C-Vol Active/Active HA

https://etherpad.openstack.org/p/mitaka-cinder-cvol-aa

Do we want High Availability Active-Active?

- **YES we do**
  - No opposition from those who don't, as long as there's no service degradation.
- Redundancy for SLAs
- Support higher workloads
  - More relevant since Cinder also has *data path*
- A little embarrassing to admit that we don't support Active-Active

Issues - *https://review.openstack.org/232599*

- Complex problem with a lot of related moving parts
- Avoid degrading service on Active-Passive configurations
- We need to pick holes in proposed solutions
  - Removal of Races on API nodes: https://review.openstack.org/207101
  - Job distribution to clusters: https://review.openstack.org/232595
  - Cleanup process of crashed nodes: https://review.openstack.org/236977
  - Data corruption prevention: https://review.openstack.org/237076
  - Remove local locks from the manager: https://review.openstack.org/237602
  - Removing local locks from drivers: https://review.openstack.org/237604

API Races - *https://review.openstack.org/207101*

- With A-A we Increase the chances
- Locks → Concerns on stale DB data
- Swap and compare (conditional updates) - Potential race on error reporting
  - Add generic error to existing errors
  - For loop & generic error → Decrease readability
  - Infinite while loop → Decrease readability & Potential endless loop if update condition and error checking are out of sync
  - Remove all specific errors and just return generic error→ *Terrible idea*

Job distribution - *https://review.openstack.org/232595*

- Active-Passive: 1 host → 1 storage *backend-pool*
- Active-Active: N hosts → 1 storage *backend-pool*
- Group hosts sharing storage → Use same *host* topic queue
- Identify individual hosts inside group for cleanup → Cannot use same *host*
- Add *cluster* logical grouping
  - New configuration option → Defaults to host
  - General rule: change where we use host to cluster
    - Potential problem for rolling upgrades → DB field rename in some cases
- Independent DB heartbeats
  - Aggregation on schedulers
  - DB Model changes
  - API impact

Cleanup - https://review.openstack.org/236977

- On node crash → Clean up DB and storage backend if needed
- Spec proposal: (performance concerns)
  - Use new *workers* DB table to store in flight operations
  - Detect crash using DB heartbeats & Perform cleanup when
    - Node is respawned with same *host* configuration → From the node
      - Needed in case scheduler doesn't notice the crash (time < DB heartbeat)
    - Node is lost → Cleanup from scheduler
  - Works even with multiple schedulers and the node doing cleanup simultaneously
- New idea now that DLM can be a hard requirement
  - Heartbeats → Group membership - Watch group leave = crash
  - Only 1 cleaner→ Leader election - Cleans crashed node's ops where updated < crash time
  - Needs more research → Concern with performance/scalability

Data corruption - *https://review.openstack.org/237076*

- On node crash detection→ Cleanup
- Detection: Heartbeat connection is lost (DB or DLM) but node is not dead
- Must avoid concurrent access from node and cleanup processes
- Access from node to the storage must stop before cleanup
- Options:
  - Autofencing inside Cinder → Quite some work
  - Autofencing outside Cinder → Ugly?
  - Force STONITH mechanism in place for A-A configurations with timing < crash detection → Requires good documentation to prevent misconfigurations.

Local locks

- Manager: *https://review.openstack.org/237602*
- Drivers: *https://review.openstack.org/237604*
- Must preserve queuing → Implicit API contract
- Specs proposal: Use *workers* table from cleanup → Performance concerns
- New idea now that DLM can be a hard requirement
  - Use DLM locks
  - Needs more research → Concern with performance/scalability
- Drivers should try to remove unnecessary locks