HPC

The HPC cluster and how to use it

Eric Spangler

Accessing the HPC's cluster

- Windows
  - Putty (As your terminal)
  - WinScp (for transferring information between your computer and the cluster)
  - Alternatives (Cygwin, MobaXterm, mRemoteNG, Xshell, Bitvise, VNC, etc…)
- Mac OS X and Linux
  - Built in commands
  - ssh (As your terminal)
  - scp (for transferring data between your computer and the cluster)
  - Usually have to enable ssh port to utilize the commands

- Hostname is '**hpclogin.memphis.edu**' or '**hpc18login2.memphis.edu**'
- **X2go** can be used to connect to the cluster with a desktop environment (XFCE)
- On Mac and Linux, open terminal and type:
  - '**ssh [username]@hpclogin.memphis.edu**'
- Can only access it on campus or through the campus VPN off campus
  - http://www.memphis.edu/its/network/vpn.php

HPC Terminal

- BASH (Borne Again Shell)
  - Case sensitive
    - 'ls', 'Ls', 'lS', and 'LS' are all different commands
  - Special characters can change the interpretation of a command
    - I suggest you avoid using spaces, ~, ?, <, >, &, \, %,and * characters in your file names
    - If you have to, use C style escape sequences, such as '\&' for &.
  - Special sequences:
    - 'command > [filename]' - creates a new file with results of command
    - 'command >> [filename]' - appends a file with results of command
    - 'command ~' - tilde (~) is a shortcut to home directory
  - **Many more, just look up BASH documentation**

- Basic file system commands:
  - 'ls' - list directory contents
  - 'cd [directoryName]' - change directory to directory name
  - 'pwd' - print working directory
  - 'mkdir [directoryName]' - make a directory from within the current directory
  - 'cat [filename]' - display what is in file
  - 'head [filename]' - display first 10 lines of file
  - 'tail [filename]' - display last 10 lines of file
  - 'cp [filename1] [filename2]' - copy a file to another file
  - 'mv [filename1] [filename2]' - move a file to another file
  - Basic File editors:
    - 'vi [filename]' - the VIM file editor
    - 'emacs [filename]' - the EMACS file editor
    - 'nano [filename]' - the Nano file editor

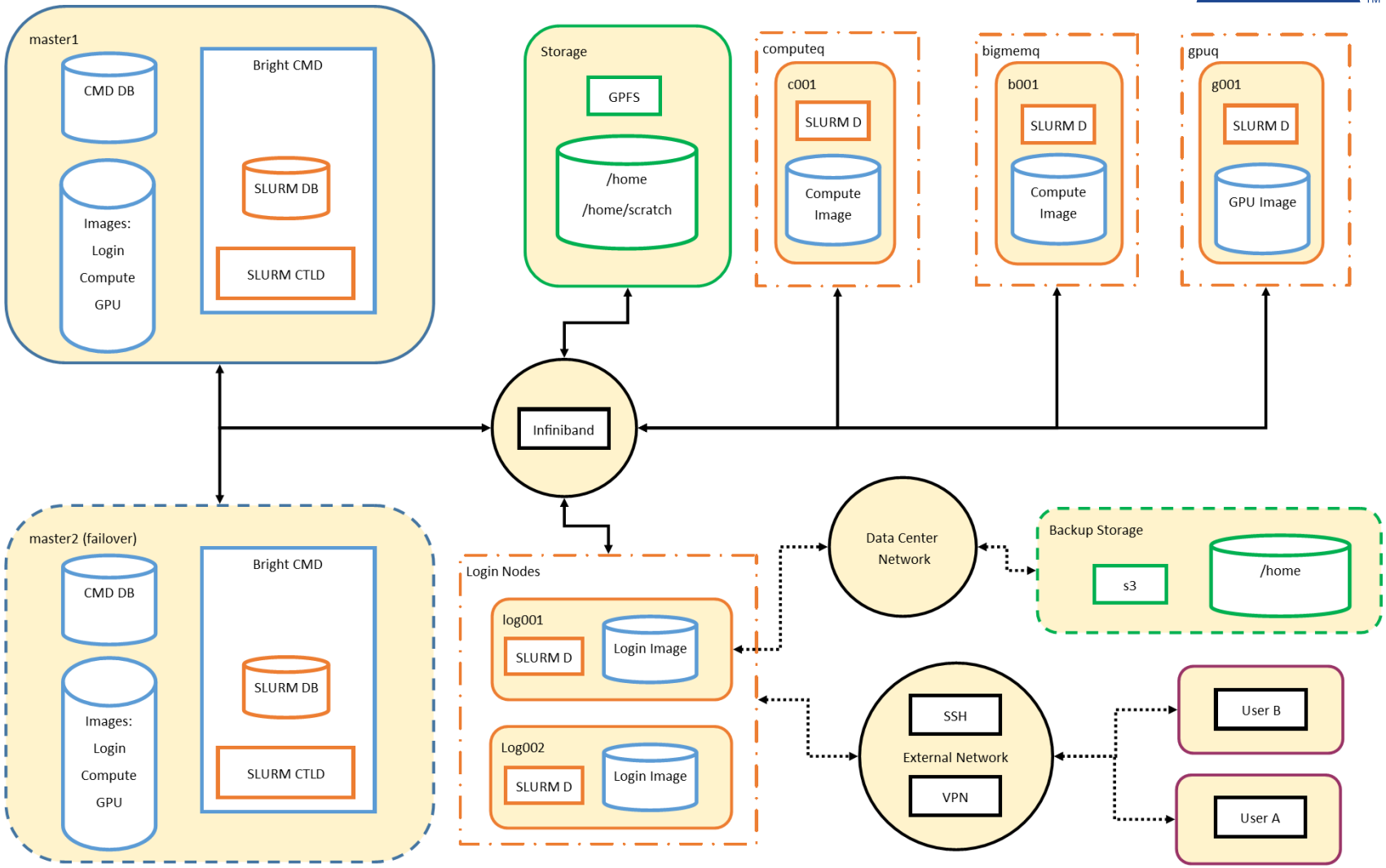
HPC Cluster

- 92 Nodes
  - 2 ‘master’ nodes for SLURM scheduler
  - 2 ‘login’ nodes for submitting jobs
  - 72 ‘computeq’ nodes
    - 2x20 Core Intel Xeon® Gold 6148 CPUs
    - 192 GB DDR4 RAM (2666 MHz, 6 Channels per CPU)
    - EDR Infiniband (100 Gb/s, 610 ns latency)
  - 4 ‘shortq’ nodes
    - Same as computeq but with 3 day timelimit
  - 2 ‘wholeq’ nodes
    - Same as computeq but with 5 day timelimit and whole node allocation
  - 6 ‘gpuq’ nodes
    - Same as ‘computeq’
    - And 2 NVIDIA Tesla® V100 GPUs with 5120 ‘cores’ and 16 GB RAM per GPU
  - 4 ‘bigmemq’ nodes
    - Same CPUs and Infiniband as ‘computeq’
    - 2 nodes have 768 GB of RAM each
    - 2 nodes have 1536 GB of RAM each

HPC Cluster

- Storage
  - DDN GS7K GRIDScalar® (GPFS)
    - Connected through Infiniband @ 8.5 GB per second
    - Contains 120x4 TB Drives (480 TB Raw, 348 TB Available)
    - Mounted as /home (this is backed up through S3)
    - Mounted as /home/scratch (this is not backed up)
  - Archive (S3 StorageGRID®)
    - This is connected through the data center network (Ethernet)
    - Contains >1 PB
    - Users can also archive their own data here (contact hpcadmins@memphis.edu)
    - Uses AWS commands, very slow @ ~100-500 MB per second

HPC Cluster


master1
CMD DB
Images:
Login
Compute
GPU
Bright CMD
SLURM DB
SLURM CTLD
Storage
GPFS
/home
/home/scratch
computeq
c001
SLURM D
Compute
Image
bigmemq
b001
SLURM D
Compute
Image
gpuq
g001
SLURM D
GPU Image
Infiniband
master2 (failover)
CMD DB
Images:
Login
Compute
GPU
Bright CMD
SLURM DB
SLURM CTLD
Login Nodes
log001
SLURM D
Login Image
Log002
SLURM D
Login Image
Data Center
Network
Backup Storage
s3
/home
SSH
External Network
VPN
User B
User A

HPC Software

- Usually installed in /cm/shared/public/apps or /public/apps
- Usually has a module associated with it:
  - ‘module avail’ to see available modules
  - ‘module load modulename’ to load module
  - ‘module list’ to see currently loaded modules
- If a particular software isn’t present, please contact hpcadmins@memphis.edu
- Commonly used:
  - GCC/Intel compilers ‘module load gcc/8.2.0’ or ‘module load intel/2022.2’
  - MATLAB ‘module load matlab’
  - Python ‘module load python/2.7.15’ or ‘module load python/3.7.0’
  - CUDA ‘module load cuda11.2/toolkit/11.2.2’ only on ‘gpuq’ nodes

SLURM

- HPC cluster uses SLURM for job scheduling
  - '**sbatch**' command to submit batch jobs
  - '**srun**' command to submit interactive jobs
  - 'salloc' and 'sattach' to allocate a job and attach to any job
  - '**squeue**' to view running and pending jobs
  - '**scancel**' to cancel running and pending batch jobs
  - 'sacct' to view completed and failed jobs
  - 'scontrol' to view or modify submitted jobs
  - 'sacctmgr' to view account resources
  - 'sprio' to view pending job's priority
  - 'sshare' to view current user's fairshare priority

SLURM Partitions

- ‘computeq’ is for general CPU bound jobs
  - This is our largest partition
  - 4.8 GB RAM per CPU core
- ‘gpuq’ is for general GPU bound jobs
  - This uses ‘--gres=gpu:N’ for submissions
  - 4.8 GB RAM per CPU core
- ‘bigmemq’ is for general memory bound jobs
  - 38.4 GB RAM per CPU core
  - 76.8 GB RAM per CPU core
- ‘shortq’ is for general short jobs
  - 3 day maximum ‘--timelimit’ for submissions
- ‘wholeq’ is for whole node jobs
  - Must use ‘--ntasks=40’ or ‘--cpus-per-task=40’ for submissions

SLURM Accounting

- You can submit jobs only under your account
- We create a default account for every new user
  - This account is part of a tree:
    - Cluster →uom →college →department →content area →lab
    - Users are part of the lab branches
    - **Coordinators (usually the PI) can modify any job submitted to their lab account**
  - We have restrictions on this account:
    - Users: **1024 CPU cores, 3 TB RAM**
    - Accounts: Fairshare Algorithm+priority weights
    - These restrictions may change at any time due to Load, reservations, user requests, or problems

SLURM Priority

- Jobs are assigned a priority when they are submitted where:
  - Priority=Fairshare+JobSize+Age+Partition+QOS+Nice+TRES
- Fairshare Priority:
  - Limits users and accounts to a fraction of active cluster jobs
  - Fairshare priority penalty decays over 24 hours
- JobSize Priority:
  - Larger jobs have higher priority (Fairshare can negate this)
  - Shorter jobs have higher priority
- Age
  - Longer pending times increase Age priority
- Partition (currently 0)

SLURM Priority (cont)

- Jobs are assigned a priority when they are submitted where:
  - Priority=Fairshare+JobSize+Age+Partition+QOS+Nice+TRES
- QOS (quality of service)
  - Each job has a normal QOS currently (0)
- Nice
  - Administrative (we increase this if a job has scheduling difficulties)
- TRES (Trackable RESources)
  - Each resource (CPU and Memory) has just 1 weight
  - GPU (for --gres) has a very high weight

SLURM Batch Scripts

- Typically BaSH shell script (can use TK/TCL)
  - '#!/bin/bash' header for bash
  - '#!/usr/bin/tclsh' header for tcl
- 'sbatch' options can be added with special comment:
  - '#SBATCH option'
- Built in SLURM environment variables:
  - '$SLURM_SUBMIT_DIR' for full path of submission directory
  - '$SLURM_JOBID'
  - '$SLURM_ARRAY_TASK_ID'
  - Try 'man sbatch' for more details

SLURM Jobs

- Minimum requirements:
  - Partition (‘**--partition**’, ‘-p’)
  - CPUs, tasks, or nodes
    - ‘**--cpus-per-task**’ or ‘-c’ for multiple threads/cores per a node/task (pthreads/OpenMP)
    - ‘--ntasks’ or ‘-n’ for multiple message passing tasks (MPI)
    - ‘--nodes’ or ‘-N’ for multiple nodes (MPI)
  - Time (‘--time’ or ‘-t’)
  - Memory
    - ‘**--mem-per-cpu**’ for memory per CPU core
    - ‘--mem’ for memory per node

SLURM Jobs

- Useful options:
  - Job name (‘--job-name’ or ‘-J’) for identification in ‘squeue’
  - Job array (‘--array’ or ‘-a’) for many job submissions
  - E-mail job status (‘--mail-user’ and ‘--mail-type’)
  - Tasks per node (‘--ntasks-per-node’) for requesting a specific number of tasks per node (MPI/OpenMP hybrid)
  - Output and error (‘--output’ or ‘-o’ and ‘--error’ or ‘-e’) to redirect script standard output and error (‘stdout’ and ‘stderr’)
  - Generic RESource (‘--gres’) used for **gpus**, licenses, and interconnects

SLURM Job Examples