Taking Advantage of Multi-cores for the Lustre Gemini LND Driver

James Simmons

Oak Ridge National Laboratory
Leadership Computing Facility

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**
  - **6 to 7 GB/s bulk messages**
  - **3 GB/s small messages**
- **Gemini 1.8 LND driver real numbers**
  - **2.6 GB/s bulk messages**
  - **1.6 GB/s small messages**

Causes

- **Checksumming**
  - **On – node to node gives 1.6 GB/s**
  - **Off – node to node gives 3.8 GB/s**
- **Kernel threads not optimized**
  - **Are their enough?**
  - **Threads free to migrate to any core**
  - **Memory allocation not NUMA aware**
- **Has this been solved before?**

Lustre 2.4

- **Lustre had the same challenges**
  - **New crypto api used for check summing.**
    - **future work for gemini LND**
  - **SMP scaling enhancements**
    - **Results covered in this talk.**

Gnilnd SMP scaling enhancements

- 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

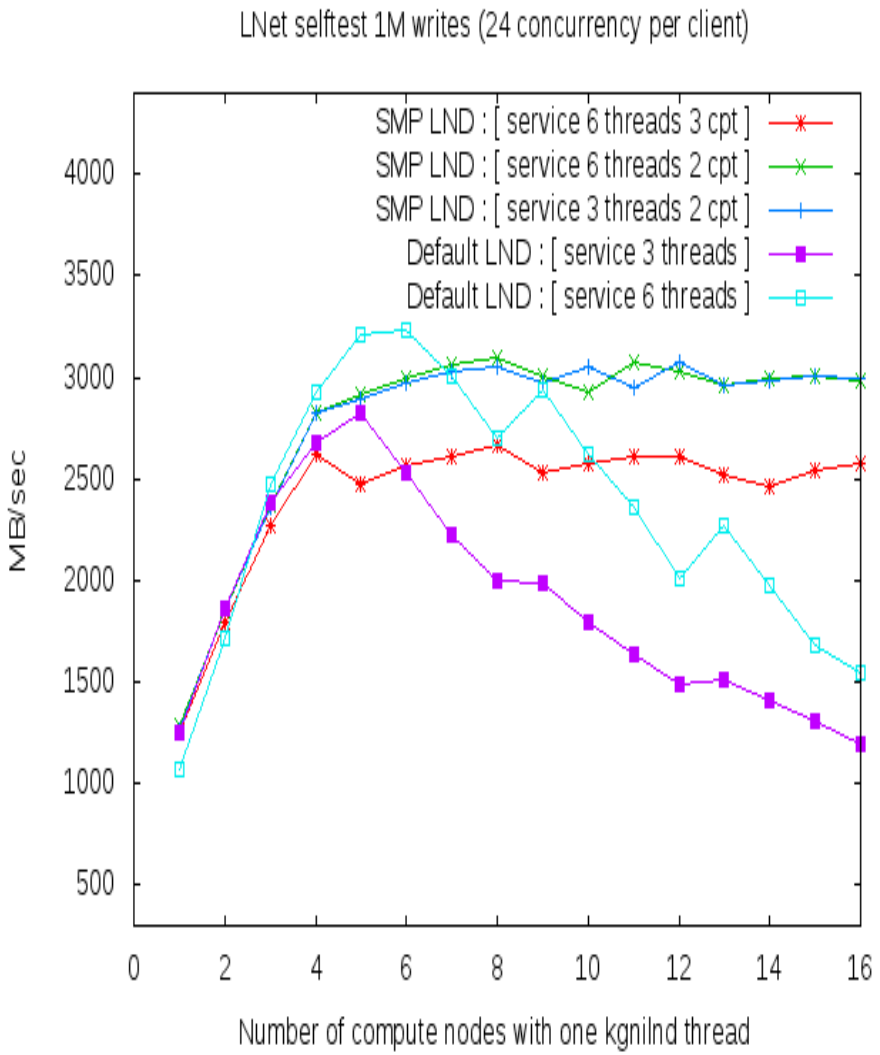
SMP API gives greater control

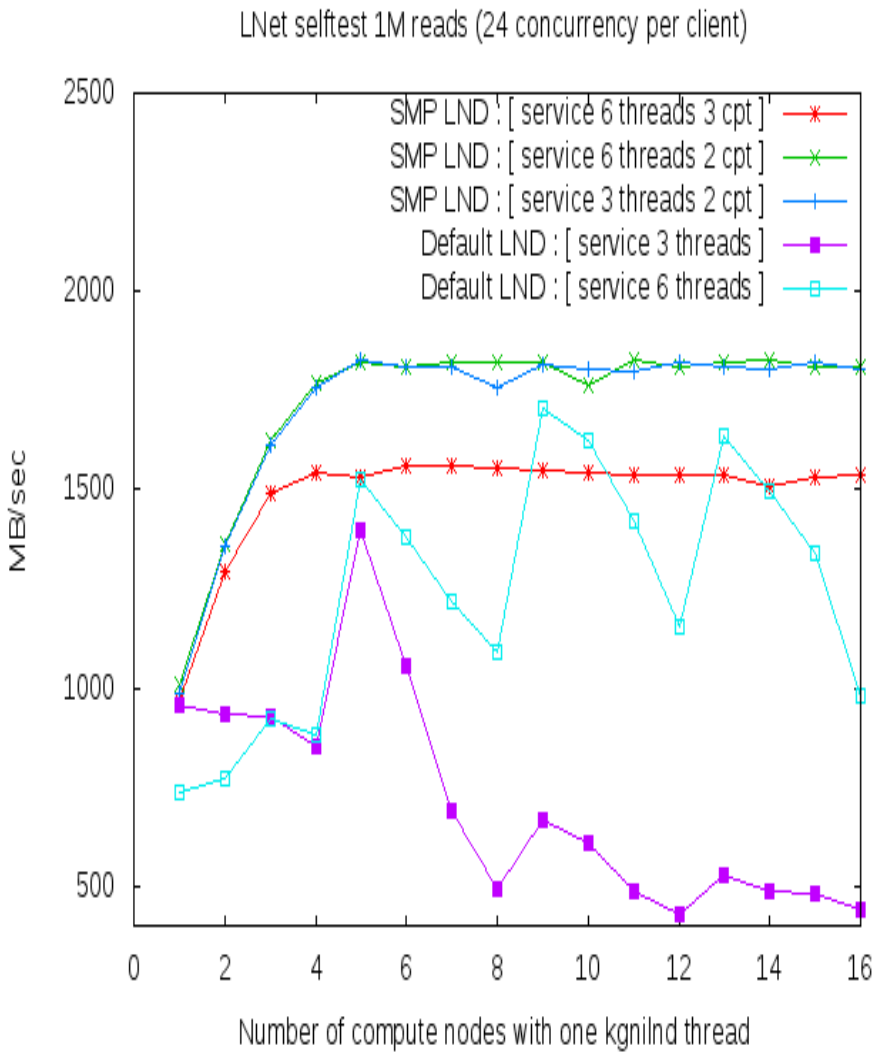
- **You can control which cores belong to which CPT**
  - **Don't need to use all cores**
- **You can map LNET interfaces to specific CPT**
  - **Use this to limit compute node noise**

Hardware influences configuration

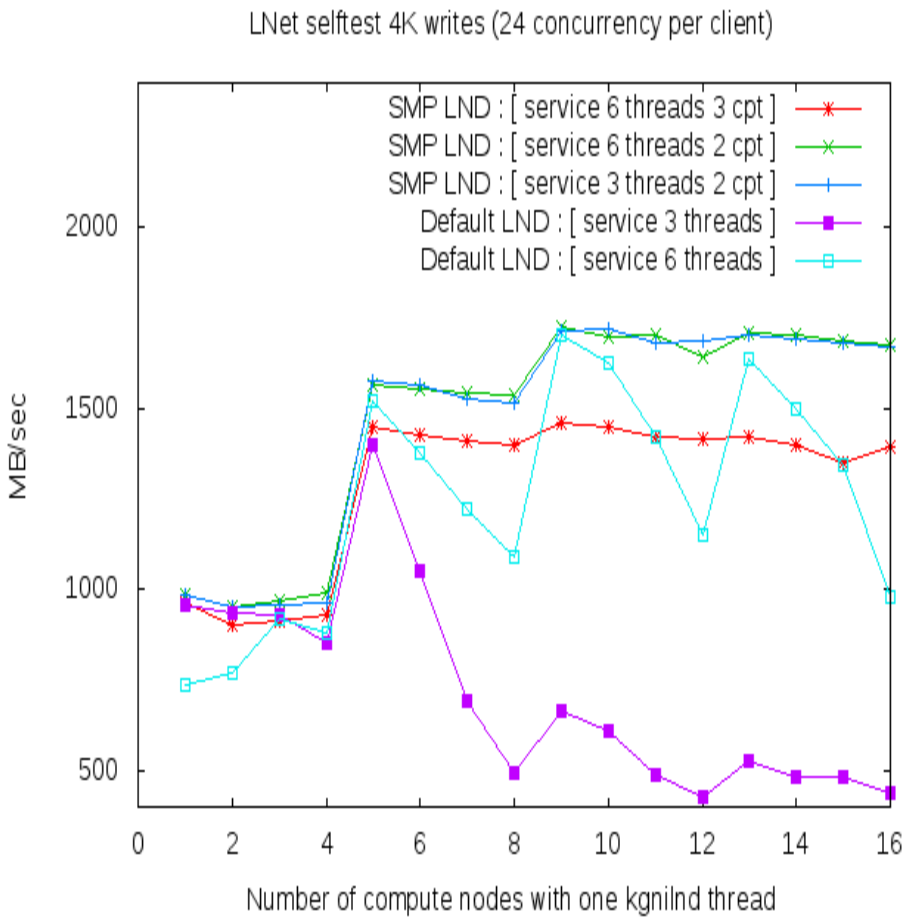
- **Processor properties**
  - **NUMA and cache shared between cores**
  - **Some AMD processors shares the FPU between 2 cores**
  - **Exploit instruction set for hardware checksumming**
- **Gemini hardware attached to one socket via the HyperTransport**
  - **Socket has two NUMA nodes. Using wrong one gives penalty.**

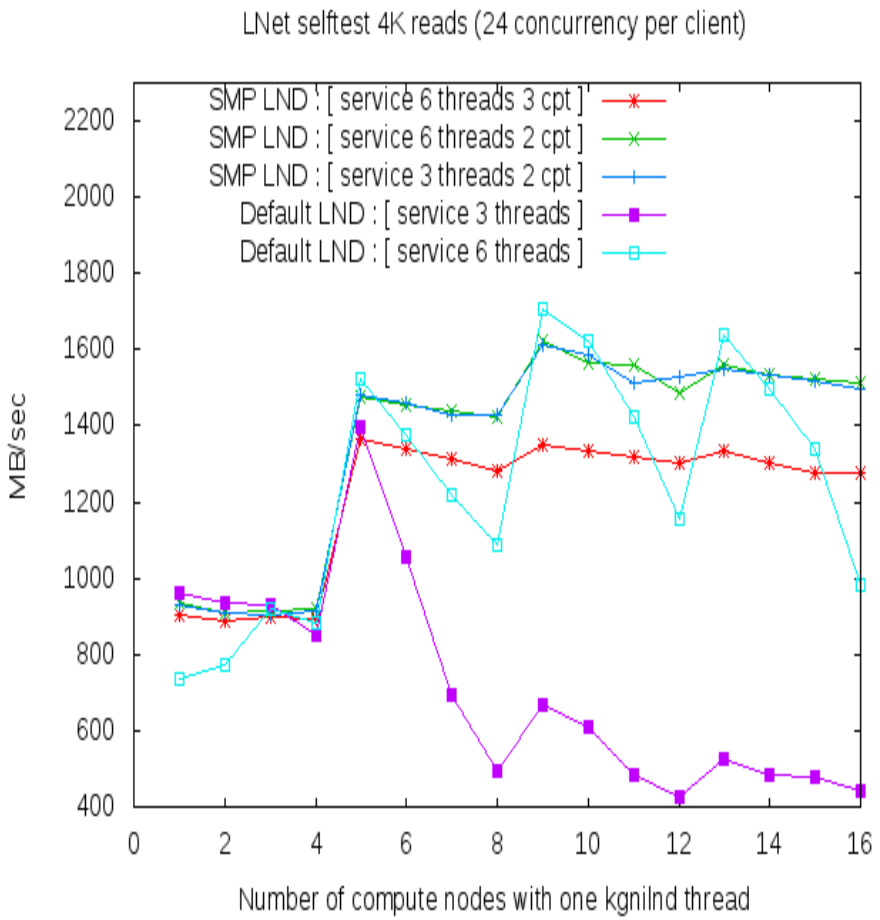
16 compute nodes to 1 router – 1MB transfers



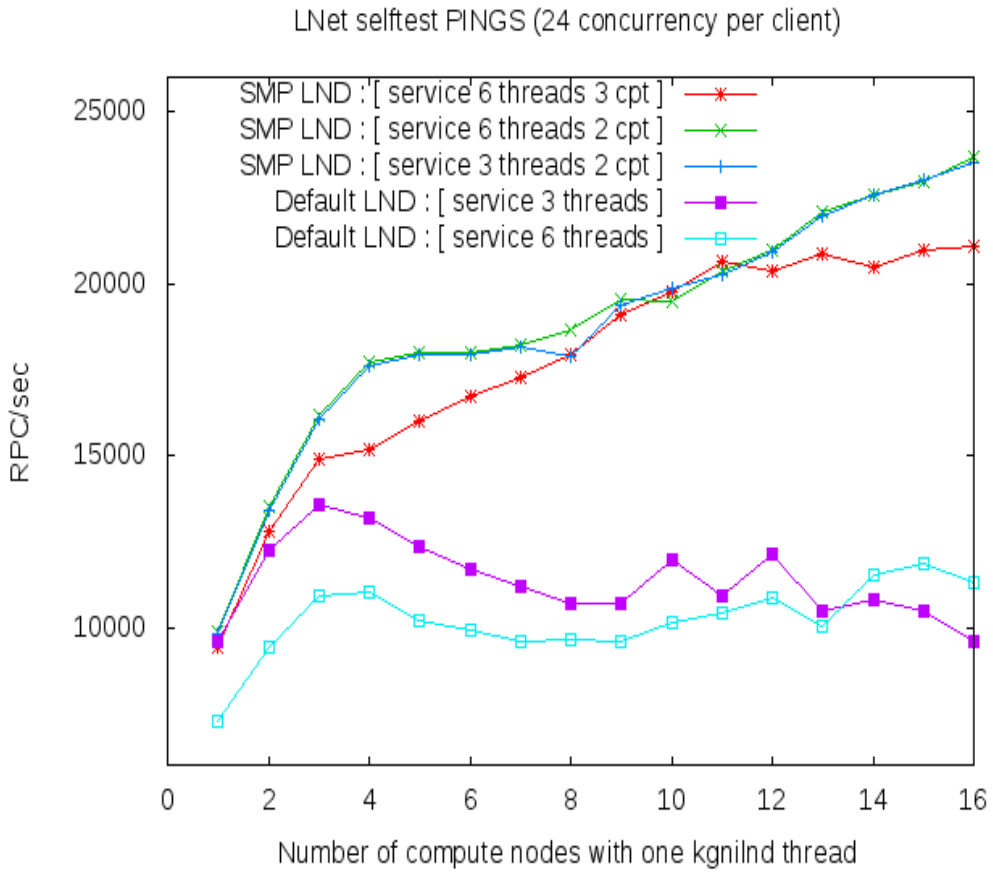


16 compute nodes to 1 router – 4K transfers



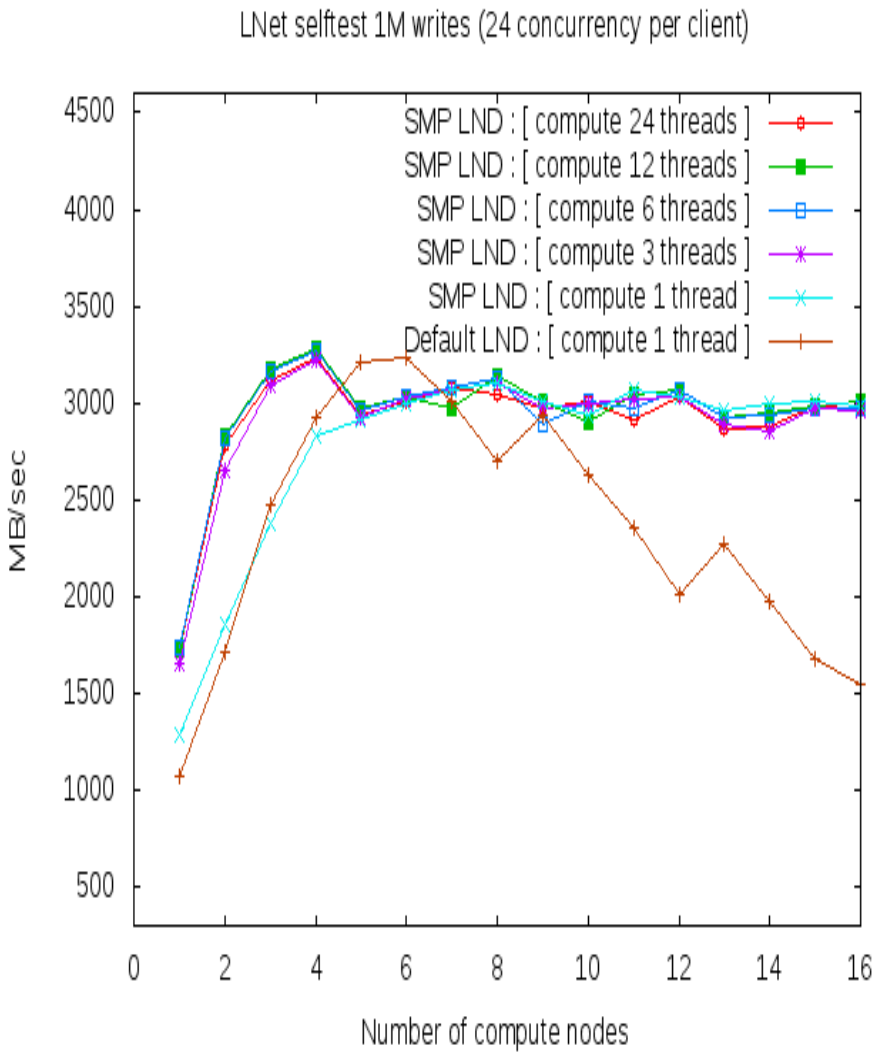


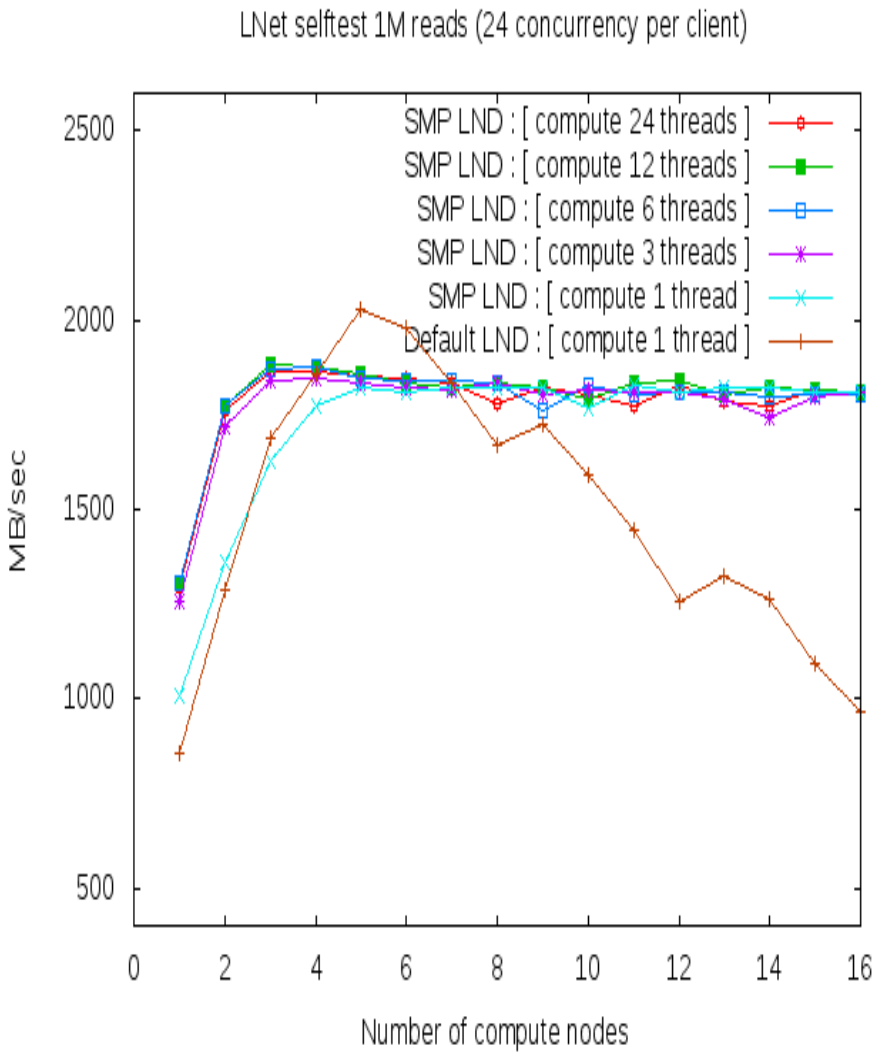
16 compute nodes to 1 router – pings



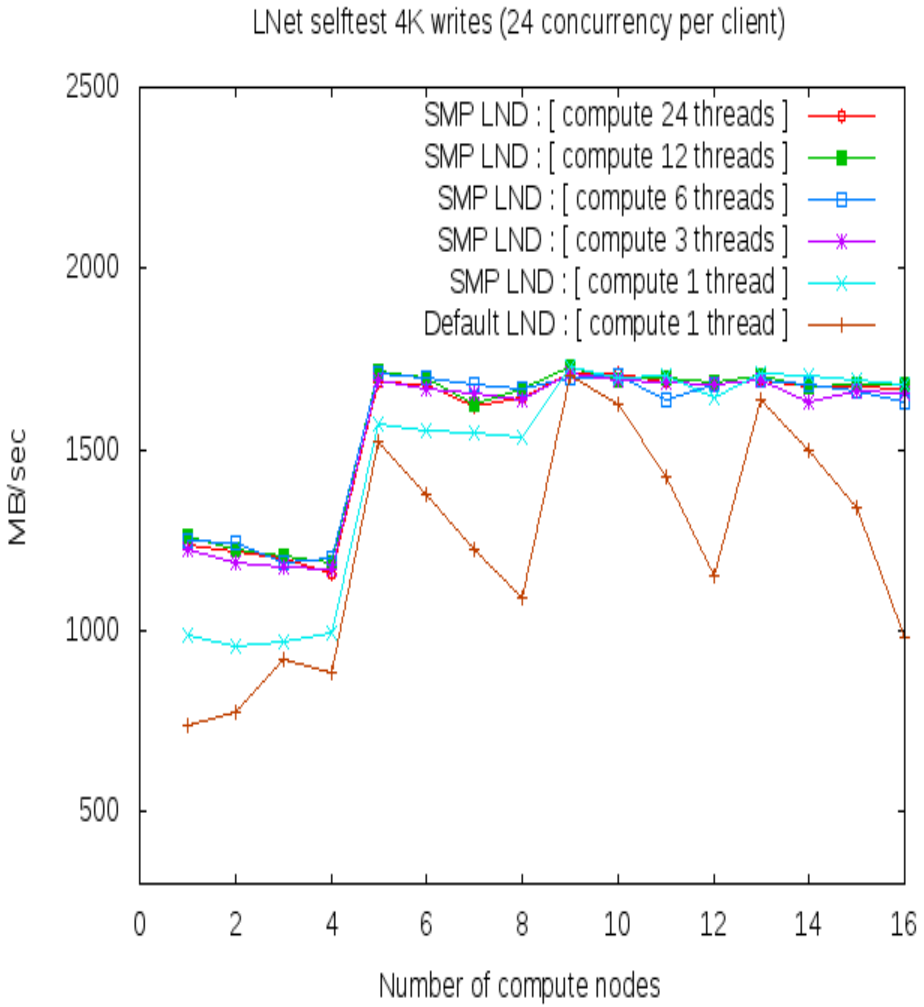
- **Adding 3 threads does not give us a gain.**
- **Creating more CPTs degrades performance**
- **We get consistent performance at all scales**
- **Pings show without checksumming we should have linear scaling**

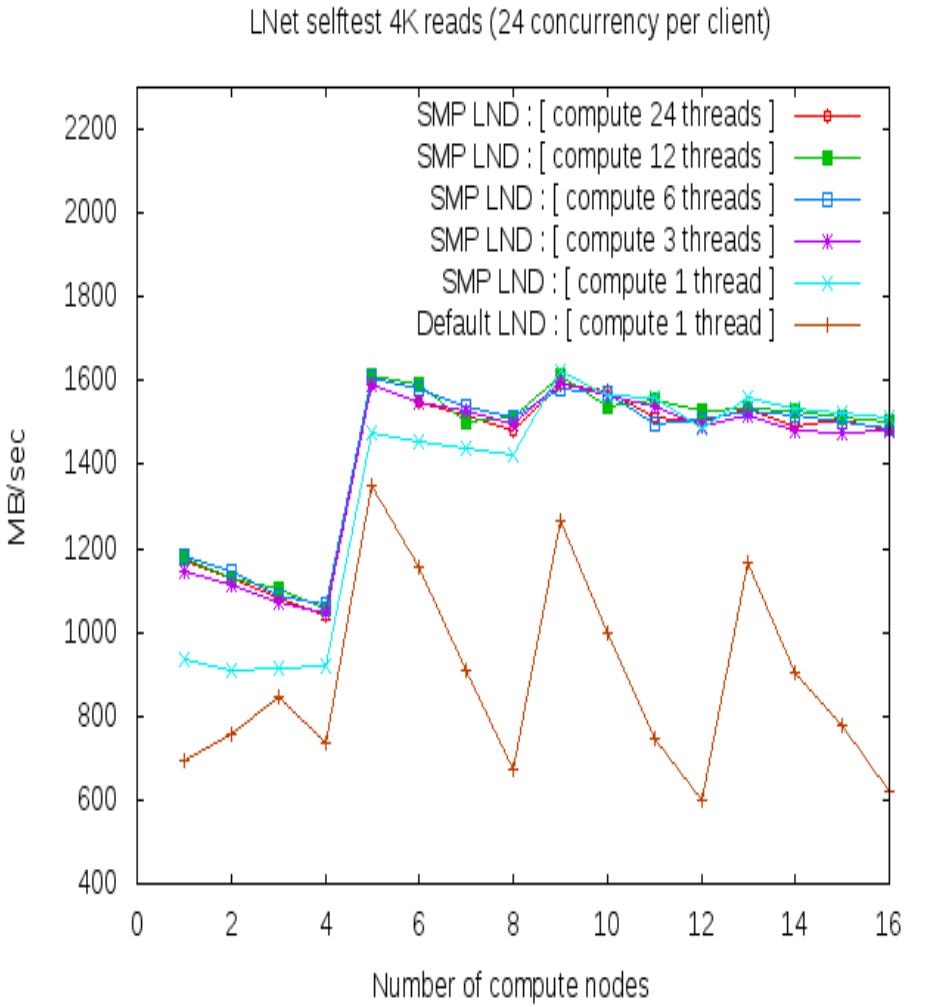
Many compute nodes with increasing kernel threads to one router – 1MB transfers



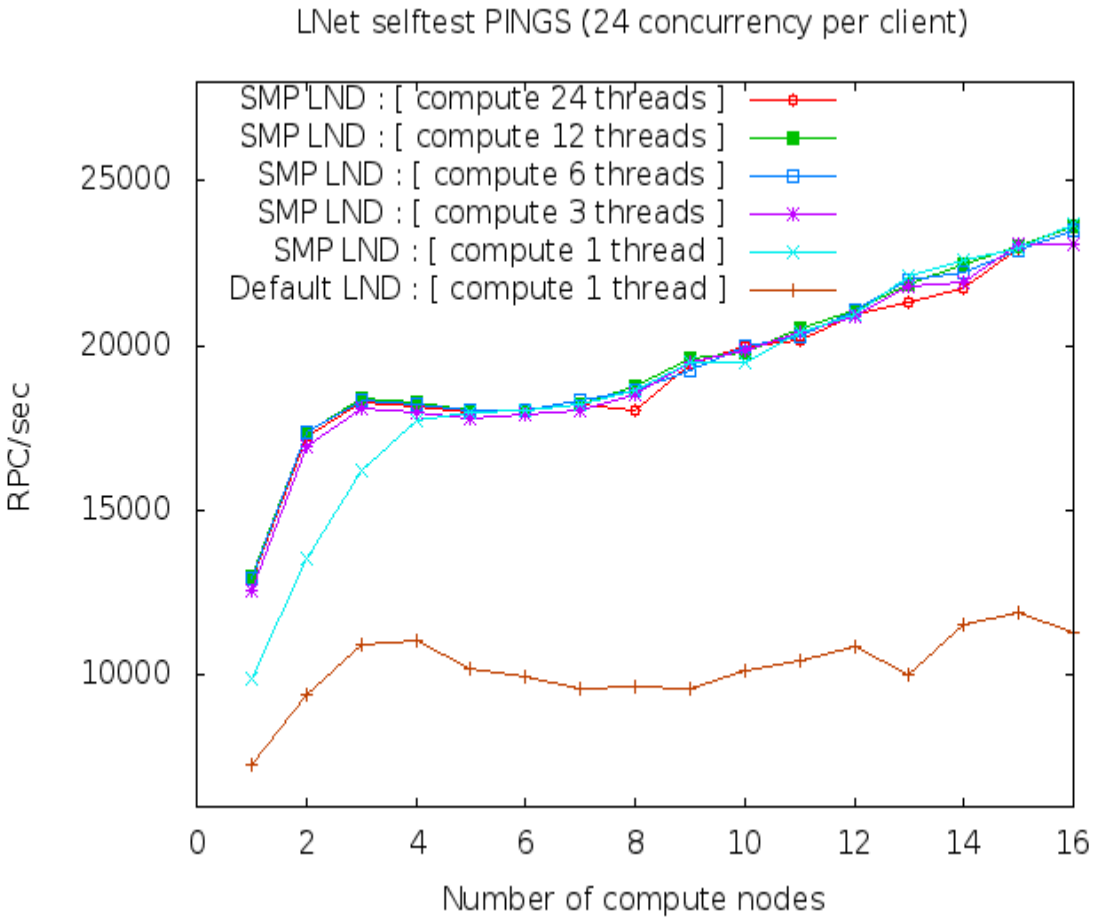


Many compute nodes with increasing kernel threads to one router – 4K transfers



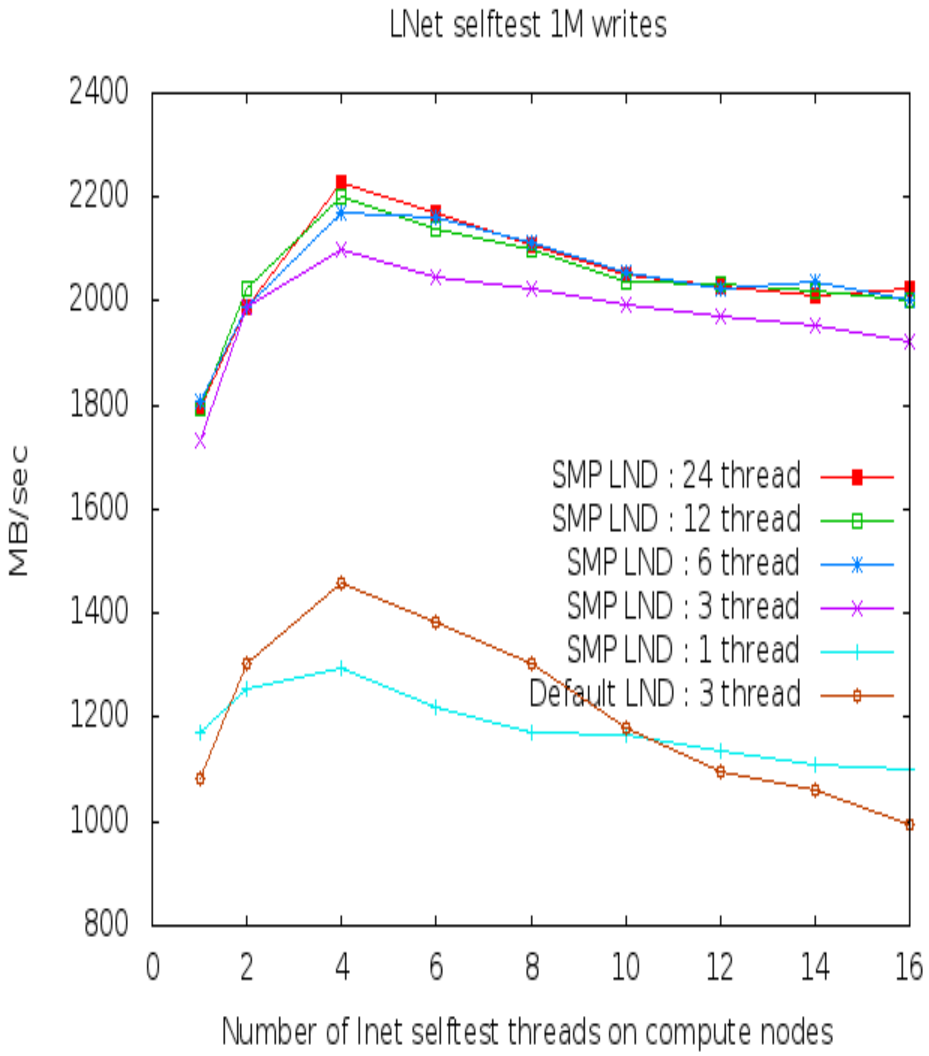


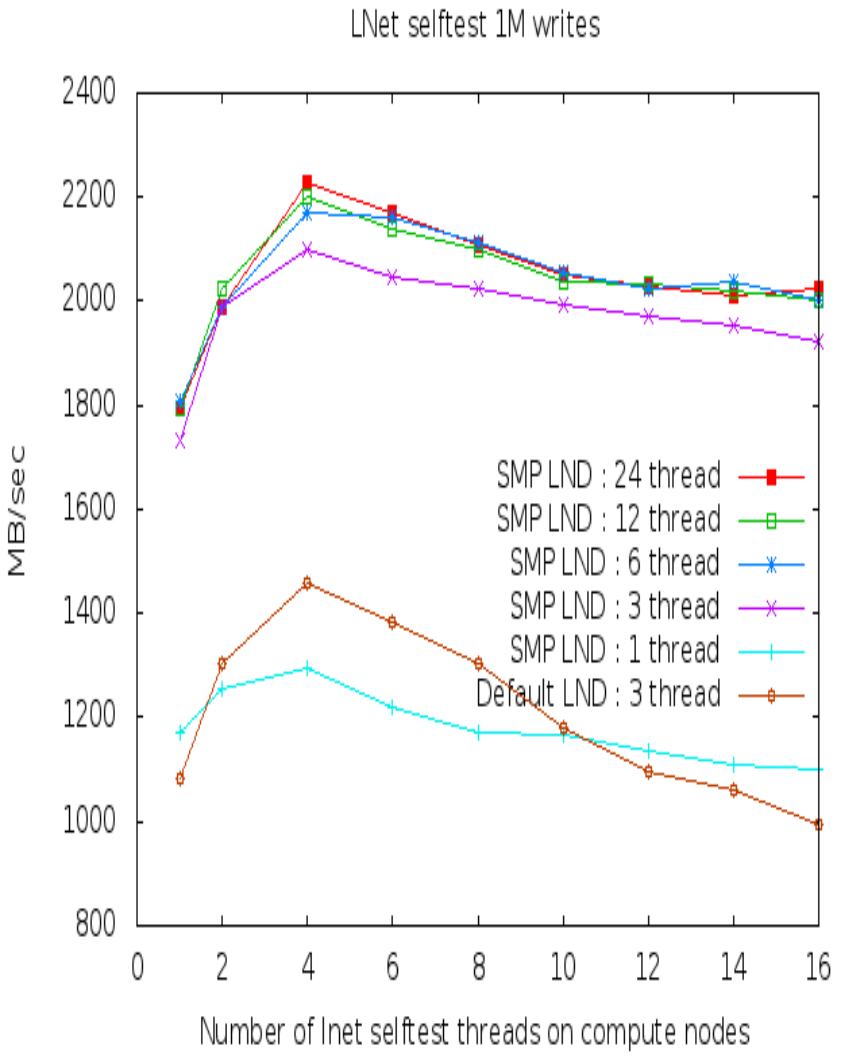
Many compute nodes with increasing kernel threads to one router – pings



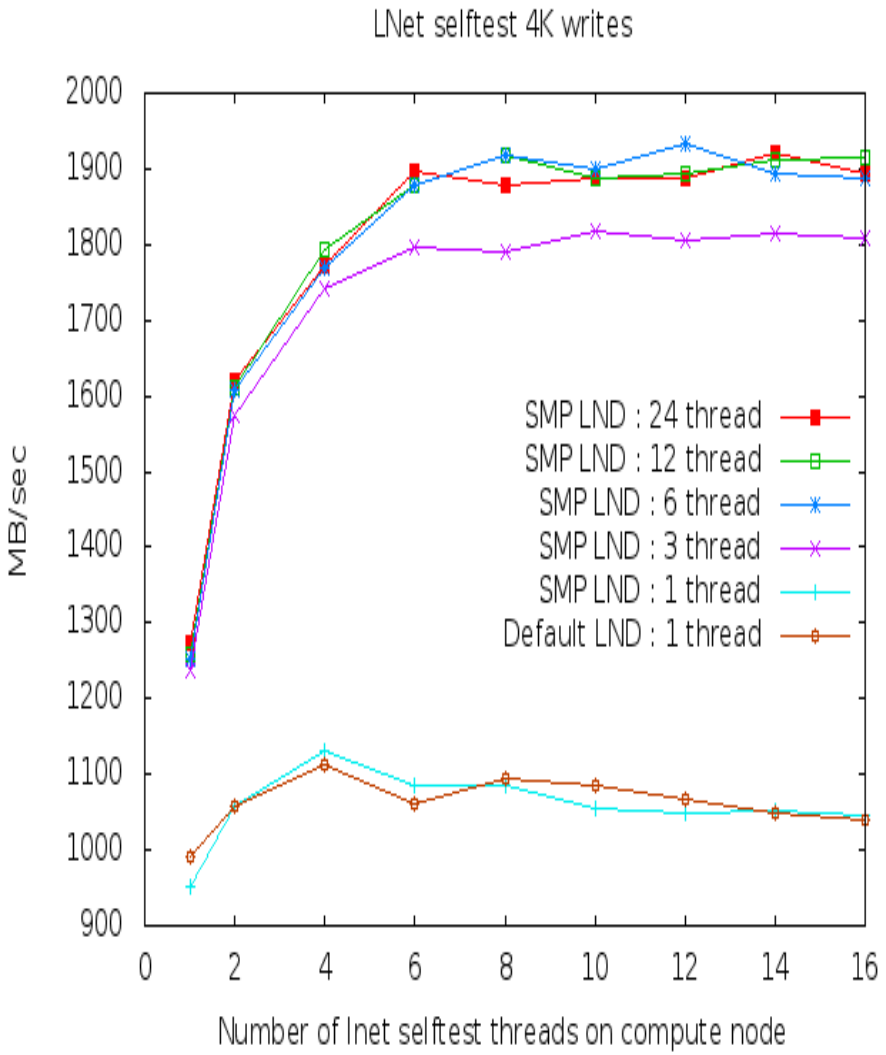
- **More than 3 threads gives no gain**
- **3 threads only small gain over one at small compute pool size**
- **Consistent behavior**
- **Pings reveal linear scaling**

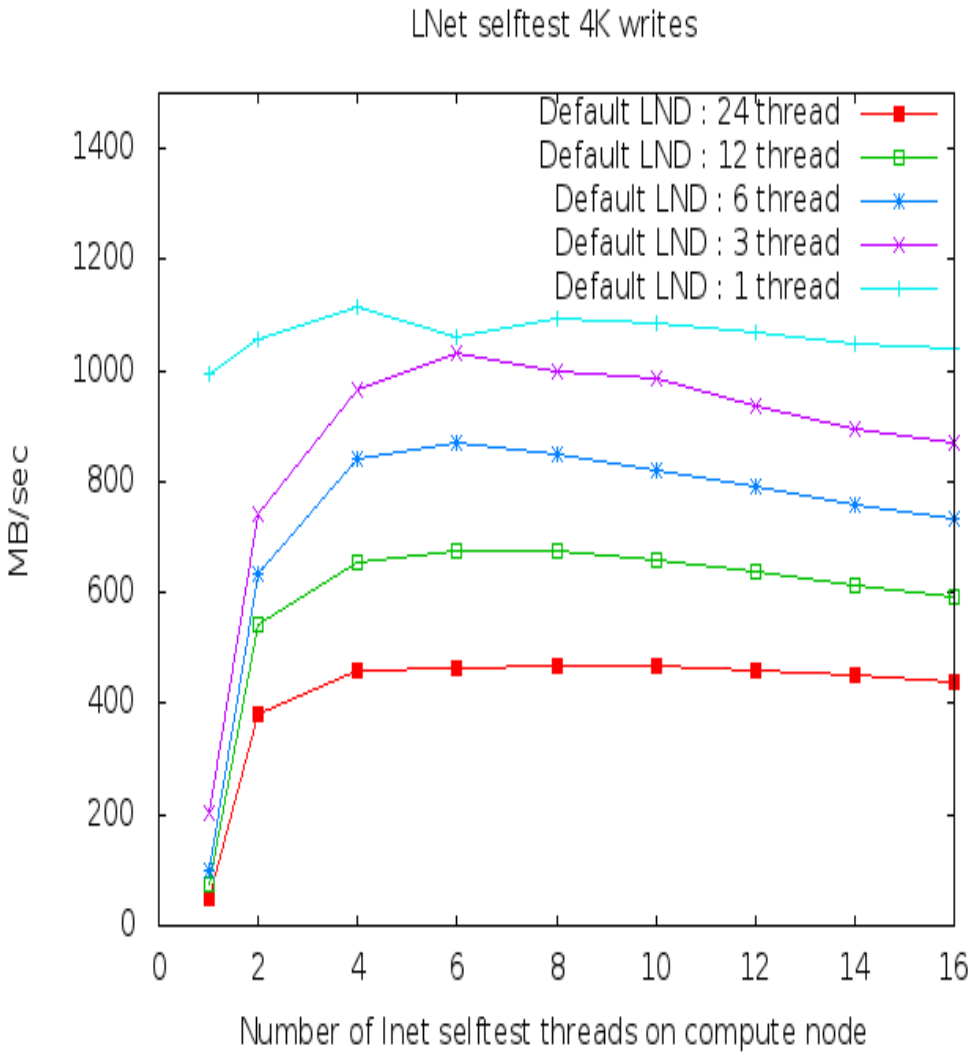
Are more kernel threads worth it on compute nodes compute to compute improvements





Are more kernel threads worth it on compute nodes compute to compute improvements





Future work

- **Testing on AMD Interlogos**
- **Two thread testing on computes**
  - **Recent testing shows behavior like three threads**
- **Lustre Crypto api**
  - **Test other checksum algorithms**
  - **Hardware accelerate checksum if platform not supported**
  - **Only do LND checksum for small packets or DVS**
  - **Other more long term solutions.**