High Performance Computing using Linux:

The Good and the Bad

Christoph Lameter

HPC and Linux

- Most of the supercomputers today run Linux.
- All of the computational clusters in corporations that I know of run Linux.
- Support for advanced features like NUMA etc is limited in other Operating systems.

Use cases: Simulations, visualization, data analysis etc.

History

- Proprietary Unixes in the 1990s.
- Beginning in 2001 Linux began to be used in HPC. Work by SGI to make Linux work on supercomputers.
- Widespread adoption (2007-)
- Dominance (2011-)

Reasons to use Linux for HPC

- Flexible OS that can be made to behave like you want.
- Rich set of software available.
- Both open source and closed solutions.
- Collaboration yields increasingly useful tools to handle cloud based as well as computing grid style solutions.

Main issues

- Fragile nature of proprietary file systems.
- OS noise, faults, etc etc.
- File system regressions on large single image systems.
- Difficulties of control over large amount of Linux instances.

HPC File Systems

- Open source solution
  - Lustre, Glustre, Ceph, OpenSFS
- Proprietary filesystems
  - GPFS, CXFS, various other vendors.

Storage Tiers

Exascale issues in File systems

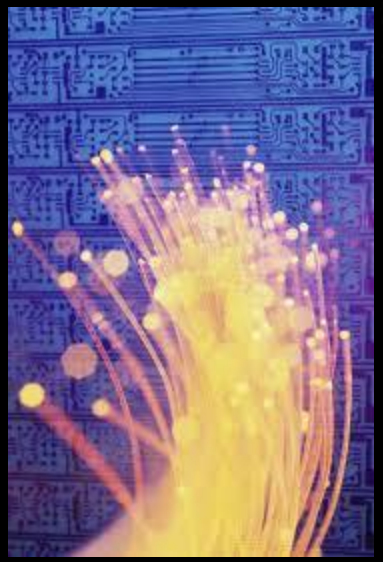
Local SSDs (DIMM form factor, PCI-E)

Remote SSD farms (Violin et al.)

Filesystem issues

- Block and filesystem layers etc does not scale well for lots of IOPS.
- New APIs: NVMe, NVP
- Kernel by pass (Gluster, Infiniband)
- Flash, NVRAM brings up new challenges
- Bandwidth problems with SATA. Infiniband, NVMe, PCI-E SSDs, SSD DIMMS

Interconnects



- Determines scaling
- Ethernet 1G/10G (Hadoop style)
- Infiniband (computational clusters)
- Proprietary (NumaLink, Cray, Intel)
- Single Image feature (vSMP, SGI NUMA)
- Distributed clusters

OS Noise and faults

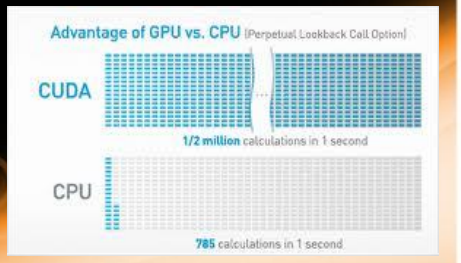
- Vendor specific special machine environment for low overhead operating systems
  - BlueGene, Cray, GPU “kernels”
  - Xeon Phi
- OS measures to reduce OS noise
  - NOHZ both for idle and busy
  - Kworker configuration
  - Power management issues

- Faults (still an issue)
  - Vendor solutions above remove paging features
  - Could create special environment on some cores that run apps without paging.

Command and control

- Challenge to deploy a large number of nodes scaling well.
- Fault handling
- Coding for failure.
- Hardware shakeout/removal.
- Reliability

GPUs / Xeon Phi



- Offload computations (Floating point)
- High number of threads. Onboard fast memory.
- Challenge of host to GPU/Phi communications
- Phi uses Linux RDMA API and provides a L:inux kernel running on the Phi.
- Nvidia uses their own API.
- The way to massive computational power.
- Phi: 59-63 cores. ~250 hardware threads.
- GPUs: thousands of hardware threads but cores work in lockstep.

Conclusion

- Questions?
- Answers?
- Opinions?