Objectives

- Overview: Farber Community Cluster
- Part I: Get your feet wet
- Part II: Jump in
Overview: Farber HPC Basics

Farber Community Cluster

http://docs.hpc.udel.edu/clusters/farber/start
Background

What is the Farber cluster?

- It is the second UD 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 sponsored by a stakeholder
Farber Cluster

farber.hpc.udel.edu
Part I: Farber 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
Farber Accounts

Username and Password

- UD = UDelNet ID and password; can only be changed via the on the Network page.
  www.udel.edu/network

- non-UD = hpcguest<uid> and password is generated by IT staff and securely sent via the UD Dropbox; please change it using:
  www.hpc.udel.edu/user?authn=login
Connecting with SSH
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
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 Farber

```
ssh -Y username@farber.hpc.udel.edu
```

Using username "traine".

Farber cluster (farber.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://farber.hpc.udel.edu/

For support, please contact consult@udel.edu

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

Bash prompt

- user name = referred to as $USER
- cluster name = head (login) node
- ~ = current working directory
- $ = end of prompt

[traine@farber ~]$
At login, you start in your home directory (~)

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

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

- Home directory (/home)

Other file storage available:

- Lustre (/lustre/scratch)
- Node-local scratch (/scratch)
Groups and Workgroups
Farber Groups

Group names determine access level to specific system resources

groups

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

[traine@farber ~]$ groups
everyone students ud-users it_css
Groups in the *investing-entity category* are used to control access to compute nodes, queues and storage.

```
workgroup -g <investing_entity>
```

starts a new shell in your workgroup. You must set your workgroup to run a job on the cluster.

```
[traine@farber ~]$ workgroup -q workgroups
  1002  it_css
[traine@farber ~]$ workgroup -g it_css
[(it_css:traine)@farber it_css]$ echo $WORKDIR
/home/work/it_css
```
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
- Users can also install their own software and create a VALET package for it.
VALET Commands

vpkg_list

● a list of all available software packages installed by IT

Available packages:
in /opt/shared/valet/2.0/etc
    acml
    acpype
    apache-ant
    arcgis
    atlas
    blacs
...

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

```
vpkg_versions <package_id>
```

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

```
[traine@farber ~]$ vpkg_versions intel
Available versions in package (* = default version):

[/opt/shared/valet/2.0/etc/intel.vpkg_json]
intel		Intel Compiler Suite
2013-sp1.3.174	Version 2013 (SP1.3.174) for x86-64
2013		alias to intel/2013-sp1.3.174
2015.0.090	Version 2015 (first release) for x86-64
* 2015		alias to intel/2015.0.090
14.0.3		alias to intel/2013-sp1.3.174
```
VALET Commands

vpkg_require <package_id>
vpkg_devrequire <package_id>

- set your environment or development environment for <package_id>

```bash
[(it_css:traine)@farber ~]$ vpkg_require intel
Adding package `intel/2015.0.090` to your environment

[(it_css:traine)@farber ~]$ vpkg_devrequire intel
Adding package `intel/2015.0.090` to your environment
```
VALET Commands

vpkg_rollback all

● undo all changes to your environment

[(it_css:traine)@farber ~]$ vpkg_rollback all
[(it_css:traine)@farber ~]$
Compiling and Running Jobs
Compilers

There are three 64-bit compiler suites on Farber:

- gcc (GNU compiler suite)
- Intel Composer SE
- PGI (Portland Group Inc.’s Cluster Development Kit)

We generally recommend that you use the gcc compilers with its rich collection of tools and libraries. If your software/application has a proven Intel/Xeon configuration, then you will get better performance by choosing the intel SE compiler suite. Intel Fortran, ifort, is a better implementation of modern Fortran standards than gfortran.
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 and compile-intel

Batch scripts for each compiler

● serial-gcc.qs and serial-intel.qs
Copy Examples

```
cp -r ~trainf/fhpcI .
cd fhpcI
pwd
ls

[traine@farber ~]$ cp -r ~trainf/fhpcI .
[traine@farber ~]$ cd fhpcI/
[traine@farber fhpcI]$ pwd
/home/1201/fhpcI
[traine@farber fhpcI]$ ls
  cmatmul  fmatmul
```
Compiling Code
Compiling Code: system cc

- Programs can be compiled on the login (head) node or compute nodes.
- Use VALET to set up your compile-time environment

This example uses the system compiler (gcc) to compile a C program to create the executable tmatmul-gcc

```
[traine@farber fhpcI]$ cd cmatmul
[traine@farber cmatmul]$ ls *.c
tmatmul.c
[traine@farber cmatmul]$ vpkg_require gcc
Adding package `gcc/system` to your environment
[traine@farber cmatmul]$ make tmatmul
cc     tmatmul.c  -o tmatmul
[traine@farber cmatmul]$ mv tmatmul tmatmul-gcc
[traine@farber cmatmul]$ ls -la tmatmul*
-rw-r--r-- 1 traine everyone  818 Sep 24 17:07 tmatmul.c
-rwxr-xr-x 1 traine everyone 8860 Sep 28 22:26 tmatmul-gcc
```
Compiling Code: intel icc

- Programs can be compiled on the login (head) node or compute nodes.
- Use VALET to set up your compile-time environment

This example uses a script, `compile-intel`, which has the commands to make an Intel version executable

```
[traine@farber cmatmul]$ . compile-intel
Adding package `intel/2015.0.090` to your environment
icc -Wall -g -debug all  tmatmul.c  -o tmatmul
debug executable in ./tmatmul-intel
[traine@farber cmatmul]$ ls -la tmatmul*
-rw-r--r-- 1 traine everyone  818 Sep 24 17:07 tmatmul.c
-rwxr-xr-x 1 traine everyone 8860 Sep 28 22:26 tmatmul-gcc
-rwxr-xr-x 1 traine everyone 8781 Sep 28 22:28 tmatmul-intel
```
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.
Interactive (session) job

qlogin

[traine@farber cmatmul]$ workgroup -g it_css
[(it_css:traine)@farber cmatmul]$ qlogin
Your job 484 ("QLOGIN") has been submitted
waiting for interactive job to be scheduled ...
Your interactive job 484 has been successfully scheduled.
Establishing /opt/shared/univa/local/qlogin_ssh session to host n038 ...
Run \texttt{tmatmul-gcc} on a compute node and exit

\begin{verbatim}
[(it_css:traine)@n038 cmatmul]$ vpkg_require gcc
Adding package `gcc/system` to your environment
[(it_css:traine)@n038 cmatmul]$ ./tmatmul-gcc
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)@n038 cmatmul]$ exit
exit
Connection to n038 closed.
/opt/shared/univa/local/qlogin_ssh exited with exit code 0
\end{verbatim}
Batch Job

qsub <job_script>

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

[(it_css:traine)@farber cmatmul]$ qsub serial-gcc.qs
Your job 410 ("serial-gcc.qs") has been submitted
Batch job output

[(it_css:traine)@farber cmatmul]$ cat *.o410

[CGROUPS] UD Grid Engine cgroup setup commencing
[CGROUPS] Setting 1073741824 bytes (vmem 1073741824 bytes) on n036 (master)
[CGROUPS] with 1 core = 0
[CGROUPS] done.

Adding package `gcc/system` to your environment

B:

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00000</td>
<td>1.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>1.00000</td>
<td>1.50000</td>
<td>2.25000</td>
</tr>
<tr>
<td>1.00000</td>
<td>2.00000</td>
<td>4.00000</td>
</tr>
<tr>
<td>1.00000</td>
<td>3.00000</td>
<td>9.00000</td>
</tr>
</tbody>
</table>

C:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.00000</td>
<td>0.00000</td>
</tr>
<tr>
<td>0.00000</td>
<td>1.00000</td>
</tr>
<tr>
<td>0.50000</td>
<td>0.50000</td>
</tr>
</tbody>
</table>

B*C with loops:

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.50000</td>
<td>1.50000</td>
</tr>
<tr>
<td>2.12500</td>
<td>2.62500</td>
</tr>
<tr>
<td>3.00000</td>
<td>4.00000</td>
</tr>
<tr>
<td>5.50000</td>
<td>7.50000</td>
</tr>
</tbody>
</table>
Batch Job template

qsub <job_script>

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

```
#$ -m eas
#$ -M <your email address>
vpkg_require intel/2015
./tmatmul-intel
```

```
[(it_css:traine)@farber cmatmul]$ qsub serial-intel.qs
Your job 485 ("serial-intel.qs") has been submitted
```
serial-intel.qs

[(it_css:traine)@farber cmatmul]$ more serial-intel.qs
#
# Template:  Basic Serial Job
# Revision:  $Id: serial.qs 523 2014-09-16 14:29:54Z frey $  
#
# Change the following to #$ and set the amount of memory you need
# per-slot if you're getting out-of-memory errors using the
# default:
# -l m_mem_free=2G
#
# 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
#
# Add vpkg_require commands after this line:
vpkg_require intel/2015

# Now append all of your shell commands necessary to run your program
# after this line:
./tmatmul-intel
[(it_css:traine)@farber cmatmul]$
Adding package `intel/2015.0.090` to your environment

\[ B: \]
\[
\begin{array}{ccc}
1.00000 & 1.00000 & 1.00000 \\
1.00000 & 1.50000 & 2.25000 \\
1.00000 & 2.00000 & 4.00000 \\
1.00000 & 3.00000 & 9.00000
\end{array}
\]

\(<\text{Same C: as above}>\)

\[ B*C \text{ with loops:} \]
\[
\begin{array}{ccc}
1.50000 & 1.50000 \\
2.12500 & 2.62500 \\
3.00000 & 4.00000 \\
5.50000 & 7.50000
\end{array}
\]
Exercise
Exercise

- Pick a compiler: gcc or intel
- Compile and batch run the Fortran example in fmatmul using
  
  ```
  compile-<compiler> to compile
  serial-<compiler>.qs to batch run
  ```

Example using Fortran program to create the executable

tmatmul-gcc using the system gfortran (gcc) compiler

```bash
[(it_css:traine)@farber cmatmul]$ cd ~/fhpcI/fmatmul
[(it_css:traine)@farber fmatmul]$ pwd
/home/1201/fhpc/fmatmul
[(it_css:traine)@farber fmatmul]$`
```
 Compile Code

- **create tmatmul-gcc fortran executable**

```
source compile-gcc

[[it_css:traine]@farber fmatmul]$ more compile-gcc
vpkg_rollback all
vpkg_devrequire gcc
export FC=gfortran
export FFLAGS='-ffree-form -Wall -g'
make tmatmul && mv tmatmul tmatmul-gcc
test -x tmatmul-gcc && echo "debug version in ./tmatmul-gcc"
```
Submit a job to run the fortran executable `tmatmul-gcc`

```
qsub serial-gcc.qs
```
Batch Job Output

[[it_css:traine)@farber fmatmul]$ more serial-gcc.qs.o612

[CGROUPS] UD Grid Engine cgroup setup commencing
[CGROUPS] Setting 1073741824 bytes (vmem 1073741824 bytes) on n038 (master)
[CGROUPS] with 1 core = 0
[CGROUPS] done.

Adding package `gcc/system` to your environment

B:
  1.0000  1.0000  1.0000
  1.0000  1.5000  2.2500
  1.0000  2.0000  4.0000
  1.0000  3.0000  9.0000

C:
  1.0000  0.0000
  0.0000  1.0000
  0.50000  0.50000

B*C with intrinsic matmul
  1.5000  1.5000
  2.1250  2.6250
  3.0000  4.0000
  5.5000  7.5000

B*C with loops
  1.5000  1.5000
  2.1250  2.6250
  3.0000  4.0000
  5.5000  7.5000
Need Help?

- **Submit** a [Research Computing Help Request](#) form; *High Performance Computing; Farber Cluster*
- **Phone:** (302) 831-6000
- **Text:** (302) 722-6820
Part II: Farber 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
File Storage on Farber

- **Home directory** 72 TB of usable space
  - Personal directory: 20GB (/home/<uid>)
    
    ```
    df -h $HOME
    ```
  - Workgroup directory: 1TB+ (/home/work/<investing-entity>)
    
    ```
    df -h $WORKDIR
    ```

- **Lustre** ~288 TB of usable space
  - Public scratch directory (/lustre/scratch)
  - IT staff will run regular cleanup procedures to purge aged files or directories

- **Node-local scratch** (/scratch)
Recovery Options
Recovering files /home

/home filesystem is a larger permanent storage with snapshots.

- Use read-only $HOME/.zfs/snapshot to recover files

```
[traine@farber cmatmul]$ ls -al tmatmul.c
-rw-r--r-- 1 traine everyone 818 Sep 24 17:07 tmatmul.c
[traine@farber cmatmul]$ rm tmatmul.c
[traine@farber cmatmul]$ $ echo 'oops !!' && ls -al tmatmul.c
oops !
ls: cannot access tmatmul.c: No such file or directory
[traine@farber cmatmul]$ pwd
/home/1201/fhpcI/cmatmul
[(it_css:traine)@farber fmatmul]$ ls ~/.zfs/snapshot
12 18 Fri Mon now prev prev-1 Sat Sun Thu Tue Wed
[traine@farber cmatmul]$ cp -p ~/.zfs/snapshot/now/fhpcI/cmatmul/tmatmul.c .
[traine@farber cmatmul]$ ls -al tmatmul.c
-rw-r--r-- 1 traine everyone 818 Sep 24 17:07 tmatmul.c
```
Recovering workgroup files

Workgroup files have their own read-only snapshots, which also span a week in

$WORKDIR/.zfs/snapshot

```
[traine@farber cmatmul]$ workgroup -g it_css
[(it_css:traine)@farber cmatmul]$ ls $WORKDIR/.zfs/snapshot/
12 18 Fri Mon now prev prev-1 Sat Sun Thu Tue Wed
```
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

.addrc in your home directory is sourced

Add lines to set your environment and start programs after the comment line in red

```
[traine@farber ~]$ 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, including your login shell via `.bash_profile`
● Add lines to create aliases and bash functions after the comment line in red

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

## 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"
Executed by each new shell
Opt into IT suggested environment changes

```bash
##
## 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 by each new shell
Opt into IT suggested environment changes

---

```bash
# By default when you login to the cluster head node you are in the
# "everyone" group and need to issue a "workgroup" command to prepare
# for submitting jobs, etc.

# Change this flag from "no" to "yes" if you want your login shell to
# automatically issue the command to change into your default
# workgroup. Your default workgroup will come from IT_DEFAULT_WORKGROUP
# if set above, or it will be the first group in the list produced by
# the command

# /opt/bin/workgroup --query workgroups

IT_SET_WORKGROUP_ON_LOGIN="no"
```
.bash_logout

- Executed at logout from the head (login) node, not the compute node when you use `qlogin`

```
[(it_css:traine)@farber ~]$ more .bash_logout
# ~/.bash_logout
# ~/.bash_logout
```
To restore all your startup files (`.bashrc`, `.bash_profile`, `.bash_udit`, and `.bash_logout`) to the system default, type

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

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

```
workgroup
```

instead of

```
workgroup -g <investing_entity>
```
Exercise (.bash_udit)

To see what aliases are defined use

alias

[(it_css:traine)@farber ~]$ alias
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
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)@farber ~]$
Exercise (.bash_udit)

Edit (vim) .bash_udit

```
([it_css:traine]@farber ~]$ 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` to set the workgroup to `it_css`.

```
[traine@farber ~]$ alias
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
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@farber ~]$ workgroup
[(it_css:traine)@farber ~]$`

By default when you login to the cluster head node you are in the "everyone" group and need to issue a "workgroup" command to prepare for submitting jobs, etc.

Change this flag from "no" to "yes" if you want your login shell to automatically issue the command to change into your default workgroup. Your default workgroup will come from IT_DEFAULT_WORKGROUP if set above, or it will be the first group in the list produced by the command:

```
/opt/bin/workgroup --query workgroups
```

IT_SET_WORKGROUP_ON_LOGIN="yes"
Exercise (.bashrc)
Exercise (.bashrc)

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

- Create a new alias whatis
- 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 (whatis)

Create an alias to determine what is the type of a command.

Example line shown in red

```bash
[it_css:traine]@farber ~]$ 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 alias whatis='builtin type -t'
```
Exercise (whatis)

Try out our new .bashrc

- Do not logout! Start a new login session.

```bash
[traine@farber ~]$ alias
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mc='. /usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias whatis='builtin type -t'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
alias workgroup='/opt/bin/workgroup -g it_css'
[traine@farber ~]$ which vpkg_require
/usr/bin/which: no vpkg_require in
(/opt/bin:/opt/shared/valet/2.0/bin:/bash:/opt/shared/valet/2.0/bin:/opt/shared/univa/current/bin/lx-amd64:/usr/lib64/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/ibutils/bin:/home/1201/bin)
[traine@farber ~]$ whatis vpkg_require
function
[traine@farber ~]$
```
Exercise (workgroup)

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

Example lines shown in red for `it_css`

```
[(it_css:traine)@farber ~]$ 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 alias whatis='builtin type -t'
 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@farber ~]$ alias
alias it_css='\workgroup -g it_css'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
alias mc='. /usr/libexec/mc/mc-wrapper.sh'
alias vi='vim'
alias whatis='builtin type -t'
alias which='alias | /usr/bin/which --tty-only --read-alias --show-dot --show-tilde'
alias workgroup='/opt/bin/workgroup -g it_css'
[traine@farber ~]$ it_css
[it_css:traine]@farber ~]$ exit
exit
[traine@farber ~]$ workgroup
[it_css:traine]@farber ~]$
```
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

- `/home/work/<investing_entity>/`
- `/lustre/scratch/`
Exercise (/home/work)

Check for your username in
/home/work/<investing_entity> or $WORKDIR

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

```bash
[(it_css:traine)@farber ~]$ cd $WORKDIR
[(it_css:traine)@farber it_css]$ ls -lad traine
ls: cannot access traine: No such file or directory
[traine@farber it_css]$ mkdir traine
[traine@farber it_css]$ ls -lad traine
drwxr-sr-x 2 traine it_css 2 Sep 29 00:10 traine
```
Exercise (/lustre/scratch)

Check for your username in /lustre/scratch/

Example shows a personal directory exists for traine in /lustre/scratch

```bash
[(it_css:traine)@farber ~]$ cd /lustre/scratch
[(it_css:traine)@farber scratch]$ ls -ld traine
drwxr-xr-x  2 traine it_css  4096 Sep 29 00:13 traine
[(it_css:traine)@farber scratch]$
```
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]@farber ~]$ 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 alias whatis='builtin type -t'
11 alias it_css='\workgroup -g it_css'
12 alias cdwork='cd /home/work/it_css/traine'
13 alias cdscratch='cd /lustre/scratch/traine'
```
Exercise (file storage)

Try out our new .bashrc

- Do not logout! Start a new login session

```bash
alias cdscratch='cd /lustre/scratch/traine'
alias cdwork='cd /home/work/it_css/traine'
alias it_css='\workgroup -g it_css'
alias l.='ls -d .* --color=auto'
alias ll='ls -l --color=auto'
alias ls='ls --color=auto'
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)@farber traine]$ cdwork
[(it_css:traine)@farber traine]$ pwd
/home/work/it_css/traine
[(it_css:traine)@farber traine]$ cdscratch
[(it_css:traine)@farber 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

```
[(it_css:traine)@farber ~]$ vpkg_versions gsl
Available versions in package (* = default version):

[/opt/shared/valet/2.0/etc/gsl.vpkg_json]
gsl         GNU Scientific Library
    1.16-intel64  Version 1.16 Intel64(2015) compilers
* 1.16        Version 1.16 with GCC(system) compilers
  intel64      alias to gsl/1.16-intel64
```
Exercise
Exercise

Using our workgroup 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

Use your new alias, *cdwork*, to get you to your workgroup storage and copy the examples.

```bash
    cdwork
cp -r ~trainf/fhpcII .
cd fhpcII/clib
```

```
[it_css:traine]@farber traine]$ cdwork
[it_css:traine]@farber traine]$ pwd
/home/work/it_css/traine
[it_css:traine]@farber traine]$ cp -r ~trainf/fhpcII .
[it_css:traine]@farber traine]$ cd fhpcII/clib
[it_css:traine]@farber clib]$ pwd
/home/work/it_css/traine/fhpcII/clib
[it_css:traine]@farber clib]$ ls
compile-gcc  compile-intel  example.c  serial-gcc.qs  serial-intel.qs
```
Compile using script `compile-intel` to generate executable `example-intel`

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

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

[(it_css:traine)@farber clib]$ source compile-intel

Adding dependency `intel/2015.0.090` to your environment
Adding package `gsl/1.16-intel64` to your environment

icc -Wall -I/opt/shared/gsl/1.16-intel64/include -L/opt/shared/gsl/1.16-intel64/lib example.c -lgsl -lgslcblas -lm -o example
```
[(it_css:traine)@farber clib]$ more serial-intel.qs
#
# Template:  Basic Serial Job
# Revision:  $Id: serial.qs 523 2014-09-16 14:29:54Z frey $ 
#
# Change the following to #$ and set the amount of memory you need
# per-slot if you're getting out-of-memory errors using the
# default:
# -l m_mem_free=2G
#
# 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
#
# Add vpkg_require commands after this line:
vpkg_require gsl/intel64
#
# Now append all of your shell commands necessary to run your program
# after this line:

echo ""
echo "---- Run Test ---------"
time ./example-intel
Batch Run (intel) and monitor

qsub and qstat

```bash
[[it_css:traine]@farber clib]$ qsub serial-intel.qs
Your job 615 ("serial-intel.qs") has been submitted
[[it_css:traine]@farber clib]$ qstat
job-ID  prior  name    user  state submit/start at  queue
jclass  slots ja-task-ID
-------------------------------------------------------------------
615 0.00000 serial-int  traine  qw  09/29/2014 01:03:28
```
More monitoring

```
[qstat -j <job_id>]

[(it_css:traine)@farber clib]$ qstat -j 615
===================================================================
job_number: 615
jclass: NONE
exec_file: job_scripts/615
submission_time: 09/29/2014 01:03:28.083
owner: traine
uid: 1201
group: it_css
gid: 1002
sgo_home: /home/1201
sgo_log_name: traine
sgo_path: /opt/shared/gsl/1.16-
            intel64/bin:/opt/shared/intel/composer_xe_2015.0.090/bin/intel64:/opt/shared/intel/composer_xe_2
            015.0.090/mpirt/bin/intel64:/opt/shared/intel/composer_xe_2015.0.090/debugger/gdb/intel64_mic/bin:
            /opt/bin:/opt/shared/valet/2.0/bin/bash:/opt/shared/valet/2.0/bin:/opt/shared/univa/current/bin/lx-amd64:
            /usr/lib64/qt-3.3/bin:/usr/local/bin:/bin:/usr/bin:/usr/local/sbin:/usr/sbin:/sbin:/opt/libutils/bin:/home/1201
            /bin
sgo_shell: /bin/bash
sgo_workdir: /home/work/it_css/traine/fhpcII/clib
sgo_host: login000
...
[(it_css:traine)@farber clib]$`
```
Look at batch run output

```
[(it_css:traine)@farber clib]$ more serial-intel.qs.o615

[CGROUPS] UD Grid Engine cgroup setup commencing
[CGROUPS] Setting 1073741824 bytes (vmem 1073741824 bytes) on n038 (master)
[CGROUPS]   with 1 core = 0
[CGROUPS] done.

Adding dependency `intel/2015.0.090` to your environment
Adding package `gsl/1.16-intel64` to your environment

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

real 0m0.005s
user 0m0.001s
sys 0m0.001s
```
Local (non-standard) Commands
Local Commands

Check /opt/bin for local commands.

These are "non-standard" commands that are specific to the Farber cluster; UD community clusters.
Local Commands

`hpc-user-info -a username`

`hpc-user-info -h`

display information about `username`

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

qnodes
qnodes -g <investing_entity>

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

```
[(it_css:traine)@farber ~]$ qnodes
n036 n037 n038 n039 n040
[(it_css:traine)@farber ~]$ exit
exit
[traine@farber ~]$ qnodes
Host group "@everyone" does not exist
[traine@farber ~]$ qnodes -g it_css
n036 n037 n038 n039 n040
```
Local Commands

qhostgrp

qhostgrp -g <investing_entity>

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

```
[(it_css:traine)@farber ~]$ qhostgrp
HOSTNAME    ARCH    NCPU NSOC NCOR NTHR NLOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
------------ --------- ----- ----- ----- ----- ------ ------- ------- ------- -------
global
n036        lx-amd64 20    2    20   20  0.00   63.0G    1.1G    2.0G     0.0
n037        lx-amd64 20    2    20   20  0.00   63.0G    1.1G    2.0G     0.0
n038        lx-amd64 20    2    20   20  0.00   63.0G    1.1G    2.0G     0.0
n039        lx-amd64 20    2    20   20  0.00   63.0G    1.1G    2.0G     0.0
n040        lx-amd64 20    2    20   20  0.00   63.0G    1.1G    2.0G     0.0
```
Local Commands

```
qhostgrp -j
qhostgrp -g <investing_entity> -j
```

will display compute nodes system information for the current workgroup or specified with `-g` and all the jobs running on each node including standby

```
[(it_css:traine)@farber ~]$ qhostgrp -j
HOSTNAME     ARCH     NCPU NSOC NCOR NTHR NLOAD  MEMTOT  MEMUSE  SWAPTO  SWAPUS
-----------------------------------------------------------------------------------------------
global        -        -   -   -    -   -    -    -      -       -       -
n036          lx-amd64 20  2  20  20  0.05  63.0G  8.0G  2.0G    1.7M

<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>master</th>
<th>ja-task-ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>385547</td>
<td>0.25590</td>
<td>openmpi-ib</td>
<td>zzhou</td>
<td>r</td>
<td>12/18/2015 09:07:12</td>
<td>standby-4h</td>
<td>SLAVE</td>
<td></td>
</tr>
<tr>
<td>385261</td>
<td>0.25632</td>
<td>JobArray</td>
<td>rpatet</td>
<td>r</td>
<td>12/18/2015 03:52:41</td>
<td>standby.q@</td>
<td>MASTER 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>standby.q@</td>
<td>SLAVE 31</td>
<td></td>
</tr>
</tbody>
</table>
```

.....
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 \(-g\)

```
[[it_css:traine]@farber ~]$ qstatgrp
CLUSTER QUEUE CQLOAD USED RES AVAIL TOTAL aoACDS csuE
it_css.q 0.00 0 100 100 0 0
standby-4h.q 0.07 0 1680 1800 120 0
standby.q 0.07 0 1680 1800 120 0
```
Local Commands

```
qstatgrp -j
qstatgrp -g <investing_entity> -j
```

will display queue information for the current workgroup or specified with `–g` and all jobs running or queued by workgroup members.

```
--snip--
```

```
385557 0.50000 openmpi-ib traine r 12/18/2015 10:07:41 it_css.q@n040
377224 0.50429 script2 trainf qw 12/09/2015 15:06:59
377225 0.50429 script3 trainf qw 12/09/2015 15:09:23
--snip--
```
Need Help?

- Submit a Research Computing Help Request form; High Performance Computing; Farber Cluster
- Phone: (302) 831-6000
- Text: (302) 722-6820