RPC/RDMA Bi-direction

Chuck Lever, Oracle

NFSv4.1 Requires A Backchannel

- Possible choices for RPC/RDMA:
  1. Client initiates a separate connection that uses an existing transport with backchannel capability (*e.g.* TCP)
  2. Plumb RPC bi-direction into the RPC/RDMA transport

- Not an exhaustive list

Linux TCP Side-Car

- CREATE_SESSION on RDMA transport
  - Client or server advertises forward channel only
- Server asserts SEQ4_STATUS_CB_PATH_DOWN
- Client sets up separate TCP transport
  - Sends BIND_CONN_TO_SESSION
  - Then used only for backchannel operations

Linux TCP Side-Car

- Pros
  - TCP bi-direction already works
  - NFSv4.1 servers are already REQUIRED to support a session using multiple transports
- Cons
  - Introduces session trunking
  - Requires non-trivial NFSv4.1 upper layer client changes

RPC/RDMA Bi-direction

- Pros
  - A single connection can be used
  - Works exactly like TCP transports
- Cons
  - Complex changes to transport required
  - Protocol changes likely needed

Bi-RPC/RDMA Prototype

- RPC/RDMA header has copy of XID to locate request data structures to process reply
- Backchannel requests must skip XID lookup
- Call direction is in RPC header, but RPC/RDMA header length varies
- Indicate call direction in RPC/RDMA header

Prototype XDR Changes

```
enum rdma_proc {
        RDMA_MSG=0,    /* An RPC call or reply msg */
        RDMA_NOMSG=1, /* An RPC call or reply msg - separate body */
        RDMA_MSGP=2,  /* An RPC call or reply msg with padding */
        RDMA_DONE=3,  /* Client signals reply completion */
        RDMA_ERROR=4, /* An RPC RDMA encoding error */
        RDMA_BCALL=5, /* A backchannel RPC call */
        RDMA_BRPLY=6  /* A backchannel RPC reply */
};

union rdma_body switch (rdma_proc proc) {
        case RDMA_MSG:
          rpc_rdma_header rdma_msg;
        case RDMA_NOMSG:
          rpc_rdma_header_nomsg rdma_nomsg;
        case RDMA_MSGP:
          rpc_rdma_header_padded rdma_msgp;
        case RDMA_DONE:
          void;
        case RDMA_ERROR:
          rpc_rdma_error rdma_error;
        case RDMA_BCALL:
          rpc_rmda_header rdma_msg;
        case RDMA_BRPLY:
          rpc_rdma_header rdma_msg;
};
```

Bi-RPC/RDMA Prototype

- Client has to know how many receive buffers to post

- Based on how many NFSv4.1 backchannel session slots are required

- Separate credit management for fore and backchannel on same RDMA transport

Bi-RPC/RDMA Prototype

- Large RPC messages need to use RDMA chunks
  - Backchannel server (*i.e.*, the client) would have to perform RDMA READ/WRITE

- Restrict backchannel to only inline operation
  - Upper layer can limit the size of callbacks
  - Kerberized backchannel calls can be large anyway, but are not used in prototype

Receive Buffer Management

- Initial transport set-up allocates transport resources for forechannel
  - Receive buffers
  - WQEs
- Backchannel set up is invoked later, allocates resources for backchannel that are shared with forechannel
- CREATE_SESSION prevents server from sending backchannel calls before client has posted receive buffers

Receive Buffer Management

- Transport posts one receive buffer for every
  - Outstanding RPC
  - Backchannel credit
- Hardware chooses the receive buffer that catches an incoming RDMA SEND

Receive Buffer Management

- After disconnect/reconnect:
  - Per-RPC receive buffers are reposted by RPC retransmit
  - Transport must repost standing backchannel receive buffers
- BIND_CONN_TO_SESSION prevents server from sending backchannel calls before client has posted receive buffers

Receive Buffer Management

- Buffer sizes (inline threshold) must match
  - Send buffers on client used for both forechannel calls and backchannel replies
  - Receive buffers on client used for both forechannel replies and backchannel calls
  - And vice versa on the server

Challenges

- Managing backchannel credits
- Selecting transport receive buffer size
- Identifying backchannel messages
- Handling large backchannel messages

Bottom Line

- I believe RPC/RDMA bi-direction requires changes to the protocol