MicroCeph: Get Ceph Up and Running in Minutes

Peter Sabaini, Canonical
2024-02-03
Why MicroCeph

1. Deploying and operating a Ceph cluster is complex
   a. Multiple nodes, distributed components and configurations
   b. Complex bootstrapping procedure
   c. Complex operation
2. Hardware requirements non-trivial,
   a. I can't just install a package on my laptop and be done
3. Operational complexity impacts Ceph adoption among users
   a. Impacts use of Ceph for focused (but smaller) use-cases
What Is MicroCeph

• Single package Ceph cluster
  • All in one package
  • 4 commands to get a functional cluster
  • Only one node and one disk required

# A functional ceph install for testing and labs

```
sudo snap install microceph            # install package
sudo microceph cluster bootstrap      # bootstrap initial node
sudo microceph disk add loop,4G,3     # add simulated drives
sudo ceph -s                           # verify status
```
MicroCeph: Snap Package

Snap package format

- **Isolation**: puts all userland in a single package
  - just need kernel and network / block devices
  - consistent environment across operating systems
- **Access to host system is confined for safety / security / robustness**
- **Standardized risk levels**: edge, candidate, stable
- **Easy to build, easy to distribute**
MicroCeph Arch

- Service management daemon
  - Cluster database (dqlite)
  - Ceph daemons
- CLI talks to service management via API
- Built from standard Ubuntu Ceph .deb LTS packages
MicroCeph Internals: Service API

- Service management daemon exposes API
- All interaction via API
  - Enable/disable/migrate services
  - Query block dev info
  - Add/remove disks
  - Add/remove nodes
  - Change config
  - etc...
- Useful for integration
- CLI is just another API client
MicroCeph Internals

- MicroCeph is built on top of the MicroCluster library
- MicroCluster provides
  - Dqlite: distributed database (RAFT)
  - Cluster membership
  - API framework
- Written in Go
MicroCeph: Scalability, Performance

- MicroCeph scales down: single node, single disk
- Automatic failure domain rules
  - Manages CRUSH rules to adapt from single-node (OSD-level) to multi-node (host-level) failure domains as clusters grow (or shrink back)
  - Single node clusters work ootb, and as you scale up you get added safety automatically
- Larger failure domains available via custom CRUSH rules
  - MicroCeph doesn't have a concept of racks or rooms
  - But won't interfere with CRUSH rules that implement wider failure domains
- MicroCeph scale up: primarily bound by RAFT (dqlite)
- Performance: same as Ceph, we're not sitting in the data path for any Ceph operations
MicroCeph: Integrations

- Cloud projects that integrate MicroCeph
  - Sunbeam (OpenStack on K8s)
  - MicroK8s
  - MicroCloud / LXD
- Juju integration: charm-microceph available to integrate into Juju clouds (beta)
- CI integration: microceph-action provides S3 for CI
MicroCeph: Demos

Single Node with RGW
https://asciinema.org/a/634408

(Demos are at 1.5x speed, 1s max idle to save time)
MicroCeph: Demos

Scaling up
https://asciinema.org/a/635478

- Choose risk level
- Add nodes
- Auto-add block devices
- Enable 2nd RGW
- Use encrypted OSD
- Remove OSDs
- Block refreshes
MicroCeph: What's Next

• Auto-Clustering via mDNS
• Built-in RGW loadbalancing and HA
• RBD mirroring support
Thank you! Questions?
MicroCeph: Resources

- Source: https://github.com/canonical/microceph
- Talk to us: https://matrix.to/#/#ubuntu-ceph:matrix.org
- Juju charm: https://github.com/canonical/charm-microceph
- MicroStack: https://microstack.run
- MicroCeph GH action: https://github.com/phvalguima/microceph-action