

Lustre Networking over OpenFabrics

Eric Barton

eeb@clusterfs.com

www.clusterfs.com

www.lustre.org



The Lustre File System

- **Scalable**
 - Tens of thousands of clients / Petabytes of data
- **Reliable**
 - Deployed on production clusters
- **Fast**
 - Deliver 95% of hardware performance
- **Cost Effective**
 - Industry – standard platforms
 - Heterogeneous networks

LNET – the lustre networking API

- **Network independent**
- **Message passing and RDMA**
- **Abstract scatter/gather network buffer descriptors**
- **Generic failure handling**
- **Routing over heterogeneous networks**

LND - the lustre network driver

- **Network specific**
- **Connection establishment**
- **Message Passing**
- **RDMA**
- **Failure handling**

Supported Networks

- **TCP/IP**
- **Quadrics**
- **Myrinet**
- **Infiniband**
 - OpenIB gen1 / Mellanox Gold (thinly disguised TopSpin ☺)
 - Silverstorm
 - Voltaire
 - OpenFabrics
- **RapidArray (Cray XD1)**
- **Cray Portals (XT3)**

OpenFabrics API Calls used

■ CM

rdma_create_id
rdma_destroy_id

rdma_connect
rdma_disconnect

rdma_create_qp
rdma_destroy_qp

rdma_bind_addr
rdma_resolve_addr
rdma_resolve_route

rdma_listen
rdma_accept
rdma_reject

■ VERBS

ib_alloc_pd
ib_dealloc_pd

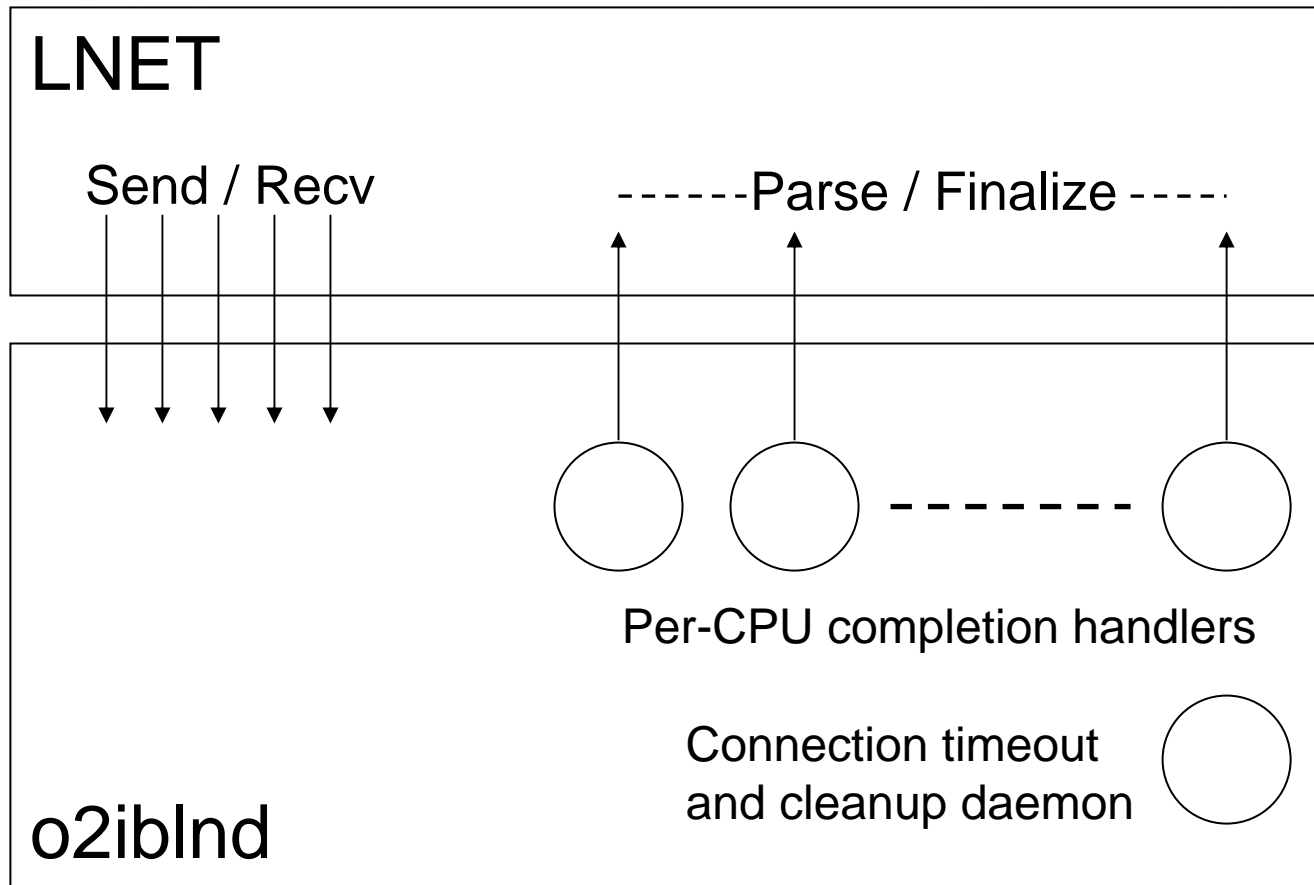
ib_create_cq
ib_req_notify_cq
ib_destroy_cq

ib_post_recv
ib_post_send

ib_get_dma_mr
ib_dereg_mr

ib_create_fmr_pool
ib_fmr_pool_map_phys
ib_fmr_pool_unmap
ib_destroy_fmr_pool

o2ibInd structure



LND state

- **Global**
 - Idle descriptor pool
 - CQ – send completions
 - Connection table
- **Per-connection State**
 - RC QP
 - CQ – receive completions
 - 2 ‘n’ rx buffers – requests + completions
 - Credits
 - Send queues
 - In-progress RDMA

Connection Setup

- **Additional information passed via CM private data in rdma_{connect,accept,reject}**
- **LND protocol version number**
 - Version interoperation
- **LNET address**
 - Who I am
- **Connection parameters**
 - Credits
 - Max allowed RDMA fragments
- **LND incarnation**
 - Is he the same guy I spoke with 10 minutes ago?

LND protocol

- **Paranoia**
 - Source / Destination address + incarnation
- **Credits**
 - Piggy-back on all messages
 - Back-pressure
- **Message Passing**
 - Noop
 - Immediate
- **RDMA setup / completion**
 - PUT
 - Request
 - ACK / NAK
 - Done
 - GET
 - Request
 - Done

Error Handling

- **RDMA descriptor == Murder Weapon**
- **Error Sources**
 - Underlying stack
 - Timeouts
- **Error Handling**
 - 1. rdma_disconnect()
 - 2. wait for all completions
 - 3. Inet_finalize()

RDMA

- **Pre-map all memory**
 - Many RDMA fragments
 - Higher Bandwidth!
 - Higher CPU?
- **Map on demand**
 - FMR pool
 - Mapping cache
 - Lower Bandwidth!
 - Lower CPU?

Performance

- **Latency**

- Can't busy-poll in the kernel ☹️
- Interrupt handling relatively HUGE ☹️ ☹️

- **Bandwidth**

- LNET MTU typically 1MByte
 - Finesses latency issues 😊😊😊
- ~600MBytes/sec - single pair of nodes
- > 900MBytes/sec into server with 2 or more clients

OpenFabrics – the bad news

- **Better Diagnostics?**
 - Vendor stacks provide more specific error codes
 - Add debug statements
- **“Backport” is a headache**
 - Customers but don't want to version-up linux

OpenFabrics – the good news

- **Single API**
 - IB
 - Other networks
- **Clean – eases implementation**
- **“Thinner” software layer**