Ping Evictor

Phil Schwan

3/22/05

0.1 Revision history

Updated 09/21/2005 by Andreas Dilger to reflect actual implementation after bug 8322 changes.

1 Requirements

Rebooted Lustre clients must be efficiently removed from the lists of connected clients maintained by the OSTs and MDTs.

2 High-level design

2.1 Functional specification

All clients should ping all servers (already completed on HEAD). In LLNL's case, because all servers are recoverable, all servers are already being pinged.

If an MDT or OST detects that it has not received any traffic on an export for some period of time (some % of the timeout value), the client is immediately evicted.

2.2 Logic description

We define the eviction threshold to be some percentage of the timeout value.

Exports should be kept in a new, ordered list. Exports at the top of this list are the soonest to be evicted. We use only check whether exports are expired when RPC requests are being processed. This prevents sprurious evictions in the face of network outages (e.g. switch or NIC failure) or if all the server threads are hung (in which case a reboot would allow successful recovery).

Any time an RPC arrives for an export, it is moved to the end of the list, and we check if the topmost export is a candidate for expiry and eviction. If this is the case, we evict all clients whose exports have not received an RPC within the eviction threshold.

2.3 State management

The last-heard-from value in the export, the new export list, and the new timer are all shared between RPC-handling threads, the thread that handles the timer IRQ, and the new worker thread. As such access must be locked.

2.4 Disk format

no changes

2.5 Configuration

no changes

2.6 Wire protocol

No changes that affect protocol compatibility, however I propose that we take Alex's change and begin to ping every (timeout / 4) seconds, instead of every (timeout) seconds, if there is not other traffic on the export. This will promote more aggressive eviction of dead clients.

2.7 Key API changes

We will likely add new APIs for adding to and reordering exports on this new list and timer. Existing APIs will be minimally affected.

2.8 Scalability and performance

I don't see any serious concerns here. Maintaining an ordered list is very cheap in this case, because it starts ordered, and the most expensive thing we ever do is move something to the end of the list.

2.9 Recovery

The likelihood of a successful recovery is dramatically improved by proactively removing records of dead clients, which are currently responsible for the illusion of "double failures" in many cases.

2.10 Alternatives

Instead of a timer and ordered list, we could scan periodically for exports which have not received traffic.

We could ping clients "in reverse", and use those pings to detect dead clients.

Neither of these alternatives is particularly attractive, in my opinion.

2.11 Post-Mortem

Design bugs

1. liblustre clients should not get ping-evicted