Introduction to the Cluster

Advanced Computing Center for Research and Education
http://www.accre.vanderbilt.edu
We will be offering an optional tour of the cluster at the end of class today.
What is a Cluster?

In a nutshell:
- A bunch of computers networked together!
- Enables users to “burst” onto idle computers

Key concepts:
- Submit jobs from gateway
- Scheduler runs jobs for you on compute node(s)
- Files are visible everywhere
- Change your password by logging into auth (type “rsh auth” from a gateway) and typing “passwd”

ACCRE CLUSTER

- Jobs are run on compute nodes by the job scheduler
- At any given time, ~2000-4000 jobs are running on compute nodes
- Users often have multiple jobs running at once
- Users do not need to log in to a compute node

Gateways are used for:
- Logging in
- Managing/editing files
- Writing code/scripts
- Submitting jobs
- Running short tests

Job Scheduler Server
- Runs the software (called SLURM) for managing cluster resources and scheduling jobs
- SLURM commands available across the cluster (e.g. from a gateway)

File Servers
Mounts/maps users’ files across all gateways and compute nodes
Our cluster is a little different than most...

* Most clusters are bought all at once.
* Therefore, all of the nodes are identical.
* Our cluster has evolved over time; it started with 120 dual-CPU P4's (and was ranked at #199 on the June 2003 “Top 500” list - [http://www.top500.org](http://www.top500.org)).
* The cluster has grown over time as researcher demand has dictated.
* Older nodes have been “retired.”
* It is now comprised of more than 600 nodes / 6000 processor cores, which isn’t enough to even make the “Top 500” list now (The current #1 system has over 3 million cores).
## Compute Node Types

<table>
<thead>
<tr>
<th>Type</th>
<th>Processor Type</th>
<th># of cores per node</th>
<th>RAM per node</th>
<th># of GPU's / cores per GPU</th>
<th># of nodes in cluster</th>
</tr>
</thead>
<tbody>
<tr>
<td>Dual-quads</td>
<td>Intel Xeon Westmere</td>
<td>8</td>
<td>24 - 128 GB</td>
<td>N/A</td>
<td>299</td>
</tr>
<tr>
<td>Dual-hex</td>
<td>Intel Xeon Westmere</td>
<td>12</td>
<td>48 - 64 GB</td>
<td>N/A</td>
<td>91</td>
</tr>
<tr>
<td>GPU nodes</td>
<td>Intel Xeon Westmere</td>
<td>8</td>
<td>48 GB</td>
<td>4 / 480</td>
<td>40</td>
</tr>
<tr>
<td>Dual-hex</td>
<td>Intel Xeon Sandy Bridge</td>
<td>12</td>
<td>128 - 256 GB</td>
<td>N/A</td>
<td>200</td>
</tr>
<tr>
<td>Dual-hex</td>
<td>Intel Xeon Haswell</td>
<td>12</td>
<td>128 GB</td>
<td>N/A</td>
<td>25</td>
</tr>
</tbody>
</table>
If you have your laptop with you, please feel free to try all of these things out as we go along...

To log in, open a terminal (Windows users should download PuTTY) - `ssh <vunetid>@login.accre.vanderbilt.edu`

You will be randomly assigned to one of our more than 1 dozen gateways via DNS round-robinning.

To change your password, after logging on to one of the gateways, `rsh auth` and once logged in there change your password by executing `passwd`.

Please do not use your e-password as your cluster password (or vice versa).

It is not permissible to share passwords to an account. Doing so may result in the account being suspended.
Cluster Storage

* Home directories (/home), scratch space (/scratch), and optional additional data storage (/data) are shared across all nodes in the cluster via IBM’s General Parallel Filesystem (GPFS).
* http://www.accre.vanderbilt.edu/?page_id=63#gpfs
* /home and /data are backed up nightly to tape; /scratch is NOT.
* If possible, use /tmp on the individual compute nodes for temporary storage while your job is running; ~30+ GB of space per core.
* When your job is done, move results back to /data, /home or /scratch and clean up /tmp.
Default Disk Quotas

* For the /home filesystem...
  * Default disk quotas are 25 GB, with a 35 GB hard limit and a 7 day grace period.
  * Default file quotas are 100,000 files, with a 200,000 file hard limit and a 7 day grace period.

* For the /scratch filesystem...
  * Default disk quotas are 50 GB, with a 200 GB hard limit and a 14 day grace period.
  * Default file quotas are 100,000 files, with a 1,000,000 file hard limit and a 14 day grace period.

* There are no default quotas for the /data filesystem since quota there is only available via purchase.
Disk Quotas

- GPFS will allow you to exceed your quota for the duration of your grace period; after that, your quota becomes your hard limit.
- GPFS will not allow you to exceed your hard limit; attempting to do so will result in an I/O error.
- You can check your quota usage at any time with the `mmlsquota` command (located in /usr/lpp/mmfs/bin).
- Example: `mmlsquota --block-size auto`
- `gpfs21 = /scratch, /gpfs22 = /home`.
- The `mmlsquota` command will not lie to you!
- Keep in mind that if you have copied data to a co-workers' home directory you are still the owner of those files and they therefore still count against your quota.
Storage Policies

Cluster disk usage and quota policies may be found at http://www.accre.vanderbilt.edu/?page_id=91

If you need to store larger quantities of data than your quota, please contact us via a help desk ticket...

Reasonable temporary increases will typically be granted.

Permanent quota increases are available for purchase on /data and /scratch.
Copying files to / from the cluster

* Linux and Mac clients can use `scp`
* Windows users can download WinSCP
* To copy a file from your client to the cluster: `scp my_file vunetid@vmplogin.accre.vanderbilt.edu:~/somedir`
* To recursively copy files from the cluster back to your client: `scp -r vunetid@vmplogin.accre.vanderbilt.edu:~/mydata ./mydir`
* `sftp` is also supported if you prefer it.
Shell Initialization Files

* `.bashrc` for the Bash shell
* `.cshrc` for the C shell
Two Very Important Environment Variables

* `PATH` is a colon delimited list of directories where the shell looks for commands.
* `LD_LIBRARY_PATH` is a colon delimited list of directories where the shell looks for libraries.
setpkgs and pkginfo

* **setpkgs** was developed by ACCRE to make updating your PATH and LD_LIBRARY_PATH variables easy ... you just add the appropriate "package" to your environment and setpkgs takes care of the rest

* **pkginfo** displays the packages which are available; there are lots of packages, so you may want to use **grep** to filter results

* Since **setpkgs** and **pkginfo** were developed by ACCRE, they will not be available on any other clusters you may use.

* Example on next slide...
Using pkginfo

* This example shows using **pkginfo** to see which:
  * compilers are installed
  * versions of the GNU Scientific Library are installed
  * versions of ATLAS are installed

```
$ pkginfo -l | grep compiler
pkginfo returns various compiler settings for the specified package or tool.
  versions to use for returning the various compiler settings.
    gcc_compiler GCC Compiler (4.6.1)
    intel_compiler Intel 12 Compiler, including fortran
    llvm_compiler LLVM/Clang (3.2)
$ pkginfo -l | grep gsl
  gsl_gcc GNU Scientific Library (GCC) [gsl]
  gsl_intel GNU Scientific Library (Intel) [gsl]
$ pkginfo -l | grep atlas
  atlas ATLAS Self-tuning BLAS (GCC) [blas]
  atlas_nehalem_gcc ATLAS Self-tuning BLAS for Nehalem (GCC) [blas]
  atlas_nehalem_intel ATLAS Self-tuning BLAS (GCC) [blas]
  atlas_x86-64_gcc ATLAS Self-tuning BLAS (GCC) [blas]
  d tatto_atlas_x86-64_gcc Dakota
  lapack_atlas LAPACK with ATLAS (GCC) [lapack]
  lapack_atlas_nehalem_gcc LAPACK with ATLAS for Nehalem (GCC) [lapack]
  lapack_atlas_nehalem_intel LAPACK with ATLAS for Nehalem (Intel12) [lapack]
  lapack_atlas_x86-64_gcc LAPACK with ATLAS (GCC) [lapack]
```

*setpkgs -a package_name* adds a package to your environment.
Erasing a package

* `setpkgs -e package_name` erases a package from your environment.
* Notice that the GCC compiler is still part of the environment.

```
$ echo $PATH
/usr/local/gcc/latest/bin:/usr/local/matlab/2012b/bin:/usr/scheduler/moab/sbin:/usr/scheduler/moab/bin:/usr/scheduler/torque/sbin:/usr/scheduler/torque/bin:/opt/ruby/bin:/usr/lib64/qt-3.3/bin:/usr/lpp/mmfs/bin:/usr/local/ddt/bin:/usr/local/bin:/usr/X11R6/bin
:/usr/local/common/bin:/usr/bin:/bin:/usr/local/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/sbin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/local/bin:

$ echo $LD_LIBRARY_PATH
/usr/local/gcc/latest/lib:/usr/local/gcc/latest/lib64:/usr/local/matlab/2012b/extern/lib/glnxa64:/usr/local/matlab/2012b/bin/glnxa64:

$ setpkgs -e matlab
$ echo $PATH
/usr/local/gcc/latest/bin:/usr/scheduler/moab/sbin:/usr/scheduler/moab/bin:/usr/scheduler/torque/sbin:/usr/scheduler/torque/bin:/opt/ruby/bin:/usr/lib64/qt-3.3/bin:/usr/lpp/mmfs/bin:/usr/local/ddt/bin:/usr/local/bin:/usr/X11R6/bin:/usr/local/common/bin:/usr/bin:/usr/bin:/bin:/usr/local/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/sbin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/bin:/usr/local/bin:

$ echo $LD_LIBRARY_PATH
/usr/local/gcc/latest/lib:/usr/local/gcc/latest/lib64
```
Replacing Packages

*setpkg -r package_name* erases all previously added packages from your environment and replaces them with the specified package.
Adding packages to your shell initialization file

*Packages added via `setpkgs` on the command line are just like aliases and variables defined on the command line - they are in effect only until you log out. Therefore, just like with aliases and variables, to make them permanent you add them to your `.bashrc` or `.cshrc`.

If you submit a job to the cluster without adding the package(s) it needs to your `.bashrc` or `.cshrc`, the job will fail!
Scheduler Basics

- The cluster currently has more than 600 compute nodes (more than 6000 processor cores) with varying amounts of RAM and is used by over 700 researchers.
- Imagine if you had to try to find a free compute node matching the resources your job needs!
- Instead, you simply write a script which describes what resources your job needs and submit that to the scheduler.
- It is the job of the scheduler to keep track of which resources are free and which are in use.
- The scheduler then starts your job on a node (or nodes) that match the resources you requested or queues your job for later execution if there are currently not enough free nodes matching your request.
How the scheduler works

* You submit a job to the scheduler.
* Each compute node is running a daemon, which keeps track of the utilization of the resources on that node.
* The job scheduler gets resource utilization information from the daemon and schedules jobs based on:
  * Fairshare contribution - from your groups’ buy-in to the cluster.
  * Job priority - calculated based primarily on fairshare, with queue time influencing priority as well.
Simple SLURM Script

First, add the appropriate package to .bashrc or to the job script.

Send mail when job begins, aborts, or ends to "Nikki.Husky@vanderbilt.edu"

We need 1 processor core (1 task) in 1 node.

The job will run no more than 2 hours.

The job needs no more than 500 MB of RAM.

Output and error messages go to python_job.out.

#!/bin/bash
#SBATCH --mail-user=Nikki.Husky@vanderbilt.edu
#SBATCH --mail-type=ALL
#SBATCH --nodes=1
#SBATCH --ntasks=1
#SBATCH --time=2:00:00
#SBATCH --mem=500M
#SBATCH -o python_job.out
setpkgs -a python2.7.8
python < vectorization.py
More Complex PBS Script

* This example asks for 8 processor cores each in 3 nodes.

* Wall time is 7 days.

* In this example, comments are colored in red.

* Job requires 3840 MB of memory per-node.

```bash
#!/bin/bash
#SBATCH --mail-user=Nikki.Husky@vanderbilt.edu
#SBATCH --mail-type=ALL
#SBATCH --nodes=3
#SBATCH --tasks-per-node=8
#SBATCH --time=7-00:00:00
#SBATCH --mem=3840M
#SBATCH -o mpi_job.log
#  MPI jobs must point to SLURM's process management library,
#  which is what this next line is for:
export I_MPI_PMI_LIBRARY=/usr/scheduler/slurm/lib/libpmi.so
date  # prints current time
echo $SLURM_JOB_NODELIST  # prints list of nodes
time srun -n 24 ./test  # run MPI job across 24 CPU cores
date  # print time again
```
**Requesting Resources**

* You should slightly overestimate the resources needed because the scheduler will kill your job if it exceeds any resource request.

* The maximum wall time you can request is 14 days (we’re actually quite generous; most sites limit jobs to 7 days or less).

* The maximum memory in any node is 256 GB; however, you cannot use all of the memory as some of it is reserved for the operating system and utilities (see next slide for details).

* If you don’t have any idea what to specify for a resource, you can ask another member of your group who might have run a similar job to get an idea of the resources your job needs.

* If you still don’t have any idea what to specify for a resource you can guess way high. Then, once the job completes, you can use `rtracejob` to see what it actually used (and adjust what you ask for on subsequent job submissions).
<table>
<thead>
<tr>
<th>Physical (GB)</th>
<th>Available (GB)</th>
</tr>
</thead>
<tbody>
<tr>
<td>24</td>
<td>20.53</td>
</tr>
<tr>
<td>32</td>
<td>28.49</td>
</tr>
<tr>
<td>48</td>
<td>43.15</td>
</tr>
<tr>
<td>64</td>
<td>61</td>
</tr>
<tr>
<td>96</td>
<td>90.39</td>
</tr>
<tr>
<td>128</td>
<td>120</td>
</tr>
<tr>
<td>256</td>
<td>252</td>
</tr>
</tbody>
</table>
But don’t way overestimate...

* Let’s just say another user has submitted a job asking for 100 processor cores for 14 days and right now there are only 50 processor cores free.

* Their job goes in the queue. Because the scheduler knows how much wall time each currently running job has asked for, it knows when those additional 50 processor cores will be free. Let’s assume that’s 3 days from now.

* You submit 10 jobs, each asking for 1 processor core for 48 hours. The scheduler will go ahead and start your jobs because it has 10 free processor cores and knows your jobs will be done (or killed) before the 100 processor core job can start. This is called “backfilling.”

* However, if you had way overestimated and asked for 7 days for each job, your job would have had to wait in the queue.
Using the Scheduler

* To submit your job - `sbatch myscript.slurm`
* To view the queue - `squeue` (or `squeue -u <vunetid>`).
  * `q3` and `qSummary` will break down jobs by user/group.
  * `SlurmActive.pl` will show overall cluster utilization.
  * Use `showLimits` to see groups' core, memory, and CPU time limits.
* To check on a job while it is queued / running - `scontrol show job job_id` (where `job_id` is the job number returned by `squeue`).
* To check on a job after it has finished - `rtracejob job_id`
  * Allows you to compare allocated versus used resources
  * Extremely useful for troubleshooting! Remember this command!
* Review our SLURM documentation to see more examples of useful commands: [http://www.accre.vanderbilt.edu/?page_id=2154](http://www.accre.vanderbilt.edu/?page_id=2154)
Self-diagnosing problems

* Use `rtracejob` and notice the exit status; if it’s non-zero, the job did not complete successfully.
* Compare requested resources with actual resource utilization; remember that the scheduler will kill jobs which exceed any resource request.
* If you have saved both output and errors (you did, didn’t you?), then check your logs.
* A segfault (segmentation fault) is always an application error.
If you still can’t figure out why your job died...

* Open up a help desk ticket.
* The most important piece of information we need to begin looking into the problem is the job number of the job(s).
* Sometimes jobs do get killed thru no fault of your own...
* Nodes can crash (ex: hard drive failure in the node).
* Another users’ job running on the same node may have run the node out of memory so quickly that the scheduler didn’t even have time to kill the offending job (we identify the offending user and work with them to resolve the issue with their jobs).
As an aside...

* The fact that your job can potentially be killed thru no fault of your own is an unavoidable fact of life in HPC.
* Imagine you’re 12 days into a 14 day, 100 processor core job when a hard drive fails in one compute node.
* Your job just died ... do you really want to have to start over from the beginning?
* If you checkpoint your jobs, you won’t have to.
* Checkpointing involves periodically (the interval between checkpoints is up to you) writing out intermediate results to disk (/scratch is a good place for checkpoint files).
* If you have to restart your job, it can then read the checkpoint file and resume execution from that point.
Making the most of the cluster

* Know your code; available resources versus required resources.
* Know cluster policies on runtime and resource limitations.
* Plan ahead for long jobs or large numbers of jobs (i.e. if you’ve got a deadline at the end of the month, don’t wait until 7 days before the end of the month to submit 1,000 6 day jobs).
* Ask experienced group members for help.
* Ask us if you need help.
Software Pages

* ACCRE staff have developed how-to webpages for using/accessing widely used software packages
  * http://www.accre.vanderbilt.edu/?page_id=2756
  * Matlab, Python, R, Ruby
  * Github Repositories (www.github.com/accre) available with examples for each language; users can contribute their own examples!
X-Windows Remote Display

* Displaying graphics over the network can be very slow and should therefore only be done when necessary (ex: using the DDT debugger to debug an application).

* Linux and Mac clients - `ssh -X vunetid@login.accre.vanderbilt.edu`

* Mac users may be prompted to install X11 if not done previously.

* Windows clients require X-Window emulation software (ex: Xming).
You are permitted to use ssh key pairs to log on to the cluster.
Most ssh clients support key pairs.
You should use the cluster to generate your public / secret key pair.
**You must password protect your secret key!!!**
If you do not password protect your secret key then anyone who hacks your client machine (it has happened!) will have access to all of your files stored on the cluster.

Running DDT
Getting help from ACCRE

* ACCRE website FAQ and Getting Started pages:  
  http://www.accre.vanderbilt.edu/support

* ACCRE Help Desk:  http://www.accre.vanderbilt.edu/support/contact/submit_RT.php

  * Tickets default to non-rush queue, which is for tickets about an issue which only impacts you and / or which can wait until the next business day.
  
  * Rush tickets are for issues which would impact the cluster as a whole.

  * Rush queue pages ACCRE personnel (we normally only have staff on site during normal business hours), so please think before opening up a rush ticket at 3 AM!

  * If you open up a rush ticket - day or night - please plan on being available to work with ACCRE personnel to resolve the issue.

  * Once the issue (rush or non-rush) is resolved we would appreciate it if you would let us know.
Getting help from ACCRE

* accre-forum mailing list.
* ACCRE office hours: open a help desk ticket and mention that you would like to work personally with one of us in the office. The person most able to help you with your specific issue will contact you to schedule a date and time for your visit.