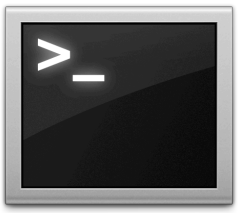
Running Jobs, Submission Scripts, Modules

Matt Gitzendanner: magitz@ufl.edu
Alex Moskalenko: om@hpc.ufl.edu

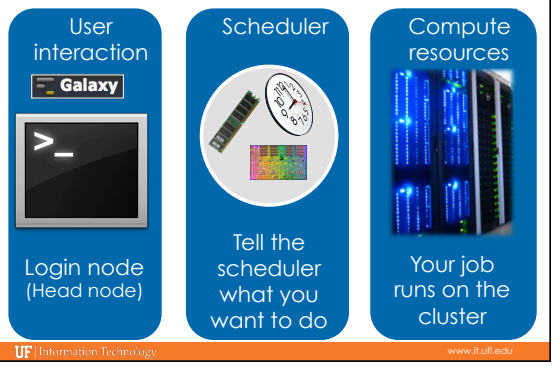
9/24/12

UF Research Computing

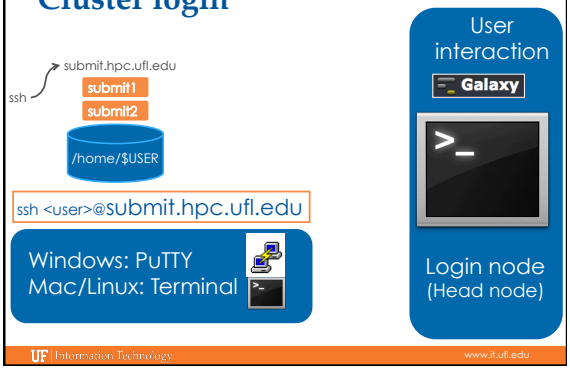


How do I get my jobs started?

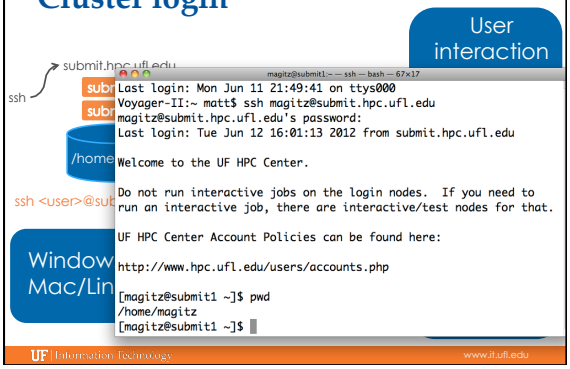
Cluster basics



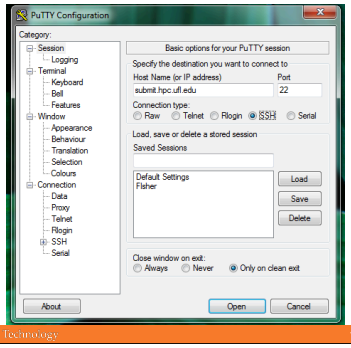
Cluster login



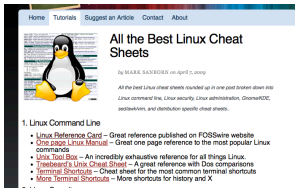
Cluster login



Logging in



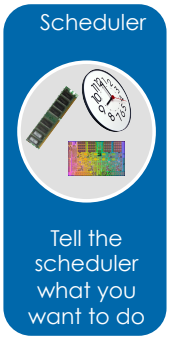
Linux Command Line



- Lots of online resources
  - Google: Linux cheat sheet
- Training sessions
- User manuals for applications

Scheduling a job

- Need to tell scheduler what you want to do
  - **How many CPUs** you want and how you want them grouped
  - **How much RAM** your job will use
  - **How long** your job will run
  - The commands that will be run



UF Research Computing

- Ordinary Shell Script

```
#!/bin/bash

date
module load test_app
test_app –i file.txt
```

Read the manual for your application

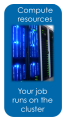
Commands typed on the command line can be put in a script.

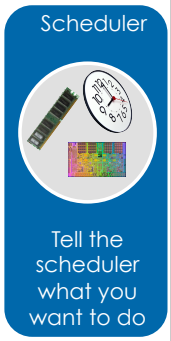
UF Research Computing

- Submission Script

```
#!/bin/bash
#
#PBS -N My_Job_Name
#PBS -M Joe_Shmoe@ufl.edu
#PBS -m abe
#PBS -o My_Job_Name.log
#PBS -j oe
#PBS -l nodes=1:ppn=1
#PBS -l walltime=00:05:00
#PBS –l pmem=900mb

cd $PBS_O_WORKDIR
date
module load test_app
test_app –i file.txt
```





Nodes and processors

```
#PBS –l nodes=1:ppn=4
#PBS –l nodes=2:ppn=8
```

Heterogeneous cluster

- There is a wide mix of nodes on the cluster
  - From 4 cores per node
  - To many with 12-16 cores
- The more ppn you ask for, the smaller the pool of nodes that can service your job
- Generally 16 is the most to request for :ppn=

RAM

#PBS –l pmem=900mb

- Lots to consider, but do your best at estimating RAM needed for job
- Over about 3GB of RAM, "costs" toward CPU allocation

Wasted RAM leads to idle CPUs and low job throughput

Processor equivalents

- Accounts for large RAM requests
- Average ~3GB RAM/core

1 core, 10GB RAM: ~3 PEs
1 core, 60GB RAM: ~18 PEs

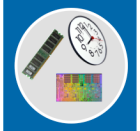
- Non-investors limit: 8 PEs
- Investor limits are based on PEs

Walltime

#PBS –l walltime=00:50:00

- Fairly straight forward
- As with all resource requests, accuracy helps ensure ***your*** jobs and all other jobs will run sooner

Scheduler



Tell the scheduler what you want to do

UF Research Computing

- Job Management
  - `qsub <file_name>`: job submission
  - `qstat –u <user>`: check queue status
  - `qdel <JOB_ID>`: job deletion

Lots of jobs

- You can script your job submission, ***BUT***:
  - How long will each job run?
    - Many short (<20 minutes) jobs are inefficient
      - Scheduling overhead

Pipettes only $1.99 each! *

* Plus $1.50 shipping per order

Would you order one at a time or place one order for 100?

Lots of jobs

```
#!/bin/bash
#PBS -l nodes=1:ppn=1
#PBS -l walltime=00:05:00
#PBS –l pmem=900mb

cd $PBS_O_WORKDIR
date
module load test_app
test_app –i file1.txt
test_app –i file2.txt
test_app –i file3.txt
…
test_app –i fileN.txt
```

You can very easily run multiple tasks in a single job script

Lots of jobs

- If you do submit lots of individual jobs:

  There is probably a better way…ask for help

  - **2,000-3,000** jobs maximum at a time
  - Add a ½ second pause between each job:

```
for i in $LIST
do
  qsub job_$i
  usleep 500
done
```

  - Consider how many jobs will run at once: what is your group's PE limit? Will they like you when you submit 2000 jobs?

Some helpful environment variables

- `$PBS_O_WORKDIR` : the directory where you typed qsub
- `$PBS_JOBID` : the unique job id: e.g. 24461774.torx.ufhpc
- `$TMPDIR` : temporary directory for each job on compute node's local disk, good for jobs with lots of I/O
- `$PBS_NP` : Number of processors for single node job, use this when starting a threaded application to tell it how many processors to use. Prevents needing to change in multiple places. E.g. nodes=1:ppn=4 , `bastn -num_threads $PBS_NP`
- `$PBS_JOBNAME` : Name your gave your job with #PBS –N

Let's look at some examples

Training Schedule

- ✓ Aug 28: Intro to UFHPC, getting started
- ✓ Sept 10: Modules, RHEL6 Transition, User Q&A
- ✓ Sept 17: The Linux/Unix Shell - An Introduction
- ✓ Sept 24: Running Jobs, Submission Scripts, Modules
- Oct 1: Galaxy Overview, The Basics
- Oct 8: NGS Data Techniques: General Methods and Tools
- Oct 15: NGS Data Techniques: Reference Based Mapping
- Oct 22: NGS Data Techniques: de Novo Assembly
- Oct 29: Phylogenetic Analyses
- Nov 5: Multiprocessing at the HPC Center
- Nov 12: Using Git and CMake to Organize and Drive Data Analysis Pipelines
- Nov 19: Introduction to GPU Nodes
- Nov 29: NGS Data Techniques: RNA-Seq
- Dec 3: NGS Data Techniques: Alternative Splicing Analysis

UF Research Computing

- Help and Support
  - Help Request Tickets
    - https://support.hpc.ufl.edu
    - For any kind of question or help requests
    - Searchable database of solutions
  - We are here to help!
    - support@hpc.ufl.edu

UF Research Computing

- Help and Support (Continued)
  - http://wiki.hpc.ufl.edu
    - Documents on hardware and software resources
    - Various user guides
    - Many sample submission scripts
  - http://hpc.ufl.edu/support
    - Frequently Asked Questions
    - Account set up and maintenance