Computing Environment

David Turner
NERSC User Services Group

NUG New User Training
February 23, 2015
Node Types

• **Login nodes**
  – Shared
  – Job preparation and submission

• **MOM nodes**
  – Shared
  – Where batch script executes
    • Parallel job launcher
      – Hopper/Edison: aprun
      – Carver: mpirun

• **Compute nodes**
  – Not shared
    • Except serial queue
Login Nodes

- **Edison**
  - Twelve nodes
    - 16 cores, 2.0 GHz Intel Sandy Bridge, 512 GB
- **Hopper**
  - Eight nodes
    - 16 cores, 2.4 GHz AMD Opteron, 128GB
  - Four nodes
    - 32 cores, 2.0 GHz AMD Opteron, 128GB
- **Carver**
  - Four nodes
    - 8 cores, 2.66 GHz Intel Nehalem, 48 GB
- **Genepool**
  - Four nodes
    - 8 cores, 2.3 GHz Intel Sandy Bridge, 32 GB
- **PDSF**
  - Three nodes
    - 16 cores, 2.6 GHz Intel Sandy Bridge, 125 GB
Login Node Access

• Connect (via ssh) to *load balancer*
  
  % ssh edison.nersc.gov
  % ssh hopper.nersc.gov
  % ssh carver.nersc.gov
  % ssh genepool.nersc.gov
  % ssh pdsf.nersc.gov

• Load balancer selects login node based on:
  
  – Number of connections
  
  – Memory of previous connections from same IP
    
    • If you login everyday, you’ll probably end up on the same login node every time.
Login Node Usage

• Login nodes are shared by many users, all the time
• Edit files, compile programs, submit batch jobs
• Some post-processing/data analysis
  – IDL
  – MATLAB
  – NCL
  – python
• Some file transfers
  – Use data transfer nodes for large/long-running transfers
• Please use discretion
  – All users get frustrated by sluggish interactive response
Login Node Monitoring

• Determine number of available cores
  \% grep processor /proc/cpuinfo | wc -l

• Determine amount of physical memory
  \% grep MemTotal /proc/meminfo

• Use “top” command to view process activity
Login Node Guidelines

• Use *no more* than 50% of available cores
• Use *no more* than 25% of available memory
• Limit use of parallel “make”
  \%
  make -j 4 all
• NERSC will kill user processes if response becomes unacceptable
• Terminate idle sessions of licensed software
  – IDL
  – MATLAB
  – Mathematica
Shell Initialization Files

• **Standard dot files**
  – .bashrc, .profile, .cshrc, .login, etc.
  – Symbolic links to read-only files
    • Allows NERSC to provide common environment

• **Personal dot files**
  – Aliases, environment variables, modules, etc.
  – Use .ext suffix (".ext files")
  – .bashrc.ext, .profile.ext, .cshrc.ext, .login.ext, etc.

• **Use “fixdots” to start over**
  – Creates $HOME/KeepDots.<timestamp>
  – Restores all dot files to current default state
  – If PATH corrupted: /usr/common/usg/bin/fixdots

• **Use NIM to change default login shell**
NERSC Supported Software

• **NERSC provides a wide range of software**
  – Scientific Applications
    • VASP, Amber, NAMD, ABySS, ...
  – Compilers
    • PGI, Intel, GCC, Cray
  – Scripting Languages
    • perl, python, R
      – and packages for each!
  – Software Libraries
    • blas/lapack (MKL), boost, hdf5, netcdf, ...
  – Utilities
    • gnuplot, git, mercurial, cmake, ...
  – Debuggers and Profilers
    • CrayPat, DDT, TotalView, gdb, MAP, darshan
  – Visualization
    • Visit, ParaView, VMD, ...

• **See complete list**
  [http://www.nersc.gov/users/software/](http://www.nersc.gov/users/software/)
Software is Managed by Modules

- NERSC provides many versions of many software packages
  - To support diverse workload on systems
- Maintaining all these separate software installations on heterogeneous systems is a major challenge!
  - Software can’t just be installed in the base operating system
    - How many copies of /usr/bin/vasp could be supported?
  - Each software package installed in its own directory
    /usr/common/usg/vasp/5.3.5

Modules is the user interface to software at NERSC
How to Access NERSC Software

• Identify the software you need
  – Use the NERSC website
    http://www.nersc.gov/users/software/
  – Use "module avail"
    • Lots of output
      – All module output goes to stderr, not stdout
    • Each system has different modules!

• Load the module
  % which idl
  idl: Command not found.
  % module load idl
  % which idl
  /usr/common/usg/idl/idl82/bin/idl
Loading Modules

• Different module for each version of software
  – Syntax: `<name>/<version>`
  – Default provided if no `<version>` supplied
  % module avail idl
  idl/7.1  idl/8.0  idl/8.2 (default)
  % module load idl/7.1

• Load modules in every batch script
  – Ensure correct run-time environment
  – Self-documenting for troubleshooting and reproducibility
Other Useful Module Commands

**module unload <modulename>**
- Remove the module from your environment

**module swap <module1> <module2>**
- Unload one module and replace it with another
  
  module swap pgi gcc

**module list**
- See what modules you have loaded right now

**module show <modulename>**
- See what the module actually does

**module help <modulename>**
- Get more information about the software
Default Modules

• When you login, many *default* modules are loaded automatically
  – Usually foundational modules which are required to get proper function from the system
    • Build environment, required libraries and applications, batch environment
  – Use caution in unloading these

• Swapping to functionally equivalent module may be OK
  carver% module swap pgi gcc
  hopper% module swap PrgEnv-pgi PrgEnv-gnu

• Each NERSC system has different default modules
Types of Modules

• Applications
  – VASP, amber, blast, ...
  – Usually only set PATH, LD_LIBRARY_PATH

• Libraries
  – Set LD_LIBRARY_PATH
    • but probably not on Crays
  – Set “helper” environment variable for building software
    • Header/include file search paths
    • Library search paths
    • Library names

% module load hdf5
% mpicc mycode.f $HDF5
Cray Programming Environment

• Compiler specific
  PrgEnv-pgi, PrgEnv-intel, PrgEnv-cray, PrgEnv-gnu
  – Intel is default on Edison, PGI is default on Hopper

• Meta-modules
  – Organize a set of modules
    • Compiler (intel, pgi, cray, gnu)
    • Libraries (including MPI) tuned for compiler

• Swapping Programming Environments
  module swap PrgEnv-pgi PrgEnv-intel
  – swaps compiler
  – no need to swap libraries!
Carver “Programming Environment”

- Not as sophisticated as Cray PrgEnv
- Separate compiler and OpenMPI modules

<table>
<thead>
<tr>
<th>Compiler module</th>
<th>OpenMPI module</th>
</tr>
</thead>
<tbody>
<tr>
<td>pgi</td>
<td>openmpi</td>
</tr>
<tr>
<td>intel</td>
<td>openmpi-intel</td>
</tr>
<tr>
<td>gcc</td>
<td>openmpi-gcc</td>
</tr>
</tbody>
</table>

- Must keep libraries consistent with compiler!
Compiler Wrappers

• **Edison/Hopper**
  - Defined by PrgEnv modules
  - *ftn, cc, CC*
  - Provides include and library search paths for MPI and some common math libraries (e.g., Cray’s libsci)
  - Provides consistent level of optimization across compilers

• **Carver**
  - Defined by openmpi modules
  - *mpif90, mpicc, mpiCC*
  - Provides include and library search paths for OpenMPI

• **Seldom need native compilers!**
• Provides different OS environments
  – Often different third-party software
    • Some software packages have specific OS requirements
      – Possibly due to validation requirements

• Used on Carver, Genepool, and PDSF

• Transparent
  – Default configuration for most users
  – Alternate configurations for some users

• Details on website
  http://www.nersc.gov/users/computational-systems/carver/user-environment/
  http://www.nersc.gov/users/computational-systems/pdsf/software-and-tools/chos/
Resources

http://www.nersc.gov/users/software/nersc-user-environment/

http://www.nersc.gov/users/software/nersc-user-environment/modules/

http://www.nersc.gov/users/computational-systems/edison/programming

http://www.nersc.gov/users/computational-systems/hopper/programming/

http://www.nersc.gov/users/computational-systems/carver/programming/
Thank you.