Lustre Ping Evictor Scaling in LNET
Fine Grained Routing Configurations

Nic Henke nic@cray.com
Cory Spitz spitzcor@cray.com
Chad Zanonie chadz@cray.com
Overview

- FGR configurations
- IOR and “dead time”
- Data collection & analysis
- Tuning
- Conclusions & Discussion
FGR Configurations

- For more details see “I/O Congestion Avoidance via Routing and Object Placement” from our friends at ORNL

- We are using FGR groups
  - Balance bandwidth, resiliency
IOR and the “Dead Time”
Data Collection & Visualization

- **Instrumented IOR**
  - Only gives us single number, rates varied
  - sub-second sampling, post processing

- **Collectl**
  - Enhanced to collect LNet data, OSS data

- **Ganglia/Graphite to visualize**

- **LNet data not all that helpful**
  - Especially LND
  - Lack of directional information
The Pinger Hurts Us

- Usually 3-8 seconds, I/O stops
  - Some over 10 seconds!
- 4% to 11% reduction in throughput
- Instantaneous loading
- Math for low petascale
  - 25000 clients
  - 4 OSTs per OSS
  - 360 OSS
  - 36M pings every 75s
  - With 4:3 FGR, 75k per RTR, 100k per OSS
- FGR makes this worse
  - Fewer IB destinations to send messages from each RTR
- No real value in traffic
  - Most times clients are idle with no locks to evict
  - Async journal complicates this a bit
OSS Data

OSS Write vs Ping

Pings

Writes

21:46:55

21:47:00

0

15.0 K

30.0 K

45.0 K

60.0 K

75.0 K

90.0 K

105.0 K

0

200.0

400.0

600.0

800.0

1000.0

1200.0

1400.0

1600.0
OSS Data

LNet Recv Rate

LNet Recv

0
15.0 K
30.0 K
45.0 K
60.0 K
75.0 K
90.0 K
105.0 K

21:46:55
21:47:00
Data: LNet queuing

gnilnd Peer Router Buffer Usage

- hera.esfs.nid00693.GNI_RTR
- hera.esfs.nid00682.GNI_RTR
- hera.esfs.nid00692.GNI_RTR
- hera.esfs.nid00683.GNI_RTR
Data: LNet queuing

Small Router Buffer Usage

- hera.esfs.nid00693.Bufs_1
- hera.esfs.nid00692.Bufs_1
- hera.esfs.nid00682.Bufs_1
- hera.esfs.nid00683.Bufs_1
Data: LNet queuing
Tuning

● IB LND is a bit of a PITA
  ● Especially for small messages
  ● peer_credits & concurrent_sends
    ● Use map_on_demand and others for concurrent_sends > 63
    ● peer_credits <= 2x concurrent_sends
    ● peer_credits limited to 255 in wire structure
  ● peer_credits returned explicitly in o2ibLnd

● Lots of other tuning required
  ● Small router buffers
    ● Ends up being 4k page for each ping message
  ● peer router buffer credits
  ● timeouts, keepalive, asym router failure, peer health, ntx, credits

● None of this is great for FGR
  ● Small number of destinations

● However, it has shown significant improvement
  ● Just reached end of tuning range
Conclusions & Discussion

- LNet routing not very friendly to small message size with high throughput rates
  - o2iblnd needs love too

- Quite hard to get “right”
  - Magic tuning, course statistics

- Worth exploring how this will impact other workloads
  - Metadata
  - Small files
  - Future Health Networks

- Questions or Comments ?