```
**********************************************************
  158887 Tue Apr 23 15:33:29 2013
new/usr/src/cmd/ztest/ztest.c
3747 txg commit callbacks don't work
Submitted by:   Will Andrews <willa@spectralogic.com>
Reviewed by:    Matthew Ahrens <mahrens@delphix.com>
**********************************************************
______unchanged_portion_omitted_

4418 /*
4419  * If a number of txgs equal to this threshold have been created after a commit
4420  * callback has been registered but not called, then we assume there is an
4421  * implementation bug.
4422  */
4423 #define ZTEST_COMMIT_CALLBACK_THRESH    (TXG_CONCURRENT_STATES + 2)

4425 /*
4426  * Commit callback test.
4427  */
4428 void
4429 ztest_dmu_commit_callbacks(ztest_ds_t *zd, uint64_t id)
4430 {
4431         objset_t *os = zd->zd_os;
4432         ztest_od_t od[1];
4433         dmu_tx_t *tx;
4434         ztest_cb_data_t *cb_data[3], *tmp_cb;
4435         uint64_t old_txg, txg;
4436         int i, error;

4438         ztest_od_init(&od[0], id, FTAG, 0, DMU_OT_UINT64_OTHER, 0, 0);

4440         if (ztest_object_init(zd, od, sizeof (od), B_FALSE) != 0)
4441                 return;

4443         tx = dmu_tx_create(os);

4445         cb_data[0] = ztest_create_cb_data(os, 0);
4446         dmu_tx_callback_register(tx, ztest_commit_callback, cb_data[0]);

4448         dmu_tx_hold_write(tx, od[0].od_object, 0, sizeof (uint64_t));

4450         /* Every once in a while, abort the transaction on purpose */
4451         if (ztest_random(100) == 0)
4452                 error = -1;

4454         if (!error)
4455                 error = dmu_tx_assign(tx, TXG_NOWAIT);

4457         txg = error ? 0 : dmu_tx_get_txg(tx);

4459         cb_data[0]->zcd_txg = txg;
4460         cb_data[1] = ztest_create_cb_data(os, txg);
4461         dmu_tx_callback_register(tx, ztest_commit_callback, cb_data[1]);

4463         if (error) {
4464                 /*
4465                  * It's not a strict requirement to call the registered
4466                  * callbacks from inside dmu_tx_abort(), but that's what
4467                  * it's supposed to happen in the current implementation
4468                  * so we will check for that.
4469                  */
4470                 for (i = 0; i < 2; i++) {
4471                         cb_data[i]->zcd_expected_err = ECANCELED;
4472                         VERIFY(!cb_data[i]->zcd_called);
4473                 }
```



```
4475                 dmu_tx_abort(tx);

4477                 for (i = 0; i < 2; i++) {
4478                         VERIFY(cb_data[i]->zcd_called);
4479                         umem_free(cb_data[i], sizeof (ztest_cb_data_t));
4480                 }

4482                 return;
4483         }

4485         cb_data[2] = ztest_create_cb_data(os, txg);
4486         dmu_tx_callback_register(tx, ztest_commit_callback, cb_data[2]);

4488         /*
4489          * Read existing data to make sure there isn't a future leak.
4490          */
4491         VERIFY(0 == dmu_read(os, od[0].od_object, 0, sizeof (uint64_t),
4492             &old_txg, DMU_READ_PREFETCH));

4494         if (old_txg > txg)
4495                 fatal(0, "future leak: got %" PRIu64 ", open txg is %" PRIu64,
4496                     old_txg, txg);

4498         dmu_write(os, od[0].od_object, 0, sizeof (uint64_t), &txg, tx);

4500         (void) mutex_lock(&zcl.zcl_callbacks_lock);

4502         /*
4503          * Since commit callbacks don't have any ordering requirement and since
4504          * it is theoretically possible for a commit callback to be called
4505          * after an arbitrary amount of time has elapsed since its txg has been
4506          * synced, it is difficult to reliably determine whether a commit
4507          * callback hasn't been called due to high load or due to a flawed
4508          * implementation.
4509          *
4510          * In practice, we will assume that if after a certain number of txgs a
4511          * commit callback hasn't been called, then most likely there's an
4512          * implementation bug..
4513          */
4514         tmp_cb = list_head(&zcl.zcl_callbacks);
4515         if (tmp_cb != NULL &&
4516             (txg - ZTEST_COMMIT_CALLBACK_THRESH) > tmp_cb->zcd_txg) {
4516             tmp_cb->zcd_txg > txg - ZTEST_COMMIT_CALLBACK_THRESH) {
4517                 fatal(0, "Commit callback threshold exceeded, oldest txg: %"
4518                     PRIu64 ", open txg: %" PRIu64 "\n", tmp_cb->zcd_txg, txg);
4519         }

4521         /*
4522          * Let's find the place to insert our callbacks.
4523          *
4524          * Even though the list is ordered by txg, it is possible for the
4525          * insertion point to not be the end because our txg may already be
4526          * quiescing at this point and other callbacks in the open txg
4527          * (from other objsets) may have sneaked in.
4528          */
4529         tmp_cb = list_tail(&zcl.zcl_callbacks);
4530         while (tmp_cb != NULL && tmp_cb->zcd_txg > txg)
4531                 tmp_cb = list_prev(&zcl.zcl_callbacks, tmp_cb);

4533         /* Add the 3 callbacks to the list */
4534         for (i = 0; i < 3; i++) {
4535                 if (tmp_cb == NULL)
4536                         list_insert_head(&zcl.zcl_callbacks, cb_data[i]);
4537                 else
4538                         list_insert_after(&zcl.zcl_callbacks, tmp_cb,
4539                             cb_data[i]);
```

```
4541                 cb_data[i]->zcd_added = B_TRUE;
4542                 VERIFY(!cb_data[i]->zcd_called);

4544                 tmp_cb = cb_data[i];
4545         }

4547         (void) mutex_unlock(&zcl.zcl_callbacks_lock);

4549         dmu_tx_commit(tx);
4550 }
```

_____***unchanged_portion_omitted_***

****
21488 Tue Apr 23 15:33:30 2013
new/usr/src/uts/common/fs/zfs/txg.c
3747 txg commit callbacks don't work
Submitted by: Will Andrews <willa@spectralogic.com>
Reviewed by: Matthew Ahrens <mahrens@delphix.com>
****
_____***unchanged_portion_omitted_***

```
 388 /*
 389  * Dispatch the commit callbacks registered on this txg to worker threads.
 390  */
 391 static void
 392 txg_dispatch_callbacks(dsl_pool_t *dp, uint64_t txg)
 393 {
 394         int c;
 395         tx_state_t *tx = &dp->dp_tx;
 396         list_t *cb_list;

 398         for (c = 0; c < max_ncpus; c++) {
 399                 tx_cpu_t *tc = &tx->tx_cpu[c];
 400                 /* No need to lock tx_cpu_t at this point */

 402                 int g = txg & TXG_MASK;

 404                 if (list_is_empty(&tc->tc_callbacks[g]))
 405                         continue;

 407                 if (tx->tx_commit_cb_taskq == NULL) {
 408                         /*
 409                          * Commit callback taskq hasn't been created yet.
 410                          */
 411                         tx->tx_commit_cb_taskq = taskq_create("tx_commit_cb",
 412                             max_ncpus, minclsyspri, max_ncpus, max_ncpus * 2,
 413                             TASKQ_PREPOPULATE);
 414                 }

 416                 cb_list = kmem_alloc(sizeof (list_t), KM_SLEEP);
 417                 list_create(cb_list, sizeof (dmu_tx_callback_t),
 418                     offsetof(dmu_tx_callback_t, dcb_node));

 420                 list_move_tail(cb_list, &tc->tc_callbacks[g]);
 420                 list_move_tail(&tc->tc_callbacks[g], cb_list);

 422                 (void) taskq_dispatch(tx->tx_commit_cb_taskq, (task_func_t *)
 423                     txg_do_callbacks, cb_list, TQ_SLEEP);
 424         }
 425 }
```

_____***unchanged_portion_omitted_***