Lustre Test Plan
Version recovery phase 2
Large Scale Test Plan

Author	Date	Description of Document Change	Client Approval By	Client Approval Date
Elena Gryaznova	2008-10-17	draft		

I. Test Plan Overview

Executive Summary

- Statement of the problem trying to solve:
  Test at scale the following version based recovery feature landed into b1_8_gate branch
  -- vbr_interop
  -- vbr_exp
  -- vbr_orphans
- Required inputs:
  -- b1_8_gate branch, with landed :
  vbr_interop: attachment 17435, attachment 17694 bug 15942
  vbr_exp: attachment 18379 bug 15391
  vbr_orphans: attachment (still in code phase) bug 15292
  -- the packages: Cray build
- Hardware to be used:
  -- Cray systems
- Expected output:
  -- The current status of the listed features
  -- The following tickets will be used for summary and a status of testing.
  vbr_interop: bug 15942
  vbr_exp: bug 15391
  vbr_orphans: bug 15392
  large-scale VBR tests: bug 17195

Problem Statement

I. We need to test VBR at scale. This test plan lays out what tests need to be run to verify this feature.

Goal

Verify that VBR functions with a large system

Success Factors

All tests need to run successfully.

Testing Plan

Pre-gate landing

Define the setup steps that need to happen for the hardware to be ready? Who is responsible for these tests?
Get system time at Cray on a small and then large system. Pre-feature testing has been completed and signed off by SUN QE for this feature.

Specify the date these tests will start, and length of time that these test will take to complete.
Date started: 2008-10-30
The time estimation for new test creation: 1 week The time estimation of 1 run: VBR feature tests (large-scale) : 2 days

Common recovery tests : 4 days Summary for 1 post-landing tests cycle: 6 days * In the case of defects found the tests should be repeated. The estimated time of completed testing depends on: -- the number of defects found during testing; -- the time needed by developer to fix the defects;

Specify (at a high level) what tests will be completed?
Functional tests: new acceptance-small tests: large-scale, recovery-scale (without lustre reformatting)

Specify how you will restore the hardware (if required) and notify the client your testing is done.
We will need feedback from Cray, recommend we use BZ for outputs.
The bugzilla ticket is filed for each failure
Summary and status report are printed in the bug that we create for this test

Test Cases

Test Cases
Post-gate landing
All these tests are (will be) integrated into acceptance-small as large-scale.sh (LARGE_SCALE). To run this large scale test: 1. Install lustre.rpm and lustre-tests.rpm on all cluster nodes. 2. Specify the cluster configuraion file, see cfg/local.sh and cfg/ncli.sh for details. 3. run the test without lustre reformatting as: SETUP=: CLEANUP=: FORMAT=: ACC_SM_ONLY=LARGE_SCALE NAME=<config_file> sh acceptance-small.sh or SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> sh large-scale.sh Requirements: 1. Installed Cray lustre build packages on all cluster nodes. 2. installed lustre user tools (lctl). 3. shared directory with lustre-tests build on all clients 4. formatted lustre fs, mounted by clients 5. the configuration file in according to formatted lustre 6. installed dd, tar, dbench, iozone
I. **Feature tests for exports:** 1.a measure N clients connection time without delayed exports (and orphans).

1.b create many delayed exports 1000/10000/100000, measure time for clients connection time - connect N new clients
large_scale.sh test_1b

1.c create many delayed exports 1000/10000/100000, measure recovery time: connect several clients, create/delete the number of files, fail mds. Print the statistic number of delayed exports/ the number of clients/ the time of connections
large_scale.sh test_1c

1.d create many delayed exports 1000/10000/100000, expire all exports and connect new clients, this will invoke massive orphans cleanup. Measure connection time.
large_scale.sh test_1d

1.e create many delayed exports 1000/10000/100000, make export expired one-by-one with delay 30 sec, measure recovery time: connect several clients, create/delete the number of files, fail mds. Print the statistic number of delayed exports/ the number of clients/ the time of connections. This should invoke contstant exports+orphans cleanup during test.
large-scale.sh test_1e

II.

Feature tests for orphans

2.b create many delayed exports 1000/10000/100000 with orphaned files (1000/10000/100000), measure time for clients connection time - connect N new clients
large_scale.sh test_2b

2.c create many delayed exports 1000/10000/100000 with orphaned files (1000/10000/100000), measure recovery time: connect several clients, create/delete the number of files, fail mds. Print the statistic number of delayed exports/ the number of clients/ the time of connections
large_scale.sh test_2c

2.d create many delayed exports 1000/10000/100000 with orphaned files (1000/10000/100000), measure the connection time.
large_scale.sh test_2d

COMMON scale tests

All these tests are (will be) integrated into acceptance-small as recovery-scale.sh (RECOVERY_SCALE).
To run this recovery scale tests over acceptance-small:
    SETUP=: CLEANUP=: FORMAT=: ACC_SM_ONLY=RECOVERY_SCALE NAME=<config_file> sh acceptance-small.sh
To run all recovery scale tests:
    SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> sh recovery-scale.sh
To run the recovery scale tests separately:
    SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> DURATION=<duration> recovery-mds-scale.sh
    SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> recovery-double-scale.sh

III.
Scale recovery tests (NC:1M:MO) (the test based on test11/17 from CMD3 project)

```
Scale recovery tests (NC:1M:MO) (the test based on test11/17 from CMD3
project)

3.a (was test11 in cmd3 project)
For defined duration (1-24 hours) repeatedly fail an MDS at defined (5-10
minutes) intervals and verify that no application errors occur. Load of
clients: dd tar dbench iozone.

recovery-mds-scale.sh

Example:
    SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> DURATION="3600" sh
recovery-mds-scale.sh


3.b (was test17 on cmd3 project)
Fail a random pair of nodes at defined (5-10 minutes) intervals and verify
that no application errors occur. Load of clients: dd tar dbench iozone.
1: failover MDS, then OST
2: failover MDS, then 2 clients
4: failover OST, then another OST
5: failover OST, then 2 clients
6: failover OST, then MDS
7: failover 2 clients, then MDS
8: failover 2 clients, then OST
9: failover 2 clients, then 2 different clients

recovery-double-scale.sh

Example:
  SETUP=: CLEANUP=: FORMAT=: NAME=<config_file> sh recovery-mds-scale.sh
```

IV. More tests can be added after discussion (see bug 17195).

Benchmarking

No benchmarks will be done.

II. Test Plan Approval

- Review date for the Test Plan review with the client:
  10/17/08 – reviewed by J.D. Neumann
- – reviewed by Ed Giesen
- Date the Test Plan was approved by the client (and by whom)
  08/11/2008 – approved by Charlie Carroll

- Date(s) agreed to by the client to conduct testing

III. Test Plan – Final Report

Test Results

Benchmarking

Conclusions

Summary of the test:

-
-

Next Steps

Define any next steps as a result of the test. Does the test need to be performed again? Are the results sufficient to be considered a success?

-
-
-