# High Level Design for Networking in MDT/OST

Niu YaWei

2006-2-13

## Contents

## 1 Requirements

Our future metadata server would layer a target handler, the MDT, similar to the OST on a local metadata driver, and all elements of networking would be moved into the MDT and OST layers, these elements are:

- Exports and Imports;
- Network recovery;
- DLM locks;

# 2 Functional specification

## 2.1 DLM locking

All DLM locking should be done in MDT/OST layer, underlying MDD/OSD should know nothing about DLM locks. The local enqueue/cancel DLM lock code in old MDS will be moved into MDT layer.

## 2.2 Export/Import encapsulation

Export/Import should strictly belong in MDT/MDC, OST/OSC layers. That means obd_export will no longer be used by local object device accessing, and it will only be used for exporting the MDT/OST to remote MDC/OSC. Each MDC/OSC will connect to individual MDT/OST, and each MDD/OSD is attached to network through its own MDT/OST.

## 2.3 Message pack/unpack

Request/reply message pack/unpack work must be done in MDT/MDC, OST/OSC layers. Our current implementation just follow this nicely, nothing need be changed for it.

## 2.4 Network recovery

Current network recovery is done in MDT/OST layer, but cleanup is necessary for this part of code. Network recovery cleanup work is clarified in other HLDs.

# 3 Use cases

## 3.1 mdt_mkdir

1. Unpack mkdir request message.
2. If it's a resent request, call proper mdt API to process this resent mkdir then return.
3. If it's a replay request, call proper mdt API to process this replay mkdir then return.
4. Lock the parent by FID.
5. Forward this mkdir to underlying layer.
6. Save the parent lock for reply ACK.
7. Pack reply message, then reply to remote MDC.

### 3.2 mdt_unlink

1. Unpack unlink request message.
2. If it's a resent request, call proper mdt API to process this resent unlink, then return.
3. If it's a replay request, call proper mdt API to process this replay unlink, then return.
4. Lock the parent by FID.
5. Lookup child FID by iam.
6. Lock the child by FID.
7. Forward this unlink to underlying layer.
8. Unlock the child.
9. Save the parent lock for reply ACK.
10. Pack reply message, then reply to remote MDC.

### 3.3 MDC/OSC connect

The import target uuid of MDC/OSC should be the uuid of MDT/OST now, MDC/OSC will connect to MDT/OST but not obdfilter/MDS anymore.

## 4 Logic specification

### 4.1 DLM locking

All the local DLM lock enqueue/cancel code in old MDS will be moved into MDT, and the locking follows "ancestor first" order.

- For locking parent/child pair, MDT locks parent firstly, then lookup child by iam and locks the child.
- For locking two parent/child pairs, MDT locks two parents in proper order, then lookup children by iam, then locks children in proper order. The "ancestor-offspring" relationship of two parents or two children can be checked by some API exported by underlying layer. (see the "CMD rename locking HLD" for detail)
- Saving DLM lock for reply ACK is done in MDT.

### 4.2 Export/Import encapsulation

OSD/MDD are designed to be pure local driver, they are invisible for remote OSC/MDC and they know nothing about network. OST/MDT connect to remote OSC/MDC and forward the requests to OSD/MDD. Thus, following significant changes will take place:

- Define a new class mdt_obd to contain connection and network recovery data. And the union mds_obd in obd_device will be changed to mdt_obd.
- All network related elements in obd_device will be moved into subclass mdt_obd/ost_obd, that include:
    1. Move exports data of obd_device into mdt_obd/ost_obd.
    2. Move network(remote) llog stuff of obd_device into mdt_obd/ost_obd.
    3. Move network recovery data of obd_device into mdt_obd/ost_obd.
    4. Move ptlrpc service data of mds_obd into mdt_obd.
- mdt_obd manages MDSs service threads, ost_obd manages OST service threads.
- All local device accessing will use obd_device pointer directly (and get refcount), but not obd_export anymore. These local accessing are: MDT/OST accessing MDD/OSD, llite accessing LMV/LOV, LOV accessing OSC, LMV accessing MDC, etc.

## 5 State management

This is code reorganization and cleanup, no protocol/disk format changes, no scalability/performance issues.

## 6 Focus for inspections

Anything important is not covered by this HLD?