Linux Cluster Computing
An Administrator's Perspective

Robert Whitinger

Traques LLC
and
High Performance Computing Center
East Tennessee State University
:
http://lxer.com/pub/self2015_clusters.pdf

2015-Jun-14

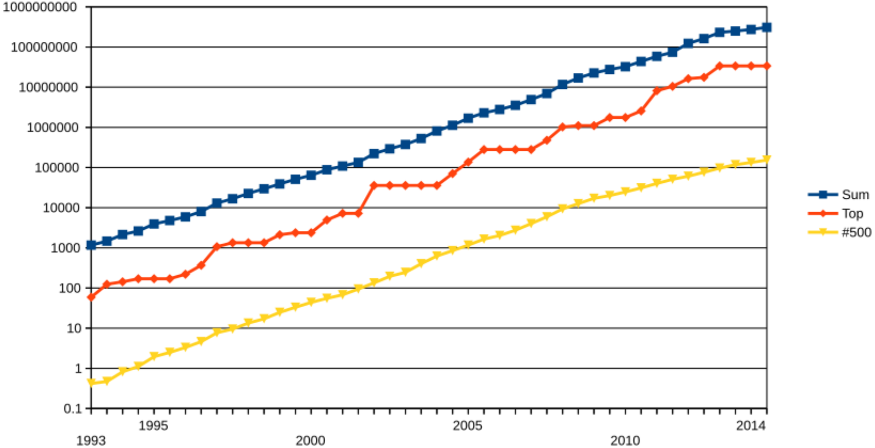
Supercomputing – Where do we stand today?

Linux is nearly universally accepted in the Supercomputing space.

- Linux runs 485 (97%) of the top 500 supercomputers
- Unix runs 13 (2.6%)
- Windows runs 1 (0.2%)
- and one system is classified as "Mixed"

Source: www.top500.org (November 2014)

Supercomputer performance



Source: www.top500.org

Top ranked Supercomputers

- 2013: 33 PFlops – Tianhe-2, Guangzhou, China
- 2012: 17 (27) PFlops – Cray Titan, Oak Ridge National Labs, Tennessee
- 2010: 2.5 Pflops – Tianhe-1A, Tainjin, China
- 2009: 1.7 Pflops – Cray Jaguar, Oak Ridge, Tennessee

Planned for 2017: $\approx 100+$ PFlops – IBM Summit, Oak Ridge, Tennessee

TITAN – Fastest Supercomputer in the USA

Early Supercomputing Architecture

- 1964: CDC 6600
- 1975: Cray-1
- Vector machine, 64 bit
- 12 pipelines
- Over 80 sold
- Performance: 80 MFlops
- Power: 115 kW
- Limited by
  - distance (light speed)
  - cooling
- In comparison:
  - RPi 2: $\approx$ 1GFlop

Distributed Architecture

- 2004: IBM Blue Gene
- 70 TFlops
- Small processors
- In very large numbers
- With fast networking
- Led to PFlop systems

Linux in Early Cluster Computing

- 1994: Beowulf cluster project
- Thomas Sterling and Donald Becker at NASA
- A High Performance Computing (HPC) Cluster
  - using commodity off the shelf systems
  - network connected
  - message passing between nodes
  - shared file system (NFS)
  - open source software
  - each system is complete usually with identical operating system configurations
- Quickly spread through NASA and academic/research organizations
- HPC was now affordable for scientific computing

Elements of a Cluster

- Compute nodes (the more the better)
- Networking
  - Ethernet 100/1000 Mb/s (inexpensive)
  - Infiniband 23 GB/s in Summit
- Shared /home directory usually using NFS
- Message passing facilities
  - openMPI
  - mpich2

Typical HPC Cluster (ETSU – Johnson City TN)

Inside View

Cluster-oriented Linux Distributions

- Kylin Linux (used in China)
- ClusterKnoppix
- openMosix
- Scyld
- Quantian
- in house adaptation of a mainline distribution
- various commercial Linux distributions (IBM, Cray)
- Rocks Cluster Distribution

Rocks Cluster Distribution

- Intended specifically for HPC Clusters
- 2000: Created at San Diego Supercomputer Center (SDSC)
- Open source
- Runs clusters ranging from 10 to 8000+ nodes
- actively maintained (version 6.2 released 2015-May-10)
- based on CentOS (RHEL) but adds cluster essentials
  - MPI – Message passing interface
  - Cluster-aware installer
  - Kickstart integration
  - Monitoring (Ganglia)
  - Scheduling (SGE)

Administration essentials

- Scripting or Point & Click?
- Keeping 1000's of nodes in sync
- User community, training and support
- Quickstart documentation
- Job Scheduling and Load Balancing
- Uptime
- Benchmarking
- Storage strategies
- Power strategies
- Third party software administration
- Software updates
- Administration team

Scripting or Point & Click

- If you have a task that you will do only once then an intuitive graphical environment is convenient
- But what if you need to do the task twice?
- or 50 times?
- or in the case of Supercomputing 500,000 times?
- Then scripted task automation rules the game
- This is why we don't find Windows on Supercomputers and Clusters
- Windows was designed first as a graphical environment with scripting added on top
- Linux was designed first as a scriptable environment with graphical added on top

How to make 100's or 1000's of nodes look the same

- Doing it by hand only works for a handful of nodes
- Rocks/RedHat solution: RPM/Kickstart
- and for ad-hoc admin tasks: pdsh is your friend

```
pdsh -w compute-0-[0-49] uptime
compute-0-0:  13:16:44 up 5 days, 10:29,  9 users,  load average:
<snip ...>
compute-0-49:  13:16:44 up 5 days, 10:32,  9 users,  load average:

pdsh -w pi@rpi[1-4] uptime
rpi1:  15:55:49 up 2 days, 22:20,  0 users,  load average: 0.00, 0
rpi3:  15:55:49 up 2 days, 22:20,  0 users,  load average: 0.02, 0
rpi4:  15:55:49 up 2 days, 22:20,  0 users,  load average: 0.01, 0
rpi2:  15:55:49 up 2 days, 22:20,  0 users,  load average: 0.01, 0
```

Building community – communicating

- Cluster Wiki – users support users
- Wiki pages communicate version update plans to users
- Code exchange
- Best practices
- Hello world quickstarts in each language

Sample quickstart – python

```
# launch with:  mpirun -np 50 python mpi_test.py

from mpi4py import MPI
import numpy as np
import platform

comm = MPI.COMM_WORLD

size = comm.Get_size()
rank = comm.Get_rank()
node = platform.node()

if rank== 0:
  data = np.arange(100, dtype=np.float)
  data[0] = 1.0
  for cn in range(1,size):
    comm.Send(data, dest=cn, tag=13)

if rank != 0:
  data = np.empty(100, dtype=np.float)
  comm.Recv(data, source=0, tag=13)

print "%s, rank: %d  size: %d  data: %f" \
  % (node,rank, size, data[0]*np.pi)
```

Sample quickstart – python

```
$ mpirun -np 50 python mpi_test.py

compute-0-0, rank: 0  size: 50  data: 3.141593

<snip...>

compute-0-49, rank: 49  size: 50  data: 3.141593
```

Job Scheduling and Load Balancing

- How to control the "computing hogs"
- Scheduler
- Grid Engine or SGE: an open source solution
- Allocate resources to applications and users
- Balance those resources with a fair strategy

Uptime

- Monitoring and text alerts
- ganglia
- SNMP
- KVM remote access
- IPMI remote power control for node restarts
- Node kickstarts concurrent with operations (SGE)

Benchmarking

- Linpack
  - for benchmarking computational performance
  - used in top500 ranking
- bonnie++ for disk I/O

```
Version  1.96       ------Sequential Output------ --Sequential Input- --Random-
Concurrency   1     -Per Chr- --Block-- -Rewrite- -Per Chr- --Block-- --Seeks--
Machine        Size K/sec %CP  K/sec %CP K/sec %CP K/sec  %CP K/sec %CP  /sec %CP
localhost.lo 15792M   733  99 149729   8 156620  10  3931   98 608665  18  9518 126
```

Power strategies

- Backup power on frontend systems and storage
- Mains on nodes
- Typical power configuration 208V three phase
- Titan saved $1M in copper cost going from 208V to 480V power

Third party software administration

- module
- RPMs for everything
- swtools (ORNL)
- smithy

Administration Team – staying on the same page

- Admin journal: `/root/admin.log`
- System configuration files checked in using mercurial version control

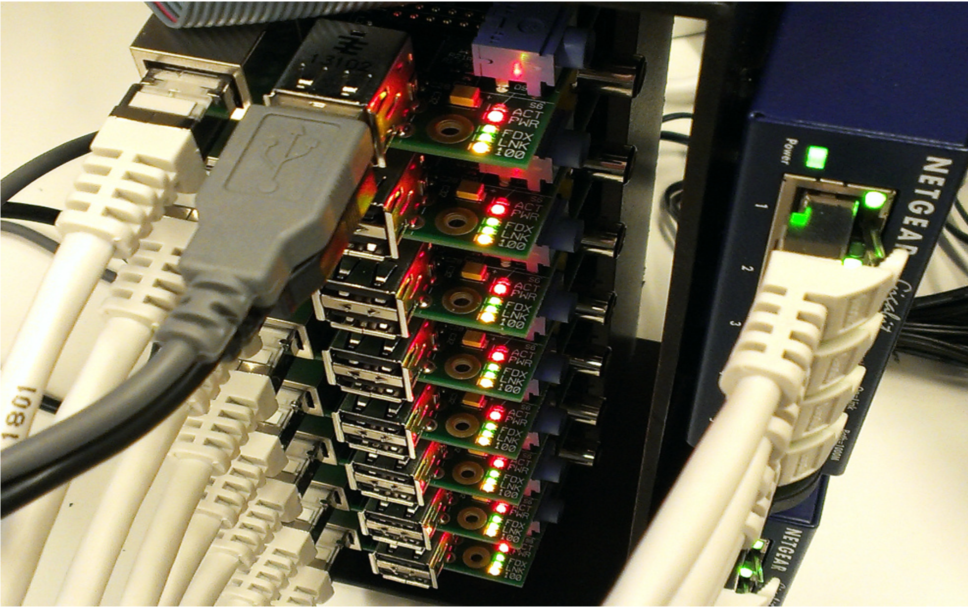
```
cd /etc
hg init
hg commit -Am 'initial commit'
```

What's next?

- Hybrid CPU/GPU system
- "Summit" to be delivered to ORNL in 2017, in production 2018
- 3400 nodes each with multiple CPUs and multiple GPUs
- 512GB of high bandwidth memory addressable from all nodes
- 5X faster than Titan
- 120 PB disk capacity with 1TB/s bandwidth
- Operating system: IBM Linux
- Software: openMPI, openACC, LSF scheduler
- Compilers: PGI, GCC, XL, LLVM
- Power: 10 MW

If you work with one of these...

If not maybe you could build one of these...



Questions?

These slides are located here:

`http://lxer.com/pub/self2015_clusters.pdf`

Contact: robert@whitinger.org