Redundancy does not imply fault tolerance

Experiment with Ceph on CORDS

CORDS:

CORDS is a file-system fault injection framework. CORDS always inserts exactly a single fault into a single file-system block at a single node. The fault is only inserted into the application-level on-disk structures. The filesystem metadata is not changed; however, if there is an error in the user data or it becomes inaccessible for some reason then the application is either provided with a corrupt data block or with an error that the application must be able to handle.

We use CORDS with the Ceph file storage to assess the fault-tolerance capabilities of the file system. We use Ceph in the **File Store** mode. We first mount a user-level filesystem, 'errfs' on top of the Ceph file system using FUSE and then obtain the traces for the read, write and append workloads. We then provide these traces to 'cords' that looks into the blocks that have been affected by the various workload operations and injects errors in them through 'errfs' when the same operations are being re-run by it. CORDS then observes the resultant system behavior.

The following types of errors are studied by CORDS:

1. Read Eio: Applications receive EIO on read when there is an underlying latent sector error for one of the sectors from which the data is being read.
2. Write/Append Eio: Applications receive EIO on write if the disk sector onto which data is being written is either unwritable, if the file system is mounted in read-only mode or if the file being written is already corrupted.
3. Append Enospace: If the underlying disk system is full or if the end of the user segment is reached during an append write, then applications receive an Enospace error.
4. Read cg/cz: These are the block-level corruptions that 'errfs' injects. 'errfs' reads the block of data that is to be corrupted, changes its contents and then returns the corrupted block to the application.

These errors have been injected into the Ceph file system when we ran CORDS on top of it. We studied these errors for the following data blocks:

- Superblocks: These blocks are present at the beginning of the filesystem and contain inodes and other metadata information about the filesystem.
- FSID: This is the application-visible file system ID.
- Data: This consists of the contents inside the file.
- Fiemap: These contain the details (start, length etc.) of the extents of the logical volume that are being appended by the file system.

Results:

	Superblock	**FSID**	**Data**	**Fiemap**
Read eio	HW / 1 osd down	HW / 1 osd down	HW / 1 osd down	
Read cg	undetected	HW / 1 osd down	HW / 1 osd down	
Read cz	undetected	HW / 1 osd down	HW / 1 osd down	
Write eio	HO	HW / 1 osd down	HW / 1 osd down	
Append eio				HW / 1 osd down
Append enospace				HW / 1 osd down

HW - Health Warning indicating reduced availability
HO - Health OK indicating full cluster availability
Undetected - no checksum to detect corrupted reads
Blank - not applicable

Experimental setup:

Ceph cluster with 3 OSDs with 100 pgs and each pg has 3 replicas. Ceph is mounted on filestore mode.

Observations:

1. Any local error did not cause a global corruption - No concept of a leader and hence not prone to corruption propagation from follower.
2. For any error, only the OSD in which the error was induced went down. Replicas are still fine, and the cluster is up and running, i.e. read and write are still happening through the other OSDs.
3. For read corruption, only in the superblock the corruption went undetected. In other places it detects corrupted values with checksum and the faulty OSD crashes.
4. For EIO during read/write, the operation is serviced through other replicas while the corrupted OSD crashes in response to error.