Fermi File Transfer Service - Feature #10537

If no layer 4 info from enstore, use layer 1 to look up the BFID and retrieve size, checksum (if relevant) and tape info direct from enstore

10/14/2015 07:02 PM - Robert Illingworth

Status:	Feedback	**Start date:**	10/14/2015
Priority:	Normal	**Due date:**	
Assignee:	Robert Illingworth	**% Done:**	100%
Category:		**Estimated time:**	0.00 hour
Target version:	v6_0_0	**Spent time:**	0.00 hour

Description

If there is no layer 4 info, but the BFID is available from layer 1, then use it to look up the file directly in enstore with the enstore info command.

History

#1 - 10/29/2015 05:25 PM - Dennis Box

- Status changed from Assigned to Feedback

- Assignee changed from Dennis Box to Robert Illingworth

I have pushed branch 10537 to the origin for review and possible merge to master.
I started with branch #10046 and modified it to read layer1 if layer4 was not readable. If layer1 is used it reads the bfid to query all the information from enstore that layer4 should have returned.

The test program test_pnfs.py reads file metadata using both layer1 and layer4 and fails if they are different. I have not found a file yet where this is the case.

To run the test case:
. /grid/fermiapp/products/common/etc/setups.sh
setup python v2_7_6
setup pyOpenSSL v0_15_1
setup twisted v15_2_1
setup ifdhc v1_8_5
setup cpn v1.7
setup sam_cp v9_0_7
setup cffi v1_1_2
setup six v1_9_0
setup sam_web_client v1_9
setup encp v3_11 -q stken
export FILETRANSFERSERVICE_DIR=$HOME/filetransferservice
export PATH=$FILETRANSFERSERVICE_DIR/bin:$PATH
export PYTHONPATH=$FILETRANSFERSERVICE_DIR/python:$PYTHONPATH
export PNFS_TEST_FILE=(some file mounted where you can see it)

cd $FILETRANSFERSERVICE_DIR/python/test
trial test_pnfs.py

changes: (some master and some from #10046):

config.py:readDCacheChecksums() Looks for an entry 'read-dcache-checksums' in the [main] section
of the config file. If entry is True or absent, method returns True.
If the entry is False, method returns False.

pnfs.py:getEnstoreFileDetails() New optional parameter, readFromDCache=True . If true,
reads the adler32 checksum from dcache metadata ,
and puts it in the dictionary that the method returns
IF layer4 metadata is not readable, read bfid from layer1
and use this to query file metadata details directly from
enstore. Returns same information that would have come
from layer4 in this case.

filestate.py:_checkLabel() Calls getEnstoreFileDetails() with config.readFromDCache() value

filestate.py:_handlePnfsResults(): Flow of control used to switch on whether getEnstoreFileDetails() returned
an empty dictionary or not. Since enstore (layer4) and dcache (get)
metadata can both now be called, returned dictionary can have one entry

in it if layer4 unreadable. Flow of control now switches on whether number
of entries in dictionary is greater than 1.

#2 - 02/18/2016 07:07 PM - Dennis Box

I have pushed changes to branch 10537 to the fts repository for review. I think it correctly implements all of changes required by the 3 fts tickets assigned to me, namely

- looking up BFID from level 1 if level 4 read doesn't work (this ticket #10537)
- generating and storing adler32 checksums using enstore checksum and filesize (#1047)
- read adler32 checksums directly from dcache (#1046)

Testing all this:

There are two test directories now, filetransferservice/test (integration tests) and filetransferservice/python/test (unit tests). Some of the integration tests require setting up a 'fake' /pnfs test area which requires root access on the test machine. The 'recipe' I used for making the 'fake' pnfs is in comments in filetransferservice/test/run_server_test3.sh

The integration tests are:

- run_server_test.sh : original test
- run_server_test2.sh: test that 'allowed-checksum-types = enstore adler32' adds adler32 to SAM metadata after enstore checksum calculation #1047
- run_server_test3.sh: test that 'read-dcache-checksums = True' reads adler32 checksum from pnfs metadata in scan-dir #1046
- run_server_test4.sh: test that level4 pnfs metadata in transfer-to ends up putting tape label into sam metadata
- run_server_test5.sh: test that level1 pnfs metadata in transfer-to ends up putting tape label into sam metadata #100537

filetransferservice/python/test contains two tests, test_config.py and test_pnfs.py.
they are run with 'trial test_pnfs.py' and 'trial test_config.py'

There are two new entries in the .ini file which controls the behavior of these features:
allowed-checksum-types = list of allowed types, default is 'enstore' , tested values are 'enstore adler32' .

#3 - 10/20/2017 09:49 AM - Robert Illingworth

- Target version changed from 818 to v6_0_0

#4 - 10/20/2017 09:49 AM - Robert Illingworth

- % Done changed from 0 to 100