lustre_msg v2

Lai Siyao

Jan 11, 2006

1 Requirements

- Support lustre_msg embedding
- Hide RPC-specific data
- Security support
- Keep protocol compatible

2 Specification

2.1 Support lustre_msg embedding

This means a packed lustre_msg could be treated as a single data buffer, and be packed inside a new lustre_msg. And in fact the current structure has supported this, except the next to address: hide RPC-specific data.

2.2 Hide RPC-specific data

The old lustre_msg contains many RPC-specific data: opc, last_xid..., to make this struct more general, those fields should be moved out and into a seperate message body.

2.3 Security support

Different security mechanisms will bring different payload on original data, and a new field will be added to mark the security mechanism applied.

2.4 Protocol compatibility

The new version of lustre_msg should guarantee interoperations in the following situations:

- new client vs. new server

- old client vs. new server
- new client vs. old server

To achieve this, we would need to have some sort of MSG_CONNECT compat flag that tells us what format to use, so that we would continue to use the old format if we are communicating with an old client or server. This means the first connect would need to use the old lustre_msg format + magic if we want to be able to commuicate with old clients or servers.

3 Logic

1. structure of the lustre_msg_v2 & ptlrpc_body

```
struct lustre_msg_v2 {
        __u32 bufcount;   /* buf count */
        __u32 secflvr;    /* sec flavor */
        __u32 magic;      /* msg magic */
        __u32 buflens[0]; /* buf lengths array */
};
struct ptlrpc_body { /* RPC-specific data */
        struct lustre_handle handle;
        __u32 type;
        __u32 version;
        __u32 opc;
        __u64 last_xid;
        __u64 last_committed;
        __u64 transno;
        __u32 status;
        __u32 flags;
        __u32 conn_cnt;
};
```

2. add wrapper to access ptlrpc_body and lustre_msg fields.

3. negotiation of the lustre_msg version in connection.

4 Alternative

- Field secflvr in lustre_msg_v2 could be put in ptlrpc_body, and this might look more reasonable. But in this way the structure ptlrpc_body has to be put in a fixed place in pending buffers, which leads to inflexibility for buffer order.
- We could specify the lustre_msg version by lconf argument, which could make the connection code more consistent. But it will bring another argument for lconf and complicate the usage of it.