LCI workshop
on high performance clustered computing

Linux Clusters Institute: Cluster Stack Basics

Derek Weitzel
Software Engineer
Holland Computing Center, University of Nebraska
Open Science Grid

A Bunch of Computers

- Users can login to any node
- User file systems aren't shared between nodes
- Work is run wherever you can find space
- Nodes maintained individually

What's wrong with a bunch of nodes?

- Competition for resources
  - Size and type of problem is limited
- Nodes get out of sync
  - Problems for users
  - Difficulty in management

Cluster Approach

- Shared file systems
- Job management
- Nodes dedicated to compute
- Consistent environment
- Interconnect

What's right about the cluster approach?

- Easier to use
- Maximize efficiency
- Can do bigger and better problems
- Nodes can be used cooperatively

The Types of Nodes

- Login
  - Users login here
  - Compiling
  - Editing
  - Submitting and Monitoring jobs
- Compute
  - Users *might* login here
  - Run jobs as directed by the scheduler
- Support
  - Users *don't* login here
  - Do all the other stuff

Technology Choices

- I'm going to be listing a lot of different technologies

- Reasons to choose a technology:
  1. Best for a job
  2. Knowledge of the technology
  3. Mindshare – How big is the community? How easy is it to "StackOverflow" a problem?
  4. Institutional momentum – Is the institution already using it?

Cluster Design

- Each cluster is different
- Different OS, different tools…
- Even very different design decisions. Some clusters are ‘stateless’, with no local disk.
  - Every reboot is a re-install.
- There’s no easy-button that says: Make your cluster this way!

What a cluster needs – the mundane

- Network services – NTP, DNS, DHCP
- Shared Storage – NFS and beyond
- Logging – Consolidated Syslog as a **starting** point
- Licensing – FlexLM and the like
- Database – User and Administrative Data
- Boot/Provisioning – PXE, build system
- Authentication – LDAP

Specialized Needs of a Cluster

- Interconnect → An ideally low-latency network
- Job manager → Resource manager/ scheduler
- Parallel Storage → Get around the limitations of NFS

Network Services

- NTP – Network Time Protocol, provides clock synchronization across all nodes in the cluster
- DHCP – Dynamic Host Configuration Protocol, allows central configuration of host networking
- DNS – Provides name to address translation for the cluster
- NFS – Basic (Easy) UNIX network filesystem

Logging

- Syslog
  - The classic system for UNIX logging
  - Application has to opt to emit messages
- Monitoring
  - Active monitoring to catch conditions elective monitoring doesn't catch
    - Resource manager
    - Nagios/cacti/zabbix/ganglia
- IDS
  - Intrusion detection
  - Monitoring targeting misuse/attacks on the cluster

Basic Services

- Licensing – FlexNet/FlexLM or equivalent, mediates access to a pool of shared licenses.
- Database – Administrative use for logging/monitoring, dynamic configuration. Requirements of user software.
- Boot/Provisioning – For example PXE/Cobbler, PXE/Image or part of a cluster management suite

Authentication

- Flat files → passwd, group, shadow entries
- NIS → network access to central flat files
- LDAP → Read/Write access to a dynamic tree structure of account and other information
- Host equivalency

Cluster Networking

- Hardware Management – Lights out management
- External – Public interfaces to the cluster
- Internal – General node to node communication
- Storage – Access to network file systems
- Interconnect – high-speed, low-latency for multi-node jobs, and Expensive!

Some of these can share a medium, meaning shared physical infrastructure.

Interconnect

InfiniBand – Leading RDMA network. SDN approach, run IP on top of IB, extra offloading capabilities for MPI.

- Bandwidth 40 (QDR), 56 (FDR), 100 (EDR), 200 (HDR) Gbps
- Latency of 1.3 µs (QDR) .7 µs (FDR-10/FDR), .5 µs (EDR)

Interconnect

Ethernet – Traditional network, at minimum for server management and out of band management. May be on backplane.

- Bandwidth: 1, 10, 40, 100 Gbps
- Latency of 50-125 µs (GbE), 5-50 µs (10GbE), ~5 µs RoCE

Interconnect

Omni-Path – Intel RDMA network for Intel's end to end solution

- Based on the TrueScale fabric
- Future benefits to Intel CPU

Parallel Filesystem

Lustre – http://lustre.org/
BeeGFS – http://www.beegfs.com/
PanFS – http://www.panasas.com/
Spectrum Scale (GPFS) – http://www-03.ibm.com/systems/storage/spectrum/scale/

… and others for varying workloads.

Parallel filesystems take the general approach of separating filesystem metadata from the storage. Lustre and PanFS have dedicated nodes for metadata (MDS or director blades). GPFS distributes metadata throughout the cluster

Cluster Management

- Automates the building of a cluster
- Some way to easily maintain cluster system consistency
- The ability to automate cluster maintenance tasks
- Offer some way to monitor cluster health and performance

Cluster Management Software

- Bright Cluster Manager (http://www.brightcomputing.com/Bright-Cluster-Manager) (paid)
- xCAT (Extreme Cluster/Cloud Administration Toolkit) (https://xcat.org/)
- OpenHPC Project (https://openhpc.community/) - Moving very fast with help from the HPC community.
- Cobbler: (https://cobbler.github.io/) – Older, but still used a lot.
- Foreman: (https://theforeman.org/) – Developed and maintained by RedHat (free)

Configuration Management

While it is true that booting with a central boot server can make it easier to make sure the OS on each compute node (or, at least, each type of compute node) has an identical setup/install, there are still files which wind up being more dynamic. Some such files are password/group/shadow and hosts files.

- Ansible - Agentless (Python, SSH)
- Cfengine - Agent based (Domain Specific Language (DSL))
- Chef - Agent based (Ruby-based)
- Puppet - Agent-based (Ruby)
- Salt - Agent based (Python, ZeroMQ)

Software Installation and Management

All Linux distros have some sort of package management tool. For Redhat/CentOS/Scientific based clusters, this is rpm and yum [dnf]. Debian has dpkg and apt.

In any case pre-packaged software tends to assume that it is going to be installed in a specific place on the machine and that it will be the only version of that software on the machine.

One a cluster, it may be necessary to look at software installation differently from a standard Linux machine

- Install to global filesystem
- Keep boot image as small as possible
- Maintain multiple versions

Software Installation and Management

Specially designed tools for installing software or running software:

Build Utilities:

- EasyBuild (https://hpcugent.github.io/easybuild/)
- Spack (https://github.com/LLNL/spack)
- Maali (https://github.com/Pawseyops/maali)

Containerization for HPC (Docker-like tools):

- **Singularity** (http://singularity.lbl.gov/)
- Shifter (https://github.com/NERSC/shifter)

Software Installation and Management

There are a couple of tools useful for navigating the difficulties of maintaining user environments when dealing with multiple versions of software or software in non-standard locations.

- SoftEnv (http://http://www.lcrc.anl.gov/info/Software/Softenv)

  Useful for packaging static user environment required by packages

- Modules (http://modules.sourceforge.net/)

  Can be used to make dynamic changes to a users environment.

- **Lmod Modules** (https://sourceforge.net/projects/lmod/)

USE THIS ONE, YOU WILL BE HAPPIER!!!

Resource Manager/Scheduler

- Accepts job submissions, maintains a queue of jobs
- Allocates nodes/resources and starts jobs on compute nodes
- Schedules waiting jobs
- Available options
  - Grid Engine
    - SGE (Sun Grid Engine)
    - Univa Grid Engine
    - Son of Grid Engine
  - LSF / OpenLava (Load Sharing Facility)
  - PBS (Portable Batch System)
    - PBSPro (Community or Commercial)
    - Torque (Advanced scheduling requires Maui or Moab (paid))
  - **SLURM**
    - Community contributed wrappers
    - Grown in popularity

Best Practices

Here is a quick overview of the general functions to secure a cluster

- Risk Avoidance
- Deterrence
- Prevention
- Detection
- Recovery

The priority of these will depend on your security approach

Risk Avoidance

- Provide the minimum of services necessary
- Grant the least privileges necessary
- Install the minimum software necessary

The simpler the environment, the fewer the vectors available for attack.

Deterrence

- Limit the discoverability of the cluster
- Publish acceptable use policies

Prevention

- Fix known issues (patching)
- Configure services for minimal functionality
- Restrict user access and authority
- Document actions and changes

Detection

- Monitor the cluster
- Integrate feedback from the users
- Set alerts and automated response

Recovery

- Backups
- Documentation
- Define acceptable loss

Any questions for a basic cluster stack?