Lustre Quotas Test Plan

Author	Date	Description of Document Change	Approval By	Approval Date
Walter Poxon	2008.06.10	First draft	N/A	N/A
Walter Poxon	2008.06.23	Add (single-user) test cases based on conversations re: quotas testing		
Walter Poxon	2008.06.26	Incorporate review comments from Yu Jian (Jian.Yu@sun.com)		
Walter Poxon	2008.07.10	add info on where to find quotas test scripts		
Yep	2008.9.12	Add test plan for Quotas Port for HEAD		
Ed Giesen	2008.11.05	Remove draft notes.		

I. Test Plan Overview

Executive Summary

- Statement of the problem: Test quotas using test cases
- Required inputs: Test team
- Hardware to be used:
- Software to be used: IOR (with and without O_DIRECT) run from quotas test scripts from
- Expected outputs: No errors when the test cases in this test plan are run
- Future actions:

Problem Statement

The quotas feature in Lustre is a relatively new and relatively complex feature which deserves targeted and specific testing to ensure its proper operation with all new Lustre releases.

The following test plan outlines steps for testing block and inode quotas for users and groups in Lustre.

Goal

The goals for testing are:

1. Thoroughly test the quotas feature (block and inode quotas for users and groups) of each new Lustre release during pre-release testing
2. Focus on quotas hard limits
3. Find and document (and add as release blockers) any quotas problems found
4. The ultimate goal is to deliver working quotas feature code to the users. with each new Lustre release and avoid bugs (especially regressions)

In the design and development of test cases for this test plan, focus will be placed on testing block and inode **hard** quota limits for users and groups.

Focus will not be placed on testing block and inode soft quota limits at this time due to the high priority of testing the hard quotas behavior and the relatively lower priority of testing the behavior of soft quotas at this time.

The components to be tested are:

1. ldiskfs
2. Lustre
3. lustre quotas feature code

Success Factors

Success will depend on error free running of all the testsuites and other utilities. All the tests should succeed without any errors like boundary or limit errors, kernel crashes.

Additionally, in all quotas tests, we are looking for two things: **not able to write more than 1% beyond** a quota limit and **able to write at data files when current usage is less than or equal to 99%** of user's quota limit:

Ideally, these success factors would be derived from a requirements document. Lacking a requirements document for this feature, these success criteria are derived from conversations with members of the User communtity.

Testing Plan

Define the setup steps that need to happen for the hardware to be ready? Who is responsible for these tests?

1. Boot a system with Lustre-1.4.12 (or newer) or Lustre-1.6.5 (or newer)
2. Enable quotas (lfs quotacheck) on the Lustre test filesystem. We should use “lfs quotacheck” instead of “lfs quotaon” to ensure that the quota database info matches the quotas data which is stored in the filesystem.
3. Have at least one user account available (besides root) for testing user quotas
4. Obtain copies of quotas test scripts

Specify the date these tests will start, and length of time that these test will take to complete.

Recurring testing (once per release).
Estimated time to complete: 1 day

Specify (at a high level) what tests will be completed?

The test to be completed are:
1. run quota_loop_block.sh to test block user quotas
2. run quota_loop_block_group.bash to test block group quotas
3. run quota_loop_inode.bash to test inode user quotas
4. run quota_loop_inode_group.bash to test inode group quotas

Run each of the above scripts.

Check: do quotas work for users? (block plus inode quotas)
Check: do quotas work for groups? (block plus inode quotas)

Then: run several copies of each script (With the same user but different directories where the scripts are generating their files – to simulate a user running multiple jobs at the same time)

Then: (last step): run with multiple users and multiple groups.

In all quotas tests, we are looking for two things: **not able to write more than 1% beyond** a quota limit and **able to write at data files when current usage is less than or equal to 99%** of user's quota limit:

1. When the tests intentionally try to exceed a quota limit, we want to see the I/O fail with EDQUOT error returned, and we want to see that the user is not able to write more than 1% beyond their quota limit.

2. When the tests try to write up to (but not exceed) a quota limit, we want to see the user's I/O continue to work and not get EDQUOT when the user has not yet reached their quota limit. The user should be able to write up to 99% of their quota limit without receiving EDQUOT errors returned when they try and write more data.

The test scripts mentioned above use the IOR test program to generate the I/O requests used for filling the filesystem to test quotas.

Specify hardware resources required

Large clusters with large numbers of client nodes are not necessary (at least at this point) .
On the order of 8 client nodes can be used for this testing.

The lustre configuration should be:
One MDS
One or Two OSS
Four or Eight OST's (with 40Gigabytes per OST as a minimum)
As many Clients as possible

Test Cases

Quotas tests can be divided into two groups: functional tests and performance tests.

The quotas functional tests to be run are:

1. Test block quotas for users (with and without O_DIRECT) run in single user, multi-user, and scramble modes (see below) using `quota_loop_block.bash`
2. Test block quotas for groups (with and without O_DIRECT) run in single user, multi-user, and scramble modes (see below) using `quota_loop_block_group.bash`
3. Test inode quotas for users (with and without O_DIRECT) run in single user, multi-user, and scramble modes (see below) using `quota_loop_inode.bash`
4. Test inode quotas for groups (with and without O_DIRECT) run in single user, multi-user, and scramble modes (see below) using `quota_loop_inode_group.bash`
5. Recovery testing with quotas enabled
6. Upgrade testing with quotas enabled

Mode 1 (single-user):	run each script above individually (one script running at a time, and never more than one script running at a time).
Mode 2 (multi-users):	for each script above, run several copies of that script at the same time using the same user (for user tests) or group (for group tests) but have each copy of the test writing to different directories.
Mode 3 (scramble):	run all of the script above simultaneously. Configure some to run with the same users and groups and some to run with different users and groups.

Important test script options to be aware of (please see the test scripts themselves for complete lists of all available options):

-d <directory> - directory to run in. This is where IOR will write its files. This should be a directory on a lustre filesystem. For multi-user and scramble-mode tests, different copies of the scripts should be run with different -d options.

-L <debug_dir> - directory where debug files can be written. Should probably be a non-lustre (NFS?) directory where log files will be preserved if we experience a lustre failure.

-i <iterations> - number of iterations to run

-l <lustre_fs> - the name of the lustre filesystem (toplevel mountpoint directory such as /mnt/lustre or /mnt/testfs)

-M <mds_name> - the name of the MDS (ie., nid00042)

Outline of basic Test logic for the quota_loop_{block,inode}[_group].bash scripts. This example shows the logic for a user-oriented test of block quotas, but the logic for the other (inode and group tests) is similar:

TODO: upload the quota_loop_{block,inode}[_group].bash test scripts to a bugzilla bug report so that other Lustre testers have access to these scripts.

set a quota (Example: *lfs setquota -u $user $block-softlimit $block-hardlimit $inode-softlimit $inode-hardlimit $lustre)*

run IOR in a loop to fill the fs to to just above the quota we just set and get EDQUOT error

Example: aprun -n $processor_count IOR $IOR_options -o $directory/$file

check that IOR exited for quota exceeded (EDQUOT)

Error: if “lfs quota” display on client does not show any asterisks in its output and the block count for the user exceeds the quota limit for the user (means lustre thinks the user is not over quota when the user really is over quota)

Error: if “lfs quota” display does show one or more asterisks in its output and the block count for the user does not exceed the quota limit for the user (means lustre thinks the user is over quota when this is not the case)

Error: if none of the “lfs quota” displays run on each of the OST's does not show any asterisks in its output and the block count for the user exceeds the quota limit for the user (means lustre thinks the user is not over quota when the user really is over quota)

Error: if none of the “lfs quota” displays run on each of the OST's shows one or more asterisks in its output and the block count for the user does not exceed the quota limit for the user (means lustre thinks the user is over quota when the user really is not over quota)

All the test results should be logged and saved for future reference.

Functional Test Cases

1. (single-user) Test user block quotas

```
$ ./quota_loop_block.bash -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs -M
nid00068
```

2. (single-user) Test user inode quotas

```
$ ./quota_loop_inode.bash -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs -M
nid00068
```

3. (single-user) Test group block quotas

```
./quota_loop_block_group.bash -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs
-M nid00068
```

4. (single-user) Test group inode quotas

```
./quota_loop_inode_group.bash -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs
-M nid00068
```

5. (single-user) Test user block quotas with O_DIRECT

```
$ ./quota_loop_block.bash -O -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs -
M nid00068
```

6. (single-user) Test user inode quotas with O_DIRECT

```
$ ./quota_loop_inode.bash -O -d /mnt/testfs -L /tmp -i 2 -l /mnt/testfs -
M nid00068
```

7. (single-user) Test group block quotas with O_DIRECT

```
./quota_loop_block_group.bash -O -d /mnt/testfs -L /tmp -i 2 -l
/mnt/testfs -M nid00068
```

8. (single-user) Test group inode quotas with O_DIRECT

```
./quota_loop_inode_group.bash -O -d /mnt/testfs -L /tmp -i 2 -l
/mnt/testfs -M nid00068
```

9. Test recovery with quotas enabled

test case command line details TBD

10. Test upgrade with quotas enabled

test case command line details TBD

N. TBD... the test cases for (multi-user) and (scramble) will be added to this test plan once the (single-user) test cases have been executed and debugged. They are actually relatively simple

variations to the test cases above.

Performance Test Cases
1. Test performance using IOR with quotas ON and then with quotas off. Compare results and be sure to report any performance degradations from previous release to QE management. *Several things should be considered:* *Between different releases:* *Quotas ON performance levels should not be slower than previous lustre releases with Quotas ON. For example, if the Quotas ON performance of lustre-1.6.6 is slower on any tests than lustre-1.6.5 with Quotas ON, this represents a performance regression and should be discussed with QE management and most likely a bug should be opened to provide information to developers so they may investigate the cause.* *Within a release:* *Quotas ON performance should be within X% of Quotas OFF performance for all tests below.* *If Quotas ON performance is more than X% slower than Quotas OFF performance for the same test on the same version of the lustre code, these results should be discussed with QE management and most likely a bug should be opened to provide information to developers so they may investigate the cause.*

1. Performance test:

```
$ ./IOR...
```

N.

II. Test Plan Reviews and Approval

Reviews:

- Reviewed by Yu Jian on 2008.06.26
- Reviewed by CR Schult on 2008.07.07

Quotas Port for HEAD

1..QE: peng.ye@sun.com
2.Time plan:
End Date:09/23/2008
3.Tracking Tickets on Bugzilla: 13058
4.Test Matrix:

	rhel5/x86_64	rhel5/x86_64
sanity-quota	[1]	[2][3][4]
Ior with quota on	[1]	
Pios with quota on	[1]	

[1] Test b_head_quota with GSS/krb5 disabled
[2] Test b_head_quota with GSS/krb5 enabled
[3] Test b_head_quota MDS/OSS capabilities security level = 1
[4] Test b_head_quota with enabled remote_client

III.Test Plan – Results Reporting

Testing of quotas should precede each new release of Lustre. Due to the criticality of this feature for certain customers, quotas test failures should be opened as release-blocker bugs for any pre-releases of Lustre which are undergoing testing.

Test Results

Test results should be checked for failures.

Any test failures should be documented as bugs in the Lustre Group bugzilla tool. Please search first for existing bugs before opening new and possibly duplicate bugs.