The Cray Linux Environment

Kevin Roy

Agenda

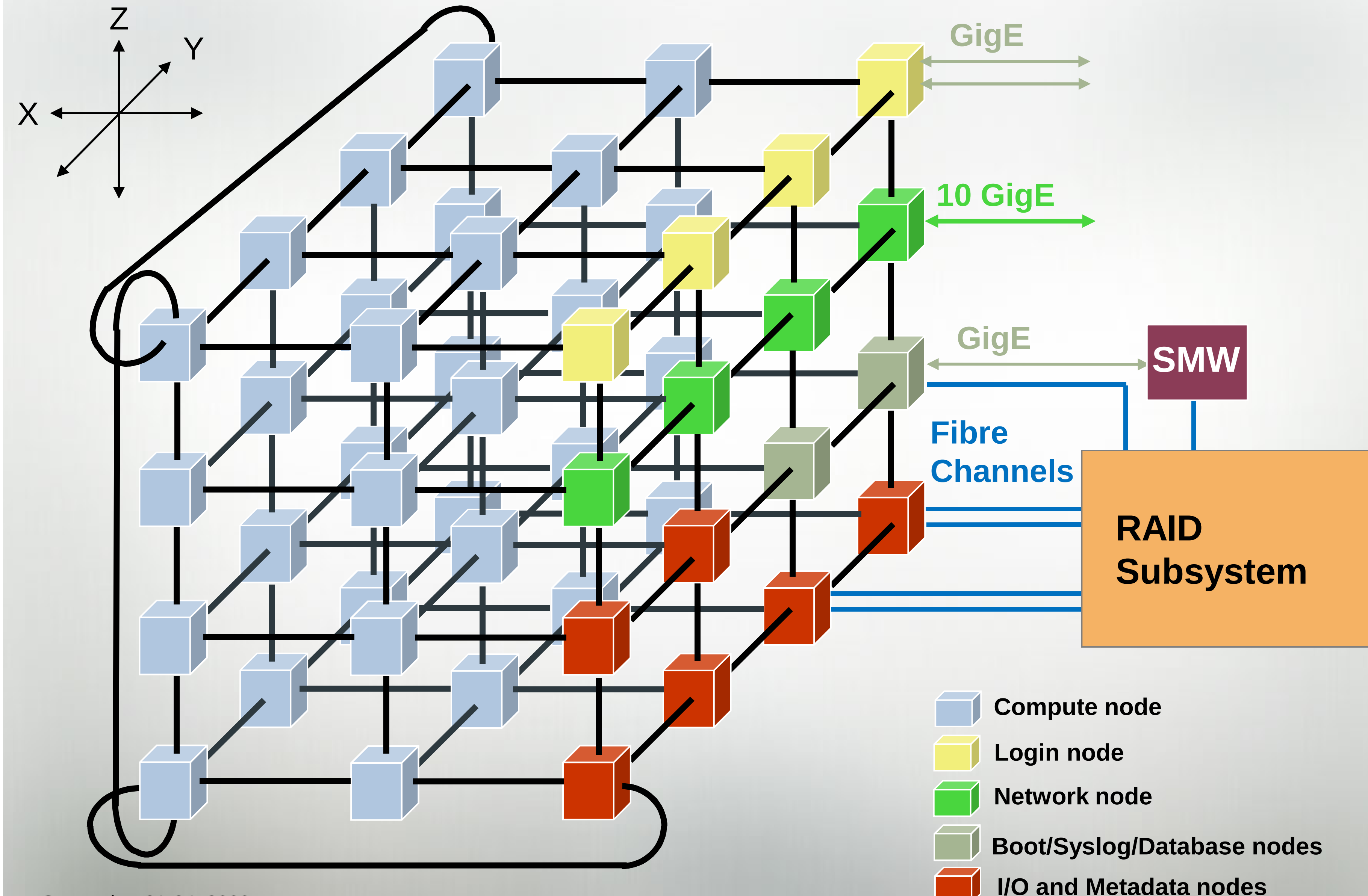
- Overview
- CLE Features
- CLE Programming
- The Storage Environment

What is CLE?

- CLE is the Cray Linux Environment.
- There is:
  - A full Linux run on the service and login nodes
  - A stripped down Linux kernel runs on the compute nodes and is CNL – Compute Node Linux.
- The combination of these two environments is called CLE.

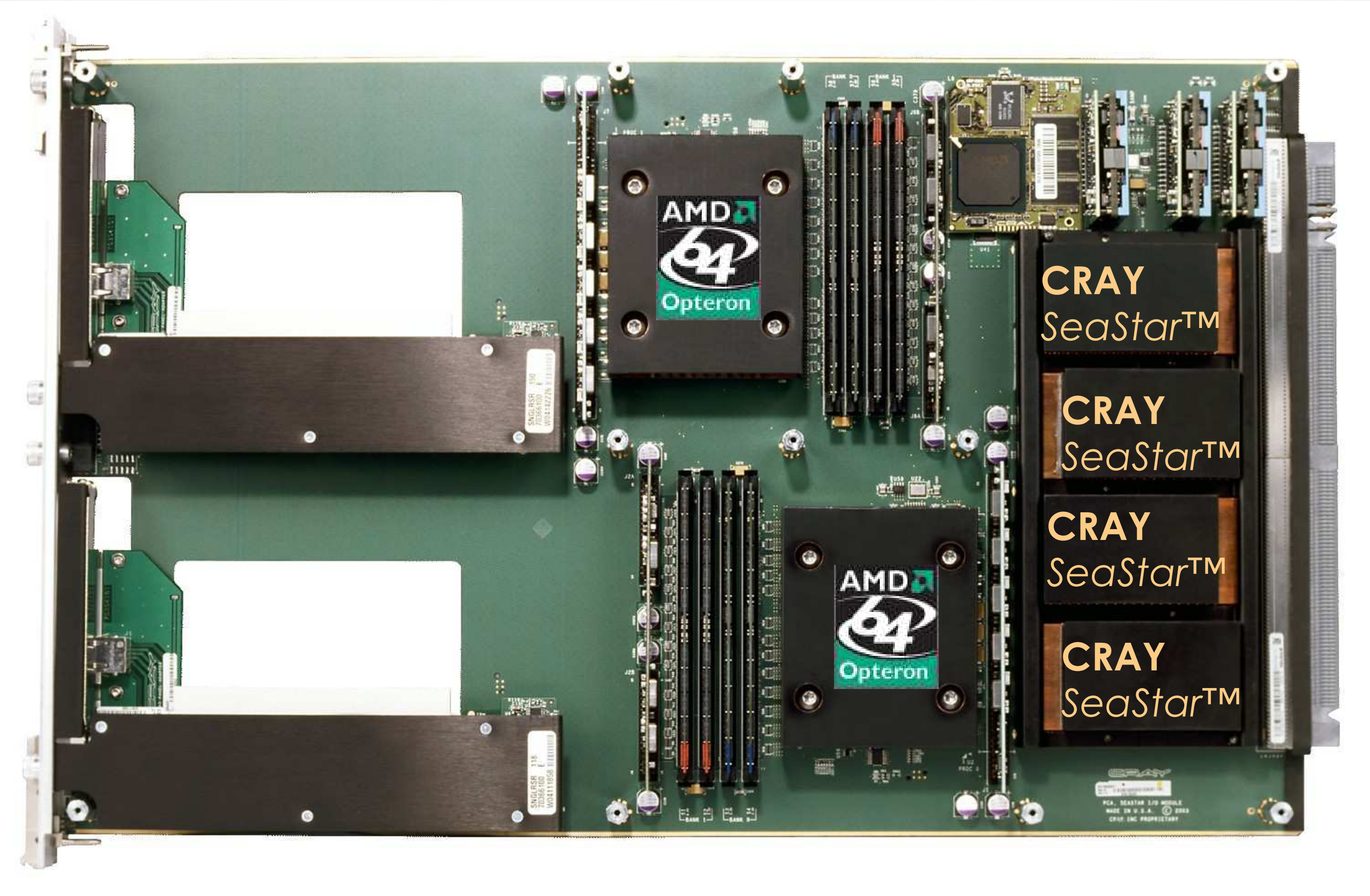
- We talk little of the Linux running on the service nodes because it is a full Linux.

- The most recently available version is CLE 2.2
- Many sites still run CLE 2.1

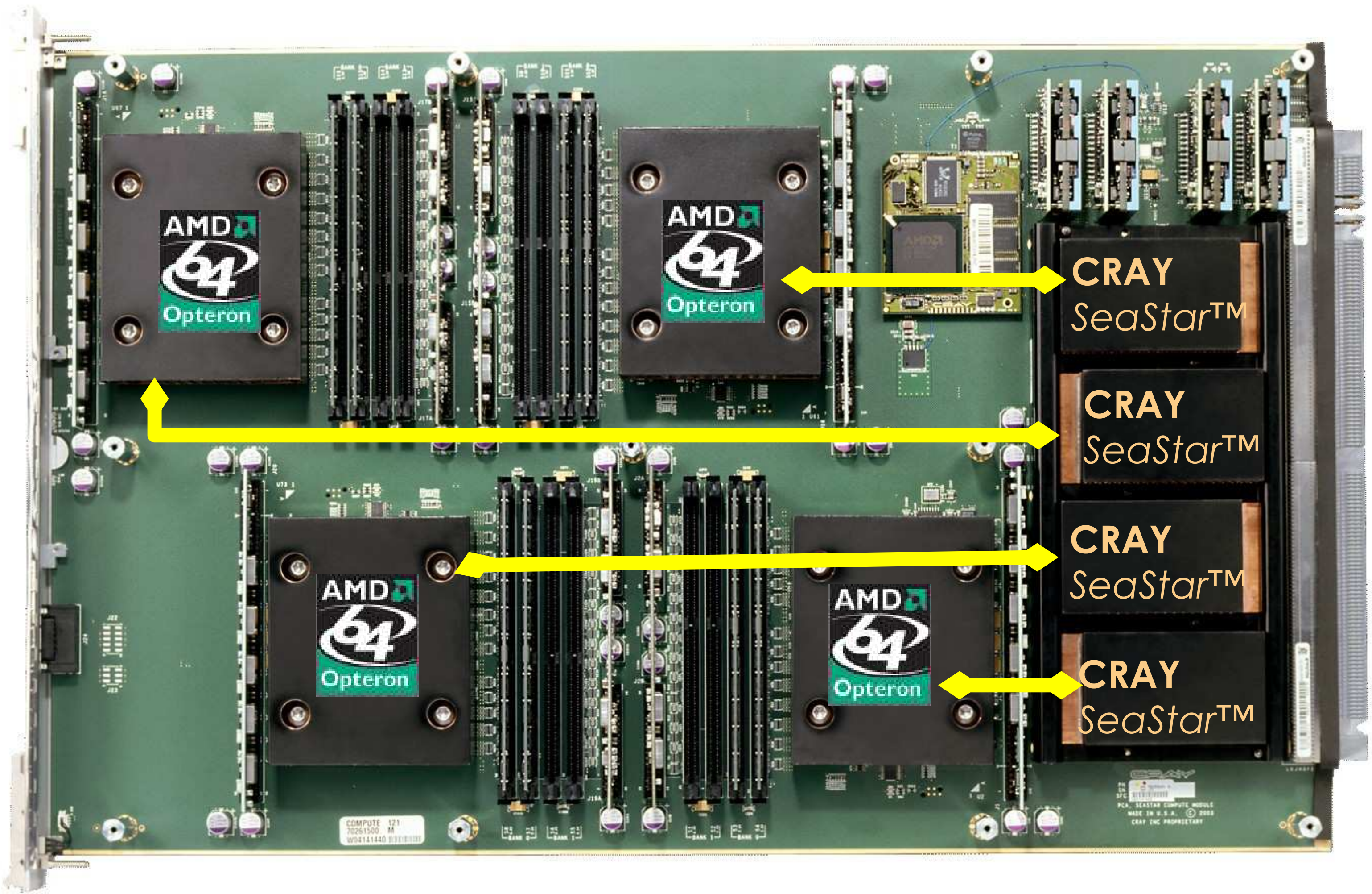


Z
Y
X
GigE
10 GigE
GigE
SMW
Fibre Channels
RAID Subsystem
Compute node
Login node
Network node
Boot/Syslog/Database nodes
I/O and Metadata nodes

Where do these OS' run? Service Blade



Where do these OS' run? Compute Blade



The CLE feature list

- Version 2.1
  - Based on SLES 10 SP1
  - Contains Lustre 1.6
  - DVS
  - First phase of the node health checker
  - CSA
  - VC2 awareness
  - NUMA Kernel changes
  - EAL3 evaluation
- Released in 2008 Q4
- There are regular updates

- Version 2.2
  - Based on SLES 10 SP2
  - Feature and performance upgrades to the Node Health Checker
  - Attribute Management
  - Checkpoint/Restart
  - LDAP integration with CSA
  - Infiniband Support
- Released in 2009 Q2
- There will be regular updates

CLE Features

- In order to maintain high performance and increase system reliability certain services, features and hardware is not available or have restricted availability on compute nodes.

- The ones that may be relevant to application programmers are:
  - Paging and Virtual Memory
  - NFS – Home file system.
  - Sockets
  - Dynamic Libraries
  - System calls

Paging & Virtual Memory

- Virtual Memory is much larger than physical memory.
- In some cases data is overflowed to disk and performance suffers.
  - This is called Paging
- Paging is not available on the XT series as there are no local disks.
  - Less moving parts per node
  - For most applications this is a unnecessary component
  - Local “file system” is placed in memory. This is why file system is kept small.
  - Use of /tmp consumes application memory and is not recommended and at some sites is restricted.
  - One effect of this can be seen when using PGI scratch files which by default are created in /tmp.

NFS - The Home Filesystem

- The home file system is available from servers within the XT infrastructure and is mounted via NFS.
- NFS is available on the login nodes
- The NFS service at this time is considered intrusive for inclusion on the compute node OS.

- Besides the effect on the compute node OS the NFS service is not a parallel high performance one.
- Data needs to be moved to Lustre so that it is available to the compute nodes.

- DVS can make this available.
  - This is a decision made by individual sites.

TCP/IP Programming

- TCP/IP does work on the XT but is not supported on compute nodes.
- You may see warnings when compiling using the Cray compiler drivers.
- There is no direct connection from compute nodes to systems outside the XT infrastructure so TCP/IP connections are not possible.
- This is possible with RSIP, but you should discuss your needs with CSC, as there are implications.

Dynamic Libraries

- By default the Cray compiler drivers build static executables.
- Some applications are built expecting dynamic libraries
  - These should be switched off with the compiler and moved to static linking.

- Dynamic libraries are possible at this time if:
  - Dynamic libraries are made available to the compute nodes (placed in Lustre)
  - The runtime link path is set to this new location.
- OpenFOAM works in this way.
- This will be better packaged in the future.

System Calls

- Some codes still use

```
Call system('rm old_checkpoint')
```

- This requires shell features and commands.
- These commands are not always available
  - Can be too big
  - Can depend on shared libraries that are not normally available
- There is a command called busybox which encapsulates many useful shell functions
- In many cases calling a c code equivalent is much more efficient

Busybox

- Busybox is a space efficient implementation of the most common shell commands.
- Can be launched directly on compute nodes (using ALPS) to make various enquiries

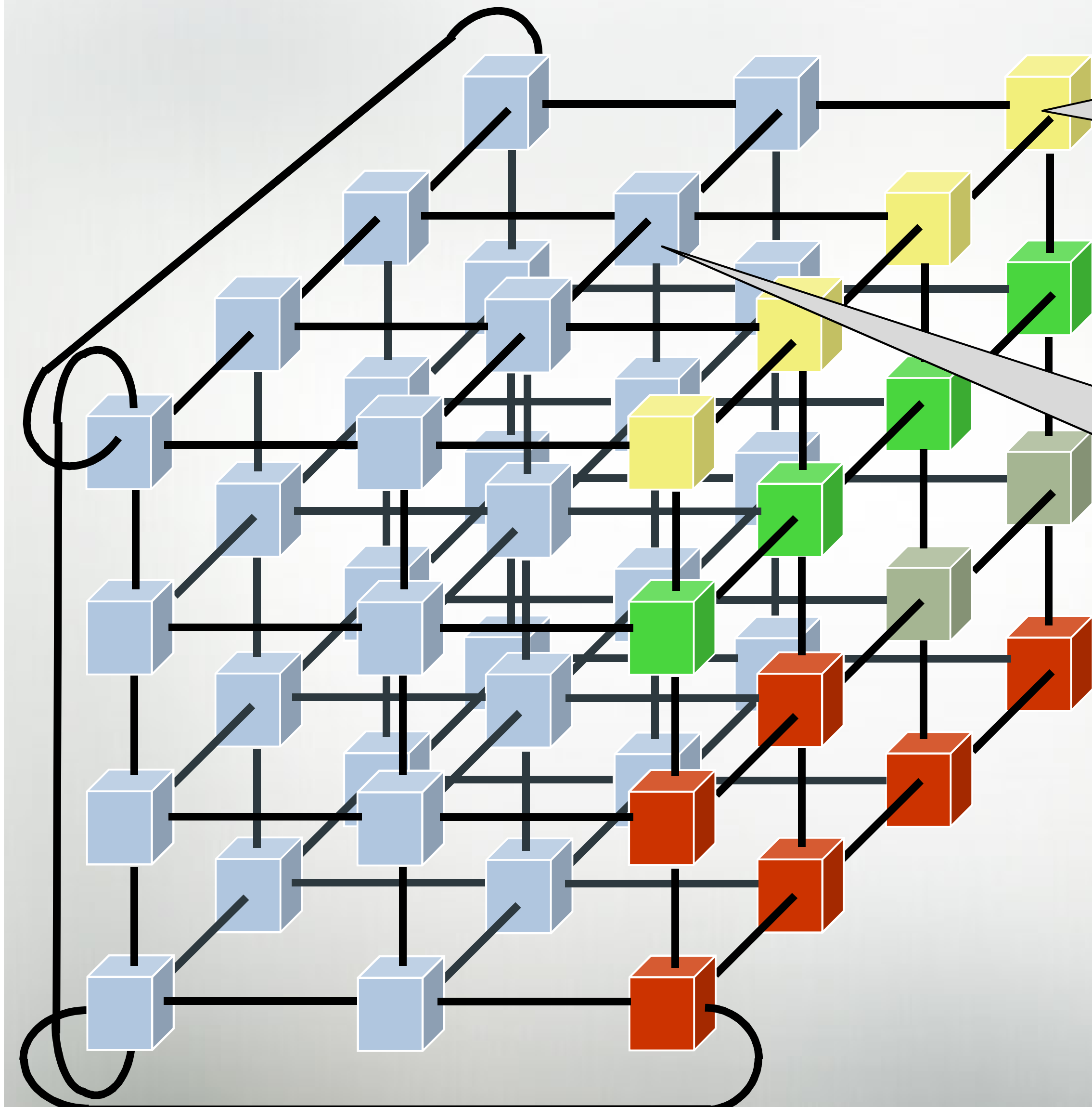
- To run “ls” on the compute nodes:
- aprun –n 1 /usr/bin/busybox ls

- See “man busybox” for the full list of available commands

The Storage Environment

- We have already discussed some aspects of the storage environment.
  - We will also revisit Lustre on Thursday and talk about performance.

- In this section we will cover
  - Scratch file systems
  - Home file system
  - Parallel file system (Lustre)



- Can view all available file systems
- /var, /usr /etc is private to each node, but mounted from elsewhere
- /tmp is local memory
- /home is available

- Has a restricted view of most file systems
- /tmp is local memory and is often ro
- /home is not available
- Lustre is the main file system

Compute node

Login node

Network node

Boot/Syslog/Database nodes

I/O and Metadata nodes

File Systems

- Cray XT systems have separated service work from compute intensive batch work.

- You login in to anyone of a number of login or service nodes.
  - `hostname` can be different each time
  - `xthostname` usually gives the "machine name"
  - Load balancing is done to choose which node you login to

- You are still sharing a fixed environment with a number of others
  - Which may still run out of resources

- Successive login sessions may be on different nodes
  - I/O needs to go to disk, etc.

File Systems

- You start in your home directory, this is where most things live
  - ssh keys
  - Files
  - Source code for compiling
  - Etc
- The home directories are mounted via NFS to all the service nodes

- The /work file system is the main lustre file system,
  - This file system is available to the compute nodes
  - Optimized for big, well formed I/O.
  - Small file interactions have higher costs.

- /opt is where all the Cray software lives
  - In fact you should never need to know this location as all software is controlled by modules so it is easier to upgrade these components

File Systems

- /var is usually for spooled or log files
  - By default PBS jobs spool their output here until the job is completed (/var/spool/PBS/spool)

- /proc can give you information on
  - the processor
  - the processes running
  - the memory system

- Some of these file systems are not visible on backend nodes and maybe be memory resident so use sparingly!
  - You can use commands to investigate what is actually on the compute nodes:
  - aprun -n 1 /usr/bin/busybox -l /

Lustre

- Lustre
  - Designed for parallel I/O
  - Is most probably the largest file system
  - Could be the only writable file system from a compute node
  - Designed for large data transactions
  - The performance is easily tuneable dependent on requirements of the application
    - Large data files could be spread across many I/O nodes to increase performance
    - Large numbers of files could be stored one per storage node to increase concurrency

Increasing Lustre Performance

- This is covered in greater depth later but in order to get a flavour of Lustre performance …
- We apply attributes to files or directories
  - For directories the attributes apply to all files contained in it
- We can describe
  - A stripe size
  - A stripe count (how many lustre nodes to spread a file across)

- As a quick test:
  - we can create two directories
  - Apply “lfs setstripe 0 -1 16” to one of them
  - Create two identical files and put one in each directory
  - In each directory simply copy the file to a new file name and measure the performance

Increasing Lustre Performance

- The previous example shows good speed up if the file is large
- For small files this will not be the case
- For many files this may not always be the case

- Later this week we will discover how to best use the parallel Lustre file system

The Cray Linux Environment

Questions / Comments
Thank You!