Tracker ticket for project FLR (LDEV-424)

[LDEV-605] Server Local Clients(SLC) implementation Created: 22/Mar/17 Updated: 06/Apr/17

Status:	Open
Project:	Lustre Development
Component/s:	None
Affects Version/s:	None
Fix Version/s:	None

Type:	Sub-Task	**Priority:**	Minor
Reporter:	Jinshan Xiong	**Assignee:**	Jinshan Xiong
Resolution:	Unresolved	**Votes:**	0
Labels:	None		

Description

Definition
========
SLC are full functional clients mounted on Lustre nodes. The ultimate purpose of SLC is to reduce the overhead of Lustre protocol when dealing with bulk data move. Typical use case for SLC would be data migration, data resync for FLR, etc.
In order to avoid interference with server workload, SLC should use direct IO to read and write data to local and remote OSTs. No regular jobs should be scheduled to SLC, instead they will be used to handle housekeeping jobs.

Operational Model
==========
SLC are mounted by `-o slc` flag. When this flag is seen by MDC and OSC, it will detect if the target going to connect is on the same node of the SLC. In that case, a connect flag OBD_CONNECT_SLC will be set so that the target knows the export is from local.

Since SLC and target are on the same node, this implies the recovery will never succeed if the server node crashes because SLC is dead at the same time. When targets see OBD_CONNECT_SLC flag, they will disable recovery from that particular client. For the same reason, the pinger on the SLC should be disabled for the local targets.

SLC are indistinguishable from normal clients when talking to non-local targets.

Locking and IO operations
==========
SLC are able to cache ldlm locks as normal clients do. However, when the SLC issue write operations, targets must complete the operations in SYNC trans. This is a strong requirement that implies only carefully chosen jobs can be performed by SLC; or jobs have to designed carefully to make it be able to perform by SLC.

Usage for FLR
==========
FLR uses SLC for data resync. MDS will send a request, via non-lustre specific data connection, to the SLC where the valid replica resides. If the valid replica is consisted of multiple stripes, the SLC should parse the layout and use stride read and then transfer to remote targets. This should be easy to implement because SLC are fully functional clients and they understand the layout.

Open Issues & discussions
==========
I thought about extending LWP for SLC but it turned out to be a bad idea because it seems hard to extend LWP to issue BRW IO requests.
Ideally OSC should talk to OFD directly for SLC, this becomes difficult in CLIO. In order to simply the implementation, we should only bypass LNET but keep most of the PTLRPC functionalities. However, special interfaces should be introduced to put RPC requests in server queue directly.

Comments

Comment by Andreas Dilger [23/Mar/17]

The "-o slc" mount option is not really needed. Even with this option, the client still needs to scan all of the imports to find local connections (0@lo, or if the local and remote NID are the same) and send extra connection flags. The main reason to add it would be to keep the current async behavior for local client mounts. That said, I think "slc" is not a very descriptive name, since there may be other reasons for users to use this. Better would be "-o local" or "-o sync".

We could just have the client and server automatically detect local connections and handle the recovery issues automatically (O_DIRECT reads and writes on the client, not adding the client to the last_rcvd file on the server).

Also, I don't think there is a requirement for the MDS->SLC communication to be done by the kernel. This could all be done in userspace via lfs. It can also know the file layout and do sparse reads and writes to replace a single failed object in a component. This is the "Optimized Resync Tool" I described in the use cases document.

Comment by Jinshan Xiong [23/Mar/17]

> The "-o slc" mount option is not really needed. ... Better would be "-o local" or "-o sync".

Yes, I will pick `-o local`.

> Also, I don't think there is a requirement for the MDS->SLC communication to be done by the kernel. This could all be done in userspace via lfs.

`lfs` won't be good enough for this job. I'm thinking about starting daemons on both MDS and OSS. The daemon on the MDS will talk with daemons on OSS for task distribution. And daemons access Lustre via SLC.

This will be described in detail in ~~LDEV-606~~.

Comment by Malcolm Cowe [04/Apr/17]

The description for the SLC says that the mount will appear to work like a regular client. Does this mean that it could be used to host other applications, e..g NFS exports? I have a task in mind to create a blue print for Lustre where NFS and other protocols can be configured on the servers – being careful, of course, not to overload the systems.

Comment by Jinshan Xiong [06/Apr/17]

No, only carefully designed work can be run on SLC due to lack of recovery to local OSTs. NFSD that can only generate regular IOs is not feasible running on SLC because node crash will cause data corruption.