file system
- Efficient
  - Fast creation/deletion
  - Snapshots only consume space when changes are made

```
$ cd any/cephfs/directory
$ ls
foo bar baz/
$ ls .snap
$ mkdir .snap/my_snapshot        ←
$ ls .snap/
my_snapshot/
$ rm foo
$ ls
bar baz/
$ ls .snap/my_snapshot
foo bar baz/
$ rmdir .snap/my_snapshot        ←
$ ls .snap
$
```

CEPHFS RECURSIVE ACCOUNTING

- MDS maintains recursive stats across the file hierarchy
  - File and directory counts
  - File size (summation)
  - Latest **ctime**
- Visible via virtual xattrs
- Recursive bytes as directory size
  - If mounted with ‘rbytes’ option
  - Unfortunately this confuses rsync; off by default
  - Similar to ‘du’, but free

```
$ sudo mount -t ceph 10.1.2.10:/ /mnt/ceph \
-o name=admin,secretfile=secret,rbytes
$ cd /mnt/ceph/some/random/dir
$ getfattr -d -m - .
# file: .
ceph.dir.entries="3"
ceph.dir.files="2"
ceph.dir.subdirs="1"
ceph.dir.rbytes="512000"
ceph.dir.rctime="1474909482.0924860388"
ceph.dir.rentries="17"
ceph.dir.rfiles="16"
ceph.dir.rsubdirs="1"
$ ls -alh
total 12
drwxr-xr-x  3 sage sage 4.5M Jun 25 11:38 ./
drwxr-xr-x 47 sage sage  12G Jun 25 11:38 ../
-rw-r--r--  1 sage sage   2M Jun 25 11:38 bar
drwxr-xr-x  2 sage sage 500K Jun 25 11:38 baz/
-rw-r--r--  1 sage sage   2M Jun 25 11:38 foo
```

OTHER CEPHFS FEATURES

- Multiple file systems (volumes) per cluster
  - Separate ceph-mds daemons
- xattrs
- File locking (flock and fcntl)
- Quotas
  - On any directory
- Subdirectory mounts + access restrictions
- Multiple storage tiers
  - Directory subtree-based policy
  - Place files in different RADOS pools
  - Adjust file striping strategy
- Lazy IO
  - Optionally relax CephFS-enforced consistency on per-file basis for HPC applications
- Linux kernel client
  - e.g., mount -t ceph $monip:/ /ceph
- ceph-fuse
  - For use on non-Linux hosts (e.g., OS X) or when kernel is out of date
- NFS
  - CephFS plugin for nfs-ganesha FSAL
- CIFS
  - CephFS plugin for Samba VFS
- libcephfs
  - Dynamically link with your application

SQUID: MAJOR DELIVERIES

- quiesce
  - crash consistent snapshots
- log trimming and scaling improvements
  - https://tracker.ceph.com/issues/61908
- automatic balancer disabled by default

UPCOMING IN TENTACLE

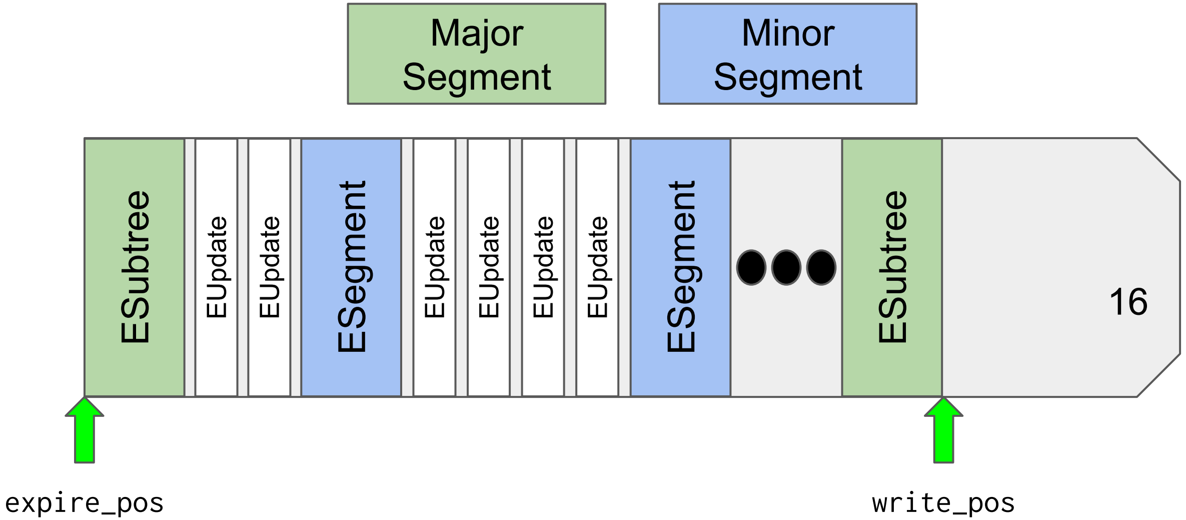
- case insensitive directory trees / subvolumes
- efficient hardlink management (referent inode)
- user-space fscrypt
- *libcephfs* async-io and zero-copy interfaces

MDS Improvements

MDS LOGGING IMPROVEMENTS

MDS Logging Improvements

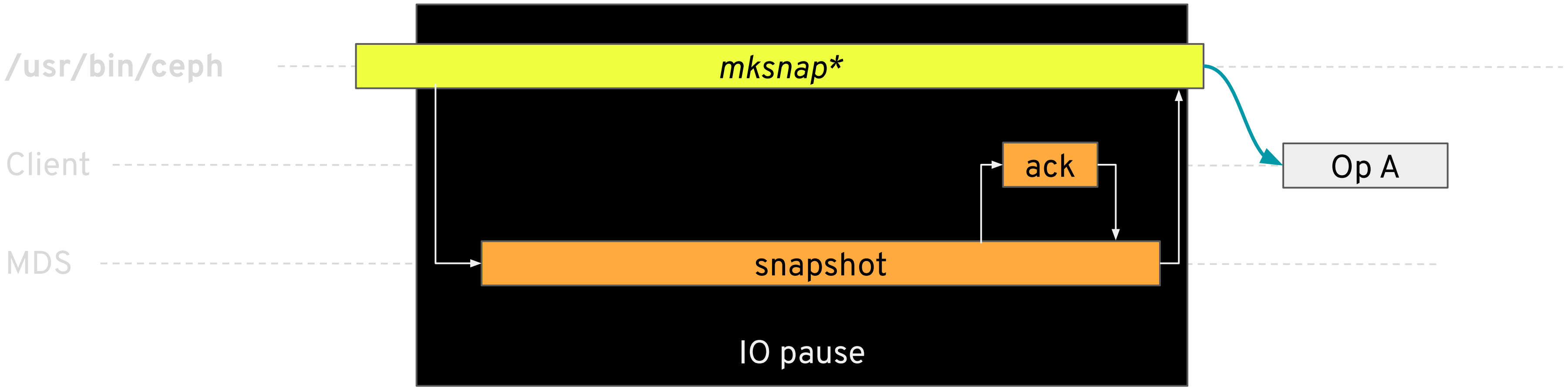
- New segment event
- Avoid journaling subtree map frequently



QUIESCE INTERFACE

MDS (squid,tentacle): More crash-consistent snapshots

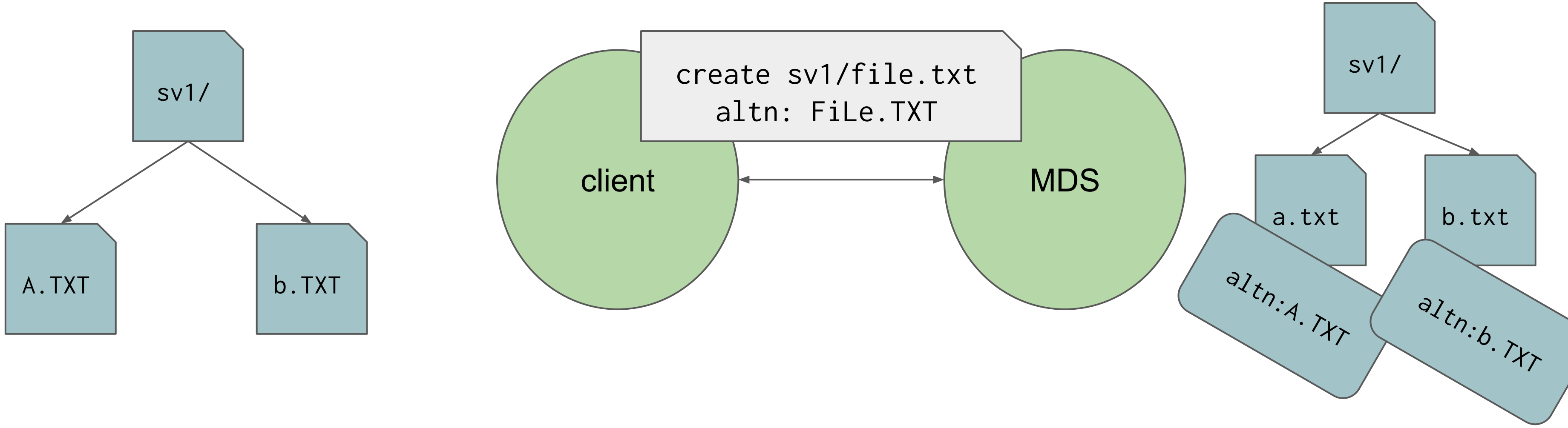
- New **quiesce** mechanism in the MDS to recursively recall all metadata/data mutation permissions from clients.
    - An admin socket command is exported to manipulate an internal database of quiesced **roots.** (This command is not meant to be user-facing.)



CASE-INSENSITIVE SUBVOLUMES

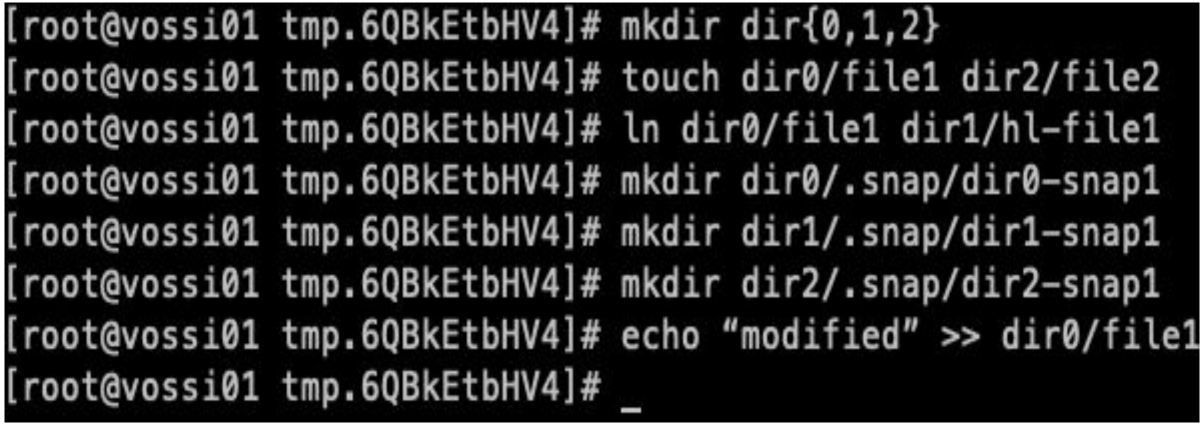
MDS (squid,tentacle): case insensitive lookup

- Reuse **alternate_name** internal metadata on dentries to preserve case information.

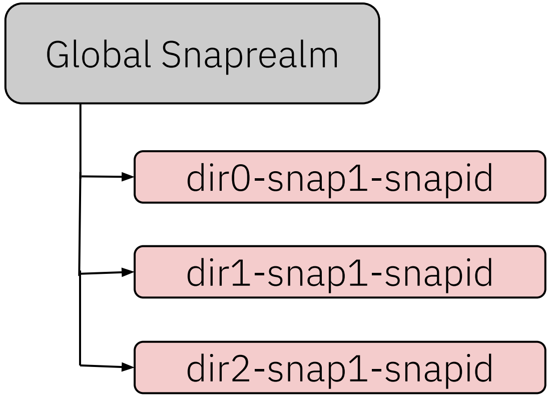


REFERENT INODES

- Upcoming: More efficient handling of hard links
  - current hardlink management uses a “global snaprealm”
  - Replace it with a “referent inode”



```
[root@vossi01 tmp.6QBkEtbHV4]# mkdir dir{0,1,2}
[root@vossi01 tmp.6QBkEtbHV4]# touch dir0/file1 dir2/file2
[root@vossi01 tmp.6QBkEtbHV4]# ln dir0/file1 dir1/hl-file1
[root@vossi01 tmp.6QBkEtbHV4]# mkdir dir0/.snap/dir0-snap1
[root@vossi01 tmp.6QBkEtbHV4]# mkdir dir1/.snap/dir1-snap1
[root@vossi01 tmp.6QBkEtbHV4]# mkdir dir2/.snap/dir2-snap1
[root@vossi01 tmp.6QBkEtbHV4]# echo “modified” >> dir0/file1
[root@vossi01 tmp.6QBkEtbHV4]# _
```



CLIENT IMPROVEMENTS

KERNEL CLIENT: FSCRYPT

Client-side encryption!

- Merged in Linux 6.6.
- Encrypts dentry names using the **fscrypt** kernel library (already used by ext4/etc.)
  - Uses **`alternate_name`** dentry metadata to store ciphertext of long dentry names.
- File data is encrypted with per-file keys stored in the inode metadata.
- Userspace tools for fscrypt volumes are compatible.

```
# fscrypt setup /mnt
# dd if=/dev/urandom of=my.key bs=32 count=1
# fscrypt encrypt –source=raw_key --key=my.key /mnt/cephfs/sv1/
# fscrypt unlock --key=my.key /mnt/cephfs/sv1
# ls /mnt/cephfs/sv1/
a.txt b.txt
```

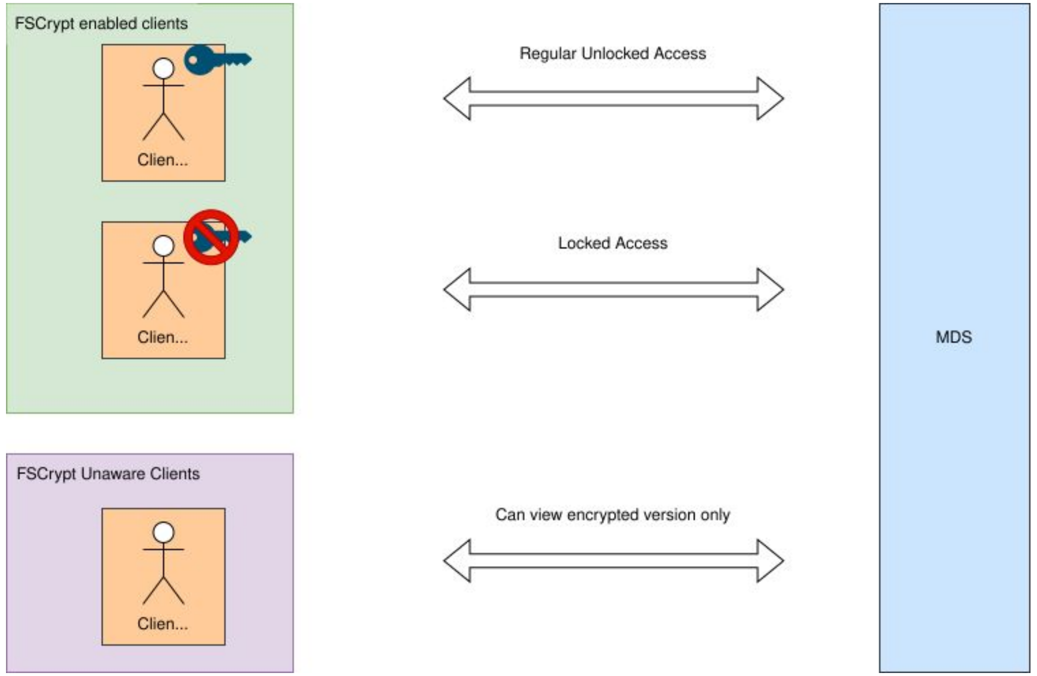
LIBCEPHFS PROXY

- Samba forks a new process for each samba client connection
  - This means there’s a new CephFS client for each samba client! :(:(:(
- New proxy daemon so each samba process on a node speaks to a single CephFS client
- https://github.com/ceph/ceph/tree/main/src/libcephfs_proxy

USERSPACE CLIENT: FSCRYPT

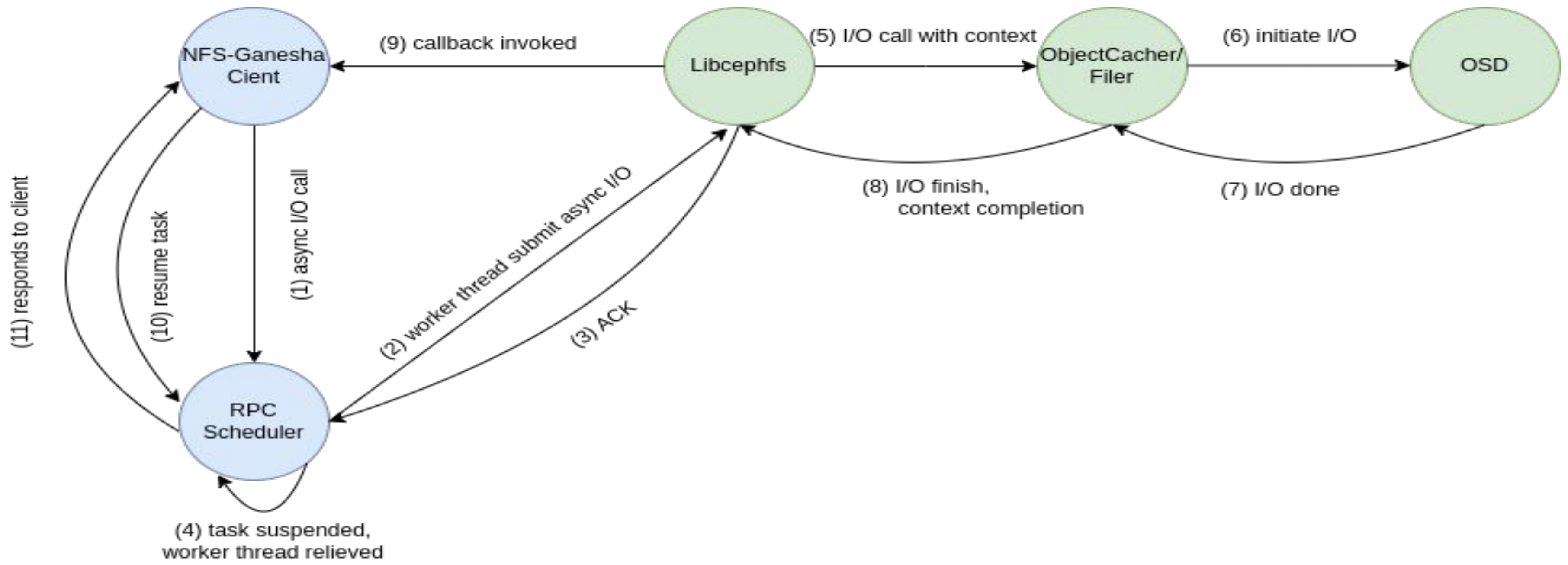
Upcoming in Tentacle

- A port of the kernel client's implementation of fscrypt to userspace.
- The kernel fscrypt library is reimplemented in C++.
- Same primitives in the protocol (alternate_name) are in use and cross-compatible with the kernel.



Bringing fscrypt Encryption to CephFS Userspace: Christopher Hoffman

LIBCEPHFS: ASYNCHRONOUS I/O



Async I/O with LibCephFS: Dhairya Parmar

CEPH-MGR/PLUGIN IMPROVEMENTS

SCALING IMPROVEMENTS TO PLUGINS

- Technical improvements further isolate plugins from each other
    - Per-plugin “finisher” thread
    - Identified incorrect (too-long) locking in some python binding calls
- Prevents a lot of downhill spirals

MGR/VOLUMES PLUGIN

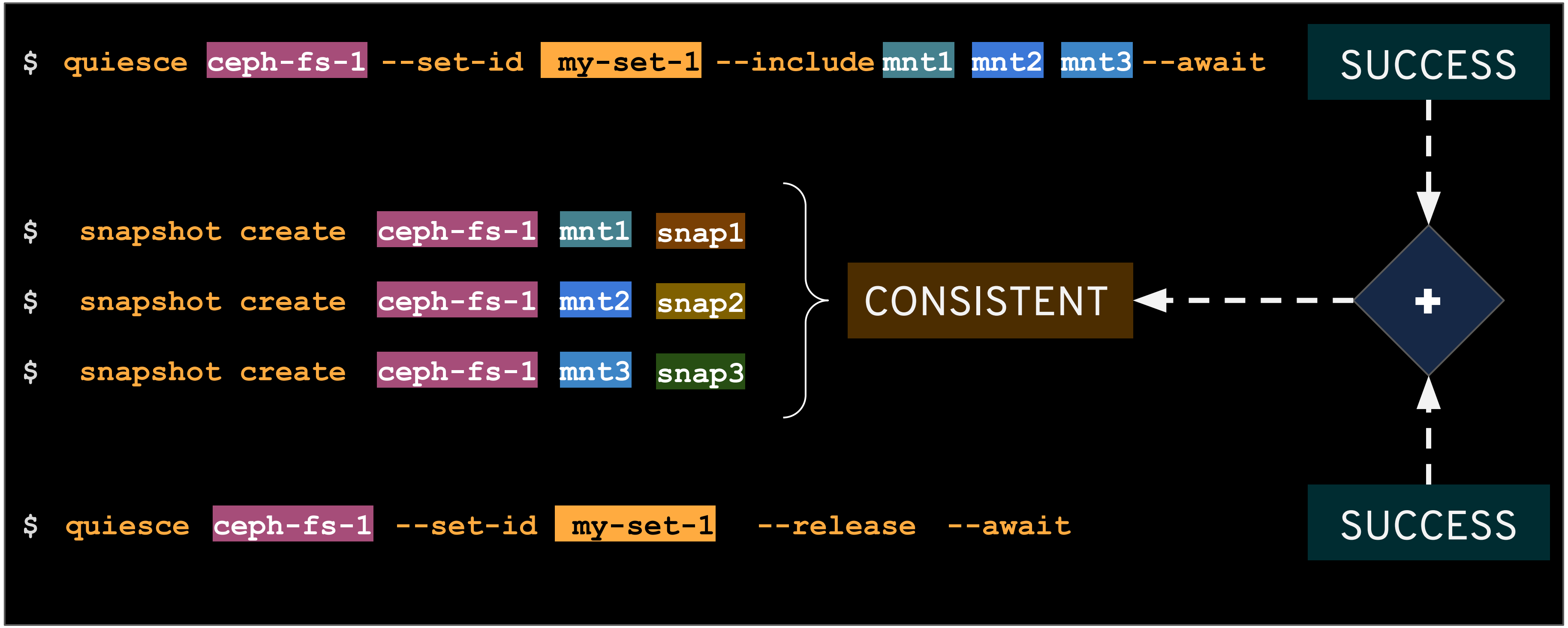
- Integration point for platforms to allocate “subvolumes” programmatically
  - Ceph-CSI, Rook, Project Manila
- https://docs.ceph.com/en/latest/cephfs/fs-volumes/

```
# ceph fs subvolume create <vol_name> <subvol_name> [--size
<size_in_bytes>] [--group_name <subvol_group_name>] [--pool_layout
<data_pool_name>] [--uid <uid>] [--gid <gid>] [--mode <octal_mode>]
[--namespace-isolated] [--earmark <earmark>]
```

QUIESCE INTERFACE

- Not meant for end users, but…
- Integration point for platforms
  - Kubernetes/Rook, OpenStack/Manila, etc
- Enables quiescing of *arbitrary* subvolumes in groups
- Then you snapshot them and get a point-in-time consistent group of snapshots across all relevant clients!

Ceph File System



```
$ quiesce ceph-fs-1 --set-id my-set-1 --include mnt1 mnt2 mnt3 --await

$  snapshot create  ceph-fs-1 mnt1  snap1
$  snapshot create  ceph-fs-1 mnt2  snap2
$  snapshot create  ceph-fs-1 mnt3  snap3

$ quiesce ceph-fs-1 --set-id my-set-1  --release  --await
```

Ceph File System

```
$ ceph fs subvolume quiesce <vol_name>
      [--set-id=<qsid> [--await|--await-for=<await_timeout>] | --release | --cancel]
      [--if-version=<v>]]
      [--query [--all]]
      [--group=<sub_group>]
      [--quiesce-timeout=<sec>] [--quiesce-expiration=<sec>]
      [([--include]|--exclude|--reset) <[sub_group/]sub_name>[, …]]
      → {"qsid": <status>|null}
      : [SUCCESS|EPERM|EACCES|ENOENT|EINVAL|EINTR|EINPROGRESS|ETIMEDOUT|ECANCELED|ESTALE]
```

```
$ ceph fs subvolume snapshot create <vol_name>                <snap_name>
      (--quiesce-set-id=<qsid> [--quiesce-await] [--quiesce-release] |
       --consistent [--quiesce-timeout=<sec>] [--quiesce-expiration=<sec>])
      [--members=<[sub_group/]sub_name>[, ...]]
      [--group_name=<sub_group>]
      : [SUCCESS|EPERM|EACCES|ENOENT|EINVAL|EINTR|EINPROGRESS|ETIMEDOUT|ECANCELED|ESTALE]
```

Ceph File System

Upcoming: V3 (the “***finale***”) subvolume layout

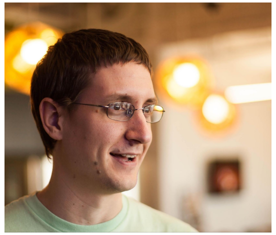
Ceph File System

Future work

- fast cloning (layering)
  - build on referent inode work
- QoS based on dmclock
  - https://github.com/ceph/ceph/pull/52147

Ceph File System

THANK YOU and Q/A



Greg Farnum
IBM, Inc.
CephFS engineering manager
`gfarnum@IBM.COM`

- Quiesce:
  - Docs: https://docs.ceph.com/en/latest/cephfs/fs-volumes/#subvolume-quiesce
  - Talk @ FOSDEM’24
- Case insensitive directory trees: https://github.com/ceph/ceph/pull/60746
- QoS: https://github.com/ceph/ceph/pull/52147