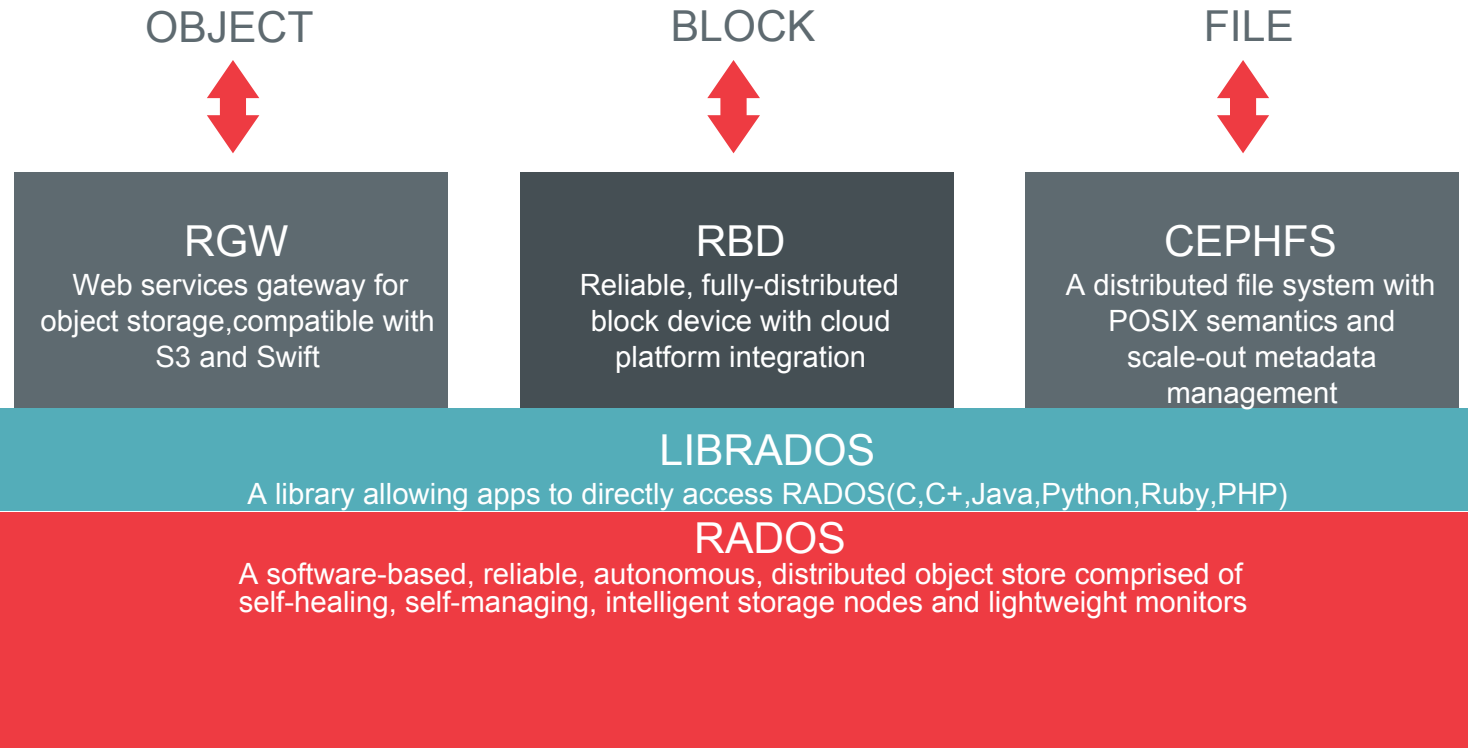
Ceph Rados Block Device

Venky Shankar
Ceph Developer, Red Hat
SNIA, 2017

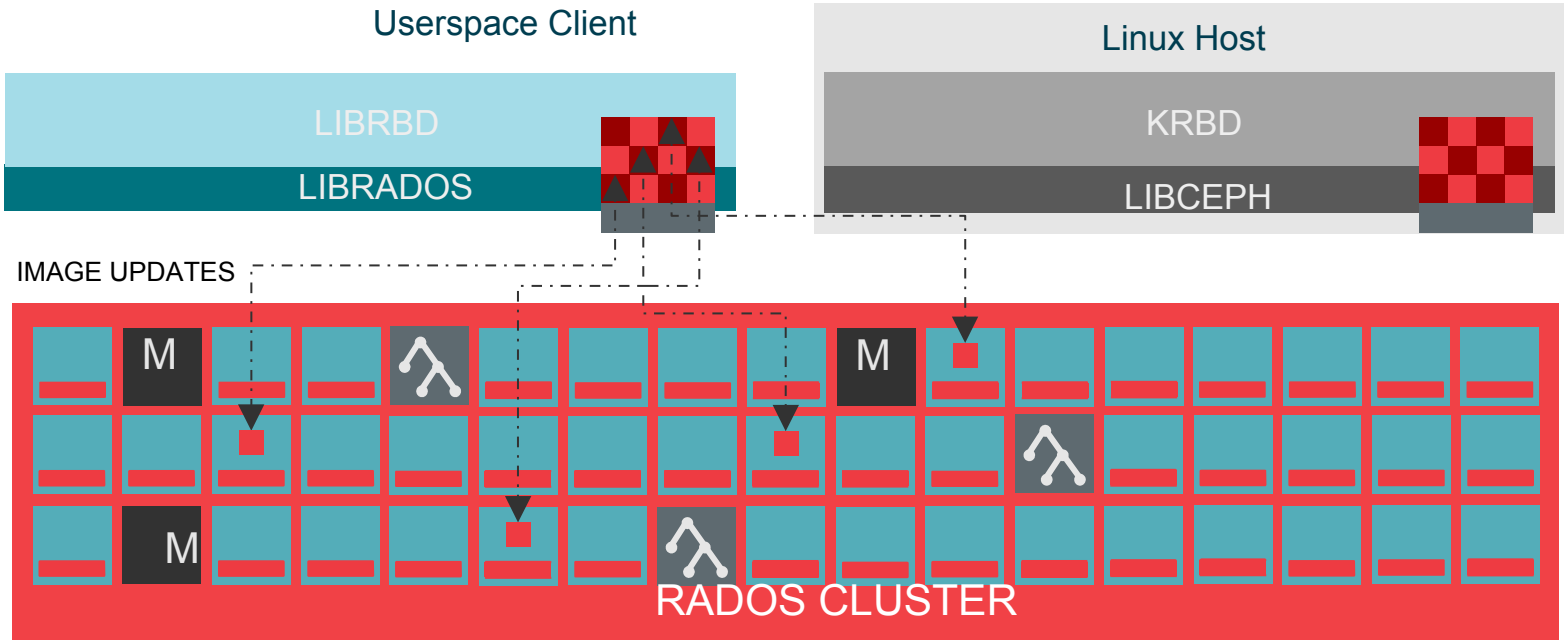
WHAT IS CEPH?

- Software-defined distributed storage
- All components scale horizontally
- No single point of failure
- Self managing/healing
- Commodity hardware
- Object, block & file
- Open source

CEPH COMPONENTS



RADOS BLOCK DEVICE



LIBRADOS

- API around doing transactions
  - single object
- Rich Object API
  - partial overwrites, reads, attrs
  - compound “atomic” operations per object
  - rados classes

RBD FEATURES

- Stripes images across cluster
- Thin provisioned
- Read-only snapshots
- Copy-on-Write clones
- Image features (exclusive-lock, fast-diff, ...)
- Integration
  - QEMU, libvirt
  - Linux kernel
  - Openstack
- Version
  - v1 (deprecated), v2

IMAGE METADATA

- rbd_id.<image-name>
  - Internal ID - locatable by user specified image name
- rbd_header.<image-id>
  - Image metadata (features, snaps, etc…)
- rbd_directory
  - list of images (maps image name to id and vice versa)
- rbd_children
  - list of clones and parent map

IMAGE DATA

- Striped across the cluster
- Thin provisioned
  - Non-existent data object to start with
- Object name based on offset in image
  - rbd_data.<image-id>.*
- Objects are mostly sparse
- Snapshots handled by RADOS
  - Clone CoW performed by librbd

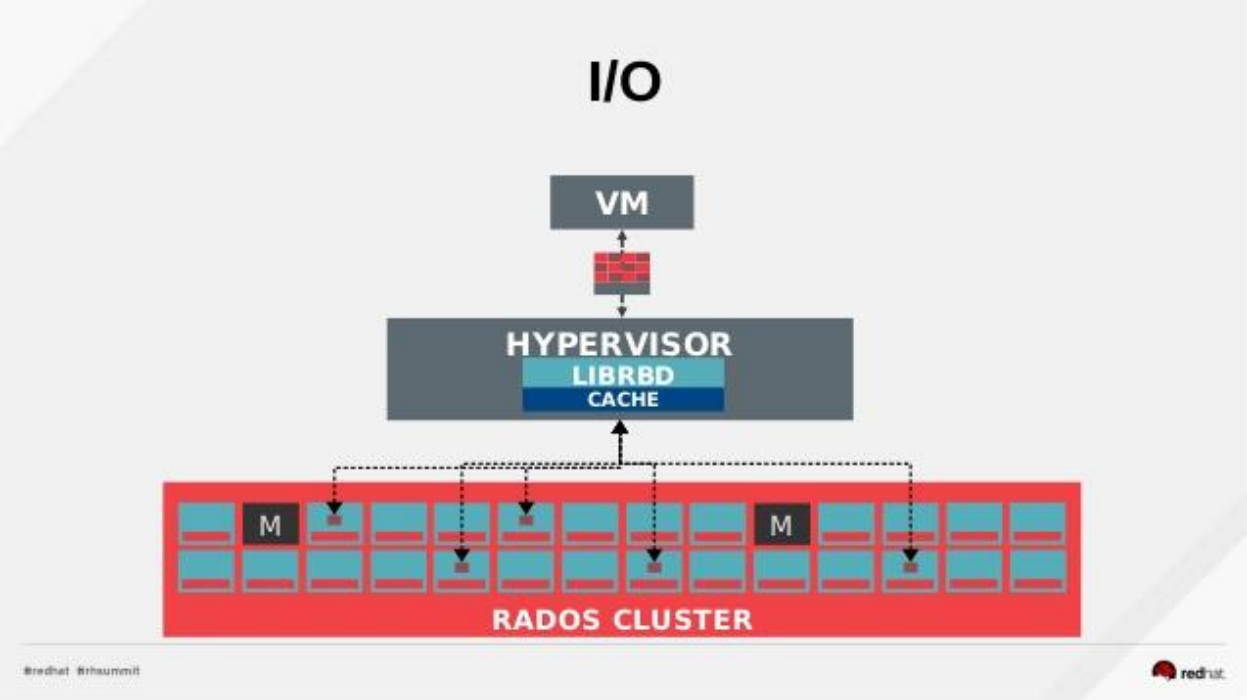
RBD IMAGE INFO

```
rbd image 'r0':
        size 10240 MB in 2560 objects
        order 22 (4096 kB objects)
        block_name_prefix:
rbd_data.101774b0dc51
        format: 2
        features: layering, exclusive-lock,
object-map, fast-diff, deep-flatten
        flags:
```

STRIPING

- Uniformly sized objects
  - default: 4M
- Objects randomly distributed among OSDs (CRUSH)
- Spreads I/O workload across cluster (nodes/spindles)
- Tunables
  - stripe_unit
  - stripe_count

I/O PATH



SNAPSHOTS

- Per image snapshots
- Snapshots handled by RADOS
  - CoW per object basis
- Snapshot context (list of snap ids, latest snap)
  - stored in image header
  - sent with each I/O
- Self-managed by RBD
- Snap spec
  - image@snap

CLONES

- CoW at object level
  - performed by librbd
  - clone has “reference” to parent
  - optional : CoR
- “clone” a protected snapshot
  - protected : cannot be deleted
- Can be “flattened”
  - copy all data from parent
  - remove parent “reference” (rbd_children)
- Can be in different pool
- Can have different feature set

CLONES : I/O (READ)

- Object doesn’t exist
  - thin provisioned
  - data objects don’t exist just after a clone
  - just the metadata (header, etc…)
  - rbd_header has reference to parent snap
- Copy-on-Read (optional)
  - async object copy after serving read
  - 4k read turns into 4M (*stripe_unit*) I/O
  - helpful for some workloads

CLONES : I/O (WRITE)

- Opportunistically sent a I/O guard
  - fail if object doesn’t exist
  - do a write if it does
- Object doesn’t exist
  - copy data object from parent
  - NOTE: parent could have a different object number
  - optimization (full write)
- Object exist
  - perform the write operation

IMAGE FEATURES

- *layering* : snapshots, clones
- *exclusive-lock* : “lock” the header object
  - operation(s) forwarded to the lock owner
  - client blacklisting
- *object-map* : index of which object exists
- *fast-diff* : fast diff calculation
- *deep-flatten* : snapshot flatten support
- *journaling* : journal data before image update
- *data-pool* : optionally place image data on a separate pool
- stripingv2 : “fancy” striping

Kernel RBD

- “catches up” with librbd for features
- USAGE: *rbd map …*
  - fails if feature set not supported
- /etc/ceph/rbdmap
- No specialized cache, uses page cache used by filesystems

TOOLS, FEATURES & MORE

What about data center failures?

RBD MIRRORING

- Online, continuous backup
- Asynchronous replication
  - across WAN
  - no IO slowdown
  - transient connectivity issues
- Crash consistent
- Easy to use/monitor
- Horizontally scalable

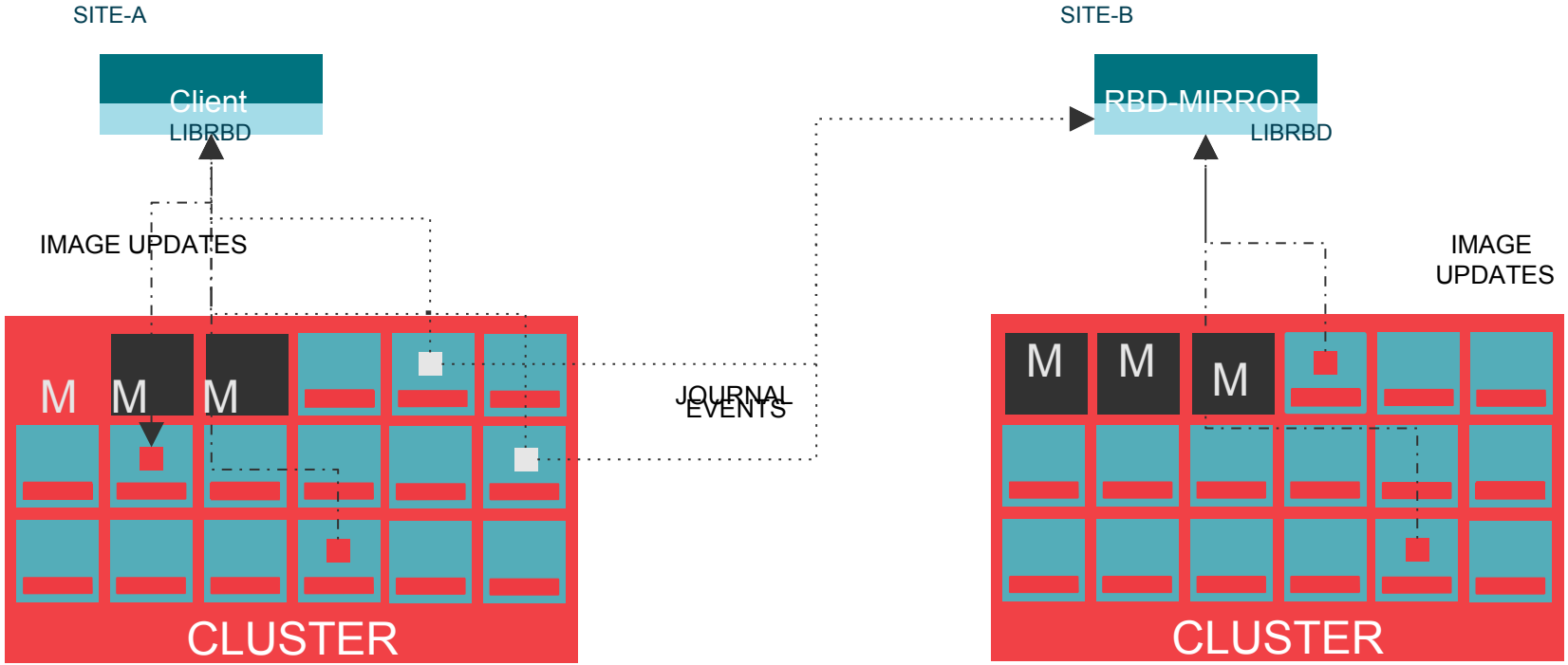
RBD MIRRORING : OVERVIEW

- Configuration
  - pool, images
- Journaling feature
- Mirroring daemon
  - rbd-mirror utility
  - pull model
  - replays journal from remote to local
- Two way replication b/w 2 sites
- One way replication b/w N sites

RBD MIRRORING : DESIGN

- Log all image modifications
  - recall : *journaling* feature
- Journal
  - separate objects in rados (splayed)
  - stores appended event logs
- Delay image modifications
- Commit journal events
- Ordered view of updates

RBD MIROR DAEMON



RBD MIRRORING : FUTURE

- Mirror HA
  - >1 *rbd-mirror* daemons
- Parallel image replication
  - WIP
- Replication statistics
- Image scrub

Ceph iSCSI

- HA
  - exclusive-lock + initiator multipath
- Started out with kernel only
  - LIO iblock + krbd
  - now TCM-User + librbd

Questions?

THANK YOU!

Venky Shankar

vshankar@redhat.com

CEPH ON STEROIDS

- Bluestore
  - newstore + block (avoid double write)
  - k/v store : rocksdb
  - hdd, ssd, nvme
  - block checksum
  - Impressive initial benchmark results
  - helps rbd a lot!