Gemini LND driver challenges

James Simmons, ORNL

Background

- ORNL largest cray system upgraded from XT5 to XK7
- Went from using SeaStar to Gemini
- Currently using modified Lustre 1.8.6 clients

Performance evaluation

- Theoretical promised raw performance
  - 3 GB/s small messages
  - 6 to 7 GB/s bulk messages
- Gemini 1.8 LND driver real numbers
  - 1.0 GB/s small messages
  - 1.6 GB/s bulk messages

Causes

- Check summing is expensive
  - On – 1.6 GB/s
  - Off – 3.8 GB/s
- Original driver was not multiple threaded.
  - Newer driver version have added threads to handle parallel check summing for service nodes
  - Threads in newer driver added for service nodes only

Possible Solutions

- Different check sum algorithm
- Avoid check summing in certain cases
- Are more threads the solution
- Have these problems been solved before ?

Lustre 2.4

- Lustre had the same challenges
  - New crypto api used for check summing.
  - SMP scaling

Lustre Crypto api

- More choices of check sum algorithms.
- Hardware optimized choices.
- PtlRpc does bulk checksumming
  - No need to check sum on routers
  - Double check summing is bad

Crypto challenges

- Cray default kernel lacks most crypto targets
- Lustre assumes crypto supported targets are there
- Both DVS and Lustre use LNET
- Impacts bulk check sum optimization

Gnilnd SMP scaling

- Rework LND driver according to mapping between layers.
  - X LNET interfaces : Y devices : Z CPT
- Per CPT allocations to limit cache migration
- CPU affinity to threads

SMP API gives greater control

- You can control which cores belong to which CPT
  - Don't need to use all cores
- You can map LNET interfaces to specific CPT
  - Use this to limit compute node noise

Gemini LND platform targets

- Cray platforms vary greatly
  - XE6 – AMD Magny-Cours
  - XK7 – AMD Bulldozers
  - XC30 – Intel Xeon E5-2600 Series
- Compute nodes and Service nodes for the same family of hardware need different configurations
- XC30 uses Aries interconnect. Others use Gemini
  - Both interconnects are supported with same software stack

Hardware influences configuration

- Processor properties
  - NUMA and cache shared between cores
  - AMD shares the FPU between 2 cores
- Compute nodes want as many cores for jobs as possible
  - Use ½ cores for jobs. Other ½ for LND
- Gemini hardware attached to only one of the two sockets via the HyperTransport.
  - Test if CPT on second socket adds any value
- Hardware check summing

Test configuration

- Are more CPT better.
  - Service node – 1 socket with 6 cores
- Were do threads cost us versus benefits
  - Computes have 24 cores total
  - Don't want to use all the cores
  - Optimize core usage based on NUMA
- When do we saturate the interconnect.
- Which crypto check sum algorithm is best

Progress so far

- Base line numbers finish
- SMP scaling code done and stable
  - More optimizations possible
- Checksumming code work in progress
  - Issues with lack of kernel crypto algo support
  - Have code but needs to be debug. Oops :-(
- TODO
  - SMP performance evaluation
  - Delay due to Lustre 2.4 testing which is highest priority

Thank you!