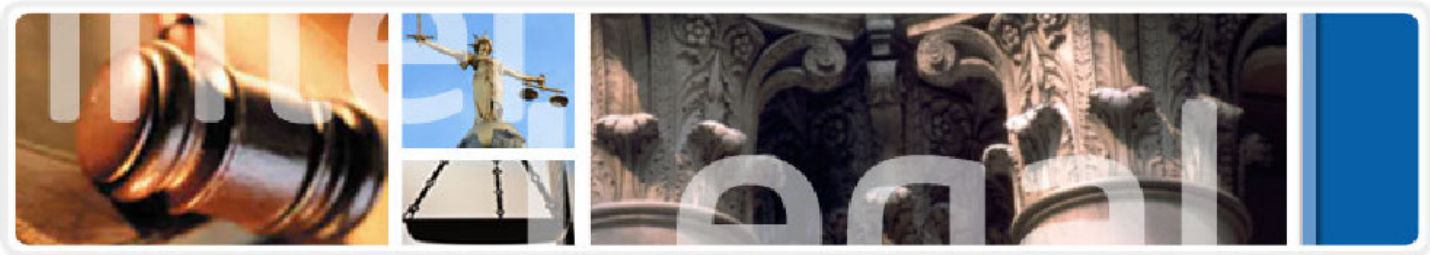
Lustre* Networking



Legal Disclaimer

- THIS DOCUMENT AND RELATED MATERIALS AND INFORMATION ARE PROVIDED "AS IS" WITH NO WARRANTIES, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO ANY IMPLIED WARRANTY OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT OF INTELLECTUAL PROPERTY RIGHTS, OR ANY WARRANTY OTHERWISE ARISING OUT OF ANY PROPOSAL, SPECIFICATION, OR SAMPLE. INTEL ASSUMES NO RESPONSIBILITY FOR ANY ERRORS CONTAINED IN THIS DOCUMENT AND HAS NO LIABILITIES OR OBLIGATIONS FOR ANY DAMAGES ARISING FROM OR IN CONNECTION WITH THE USE OF THIS DOCUMENT.
- All products, product descriptions, plans, dates, and figures are preliminary based on current expectations and subject to change without notice. Availability in different channels may vary.
- Intel and the Intel logo are trademarks or registered trademarks of Intel Corporation in the United States and other countries.
- *Other names and brands may be claimed as the property of others.
-

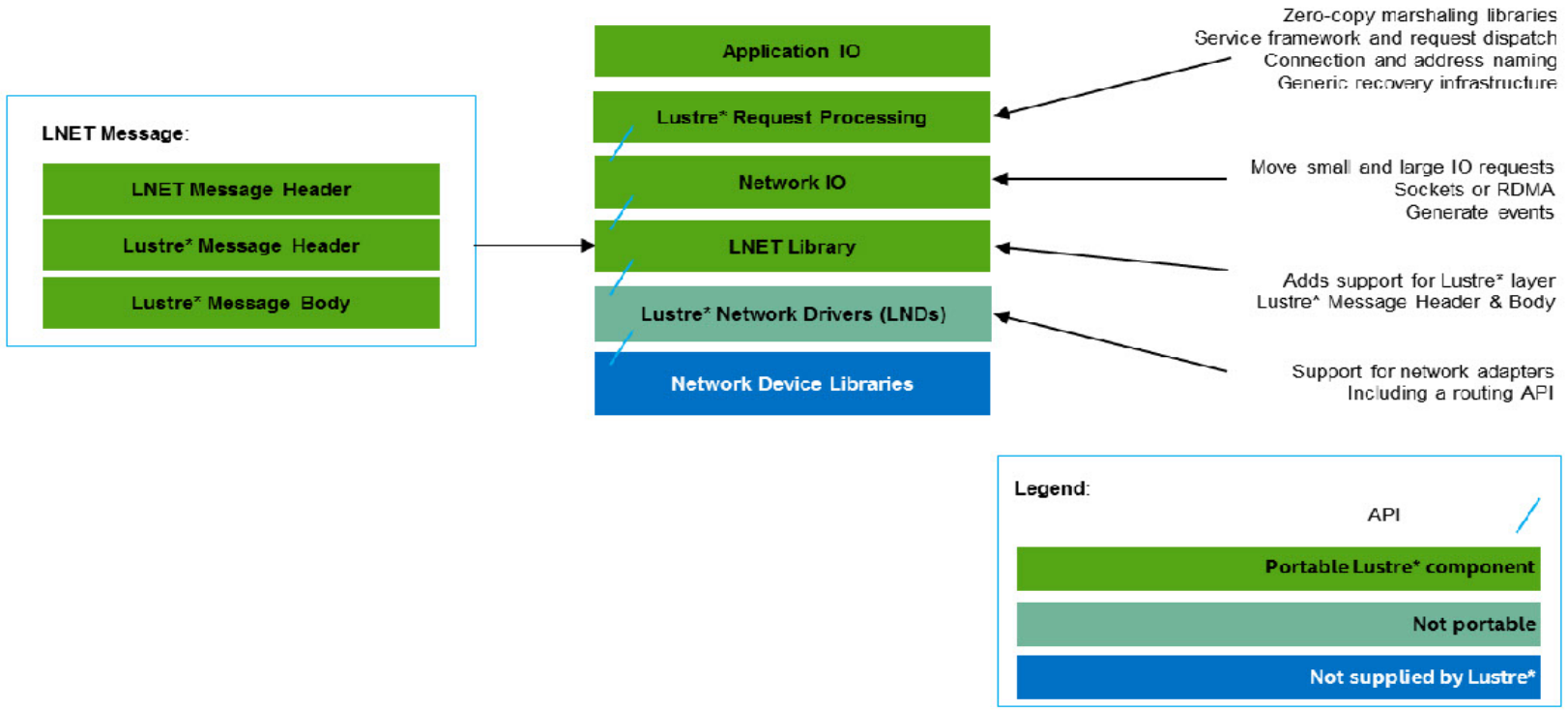
Module Overview

Topics covered in this module include:

- LNET Overview
- Network IDentifier Configuration
- Multi-[rail, path, plane]
- Multi-rail (Bonding)
- Introduction to LNET Routing
- LNET Configuration Parameters
- Summary

LNET Overview

Lustre* Network Software Stack



Interfaces Supported by Lustre* LNDs

Lustre* has provided support for many different types of network interfaces

- Each requires a supporting Lustre* Network Driver (LND)
- Key today are InfiniBand*, Intel® Omni-Path Architecture (Intel® OPA), Cray's GNI, and Sockets

InfiniBand* - uses OFED LND

- Native RDMA using Verbs, High Performance

Intel® Omni-Path - uses OFED LND

- Intel's new 100 Gbps network - leveraging QLogic's TrueScale, InfiniBand*, and Cray's Interconnects

Cray Interconnects - uses GNI LND

Sockets - uses SOCK LND on TCP port 988

- Internet Protocol over Ethernet or InfiniBand* (IPoIB)

RDMA over Converged Ethernet (RoCE) - uses OFED LND (requires a lossless Ethernet network)

- RoCE v1 is a link-layer only protocol - v2 is both routable and priority-based

Internet Wide Area RDMA Protocol (iWARP) - uses OFED LND

- RDMA over Sockets using iWARP extensions - currently implemented over TCP (offloaded)

Lustre* Network Operations

Start or Stop LNET

```
# lctl network [up|down]
```

List the Lustre* Network IDentifiers (NID)

```
# lctl list_nids
```

Check communication

```
# lctl ping <NID>
```

- When used with o2iblnd, the ping verifies RDMA functionality

Unconfigure LNET

```
# lctl network unconfigure
```

LNET functions independent of a file system

- In the case of Lustre* routers

Lustre* Network Operations - "lnetctl"

As of Lustre* 2.7, "lnetctl" works with Dynamic LNET Configuration (DLC) to configure Nodes and Routers

- Nodes:
  - Start/Stop LNET, Add/Remove/Show Networks, Add/Remove/Show Routes
- Routers:
  - Enable/Disable Routing, Enable/Disable/Show Routes, Configure Routing Buffers, Show Routing Parameters and Statistics

Start or Stop LNET

```
# lnetctl lnet [configure|unconfigure]
```

List the Lustre* Networks and Network IDentifiers (NID)

```
# lnetctl net show [--verbose]
```

Add a New Network

```
# lnetctl net add --net <LNET> --if <INTERFACE>
```

Remove a Network

```
# lnetctl net del --net <LNET>
```

Network IDentifier Configuration

Network IDentifier Configuration

Every Lustre* network needs an LNET Network IDentifier (NID)

Configured dynamically ("lnetctl"), or in `/etc/modprobe.d/`

- Actual file name is arbitrary - suggested is "`lnet.conf`"

Without an explicit configuration

- Starting Lustre* configures and enables LNET on the first non-loopback interface

LNET works with the per-interface network configuration

- Frame size, buffer sizes, TCP settings, etc.

Multiple LNETs can exist on one physical network

- e.g. Both o2ib1000 and o2ib1001 on ib0
- May be useful with fine-grained routing, or some types of load balancing

NID Configuration (cont)

Two configuration options: "`networks`" or "`ip2nets`"

- Mutually exclusive – use either "`networks`" or "`ip2nets`"

Option networks

- Simpler format to read and write
- Configures LNET on a specific Linux interface
- May include multiple LNET configurations

Option ip2nets

- More complex to read and write
- Supports deploying the same "`lnet.conf`" to nodes with different network interfaces
- LNETs are configured by matching their interface(s) and/or IP address(es)

LNET Multi-plane Example Using Networks

Server has 3 Ethernet ports and 1 IB port

- The first Ethernet interface (eth0) is for management - it will not be used by Lustre*
- The remaining ports, eth1, eth2 and ib0, will all be configured to use LNET

```
options lnet networks="tcp1(eth1),tcp2(eth2),o2ib0(ib0)"
```

- `networks` options are parsed in the order shown above

In the above configuration:

- `tcp1` will run over `eth1`
- `tcp2` will run over `eth2`
- `o2ib0` will run over `ib0` (and will sometimes show as "`o2ib`" – minus the numerical suffix)
- Lustre* will not configure `eth0` with an LNET NID

LNET Multi-plane Example Using ip2nets

Servers

- Server1: eth0 IP address: 192.168.0.2; ib0 IP over InfiniBand* (IPoIB) address: 132.6.1.2
- Server2: eth0 IP address: 192.168.0.4; ib0 IP over InfiniBand* (IPoIB) address: 132.6.1.4

Clients

- Ethernet/TCP clients: All have IP addresses 192.168.0.5-255
- InfiniBand* clients: All have IP over InfiniBand* (IPoIB) addresses 132.6.[2-3].2, .4, .6, .8

Configuration (semicolon between entries)

```
options lnet ip2nets="tcp1(eth0) 192.168.0.[2,4]; \
  tcp1 192.168.0.*; o2ib1 132.6.[1-3].[2-8/2]"
```

- Nodes use the first matching rule for particular network types
- Nodes with multiple matches on different interfaces support multiple LNETs
- [1-3] matches the addresses 1, 2, and 3
- [2-8/2] matches the addresses 2, 4, 6, 8

LNET Multi-plane Example - with Comments

Setup

- MDS – `10.10.120.[3,4] (eth1)`
- OSS – `10.10.[1-8].[1-253] (ib0)`
- OSS – `10.20.[1-8].[1-254/2] (ib0)`
- Clients – `10.30.x.x` (any interface)
- Remote clients use – `eth1`
- Routers – Not shown

```
options lnet ip2nets="\
  tcp1(eth1) 10.10.120.[3,4]          # MDS group; \
  o2ib1(ib0) 10.10.[1-8].[1-254]      # 1st OSS group;   \
  o2ib2(ib0) 10.20.[1-8].[1-254/2]    # 2nd OSS group;  \
  tcp2       10.30.x.x                # 10.30.0.0 clients; \
  tcp3(eth1)                          # Remote clients"
```

Multi-[rail, path, plane]

Network Connections - Multi-rail

Multi-rail (a.k.a “Bonding”)

- Multiple connections between end points, using the same network ID
  - Imagine: Train tracks, working together to transfer data
  - Example: Ethernet Bonding in Linux
- Ethernet bonding in Linux is fully supported by LNET
  - LNET uses the underlying bonding technology
- InfiniBand* bonding is not as clear cut
  - LNET over IPoIB (sockets) fully supports multi-rail (using sock LND)
  - OFED InfiniBand* does not support multi-rail for RDMA
  - LNET over RDMA-IB supports interface failover (using o2ib LND)
- Next unit will cover bonding in more detail

Network Connections - Multi-path

Multi-path

- Multiple connections providing different routes to the same end point
  - Example: A node with:
    - Two adapters with different network IDs, and
    - The adapters are connected to different switches in the fabric
- Not supported by Lustre* between nodes
- Lustre* determines one (1) NID to reach a destination, and it sticks with it
- Lustre* servers do use multi-path to storage controllers

Network Connections - Multi-plane

Multi-plane (a.k.a. “Dual-Homed”)

- Having separate connections to 2+ networks
  - Each network connection requires a unique network identifier
  - Often referred to as “dual-homed”, or “multi-homed”
- Supported by Lustre* via LNET
  - Nodes have routing table entries for each LNET Network
- Works with commonly used network protocols
  - InfiniBand*, Ethernet, etc.
  - Heterogeneous networks

Multi-plane (cont)

A use case for a multi-plane configuration

- Additional bandwidth gained by using additional networks
- Supporting multiple compute clusters
- Connected nodes are relegated into separate groups
  - Remember, LNET always uses the same local NID to communicate with a peer

Configuration example - Multi-plane InfiniBand*

- Servers each have two IB interfaces - ib0 and ib1
- Clients each have one IB interface - ib0
- Clients attach to only one of the server interfaces
- Using a "/17" network

```
option lnet ip2nets="\
     o2ib0(ib0) 10.10.[0-127].*   # subnet 1: servers + clients;\
     o2ib1(ib1) 10.10.[128-255].* # subnet 2: servers;\
     o2ib1(ib0) 10.10.[128-255].* # subnet 2: clients"
```

Multi-rail (Bonding)

Multi-rail ("Bonding")

Switch from Multi-rail terminology to the more familiar term of bonding

Not all protocols support bonding equally

- Bonding works well for Ethernet and IPoIB, as it did for ELAN and others
  - Much better bonding support in Ethernet vs IPoIB
- Bonding for RDMA data transfers is not supported by the OFED stack
  - LNET provides limited bonding support for RDMA over IB

Ethernet bonding in Linux

- Linux provides a very good bonding driver
- Provides fault tolerance and load balancing
- Supports seven (7) different bonding modes
- Lustre* supports the use of Linux bonded interfaces

Multi-rail ("Bonding") (cont)

Ethernet bonding (cont)

Creating a bonded Ethernet interface using Linux bonding techniques

```
alias bond0 bonding
options bond0 mode=balance-alb miimon=1000
  (other options exist as well)
```

For reference, this means:

- Use interface `bond0`
- Use Active Load Balancing (ALB) - mode 6
  - Has certain requirements of the Ethernet driver
- Use "`miimon=1000`" (ms) for link detection
  - Must support Media Independent Interface (MII) checking/setting

Configuring the bonded interface into LNET:

```
options lnet networks=tcp1(bond0)
```

Bonding - LNET over InfiniBand*

The `o2ib` LND provides limited support for LNET bonding over InfiniBand*

Active-passive configuration only

- If active interface fails, other interface becomes active
- Used in situations where high availability is required
- No aggregation of IO

Configure bond using InfiniBand* LND

```
options lnet networks=o2ib0(bond0)
```

Enable failover bonding using this LNET option

```
options ko2iblnd dev_failover=1
```

Multi-Rail LNET (over InfiniBand*) design

- See wiki.lustre.org/Projects

Introduction to LNET Routing

LNET Routing

The Lustre* Network protocol is a routable protocol

- Routing across a WAN
- Routing within a Data Center
  - Connect different LNET types - InfiniBand* with Ethernet or Aries/Gemini (Cray)
  - Fine Grained Routing (FGR) - router preferences, to avoid ISLs

Components

- Lustre* end-nodes
  - Lustre* clients and servers
  - Configured with routes to remote LNETs
- Lustre* routers
  - Multi-plane nodes
  - Run LNET protocol only, via Lustre* client software
  - Forward traffic between LNETs
  - Deploy multiple routers for both performance and high-availability

LNET Routing - End Nodes

End nodes have static routes defined

- Route to destination LNET via router on local LNET

Routing decision is based on destination NID

- If destination NID is on a local LNET, send direct to NID
- If destination NID is on a remote LNET, find the best route

Best route is based on hop count *(excluding FGR)*

- If one router has the shortest hops to the destination, that route is used
- If multiple routers have the same hop count to the destination, those routers are used in a round robin manner

LNET Routing - End Nodes (cont)

End nodes monitor routers

- All Lustre* nodes always monitor peers via LND
  - Cannot be disabled
  - LND detects when a peer is down - will not send to a down node
- End nodes can also monitor routers at the LNET layer

LNET monitoring of a router is performed by the Router Checker

- Though disabled by default, enabling certain LNET parameters enables the Router Checker
  - live_router_check_interval, dead_router_check_interval
- Router Checker issues an LNET_GET to the router
  - Receives a list of router NIDs and their state
  - If the route to the destination is not available, that route will be marked down
  - If any NID is down and avoid_asym_router_failure=1, router will be marked down
  - If the “dead check” is enabled, end node will detect when the router is up again

LNET Routing - Routers

Lustre* routers are basically multi-planed Lustre* clients

- Their routing tables contain the interfaces they know about, and possibly static routes to remote networks for WAN configurations
- The routers buffer incoming messages and relay them out another interface
- They have an interface specific LND with necessary customizations
- Though there will be multiple LND layers, one for each interface, there will be only one LNET thread

The LNET layer is a single kernel thread, so LNET, and thus LNET routing, is logically a single queue

- Because of that there must be some significant buffering and queuing, both in the LNET layer as well as the LNDs

Next - Routing buffers at the LNET layer

LNET Routing - Routers (cont)

LNET Routers have three (3) types of buffers

- **Tiny buffers**
  - Used for transfers that have no payload - ACK, etc.
  - Zero-byte payloads
  - Default/minimum amount is 512
- **Small buffers**
  - Used for transfers that have a small payload
  - Single page (4096 bytes) payload
  - Default/minimum amount is 4096
- **Large buffers**
  - Used for large transfers
  - 1 MB - Lustre*'s magic number
  - Default/minimum amount is 256

LNET Routing Configuration

Example configuration using the LNET networks option:

- Servers are on o2ib1010 – 10.10.0.0/16
- Clients are on tcp168 – 192.168.0.0/16
- Routers are on both o2ib1010 – 10.10.0.1-5 and tcp168 – 192.168.0.1-5

Servers:

```
options lnet networks="o2ib1010(ib0)" routes="tcp168 10.10.0.[1-5]@o2ib1010" \
live_router_check_interval="45" dead_router_check_interval="180"
```

Clients:

```
options lnet networks="tcp168(eth0)" routes="o2ib1010 192.168.0.[1-5]@tcp168" \
live_router_check_interval="45" dead_router_check_interval="180"
```

Routers:

```
options lnet networks="o2ib1010(ib0), tcp168(eth0)" "forwarding=enabled"
options ko2iblnd peer_buffer_credits="12"
options ksocklnd peer_buffer_credits="8"
```

LNET Routing Configuration - Results

```
[client]#  lnetctl net show --verbose
net:
    - net: lo (not shown here...)
    - net: tcp168
      nid: 192.168.10.111@tcp168
      status: up
      interfaces:
          0: eth0
      tunables:
          peer_timeout: 180
          peer_credits: 8
          peer_buffer_credits: 0
          credits: 256
```

LNET Configuration Parameters

Network Configuration Parameters

Lustre* networking contains many, many, many configuration parameters

- Two types of Lustre* modules - and then there are the Linux modules…
- Lustre* modules: LNET (lnet.ko) and LND (ksocklnd.ko, ko2iblnd.ko, kgnilnd.ko, etc.)
- Some parameters apply to LNET, others apply to the LND
- To determine available parameters for an LND, run:

```
$ modinfo /lib/modules/…/kernel/net/lustre/[LND module]
```

Module parameters have default values

- Sometimes hard-coded (see: o2iblnd_modparams.c)
- Other times derived
  - LNET peer_buffer_credits derived from LND peer_buffer_credits
  - LND concurrent_sends based on LND peer_credits and LND map_on_demand

What do all the parameters do? Which should be focused on?

- The most significant ones, of course…

Configuration Parameters - LND

LNET's credit system

- Rate limiting is done on SENDING information
- LNET uses a credit system to limit overloading of nodes
- Cannot send anything to a peer without approval - or "credits"

Credits, Peer Credits, Peer Credits Hi-Water, and Peer Buffer Credits

- "credits" - number of outstanding sends on a specific network interface
- "peer_credits" - number of outstanding sends to one (1) peer
- "peer_credits_hiw" - number at which credit returns are eagerly requested
- "peer_buffer_credits" - number of outstanding sends from an end node

Monitor Credits

```
/proc/sys/lnet/nis
/proc/sys/lnet/peers
```

Configuration Parameters - LNET

Mostly Routing Parameters

```
config_on_load                  - configure LNET at module load
routes                          - define routes through routers

forwarding                      - route LNET traffic: enabled|disabled
auto_down                       - mark router state up/down: 0|1
check_routers_before_use        - force a check: 0|1 (req's d_r_c_i)
dead_router_check_interval      - check if down: secs
live_router_check_interval      - check if up: secs
router_ping_timeout             - router checker timeout: secs
avoid_asm_router_failure        - down router if any route down: 0|1

peer_buffer_credits             - router buffer credits, per peer
large|small|tiny_router_buffers - buffers in each group: num
```

Summary

Module Summary

LNET Overview

NID Configuration

Multi-[rail, path, plane] Configuration

Multi-rail (Bonding)

Introduction to LNET Routing

LNET Configuration Parameters

Congratulations! You have completed:

Lustre* Networking