Objectives

● Overview: Mills Community Cluster
● Part I: Get your feet wet
● Part II: Jump in
Overview: Mills HPC Basics

Mills Community Cluster
What is the Mills cluster?
● First of its kind at UD - a community cluster
● Technical and financial partnership between UD-IT and UD faculty and researchers

Who can use it?
● UD faculty and researchers who purchased compute nodes (stakeholders)
● Researchers approved by a stakeholder
Mills Cluster

mills.hpc.udel.edu
Mills Cluster

http://www.it.udel.edu/research-computing
Part I: Mills HPC Basics

Getting your feet wet
Getting your feet wet

- Accounts
- Connecting with SSH
- Bash Shell and Working Directory
- File Storage
- Groups and Workgroup(s)
- VALET
- Compiling and Running Jobs
- Help
Accounts
Mills Accounts

Username and Password

- UD = UDelNet ID and password; cannot be changed and is not updated when you change your password on the Network page.

- non-UD = hpcguest<uid> and password is generated by IT staff and cannot be changed. Securely sent via UD Dropbox.
Connecting with SSH
Overview

Client using ssh → Public Network → Head Node → Private Network → Compute Node

Compute Node  Compute Node  Compute Node
SSH Client

- SSH is typically used to connect to the cluster's head (login) node.

- Standard Linux and Mac distributions provide an ssh client.

- Windows distributions require installation of an ssh client such as PuTTY.
SSH Public/Private Keys

- Eliminates entering your password for each remote connection - only need to remember a passphrase of your choice
- More convenient and efficient especially with other applications such as scp and sftp
SSH Help

- Follow documentation for Mac and Linux, or Windows configuration to get connected using X11 and SSH with public/private keys.

http://www.udel.edu/it/research/training/config_laptop/
Connecting to Mills

ssh -Y username@mills.hpc.udel.edu

Using username "traine".

............................................................

Mills cluster (mills.hpc.udel.edu)

This computer system is maintained by University of Delaware IT. Links to documentation and other online resources can be found at:

   http://mills.hpc.udel.edu/

For support, please contact consult@udel.edu

............................................................
Bash Shell and Working Directory
Bash Shell

Bash prompt

- username = referred to as $USER
- hostname = head (login) node
- ~ = current working directory
- $ = end of prompt

[traine@mills ~]$
Working Directory

At login, you start in your home directory (~)

- `/home/<uid>`
- Referred to as `$HOME`

```
[traine@mills ~]$ pwd
/home/1201
[traine@mills ~]$ echo $HOME
/home/1201
```
File Storage
File Storage on Mills

- Home directory 2GB (/home/<uid>)

Other file storage available:

- Lustre ~180 TB (/lustre)
- Archive ~60 TB (/archive)
- Node-local scratch (/scratch)
Groups and Workgroups
Mills Groups

Group names determine access level to specific system resources

groups

- **Class category** = everyone, ud-users, hpc-guests, facstaff and students
- **Investing-entity category** = represents a unique group name for stakeholders and its sponsored users (e.g., it_css)

[traine@mills ~]$ groups
everyone students ud-users it_css
Groups in the **investing-entity category** are controlled by using the command

```
workgroup
```

Workgroup is displayed in the bash prompt after the `workgroup -g <investing_entity>` command.

```
[traine@mills ~]$ workgroup -q workgroups
1002  it_css
[traine@mills ~]$ workgroup -g it_css
[(it_css:traine)@mills ~]$
VALET
VALET

- UD-developed software to help configure your environment for all IT-installed software packages.

- Changes environment such as `PATH`, `LD_LIBRARY_PATH` and `MANPATH`

- Changes software development environment such as `LDFLAGS` and `CPPFLAGS`

- An alternative to the Modules software used at other HPC sites

`man valet`
VALET Commands

vpkg_list

● a list of all available software packages installed by IT

[(it_css:traine)@mills ~]$ vpkg_list
Available packages:
  acml
  amd-app-sdk
  apache-ant
  blacs
  boost
  ...
  ...

● a web page of all software packages and descriptions derived from VALET
http://mills.hpc.udel.edu/software.php
VALET Commands

vpkg_versions <package_id>

- a list of versions available for <package_id>
- default version marked with *

[(it_css:traine)@mills ~]$ vpkg_versions intel
Available versions in package (* = default version):
intel Intel Compiler Suite
   2011-32bit Version 2011 for 32-bit x86
* 2011-64bit Version 2011 for x86-64
[(it_css:traine)@mills ~]$
VALET Commands

```
vpkg_require <package_id>
vpkg_devrequire <package_id>
```

- set your environment or development environment for `<package_id>`

```
[(it_css:traine)@mills ~]$ vpkg_require intel
Adding package `intel/2011-64bit` to your environment
[(it_css:traine)@mills ~]$ 
```

```
[(it_css:traine)@mills ~]$ vpkg_devrequire intel
Adding package `intel/2011-64bit` to your environment
[(it_css:traine)@mills ~]$ 
```
VALET Commands

vpkg_rollback all

• undo all changes to your environment

[it_css:traine]@mills ~]$ vpkg_rollback all
[[it_css:traine]@mills ~]$
Compilers

There are four 64-bit compiler suites on Mills:

- PGI CDK (Portland Group Inc.’s Cluster Development Kit)
- Open64
- Intel Composer XE 2011
- GNU

OpenJDK (Open Java Development Kit) is installed, but can only be used on the compute nodes.

We generally recommend that you use the PGI compilers rather than the Intel and GNU compilers. The PGI compilers exploit special features of the AMD Interlagos processors and are better supported by IT staff.
Running Applications

In general, applications should be run on the compute nodes, not on the login (head) node.

- Use VALET to set up your runtime environment; should be similar to your compile-time environment.

- Use Grid Engine's `qlogin` or `qsub` to submit an interactive or batch job to run your applications.
C and Fortran Examples

C and Fortran program examples
- cmatmul and fmatmul

Compile scripts for each compiler
- compile-gcc, compile-intel, compile-open64 and compile-pgi

Batch scripts for each compiler
- serial-gcc.qs, serial-intel.qs, serial-open64.qs and serial-pgi.qs
Copy Examples

```bash
cp -r ~trainf/mhpcI .
cd mhpcI
pwd
```

```
[[it_css:traine]@mills ~]$ cp -r ~trainf/mhpcI .
[[it_css:traine]@mills ~]$ cd mhpcI/
[[it_css:traine]@mills mhpcI]$ pwd
/home/1201/mhpcI
[[it_css:traine]@mills mhpcI]$ ls
cmatmul fmatmul
```
Compiling Code
Compiling Code

- Programs can only be compiled on the login (head) node, except for java
- Use VALET to set up your compile-time environment

Example using C program to create executable `tmatmul`

```bash
[(it_css:traine)@mills mhpcI]$ cd cmatmul
[(it_css:traine)@mills cmatmul]$ pwd
/home/1201/mhpcI/cmatmul
[(it_css:traine)@mills cmatmul]$ more compile-pgi
touch tmatmul.c
vpkg_rollback all
vpkg_devrequire pgi
export CC=pgcc
export CFLAGS='-Minform=warn -fast -tp bulldozer'
make tmatmul
[(it_css:traine)@mills cmatmul]$ source compile-pgi
Adding package `pgi/11.10` to your environment
pgcc -Minform=warn -fast -tp bulldozer tmatmul.c -o tmatmul
[(it_css:traine)@mills cmatmul]$
```
Running Jobs
Running Jobs

- **Interactively using** `qlogin`

  Grid Engine will submit an interactive job to the queuing system.

- **Batch using** `qsub <job_script>`

  Grid Engine will submit batch job `<job_script>` to the queuing system

- **Both must be done from the head (login) node**
Interactive (session) job

qlogin

[(it_css:traine)@mills cmatmul]$ qlogin
Your job 48332 ("QLOGIN") has been submitted
waiting for interactive job to be scheduled ...
Your interactive job 48332 has been successfully scheduled.
Establishing /opt/shared/GridEngine/local/qlogin_ssh session to host n017 ...
[traine@n017 cmatmul]$
Interactive Run

Run `tmatmul` on a compute node and exit

```
[traine@n017 cmatmul]$ vpkg_require pgi
Adding package `pgi/11.10` to your environment
[traine@n017 cmatmul]$ ./tmatmul

B:
1.00000 1.00000 1.00000
1.00000 1.50000 2.25000
1.00000 2.00000 4.00000
1.00000 3.00000 9.00000

C:
1.00000 0.00000
0.00000 1.00000
0.50000 0.50000

B*C with loops:
1.50000 1.50000
2.12500 2.62500
3.00000 4.00000
5.50000 7.50000

[traine@n017 cmatmul]$ exit
exit
Connection to n017 closed.
/opt/shared/GridEngine/local/qlogin_ssh exited with exit code 0
```

Batch Job

qsub <job_script>

- Sample <job_script> was copied from /opt/templates/serial.qs and modified as serial-pgi.qs

[(it_css:traine)@mills cmatmul]$ qsub serial-pgi.qs
Your job 48328 ("serial-pgi.qs") has been submitted
[(it_css:traine)@mills cmatmul]$
serial-pgi.qs

[(it_css:traine)@mills cmatmul]$ more serial-pgi.qs
#
# Template:  Basic Serial Job
# Revision:  $Id: serial.qs 393 2012-04-13 12:37:02Z frey $
#
# If you want an email message to be sent to you when your job ultimately
# finishes, edit the -M line to have your email address and change the
# next two lines to start with #$ instead of just #
# -m eas
# -M my_address@mail.server.com
#
#
# Setup the environment; add vpkg_require commands after this
# line:
source /opt/shared/valet/docs/valet.sh
vpkg_require pgi

# Now append all of your shell commands necessary to run your program
# after this line:
./tmatmul
[(it_css:traine)@mills cmatmul]$
Output in `<job_script>·<job_id>`

```plaintext
[(it_css:traine)@mills cmatmul]$ more serial-pgi.qs.o48328
Adding package `pgi/11.10` to your environment
B:
  1.00000  1.00000  1.00000
  1.00000  1.50000  2.25000
  1.00000  2.00000  4.00000
  1.00000  3.00000  9.00000
C:
  1.00000  0.00000
  0.00000  1.00000
  0.50000  0.50000
B*C with loops:
  1.50000  1.50000
  2.12500  2.62500
  3.00000  4.00000
  5.50000  7.50000
[(it_css:traine)@mills cmatmul]$
```
Exercise
Exercise

- Pick a compiler: gcc, intel, open64 or pgi
- Compile and batch run the Fortran example in fmatmul using

  \( \text{compile-}\langle\text{compiler}\rangle \) to compile
  \( \text{serial-}\langle\text{compiler}\rangle.\text{qs} \) to batch run

Example using Fortran program to create the executable tmatmul using the PGI compiler

```
[(it_css:traine)@mills cmatmul]$ cd ~/mhpcI/fmatmul
[(it_css:traine)@mills fmatmul]$ pwd
/home/1201/mhpc/fmatmul
[(it_css:traine)@mills fmatmul]$
```
PGI compiler

```
source compile-pgi
```

```
[(it_css:traine)@mills fmatmul]$ more compile-pgi

```
touch tmatmul.f
vpkg_rollback all
vpkg_devrequire pgi
export FC=pgfortran
export FFLAGS='Mfree -Minform=warn -fast -tp bulldozer'
make tmatmul

```
[(it_css:traine)@mills fmatmul]$ source compile-pgi

Adding package `pgi/11.10` to your environment

```
pgfortran -Mfree -Minform=warn -fast -tp bulldozer tmatmul.f -o tmatmul

```
[(it_css:traine)@mills fmatmul]$
```
Batch Job Run

- PGI Compiler

```bash
qsub serial-pgi.qs
```

```
[(it_css:traine)@mills fmatmul]$ qsub serial-pgi.qs
Your job 49623 ("serial-pgi.qs") has been submitted
[(it_css:traine)@mills fmatmul]$
```
Adding package `pgi/11.10` to your environment

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>1.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>1.0000</td>
<td>1.5000</td>
<td>2.2500</td>
</tr>
<tr>
<td>1.0000</td>
<td>2.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td>1.0000</td>
<td>3.0000</td>
<td>9.0000</td>
</tr>
</tbody>
</table>

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.0000</td>
<td>0.0000</td>
</tr>
<tr>
<td>0.0000</td>
<td>1.0000</td>
</tr>
<tr>
<td>0.50000</td>
<td>0.50000</td>
</tr>
</tbody>
</table>

**B*C with intrinsic matmul**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5000</td>
<td>1.5000</td>
</tr>
<tr>
<td>2.1250</td>
<td>2.6250</td>
</tr>
<tr>
<td>3.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td>5.5000</td>
<td>7.5000</td>
</tr>
</tbody>
</table>

**B*C with loops**

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.5000</td>
<td>1.5000</td>
</tr>
<tr>
<td>2.1250</td>
<td>2.6250</td>
</tr>
<tr>
<td>3.0000</td>
<td>4.0000</td>
</tr>
<tr>
<td>5.5000</td>
<td>7.5000</td>
</tr>
</tbody>
</table>
Need Help?

● **Email:** consult@udel.edu
   If you make the first line of the e-mail message
   
   Type=Cluster-Mills

   your question will be routed more quickly.

● **Phone:** (302) 831-6000
● **Text:** (302) 722-6820
Part II: Mills HPC Basics

Jumping in
Jumping in

- File Storage and recovery options
- Bash startup files
- Compiling, Running and Monitoring Jobs
- Local (non-standard) Commands
File Storage on Mills

- **Home directory** ~8 TB of usable space
  - Personal directory - 2GB (/home/\(<uid>\))
- **Lustre** ~180 TB of usable space
  - Private workgroup directory - fair share (/lustre/work/\(<investing_entity>\))
  - Public scratch directory (/lustre/scratch)
- **Archive** ~ 60 TB of usable space
  - Private workgroup directory - 1TB (/archive/\(<investing_entity>\))
- **Node-local scratch** (/scratch)
Recovery Options
Recovering files /home

/home filesystem is your default login permanent storage and is backed up.

- **Use** `recover` to recover files in `/home/<uid>`
  
  `recover>` `versions` `filename`
  `recover>` `add` `filename`
  `recover>` `recover`
  `recover>` `quit`

```
[(it_css:traine)@mills ~]$ recover
Current working directory is /home/1201/
recover> quit
[(it_css:traine)@mills ~]$`
```
Recovering files /archive

/archive filesystem is a larger permanent storage with snapshots.

- Use `.zfs/snapshot` to recover files in `/archive/<investing_entity>`
- Recovery from the previous week is available from a snapshot

```
[[it_css:traine]@mills snapshot]$ cd /archive/it_css/.zfs/snapshot
[[it_css:traine]@mills snapshot]$ ls
12 18 Fri Mon now prev prev-1 Sat Sun Thu Tue Wed
[[it_css:traine]@mills snapshot]$
```
Bash Startup Files
Keep startup files clean

- Make sure you understand what your startup files are doing.
- Environments are different for the login (head) node versus the compute nodes.
- If you make changes, test by starting a new login, don't logout.
- You can always restore your startup files to the system versions.
Startup files

- .bash_profile
- .bashrc
- .bash_udit
- .bash_logout
• Executed once at login
• `.bashrc` in your home directory is sourced
• Add lines to set your environment and start programs after the comment line in red

```bash
[(it_css:traine)@mills ~]$ more .bash_profile
# .bash_profile

# Get the aliases and functions
if [ -f ~/.bashrc ]; then
   . ~/.bashrc
fi

# User specific environment and startup programs

PATH=$PATH:$HOME/bin

export PATH
```
Executed by each new shell you start including your login shell via `.bash_profile`

Add lines to create aliases and bash functions after the comment line in red

Commands such as `xterm`, `workgroup` and `qlogin` automatically start a new shell

```bash
[(it_css:traine)@mills ~]$ more .bashrc

# .bashrc

# Source global definitions
if [ -f /etc/bashrc ]; then
  . /etc/bashrc
fi

# User specific aliases and functions
```
- Executed by each new shell
- Opt into IT suggested environment changes

```bash
((it_css:traine)@mills ~)$ more .bash_udit
#
## Change from "no" to "yes" to enable IT's suggested environment changes.
## The behaviors enabled by the remainder of this file are contingent on
## enabling IT_WANT_ENV_EXTENSIONS:
##
IT_WANT_ENV_EXTENSIONS="no"
#
## If you have multiple workgroups available to you, change this to the one
## you want to be the default; otherwise, the first one listed by
## "workgroup -q workgroups" will be your default:
##
IT_DEFAULT_WORKGROUP=""
#
## If you want the "workgroup" command to by default change your working
## directory to the $WORKDIR, change from "no" to "yes".
##
IT_WORKGROUP_CHDIR="no"
```
Executed at logout from the head (login) node, not the compute node when you use `qlogin`
To restore all your startup files (`.bashrc`, `.bash_profile`, `.bash_udit`, and `.bash_logout`) to the system default, type

```
cp /etc/skel/./bash* $HOME
```
Exercise (.bash_udit)
Exercise (.bash_uedit)

Customize our startup file .bash_uedit to opt into IT suggested environment changes by setting a default workgroup so we only need to type

\texttt{workgroup}

instead of

\texttt{workgroup -g \langle investing_entity \rangle}
Exercise (.bash_udit)

To see what aliases are defined use

```
alias

[(it_css:traine)@mills ~]$ [(it_css:traine)@mills ~]$ alias
alias mc='./usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
[(it_css:traine)@mills ~]$ ...
```
Exercise (.bash_udit)

Edit (vim) .bash_udit

```bash
[(it_css:traine)@mills ~]$ vim .bash_udit
##
## Change from "no" to "yes" to enable IT's suggested environment changes.
## The behaviors enabled by the remainder of this file are contingent on
## enabling IT_WANT_ENV_EXTENSIONS:
##
IT_WANT_ENV_EXTENSIONS="yes"

##
## If you have multiple workgroups available to you, change this to the one
## you want to be the default; otherwise, the first one listed by
## "workgroup -q workgroups" will be your default:
##
IT_DEFAULT_WORKGROUP="it_css"

##
## If you want the "workgroup" command to by default change your working
## directory to the $WORKDIR, change from "no" to "yes".
##
IT_WORKGROUP_CHDIR="no"
```
Exercise (.bash_udit)

Try out our new .bash_udit

- Do not logout! Start a new login session
- Now you only need to type workgroup

[traine@mills ~]$ alias
alias mc='.
/usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
alias workgroup='/
/opt/bin/workgroup -g it_css'
[traine@mills ~]$ workgroup
[(it_css:traine)@mills ~]$
Exercise (.bashrc)
Exercise (.bashrc)

Customize our startup file `.bashrc` to create aliases for workgroups and other file storage directories

- Create a new alias for each `<investing_entity>` to define a workgroup
- Create a new alias for each file storage personal work directory and change to it
Exercise (workgroup)

Create an alias for each `<investing_entity>` to set the workgroup

Example lines shown in red for `it_css`

```
[(it_css:traine)@mills ~]$ vim .bashrc
  1 # .bashrc
  2
  3 # Source global definitions
  4 if [ -f /etc/bashrc ]; then
  5       . /etc/bashrc
  6 fi
  7
  8 # User specific aliases and functions
  9
10 # it_css workgroup
11 alias it_css='\workgroup -g it_css'
```
Exercise (workgroup)

Try out our new .bashrc

- Do not logout! Start a new login session
- Now `it_css` and `workgroup` work the same.

```
[traine@mills ~]$ alias
alias it_css='\workgroup -g it_css'
alias mc=' . /usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
alias workgroup='/opt/bin/workgroup -g it_css'
[traine@mills ~]$ it_css
[(it_css:traine)@mills ~]$ exit
exit
[traine@mills ~]$ workgroup
[(it_css:traine)@mills ~]$
```
Exercise (file storage)

Make sure you have a your own personal directory created for each file storage area. This may vary for each `<investing_entity>` research group (eg. `users` or `projects` subdirectory may exist).

These exercises assume your username will be in the base work directories

- `/lustre/work/<investing_entity>/`
- `/lustre/scratch/`
- `/archive/<investing_entity>/`
Check for your username in `/lustre/work/<investing_entity>`

Example shows creating a personal directory for `traine` in `/lustre/work/it_css`

```bash
[(it_css:traine)@mills ~]$ cd /lustre/work/it_css
[(it_css:traine)@mills it_css]$ ls -lad traine
ls: cannot access traine: No such file or directory
[traine@mills it_css]$ mkdir traine
[traine@mills it_css]$ ls -lad traine
drwxr-xr-x 15 traine it_css 4096 Mar 23 11:27 traine
```
Exercise (/lustre/scratch)

Check for your username in /lustre/scratch/

Example shows a personal directory exists for

```
[(it_css:traine)@mills ~]$ cd /lustre/scratch
[(it_css:traine)@mills scratch]$ ls -lad traine
drwxr-sr-x 2 traine everyone 4096 Jul  3 11:08 traine
[(it_css:traine)@mills scratch]$
```
Check for your username in /
/archive/<investing_entity>

Example shows creating a personal directory for 
traine in /archive/it_css

```
[(it_css:traine)@mills ~]$ cd /archive/it_css
[(it_css:traine)@mills it_css]$ ls -lad traine
ls: cannot access traine: No such file or directory
[(it_css:traine)@mills it_css]$ mkdir traine
[(it_css:traine)@mills it_css]$ ls -lad traine
drwxr-sr-x+ 2 traine it_css 2 Jul  3 13:06 traine
```
Exercise (file storage)

Create an alias for each file storage to change to that work directory

Example lines shown in red for `traine` and `it_css`

```
[(it_css:traine)@mills ~]$ vim .bashrc
 1 # .bashrc
 2
 3 # Source global definitions
 4 if [ -f /etc/bashrc ]; then
 5     . /etc/bashrc
 6 fi
 7
 8 # User specific aliases and functions
 9
10 # it_css workgroup
11 alias it_css='\workgroup -g it_css'
12
13 # File storage work directories for traine and it_css
14 alias cdlustre='cd /lustre/work/it_css/traine'
15 alias cdscratch='cd /lustre/scratch/traine'
16 alias cdarchive='cd /archive/it_css/traine'
```
Exercise (file storage)

Try out our new .bashrc

- Do not logout! Start a new login session

```bash
[(it_css:traine)@mills traine]$ alias
alias cdarchive='cd /archive/it_css/traine'
alias cdlustre='cd /lustre/work/it_css/traine'
alias cdscratch='cd /lustre/scratch/traine'
alias it_css='\workgroup -g it_css'
alias mc='/.usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
alias workgroup='/opt/bin/workgroup -g it_css'
[(it_css:traine)@mills traine]$ cdlustre
[(it_css:traine)@mills traine]$ pwd
/lustre/work/it_css/traine
[(it_css:traine)@mills traine]$ cdarchive
[(it_css:traine)@mills traine]$ pwd
/archive/it_css/traine
[(it_css:traine)@mills traine]$ cdscratch
[(it_css:traine)@mills traine]$ pwd
/lustre/scratch/traine
```
Compiling, Running and Monitoring Jobs
C Example

C program example using a library called gsl (Gnu Scientific Library)

- `clib`

Compile scripts for valid compilers

- `compile-intel` and `compile-gcc`

Batch job scripts for each compiler

- `serial-intel.qs` and `serial-gcc.qs`
● Using VALET to set our environment

```
vpkg_versions gsl
```

● Only available for intel and gcc

```bash
[(it_css:traine)@mills clib]$ vpkg_versions gsl
Available versions in package (* = default version):
gsl             GNU Scientific Library
  * 1.15-gcc      Version 1.15 (compiled with native GCC)
      1.15-intel64 Version 1.15 (compiled with Intel 2011)
[(it_css:traine)@mills clib]$
```
Exercise
Exercise

Using our Lustre storage, compile and batch run the C program in `clib for`

- Intel
  
  `source compile-intel`
  `qsub serial-intel.qs`

- GCC
  
  `source compile-gcc`
  `qsub serial-gcc.qs`
Copy Examples to Lustre

Use your new alias, `cdlustre`, to get you to your Lustre storage and copy the examples.

```
  cdlustre
  cp -r ~trainf/mhpcII .
  cd mhpcII/clib
```

```
[[it_css:traine)@mills ~]$ cdlustre
[[it_css:traine)@mills traine]$ pwd
/lustre/work/it_css/traine
[[it_css:traine)@mills traine]$ cp -r ~trainf/mhpcII .
[[it_css:traine)@mills traine]$ cd mhpcII/clib
[[it_css:traine)@mills clib]$ pwd
/lustre/work/it_css/traine/mhpcII/clib
[[it_css:traine)@mills clib]$ ls
compile-gcc compile-intel example.c serial-gcc.qs serial-intel.qs
```
Compile intel version using

source compile-intel

```
[(it_css:traine)@mills clib]$
more compile-intel

touch example.c
vpkg_rollback all
vpkg_devrequire gsl/1.15-intel64
export CC=icc
export CFLAGS="-Wall"
export LDLIBS='-lgs1 -lgslcblas -lm'
make example

[(it_css:traine)@mills clib]$
```
Batch job script

```bash
[(it_css:traine)@mills clib]$ more serial-intel.qs
#
# Template: Basic Serial Job
# Revision: $Id: serial.qs 393 2012-04-13 12:37:02Z frey$
#
# If you want an email message to be sent to you when your job ultimately
# finishes, edit the -M line to have your email address and change the
# next two lines to start with #$ instead of just 
# -m eas
# -M my_address@mail.server.com
#
#$ -N intel-gsl

# Setup the environment; add vpkg_require commands after this
# line:
source /opt/shared/valet/docs/valet.sh
vpkg_require gsl/1.15-intel64

# Now append all of your shell commands necessary to run your program
# after this line:

echo ""
echo "---- Run Test ---------"
time ./example
[(it_css:traine)@mills clib]$
```
### Batch Run (intel) and monitor

**qsub** and **qstat**

```bash
[[it_css:traine]@mills clib]$ qsub serial-intel.qs
Your job 63153 ("intel-gsl") has been submitted

[[it_css:traine]@mills clib]$ qstat
```

<table>
<thead>
<tr>
<th>job-ID</th>
<th>prior</th>
<th>name</th>
<th>user</th>
<th>state</th>
<th>submit/start at</th>
<th>queue</th>
<th>slots</th>
<th>ja-task-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>63153</td>
<td>0.00000</td>
<td>intel-gsl</td>
<td>traine</td>
<td>qw</td>
<td>08/07/2012 13:49:03</td>
<td></td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

```bash
[[it_css:traine]@mills clib]$ qstat
```

<table>
<thead>
<tr>
<th>job-ID</th>
<th>prior</th>
<th>name</th>
<th>user</th>
<th>state</th>
<th>submit/start at</th>
<th>queue</th>
<th>slots</th>
<th>ja-task-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>63153</td>
<td>0.50500</td>
<td>intel-gsl</td>
<td>traine</td>
<td>r</td>
<td>08/07/2012 13:49:21</td>
<td>it_css.q+@n016</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
More monitoring

```
qestat -j <job_id>
```

```
[[it_css:traine)@mills clib]$ qstat -j 63153
==============================================================
job_number: 63153
exec_file: job_scripts/63153
submission_time: Tue Aug 7 13:49:03 2012
owner: traine
uid: 1201
group: it_css
gid: 1002
sge_o_home: /home/1201
sge_o_log_name: traine
sge_o_path: /usr/lib64/qt-3.3/bin:/opt/bin:/home/software/GridEngine/6.2u7/bin:/usr/local/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/home/1201/bin
sge_o_shell: /bin/bash
sge_o_workdir: /lustre/work/it_css/traine/mhpcII/lib
...
[[it_css:traine)@mills clib]$
```
[it_css:traine]@mills clib]$ more intel-gsl.o63153
Adding dependency `intel/2011-64bit` to your environment
Adding package `gsl/1.15-intel64` to your environment

---- Run Test ---------
J0(5) = -1.775967713143382642e-01

real 0m0.203s
user 0m0.000s
sys 0m0.005s
Compile (gcc)

Compile gcc version using

```
source compile-gcc
```

```bash
[(it_css:traine)@mills clib]$ more compile-gcc
touch example.c
vpkg_rollback all
vpkg_devrequire gsl/1.15-gcc gcc
export CC=gcc
export CFLAGS="-Wall"
export LDLIBS='-lgsl -lgslcblas -lm'
make example
```

```
[(it_css:traine)@mills clib]$ source compile-gcc
Adding package `gsl/1.15-gcc` to your environment
Adding package `gcc/system` to your environment
gcc -Wall -I/opt/shared/gsl/1.15-gcc/include -L/opt/shared/gsl/1.15-gcc/lib example.c -lgsl -lgslcblas -lm -o example
```
Batch Run (gcc) and output

qsub and look at batch run output

```bash
[(it_css:traine)@mills clib]$ qsub serial-gcc.qs
Your job 49479 ("gcc-gsl") has been submitted
[(it_css:traine)@mills clib]$ more gcc-gsl.o49479
Adding package `gsl/1.15-gcc` to your environment

---- Run Test ----------
J0(5) = -1.775967713143382642e-01

real    0m0.090s
user    0m0.000s
sys     0m0.002s
```
Local (non-standard) Commands
Check `/opt/bin` for local commands.

These are "non-standard" commands that are specific to the Mills cluster.

```bash
[it_css:traine]@mills ~]$ ls /opt/bin
dothill_controller_failures ldapcompare ldappasswd pdcp qstatgrp
dothill_disk_failure ldapdelete ldapsearch pdsh RCS
globusconnect ldapexop ldapurl qhostgrp workdir
hpc-user-info ldapmodify ldapwhoami qjobs workgroup
ldapadd ldapmodrdn passwd qnodes

[it_css:traine]@mills ~]$
```
Local Commands

```
hpc-user-info -a username
hpc-user-info -h
```

display information about `username`

```
[(it_css:traine)@mills ~]$ hpc-user-info -a traine
full-name = Student Training
last-name = Student Training
home-directory = /home/1201
email-address = traine@udel.edu
[(it_css:traine)@mills ~]$
```
Local Commands

qnodes
qnodes -g <investing_entity>

will display the compute nodes based on the current workgroup or specified with -g

[(it_css:traine)@mills ~]$ qnodes
n015 n016 n017
[(it_css:traine)@mills ~]$ exit
exit
[traine@mills ~]$ qnodes
Host group "@everyone" does not exist
[traine@mills ~]$ qnodes -g it_css
n015 n016 n017
Local Commands

qhostgrp
qhostgrp  -g  <investing_entity>

will display compute nodes system information for the current workgroup or specified with -g

```
[[it_css:traine]@mills ~]$ qhostgrp
HOSTNAME    ARCH      NCPU LOAD  MEMTOT   MEMUSE  SWAPTO  SWAPUS
[---------------------------------------------------------------]
  global       -        -   -     -        -      -      -
  n015  lx24-amd64  24  0.00   63.0G    1.4G  126.1G     0.0
  n016  lx24-amd64  24  0.00   63.0G    1.4G  126.1G     0.0
  n017  lx24-amd64  24  0.00   63.0G    1.4G  126.1G     0.0
```
Local Commands

qjobs
qjobs -g <investing_entity>
qjobs -a

will display the status of jobs submitted by your research group or specified workgroup with -g or all jobs with -a.
Local Commands

qstatgrp
qstatgrp -g <investing_entity>

will display queue information for the current workgroup or specified with \(-g\)
Need Help?

- **Email**: consult@udel.edu
  If you make the first line of the e-mail message
  
  Type=Cluster-Mills

  your question will be routed more quickly.

- **Phone**: (302) 831-6000
- **Text**: (302) 722-6820