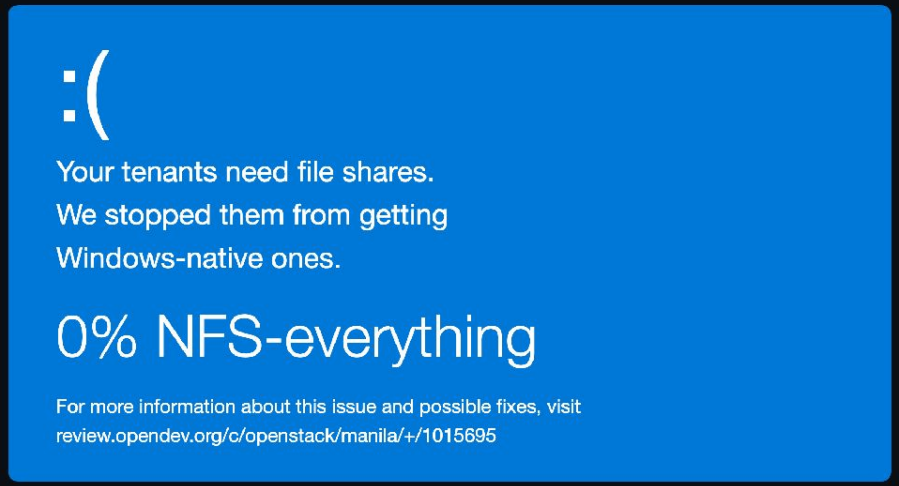



# The Blue Screen of Ceph

## An SMB experiment in OpenStack Manila

Goutham Pacha Ravi · Core Maintainer, OpenStack Manila · Red Hat

Ceph Day Chicago 2026

Project names and logos are trademarks of their respective projects (Open Infrastructure Foundation; Red Hat / Ceph).
Used for identification in this community presentation only — does not imply endorsement.

Raise your hand: do you run CephFS in production?

Keep it up: do your Windows workloads get file shares from it?

THE GAP IS REAL WORKLOADS

# SMB is the Windows native tongue

🗄️

**FSLogix / VDI**
Profile containers are the #1 cloud file-share workload on Earth — every virtual desktop mounts one at login

_structured data_

**SQL Server & IIS**
Databases and web farms want SMB shares with ACLs, not NFS-on-Windows

🎨

**macOS fleets**
Creative and engineering shops; SMB is the only first-class network filesystem macOS ships

In an OpenStack cloud with Ceph underneath, these workloads either ran on a **separate Windows file-server island** — or didn't run at all.

THE EXPERIMENT'S PREMISE

# "Surely connecting them is easy?"



**CephFS**
subvolumes, quotas, snapshots, clones — storage primitives, done for years



**Ceph Tentacle**
an SMB control plane: the ceph-mgr `smb` module, cephadm-deployed Samba, declarative resources

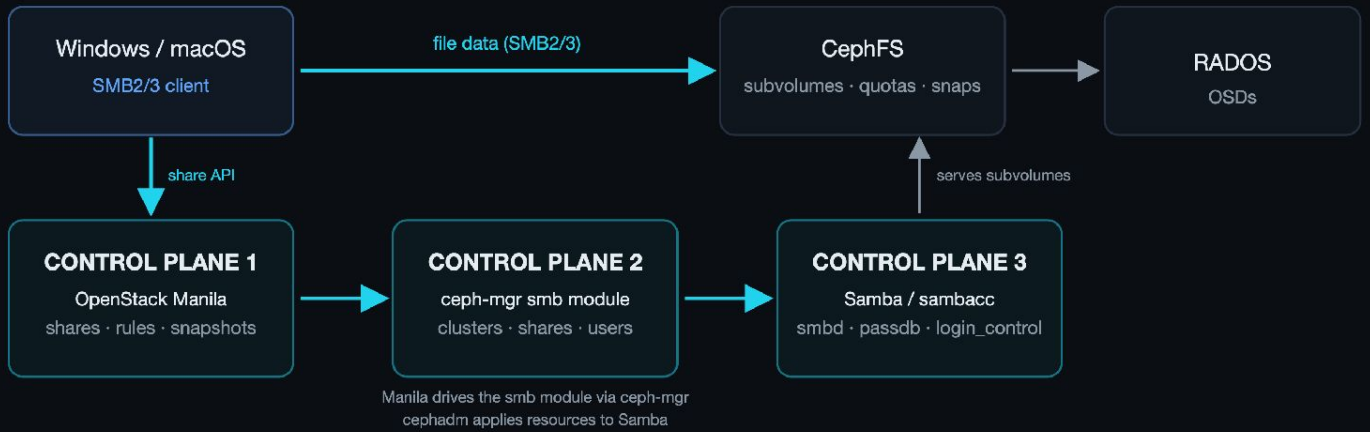


**Manila**
tenant-facing share lifecycle, access rules, snapshots — already drives Ceph's *nfs* module

Three bricks that look like they snap together.
**This talk explores that assumption.**

THE EXPERIMENT

# Three control planes, one filesystem



WHAT A SHARE BECOMES

# A decade of co-evolution, one new protocol

| MANILA CONCEPT | CEPH PRIMITIVE |
|---|---|
| share (CIFS, 100 GiB) | subvolume + `ceph.smb.share` resource · quota = `bytes_quota` |
| access rule ( `user` ) | one `login_control` entry — Samba's `valid users` + `read list` / `write list`, managed as declarative state |
| credential delivery | tenant-supplied password via `access_key` (write-through, not stored); or generated and returned once |
| snapshot | a CephFS subvolume snapshot — nothing more, nothing less |
| share deletion | resources removed idempotently; **no resource until the first access rule** — deny-by-default |

```
# the share resource:
"restrict_access": true,
"login_control": [
  {"name": "gouthamr",
   "access": "read-write"}
]

# what Samba sees:
valid users = gouthamr
write list = gouthamr
```

**restrict_access** is the deny-by-default switch — Samba denies unlisted users at tree connect.

**login_control** entries say what each listed user can do: read, read-write, admin, or denied.

Both are declarative fields on the Ceph share resource; cephadm's configwatch translates them into running Samba configuration.

Same lifecycle as NFS shares: no export until the first rule, removed with the last — deny-by-default by absence.

SHARP EDGE № 1 — AUTHORIZATION

# Deny-by-default is native — but the lifecycle isn't

- `restrict_access` + `login_control` give per-share authorization — Samba denies unlisted users at tree connect, enforces read vs read-write per operation
- Manila's rule model maps cleanly: one entry per active rule, recomputed and re-applied on every `update_access`
- **The sharp edge: a share with `restrict_access` and zero `login_control` entries is rejected by the module's schema**
- This aligns with Manila's lifecycle — share resource is created on the first rule, removed with the last — but required the driver to design around it explicitly

SHARP EDGE № 2 — USER LIFECYCLE

# Share rules propagate live… users don't

- `login_control` changes apply to running containers via configwatch — **no restart, tens of seconds**
- **New *users* are different: sambacc writes the passdb entry, but `smbd` doesn't pick it up automatically — a restart is required**
- The driver restarts the Samba service when it creates a new user — with a **crash-safe pending marker** in case the rule that triggered it fails part-way
- The restart drops connections — same as any file server reboot. Clients reconnect automatically; applications using SMB shares should already handle this (they handle monthly Windows patch cycles)
- The real issue is **when**: it triggers on any new user creation, not a maintenance window

```
# share rule change:
configwatch → smb.conf → live

# new user:
passdb entry → invisible to smbd
→ service restart → live
```

This asymmetry is upstream sambacc work

SHARP EDGE № 3 — DISAGREEMENT

# When control planes disagree

- **An SMB cluster with no shares is never deployed** — and removing the last share tears it down. Manila keeps a locked-down service share alive so the Samba service stays up
- **Resource IDs are capped at 18 characters** — share UUIDs had to be folded; **share-group flows pass instances without their group**; **RADOS commands time out at 10s** under load
- Every one of these was found by **tempest, running against real Ceph** — three of them became upstream fixes on their own

THE ANSWER TO DISAGREEMENT

# Desired state, applied declaratively

- Manila's database is the **source of truth**; Ceph resources and NTFS ACLs are its **projections**
- Every `update_access` recomputes the full allow-list and rewrites it — idempotent, last-writer-wins
- A periodic verifier recomputes each ACL **with freshly-resolved SIDs** and rewrites on drift
- Service restarts re-ensure every share; topology changes are fingerprinted into backend info

```
# the reconciliation loop, in one log line:
OK … share resource applied
with restrict_access=true and
login_control entries from active rules
```

The share resource *is* the desired state.

WHAT WE DELIBERATELY DID NOT DO

# The first boundary is narrow

- **cephadm, operator-managed SMB cluster** — Manila drives shares *within* a cluster the operator deployed; it does not deploy clusters
- **Standalone users** — Active Directory is designed but deferred: per-share ACLs for `DOMAIN\user` need domain-joined SID resolution
- **SMB2/3 only** — no SMB1, no dialect negotiation games
- **Manila is not** an identity service, a DNS server, or a gateway manager — if your Samba needs it, it was already your Samba
- Known limits, said out loud: **new-user creation restarts Samba** (same impact as a file server reboot — but unscheduled); usernames are cluster-global; the smb module is **tech preview**

EVERY CLAIM, ONE PIECE OF EVIDENCE

# What you just watched

| CLAIM | EVIDENCE IN THE CAST |
| --- | --- |
| Self-service tenant lifecycle | `share create` → available, with export location — no operator in the loop |
| Credentials without an identity service | rule `active` with generated `Access Key`; password never typed by a human |
| Quotas are real | `bytes_quota` in subvolume info; 100 GiB reported as free space by the SMB client |
| Deny-by-default | zorilla: no rule → denied; granted → access; revoked → `ACCESS_DENIED` while authenticated |
| Snapshots are CephFS snapshots | Manila snapshot appears in `fs subvolume snapshot ls` |
| Reconciliation survives sabotage | xattr deleted → fail-open → drift logged → ACL re-applied → locked out again |

WHERE THIS STANDS

# An honest ledger

**Works today** (code in review, tested end-to-end on Tentacle)

- create / delete / extend / shrink
- user access rules, rw + ro, credentials via `access_key`
- snapshots & create-from-snapshot (async clone)
- share groups, periodic ACL verification, drift healing
- full manila tempest API suite passing serially

**Not yet / by design**

- Active Directory access rules (SID resolution design in the spec)
- DHSS=True: per-tenant Samba clusters
- replication, migration, manage/unmanage
- the smb module itself is **tech preview**

ASKS

# What this experiment needs from Ceph

- **sambcc:** hot-reload the passdb when users change — share rules already propagate live via configwatch; user changes should too
- **Document** `login_control` **and** `restrict_access` — they work in Tentacle but are absent from the smb module documentation
- **Teuthology coverage** for `login_control` and `restrict_access` (tracker #76672)

# Should CephFS-backed SMB become a first-class Manila backend?

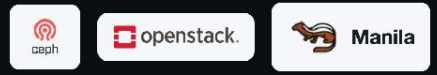
Driver

Spec

Come argue with me — here, or in #openstack-manila on OFTC

# Thank you

The Blue Screen of Ceph — an SMB experiment in OpenStack Manila




Goutham Pacha Ravi · gouthamr · #openstack-manila @ OFTC

BACKUP

# The security model, layered

- **SMB layer:** allow-list DACL on the share root (SYSTEM + one ACE per rule, no WRITE_DAC granted) — enforced by smbd per-operation
- **Filesystem layer:** share roots are POSIX 0777 (smbd runs non-root) — *only* the NTFS ACL gates access there
- **Ceph layer:** Manila's own cephx identity is path-scoped to `/volumes`; tenants never receive storage credentials or routes
- **Credential hygiene:** tenant-supplied passwords write through to Ceph without persistence; generated fallback is returned once; never argv, never logs
- **Residual risks (documented):** tree-connect precedes ACL denial (share existence observable); usernames are cluster-global; SMB encryption is a cluster-side recommendation

BACKUP

# Bugs found by integration testing

- **CephFS snapshots of grouped shares never worked** — the subvolumegroup was never passed to Ceph; group-from-snapshot clones failed three more ways → **review 1015696**, **LP #2174475**
- **Hardcoded 10s RADOS timeout** turned CI load into spurious "timed out" backend errors → configurable `cephfs_rados_timeout` → **review 1015697**
- Unit coverage on the new code: **~95%**, branch-focused; full tempest API suite passing serially
- Also upstreamed expectations: the manila-tempest-plugin now knows CIFS user rules can carry credentials

BACKUP

# The experiment rig

- Ceph **Tentacle (v20)**, cephadm-managed: 2 mons, 4 OSDs, 2 MDS (1 active + standby), smb module enabled — two hosts
- Samba service deployed by the smb module from **quay.io/samba.org/samba-server** (Samba 4.24), classic VFS provider
- OpenStack **Manila (master)** with the SMB protocol helper — `cephfs_protocol_helper_type = SMB`
- The cast: recorded live with asciinema; the only edit is a visible `sed` renaming lab hostnames — no output is fabricated
- Every act re-runnable: the demo script is in the kit ( `demo-cast.sh prep && play` )