xyratex

Advancing Digital Storage Innovation

Repair : Aux-DB

Objective

To implement any additional meta-data tables that are not needed during normal ioservice operations, but are only required for SNS repair.

Keywords

- storage devices - devices attached to data servers
- storage objects (stob) - provide access to storage device contents by means of a linear name-space associated with an object
- component object (cob) - refers to a stob and contains its metadata
- global object (gob) - is an object describing a striped file, by referring to a collection of cobs
- containers - capable of storing other objects

SNS Repair and Aux:DB

- SNS repair proceeds in gob fid order
- A single table is necessary which maps the gob fid to cob fid for every device
- This information is used by storage agents to iterate over gob-fid->cob-fid mapping for its device, selecting the next cob to process.

COB-FID MAP

- C2 database tables are key-value associations.

- A typical record in the cobfid_map will have,

  Key : (device_id, fid)
  Value: cob_fid

  where,
  - device/container id : uint64_t
  - fid (gob fid) : struct c2_fid
  - cob_fid : struct c2_uint128

Note: Tuple of {device_id, fid, cob_fid} is always unique.

Interfaces

1) Insert record to cobfid_map - executed on every ioservice when a file gets created

2) Delete record from cobfid_map - executed on every ioservice when a file gets deleted

3) Enumerate devices/containers (In future, device id will be generalized to container id) - Used by storage agents (typically storage-in agent to iterate over fid-cob_fid mapping to select next cob for repair)

4) Enumerate map (devices + fid)

Enumeration

Enumeration is implemented using the c2_db_cursor interfaces.

The sequence of operation is as follows:

- Create a cursor using c2_db_cursor_init().

- Create an initial positioning key value with the desired device-id and a file fid value of 0, and invoke c2_db_cursor_get() to set the initial position and get the first key/value record. This works because this subroutine sets the DB_SET_RANGE flag internally, which causes a greater-than-equal-to comparison of the key value when positioning the cursor.

- Subsequent records are fetched using c2_db_cursor_next().

- Traversal ends if at any time the device-id component of the returned key changes from the desired device-id, or we've exhausted all records in the database

Iterator support

Purpose:

To minimize the time for which the database lock is held. There will be background concurrent activity when the database is traversed, and one can't hold the lock while iterating through the records.

- THUNK limit for number of record fetches during enumeration operation

- After the limit is reached, release the lock, and then restart the enumeration from where it was last left.

Iterator operations

- Fetch : Loads the next batch of records into the iterator and updates the iterator state to correctly position for the next call.
- Reload : Reload the records from the current position, because the map may have been altered by an intervening call to add or delete a record
- Check for end : Determines if the record in the specified position will exhaust the iterator.

Typical usage

```
struct c2_dbenv mydbenv;
struct c2_addb_ctx myaddb_ctx;
struct c2_cobfid_map mymap;
struct c2_cobfid_map_iter myiter;

uint64_t container_id;
struct c2_fid file_fid;
struct c2_uint128 cob_fid;

/* initialize mydbenv */

/* create or open the map */
rc = c2_cobfid_map_init(&mymap, &mydbenv, &myaddb_ctx, "mycobfidmapname");

/* insert records */
rc = c2_cobfid_map_add(&mymap, container_id, file_fid, cob_fid, &mydbtx);
...

/* enumerate */
rc = c2_cobfid_map_container_enum(&mymap, container_id, &myiter, &mydbtx);
while ((rc = c2_cobfid_map_iter_next(&myiter, &container_id, &file_fid, &cob_fid, &mydbtx)) == 0) {
        /* process record */
}

/* cleanup */
c2_cobfid_map_fini(&mymap);
```

HLD, Source Code and UT

HLD : High level design of Auxiliary Databases for SNS repair
- (Author - Carl)

Source code: core/ioservice/cobfid_map.[ch]

UT code: core/ioservice/cobfid_map.c

Questions ?