# Lustre User Group 2009

Lustre Recovery

**Robert Read**
**Sun Microsystems**

# Recovery

- Recovery Overview
- Adaptive Timeouts
- Version Based Recovery
- Commit on Share

# Recovery Overview

Lustre Users Group 2009

# Imports & Exports

- An Import is the client side of the connection to a target.
- Export is the server side.
- A client has one import for every target.
- A service has one Export for every client.
- Recovery state is managed by Imports and Exports.

# Import status in /proc

- Import status /proc file (1.6.7 and above)
- Dumps current import state
  - cat /proc/fs/lustre/{mdc,mgc,osc}/*/import
  - lctl get_param '*.*.import'

# Sample Import

```
# cat /proc/fs/lustre/mdc/lustre-MDT0000-mdc-cde80000/import
import: lustre-MDT0000-mdc-cde80000
    target: lustre-MDT0000_UUID@10.0.1.180@tcp
    state: FULL
    inflight: 0
    unregistering: 0
    conn_cnt: 37
    generation: 6
    inval_cnt: 0
    last_replay_transno: 1329
    peer_committed_transno: 1338
    last_trasno_checked: 1338
    flags: replayable pingable
```

# What is Recovery?

- Client is disconnected from target
- Reconnect
- Synchronize state on client and server
- Resume normal (full) operation mode

# What causes disconnection?

- Failover
  - Node failed and target is restarted on same or new server
- Timeout
  - Router dropped message
  - Server/network too slow
- Eviction
  - Client didn’t respond to lock cancel request

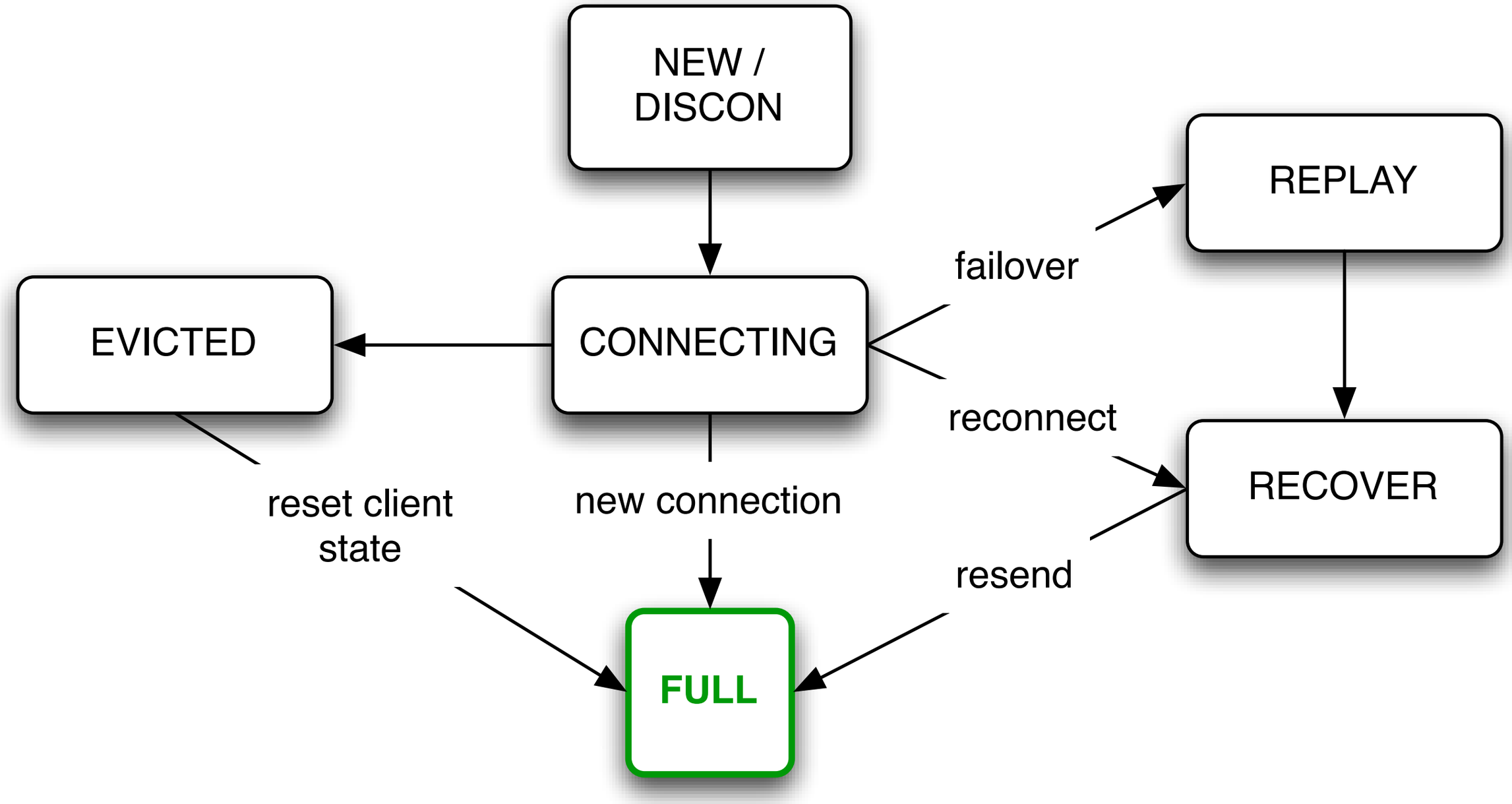
# When Disconnection Occurs

- Client re-queues in-flight RPCs
- New RPCs are delayed
  - Applications are blocked
- Client attempts to reconnect to the server (or failover node) until successful

# After Reconnection

- If server failed
  - Client replays uncommitted transactions
- If client simply reconnected
  - Resend in-flight requests
- If client was evicted
  - Client flushes state and returns IO errors

# Import States



# Failover period

- Clients usually take at least obd_timeout before they begin to reconnect
- Server waits 3x obd_timeout for clients to reconnect
- If a single client doesn’t reconnect, recovery takes entire duration

# Imperative Recovery

- Currently just an proposal
- Notify all clients server has failed and the new nid
- Server can set recovery period to be much smaller

# Adaptive Timeouts

Lustre Users Group 2009

# About Timeouts

- Timeouts determine when server is unresponsive
  - Server failure (or failover)
  - Network partition
  - Heavy load/net congestion
- Server death and load appear identical
- Longer timeouts increase recovery time

# Adaptive Timeouts

- Two Goals:
  - Automatically adjust RPC timeouts as network conditions and server load change
  - Decrease server recovery time
- Differentiate between dead server and busy server.

# Normal Operation

- Timeout set per RPC based on current info
  - server response time
  - network lag
- Server sends early replies to increase in-flight RPC timeouts
  - prevent reconnect/resend

# Server Recovery

- Initial recovery period based on multiple of obd_timeout
- As clients reconnect,  server adapts based on previous server response times
  - active clients have similar timeouts
  - need to wait until all clients timeout their RPCs

# Version Based Recovery

Lustre Users Group 2009

# Problems with Recovery

- Strict in-order replay requirement (no gaps)
- Requires all clients to reconnect during recovery period
- All clients evicted if recovery times out

# Solution

- Uses versions to detect conflicting transactions
- If version on an object is what we expect, then we can replay transaction
- Object version mismatches is a "mini-gap"
  - Only stops the clients attempting to modify that object
- Transactions on other objects can continue

# New Recovery

- Conditional out-of-order replay is allowed
- There is a still a limited recovery period
- Clients blocked by gaps are evicted at end of recovery
- Clients that have not reconnected are not evicted

# Commit on Share

# Detect Conflicts

- Checks for an uncommitted transactions from a different client before updating an object.
- Commit first, then update
- All uncommitted transactions have no dependencies

# Thank You

Robert Read
rread@sun.com