XSEDE New User Tutorial

Jay Alameda
National Center for Supercomputing Applications
XSEDE Training Survey

• Make sure you sign the sign-in sheet!
• At the end of the module, I will ask you to please complete a short on-line survey about this module. The survey can be found at http://bit.ly/CSUXSEDE
• I’ll post this URL at the end of the course as well.
Now what?

Yeah! I got an XSEDE allocation!
Learning Outcomes

After completing this tutorial, you will be able to:

• Use the XSEDE User Portal
• Access your XSEDE resources
• Manage files
• Run jobs
• Get help
XSEDE User Portal (XUP)

- URL: portal.xsede.org
- Single point-of-entry to information about XSEDE services and utilities for using them
- Anyone can create an XUP user account and access non-project features
- Only XSEDE allocation project members can access project features
Using the XUP

• Create and login to your XUP Account
• Use XSEDE resources responsibly
• Get added to your XSEDE project
• Navigate your personal My XSEDE webpage
• Navigate the information in the XUP
Create and login to your XUP account

1. From the XUP homepage, click CREATE ACCOUNT
2. Complete the User Account Form
3. Verify your account request
4. Select your username and password
5. Login to the XUP

portal.xsede.org
XSEDE Acceptable Use Policy

- Must accept the [User Responsibilities Form](#) after creating your XUP account and again at the beginning of each allocation you receive.
- Choose a strong password and protect it.
- Close SSH terminals and log out of the User Portal when you are finished with your session.
- Report Suspicious Activity: email [help@xsede.org](mailto:help@xsede.org) or call 1-866-907-2383 immediately, regardless of the time of day.

XSEDE Cybersecurity Tutorial

[http://www.citutor.org](http://www.citutor.org)
Get Added to Your XSEDE project

• PIs automatically have full access to their project’s account.

• The PI is responsible for managing users on their account.

• Ask the PI, or their allocation manager, to add your XUP username to the project.
WELCOME TO XUP
• Quick access to commonly used features.

LATEST UPDATES
• Latest information specific to your user account.

MY ACTIVE ALLOCATIONS
• Summary of the active projects for which you are either a PI or member.
Update your XUP User Profile

MY XSEDE→Profile

- View and or change your user information (organization, address).
- Make sure your email address is correct. XSEDE staff will use it to communicate with you regarding your allocation.
Navigating the XUP

- My XSEDE
- Resources
- Documentation
- Allocations

- Training
- User Forums
- Help
- About
View the XSEDE Systems Monitor

- **Resources -> Systems Monitor**
  - Provides technical and status information for all of XSEDE's resources.
  - The STATUS column indicates whether the system is up or down. If down, can click on status to find when the machine is expected to come back up.

<table>
<thead>
<tr>
<th>NAME</th>
<th>INSTITUTION</th>
<th>SYSTEM</th>
<th>PEAK TFLOPS</th>
<th>MEMORY TBYTES</th>
<th>STATUS</th>
<th>LOAD</th>
<th>RUNNING JOBS</th>
<th>QUEUED JOBS</th>
<th>OTHER JOBS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stampede</td>
<td>TACC</td>
<td>Dell PowerEdge C8220 Cluster with Intel Xeon Phi coprocessors</td>
<td>9500.00</td>
<td>205.00</td>
<td>Up</td>
<td></td>
<td>467</td>
<td>1175</td>
<td>263</td>
</tr>
<tr>
<td>Kraken</td>
<td>NICS</td>
<td>Cray XTS</td>
<td>1174.00</td>
<td>147.00</td>
<td>Up</td>
<td></td>
<td>237</td>
<td>409</td>
<td>261</td>
</tr>
</tbody>
</table>
Accessing XSEDE Resources

Authentication Methods
1. Password
   - XUP credentials
   - Site-password
   - One-time password
2. Key-based

Single Sign-On
- Enables logging in once to access all of your allocated resources

Connection Methods
1. GSI-OpenSSH
2. OpenSSH
XSEDE SSO Login Hub

An SSO enabled connection point to XSEDE resources

- SSH to login.xsede.org using your XUP credentials
- Move among resources using gsssh command
Following along with today’s tutorial:

- Verify that everyone has an ssh client on their laptop!
- For ssh to XSEDE SSO login hub (today!)
  - `ssh -l username login.xsede.org`
  - *Username* on handout
- And from there go to your XSEDE resource, for example:
  - `gsissh trestles.sdsc.edu`
Managing your XSEDE files

1. Where to store files
   - Home directory
   - Scratch directory
   - Archival storage

2. How to move files
   - Command line using globus-url-copy, uberftp, scp, or sftp
   - Globus Online
XSEDE File Systems

• **Home directory**
  - Location specified in the environment variable $HOME.
  - Use to store project files you want to keep long term such as source code, scripts, and input data sets.
  - Not backed up regularly and not purged.
  - Quotas typically set to limit amount of disk space available.

• **Scratch directory**
  - Location specified in environment variable varies among resources but will include the term SCRATCH, e.g. $SCRATCH_DIR.
  - Use to temporarily store files produced during application runs.
  - Not backed up and routinely purged.
  - No quotas. Available space depends on cumulative use by all users.

• **Archival storage**
  - Must request through allocation process
Your XSEDE Compute Environment

• Your default XSEDE compute environment provides access to the compilers, directories, and software you will need to efficiently use your XSEDE resources.

• Customize it using **Modules**
Modules Package

- A command line interface used to configure the shell for an application. Two components:
  1. Modulefiles - contain configuration information
  2. Module command - interprets modulefiles

- Pre-written modulefiles available for compilers, mpi implementations
- Pre-written modulefiles available for common software, e.g. NAMD, GAMESS
## Module Commands

<table>
<thead>
<tr>
<th>Module command</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td><code>module avail [path...]</code></td>
<td>List all modulefiles available on the system.</td>
</tr>
<tr>
<td><code>module list</code></td>
<td>List the modulefiles currently loaded in the shell environment.</td>
</tr>
<tr>
<td><code>module help modulefile</code></td>
<td>Print help information for the <code>modulefile</code> specified in the argument.</td>
</tr>
<tr>
<td><code>module display modulefile</code></td>
<td>Display the changes made to the environment when the specified modulefile is loaded.</td>
</tr>
<tr>
<td><code>module load modulefile</code></td>
<td>Interpret the commands contained within the specified modulefile.</td>
</tr>
<tr>
<td><code>module switch modulefile1 modulefile2</code></td>
<td>Remove the environment changes made by <code>modulefile1</code> and make the changes specified in <code>modulefile2</code>.</td>
</tr>
<tr>
<td><code>module unload modulefile</code></td>
<td>Remove the environment changes made by <code>modulefile</code>.</td>
</tr>
</tbody>
</table>
Module Commands Example

% module list
Currently Loaded Modulefiles:
  1) torque/2.3.13_psc  4) icc/14.0.0  7) globus/5.2.2
  2) mpt/2.04          5) imkl/10.3.3  8) xdusage/1.0-r7
  3) ifort/14.0.0      6) psc_path/1.0

% module avail gcc
------------------------- /usr/local/opt/modulefiles --------------------------
gcc/4.3.5 gcc/4.4.6 gcc/4.5.3 gcc/4.6.0 gcc/4.7.2 gcc/4.8.0 gcc/4.8.1
% module load gcc/4.8.1
% module list
Currently Loaded Modulefiles:
  1) torque/2.3.13_psc  5) imkl/10.3.3  9) mpfr/3.1.0
  2) mpt/2.04          6) psc_path/1.0  10) gmp/5.0.5
  3) ifort/14.0.0      7) globus/5.2.2  11) mpc/0.8.2
  4) icc/14.0.0       8) xdusage/1.0-r7  12) gcc/4.8.1

% module unload gcc
% module list
Currently Loaded Modulefiles:
  1) torque/2.3.13_psc  4) icc/14.0.0  7) globus/5.2.2
  2) mpt/2.04          5) imkl/10.3.3  8) xdusage/1.0-r7
  3) ifort/14.0.0      6) psc_path/1.0
Moving Files - Globus Online

- A fast, reliable, and secure file transfer service geared to the big data needs of the research community.
- Moves terabytes of data in thousands of files
- Automatic fault recovery
- Easy to use
- No client software installation
- Consolidated support and troubleshooting
- Supports file transfer to any machine
- Accounts are free - www.globusonline.org
Globus Online File Transfer
Running Jobs Overview

When you connect to a resource, you are on a login node shared by many users.

Commands for code execution, copy input files to scratch,… Specify number/type of nodes, length of run, output directory, …

Batch Script

Run jobs by submitting your batch script to the compute nodes using the "qsub" command.

Your job is submitted to a queue and will wait in line until nodes are available. Queues are managed by a job scheduler that allows jobs to run efficiently.

Batch Script

Job

Use for tasks such as file editing, code compilation, data backup, and job submission.

Login Nodes

Data

Read/write data from compute nodes to Scratch directory.

Data

File System

Home

Scratch

Store project files
Such as source code, scripts, and input data sets to Home directory.

Researcher

Login

Command Line

Compute Nodes

Your job is submitted to a queue and will wait in line until nodes are available. Queues are managed by a job scheduler that allows jobs to run efficiently.
Login Nodes

• When you login to an XSEDE resource, you connect to a login node.

• Use login nodes for basic tasks such as file editing, code compilation, data backup, and job submission.

• Do not run compute jobs on the login nodes.

Where do I run compute jobs?
Running Compute Jobs

- Jobs are run on the compute nodes by submitting a **batch script** on a login node.
- All jobs are placed in a **batch queue** after they are submitted.
- All XSEDE compute resources use a **batch scheduler** for running jobs.
- Resource User Guides on the XUP have details on your system’s scheduler.
Batch Scripts

- Batch scripts include scheduler specific directives, comments, and executable commands, e.g.:
  - Number and type of nodes needed
  - Time needed to run the job
  - Where to write output files
- Script commands are system specific – see the resource’s User Guide on the XUP for details
Batch Script for PSC’s Blacklight

1. #!/bin/csh
2. #PBS -l ncpus=16
3. #ncpus must be a multiple of 16
4. #PBS -l walltime=5:00
5. #PBS -j oe
6. #PBS -q batch
7. set echo
8. ja
9. #move to my $SCRATCH directory
10. cd $SCRATCH
11. #copy executable to $SCRATCH
12. cp $HOME/mympi
13. #run my executable
14. mpirun -np $PBS_NCPUS ./mympi
15. ja -chlst

Blacklight uses the Portable Batch System (PBS) scheduler. Lines 2, 4, 5, and 6 are PBS directives.
Submitting Batch Scripts

• Commands are machine specific, but follow general principles

• With PBS batch scripts, use the `qsub` command

  `qsub myscript.job`

• Can also specify PBS directives as command-line options:

  `qsub -l ncpus=16 -l walltime=5:00 -j oe -q batch myscript.job`

• Command-line directives override directives in your scripts.
More PBS commands

• qstat - displays the status of batch jobs.

<table>
<thead>
<tr>
<th>Option</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>-a</td>
<td>gives the status of all jobs on the system.</td>
</tr>
<tr>
<td>qstat -n</td>
<td>lists nodes allocated to a running job in addition to basic information.</td>
</tr>
<tr>
<td>qstat -f PBS_JOBID</td>
<td>gives detailed information on a particular job.</td>
</tr>
<tr>
<td>-q</td>
<td>provides summary information on all the queues.</td>
</tr>
</tbody>
</table>

• qdel – deletes a queued job or kills a running job.

• See the qsub manpage for more
Example Batch Command

```bash
csub amber.job
cstat -a
```

<table>
<thead>
<tr>
<th>Job ID</th>
<th>Username</th>
<th>Queue</th>
<th>Jobname</th>
<th>SessID</th>
<th>NDS</th>
<th>Tasks</th>
<th>Memory</th>
<th>Time</th>
<th>S Time</th>
</tr>
</thead>
<tbody>
<tr>
<td>29668</td>
<td>user1</td>
<td>batch</td>
<td>job2</td>
<td>21909</td>
<td>1</td>
<td>256</td>
<td>--</td>
<td>08:00</td>
<td>R</td>
</tr>
<tr>
<td>29894</td>
<td>user2</td>
<td>batch</td>
<td>run128</td>
<td>--</td>
<td>1</td>
<td>128</td>
<td>--</td>
<td>02:30</td>
<td>Q</td>
</tr>
<tr>
<td>29895</td>
<td>user3</td>
<td>batch</td>
<td>STDIN</td>
<td>15921</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>01:00</td>
<td>R</td>
</tr>
<tr>
<td>29896</td>
<td>user2</td>
<td>batch</td>
<td>jobL</td>
<td>21988</td>
<td>1</td>
<td>2048</td>
<td>--</td>
<td>01:00</td>
<td>R</td>
</tr>
<tr>
<td>29897</td>
<td>user4</td>
<td>batch</td>
<td>STDIN</td>
<td>22367</td>
<td>1</td>
<td>2</td>
<td>--</td>
<td>00:30</td>
<td>R</td>
</tr>
<tr>
<td>29898</td>
<td>user1</td>
<td>batch</td>
<td>amber</td>
<td>25188</td>
<td>1</td>
<td>1</td>
<td>--</td>
<td>01:10</td>
<td>R</td>
</tr>
</tbody>
</table>

```

qdel 29668

**• After job 29898 runs: user1 should get file amber.job.o29898 with output/errors (log file)**
Job Scheduling

• All XSEDE compute resources use a batch scheduler for running jobs.
• All jobs are placed in a batch queue after they are submitted.
• Resource User Guides on the XUP have details on your system’s scheduler.
Batch Schedulers

• Attempt to balance queue wait times of competing jobs with efficient system utilization.
  – Job prioritization influenced by number of cores and wall clock time requested
  – FIFO queues with fair use mechanisms to keep a single user from dominating the queue
  – Backfilling unused nodes with smaller jobs

• Will not start jobs if they will not finish before scheduled system maintenance.
Common problems encountered when running jobs:

• Invalid number of cores were requested
• Job runs out of CPU time
• Files can’t be found
• Inadequate software permissions
Improving job turnaround

• Request accurate walltime
• Use flexible walltime
• Pack your job
  – Running many small jobs places a great burden on the scheduler and is also inconvenient for you.
  – Pack many executions into a single job, which you then submit to PBS with a single qsub command.
Requesting flexible walltime

Example: Your job requests 64 cores and a walltime between 2 and 4 hours. If there is a 64 core slot available for 3 hours, your job could run in this slot. However, if your job had requested a fixed walltime of 4 hours it would not run until the larger time slot becomes available.
Packing Serial Jobs

Run each program execution in the background and place a wait command after each execution.

```
#!/bin/csh
#PBS -l ncpus=96
#PBS -l walltime=5:00
#PBS -q batch
dplace -c 0 ./myserial1 < serial1.dat &
dplace -c 32 ./myserial2 < serial2.dat &
dplace -c 64 ./myserial3 < serial3.dat &
wait
```
Packing Serial or MPI jobs

- The dplace command insures that each execution will run on its own set of 32 cores. The executions will run concurrently.
- Same approach using the dplace command can be used to pack MPI executables.

dplace -c 0 ./myserial1 < serial1.dat &
dplace -c 32 ./myserial2 < serial2.dat &
dplace -c 64 ./myserial3 < serial3.dat &
wait
Packing OpenMP Jobs

To pack OpenMP executables, replace the dplace command with the omplace command. Sample job to pack OpenMP executables:

```bash
omplace -nt 32 -c 0 ./myopenmp1 < myopenmp1.dat &
omplace -nt 32 -c 32 ./myopenmp2 < myopenmp2.dat &
omplace -nt 32 -c 64 ./myopenmp3 < myopenmp3.dat &
omplace -nt 32 -c 96 ./myopenmp4 < myopenmp4.dat &
wait
```
Managing Your Environment: Modules

- Allows you to manipulate your environment.
- `module list` shows currently loaded modules.
- `module avail` shows available modules.
- `module show <name>` describes module.

http://modules.sourceforge.net/

% module load gcc/3.1.1
% which gcc
/usr/local/gcc/3.1.1/linux/bin/gcc

% module switch gcc/3.1.1 gcc/3.2.0
% which gcc
/usr/local/gcc/3.2.0/linux/bin/gcc

% module unload gcc
% which gcc
gcc not found
For the following exercise (same steps as before):

• Check to see if connection is still live, if not:
• For ssh to XSEDE SSO login hub (today!)
  – ssh -l username login.xsede.org
  – Username on handout

• And from there go to your XSEDE resource, for example:
  – gsissh trestles.sdsc.edu
SDSC trestles Cluster & Modules

• Default environment pgi compilers, mvapich2 MPI implementation

• We will swap pgi compilers with gnu compilers
  – Demo: module swap pgi gnu
  – which gcc

• And then we’ll load the openMPI library
  – module load openmpi_ib
  – which mpicc
Module demo on trestles

[xdtr1@trestles-login2 ~]$ module swap pgi gnu
Unloading compiler-dependent module mvapich2_ib/1.9
[xdtr1@trestles-login2 ~]$ which gcc
/opt/gnu/gcc/bin/gcc
[xdtr1@trestles-login2 ~]$ module load openmpi_ib
[xdtr1@trestles-login2 ~]$ which mpicc
/opt/openmpi/gnu/ib/bin/mpicc
[xdtr1@trestles-login2 ~]$
Using modules in your job script

• First, source /usr/share/Modules/init/shell-name
• module load module-name
• Check on-line documentation
Exercise

• Make sure you are on trestles.sdsc.edu
• Run the shallow water model code provided
• No input file needed
• Copy batch script from my home directory:
  – `cp ~/ux400689/shallow-batch.sh`.
#!/bin/bash --login
PBS -q shared
PBS -N ptp_job
PBS -l nodes=1:ppn=5
PBS -l walltime=00:30:00
PBS -V

MPI_ARGS="-np 5"
if [ "-np" == "${MPI_ARGS}" ] ; then
  MPI_ARGS=
fi

source /usr/share/Modules/init/bash
module swap pgi gnu
module load openmpi_ib

cd /oasis/scratch/trestles/$USER/$PBS_JOBID
cp /home/ux400689/shallow-SR1_kepler_try2/shallow .
MYSCREXE=`basename /home/ux400689/shallow-SR1_kepler_try2/shallow`

COMMAND=mpirun

if [ -n "${COMMAND}" ] ; then
  COMMAND="${COMMAND} ${MPI_ARGS} -hostfile ${PBS_NODEFILE} ${MYSCREXE}"
else
  COMMAND="${MYSCREXE}"
fi

${COMMAND}
Exercise:

• Submit the job (*qsub shallow-batch.sh*)
• Monitor the job (*qstat –u username*)
• Make sure you have the output files at job completion

```bash
[xdtr1@trestles-login1 ~]$ ls
Input ptp_job.2011011 ptp_job.o2011011 Results shallow-batch.sh
[xdtr1@trestles-login1 ~]$ 
```

• more ptp*11 (for this case, yours will be different!)
Output files: need to show successful completion

jstart=0, jend=7, next=2, prev=4
jstart=8, jend=15, next=3, prev=1
jstart=16, jend=23, next=4, prev=2
jstart=24, jend=31, next=1, prev=3

Shallow water weather model - Distributed Memory Version 0.6

Number of points in the X direction  32
Number of points in the Y direction  32
Grid spacing in the X direction    100000.00
Grid spacing in the Y direction    100000.00
Time step                          90.000
Time filter parameter              0.001
Cycle number 1  Model time in days 0.00
    Potential energy     0.000  Kinetic Energy  48036.828
    Total Energy         48036.828  Pot. Enstrophy  0.000000e+00

Cycle number 50  Model time in days 0.05
    Potential energy 1256.284  Kinetic Energy  46526.969
    Total Energy 47783.254  Pot. Enstrophy -nan

Cycle number 100  Model time in days 0.10
Need help? Reporting and Tracking Issues

• portal.xsede.org → Help
  Submit ticket

• portal.xsede.org → My XSEDE → Tickets
  – Submit ticket
  – View past tickets (both open and closed)

• Can also email help@xsede.org or call 1-866-907-2383, at any hour (24/7)
Discussing your problems…
User Portal: User Forums

- The User Forums are a great place to ask questions, get help, or discuss ideas about XSEDE.
More “helpful” resources

xsede.org ➔ User Services

• Resources available at each Service Provider
  • User Guides describing memory, number of CPUs, file systems, etc.
  • Storage facilities
  • Software (Comprehensive Search)

• Training: portal.xsede.org ➔ Training
  • Course Calendar
  • On-line training

• Get face-to-face help from XSEDE experts at your institution; contact your local Campus Champions.

• Extended Collaborative Support (formerly known as Advanced User Support (AUSS))
XSEDE Training Survey

• Please complete a short on-line survey about this module. The survey can be found at http://bit.ly/CSUXSEDE

• We value your feedback, and will use your feedback to help improve our training offerings.
Thanks for listening and welcome to XSEDE!