High Level Design for group locks

May 5, 2006

1 Functionality specification

Possibility to lock files (inodes) for exclusive access by only processes belonging to a logical group of processes.

2 Logic specificatioN

To support certain HPC installations, Lustre supports a group I/O lock. The semantics of the lock are as follows:

1. All processes in a group of cooperating processes:
   (a) the processes share a group id, a 32 bit integer, which is generated in a way outside of the scope of this document.
   (b) mark the file as not requiring normal extent locks, and mark the file descriptor (as "usual") as blocking or non blocking.
   (c) take a concurrent GROUP lock on a [0,EOF] extent associated with a file. The concurrent GROUP lock is passed the group id.
   (d) explicitly release this lock when done with their I/O, preceeded by a flush of cached data.
   (e) when the file is closed, deliberately or through exit, the group locks are dropped
2. Readers/writers on other nodes take [a,b]R/W locks which cannot be granted when group locks are present. Such readers can receive:
   (a) can be made to wait forever, interruptably. This is good for blocking file descriptors.
   (b) can get -EWOULDBLOCK, this is good for file descriptors that have been marked as non-blocking.
   (c) group enqueues with a different group id must wait for the current group and PR/PW locks to be released.

In case (a) this behavior causes further group locks to have to wait until the read is satisfied. This is not desirable, so we will let group locks jump over the waiting lists if other group locks have already been granted.

3 State management

This lock can be held for unspecified amount of time by a client, so usual lock revocations timeouts are not applicable to these locks.

4 Protocol, APIs, disk format

New lock mode LCK_GROUP is added to support such a locking mode. This lock mode is only used for EXTENT locks. Access to this sort of locks is possible through ioctls:

- LL_IOC_GROUP_LOCK would get a group lock on a file. ioctl's arg argument reperesents 32bit "group id".
- LL_IOC_GROUP_UNLOCK woulo release a group lock previously granted on a file. ioctl's arg argument represents 32bit "group id" and should match that used at LL_IOC_GROUP_LOCK time.

5 Scalability and performance

It is believed that certain processes using this sort of locks will see speed burst, because several nodes can read and write the file at the same time without any locks bouncing around. Applications should be specially written to get use of this feature and to avoid any possible races.

6 Recovery

No implications.

7 Alternatives

None known.

8 Concerns

If a node holding such a lock would die, normal access to a file locked by this lock would be stalled until the node is evicted.