



Commit on sharing transfer of information

Alexander. Zarochentsev@sun.com
Sun Microsystems, Inc.



Introduction

- dependent uncommitted transactions lead to recovery problems in multi node failure cases
- COS eliminates inter-client dependent transactions by doing commits between them

Design and implementation details

- LDLM based COS
 - > new COS lock to protect MD updates after the transaction closed
 - > inter-client dependency detected as a LDLM lock conflict between COS lock and another client lock
- COS and REP-ACK relationship
 - > Existing infrastructure for preserving LDLM locks after the transaction is closed (AKA REP-ACK, `ptlrpc_save_lock`) is reused for COS
 - > REP-ACK is to guarantee that the client will replay the request after MDS failure, COS establishes higher level of reliability by ensuring the the transaction has reached a stable storage

Implementation details (Cont)

- PTLRPC changes
 - > disabling ACKs for difficult replies

Performance problems & solutions

- > COS got performance problems (13% loss of file creation speed) in case of no COS conflicts.
- > PTLRPC request/reply handling code was not good to massive processing of deferred reply state objects (tens of thousands each 5 seconds)
- > reply state object locking was optimized and reply handling moved to dedicated threads.

references

- BZ15939 Commit on Share
- BZ17461 COS performance degradation