Getting Started with CHPC Utah State University

Martin Čuma, Brett Milash, Anita Orendt

Center for High Performance Computing

CHPC's Mission

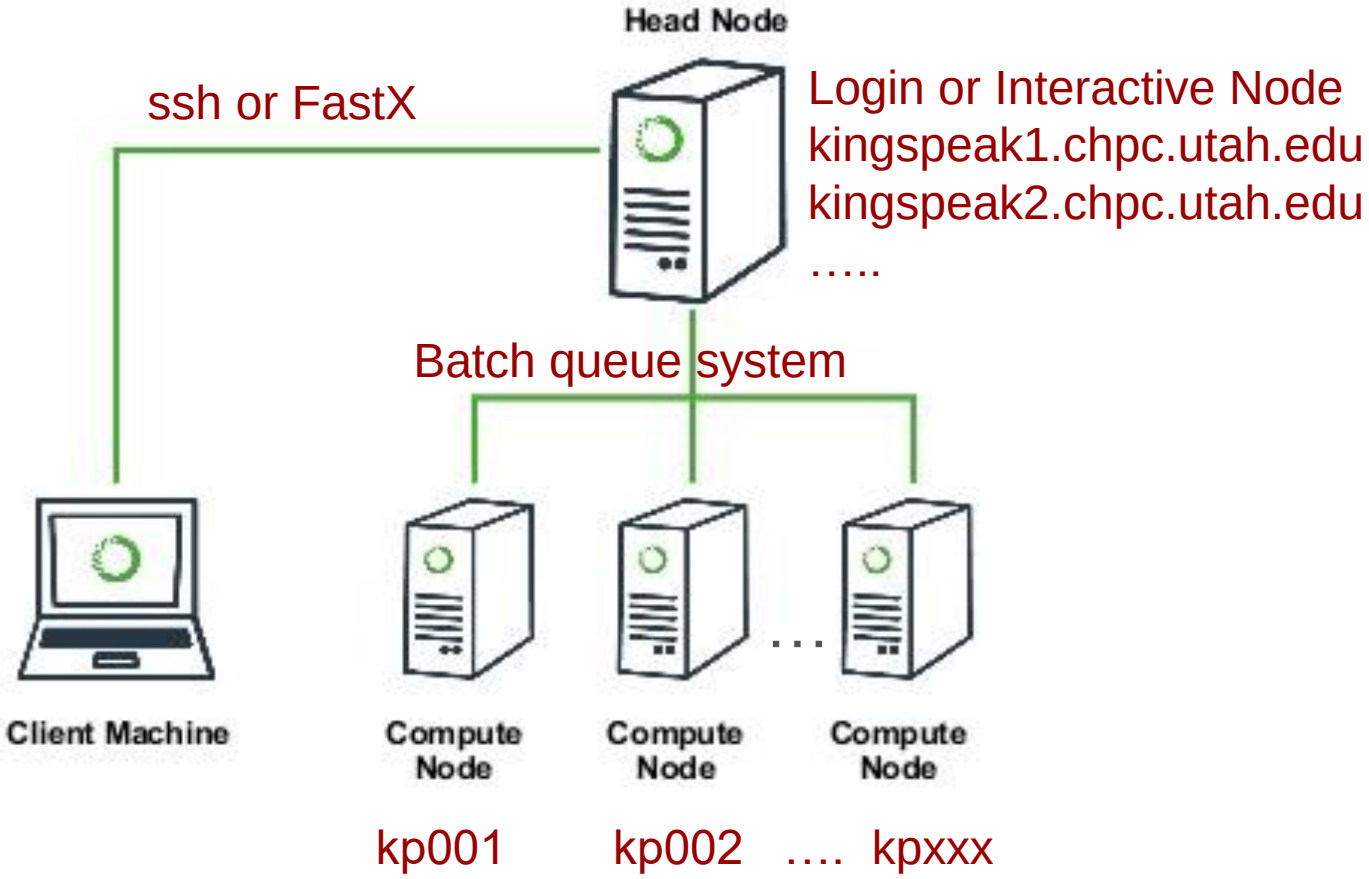
In addition to deploying and operating high performance computational resources and providing advanced user support and training, CHPC serves as an expert team to broadly **support** the increasingly **diverse research computing needs** on campus. These needs include support for big data, big data movement, data analytics, security, virtual machines, Windows science application servers, protected environments for data mining and analysis of protected health information, and advanced networking.

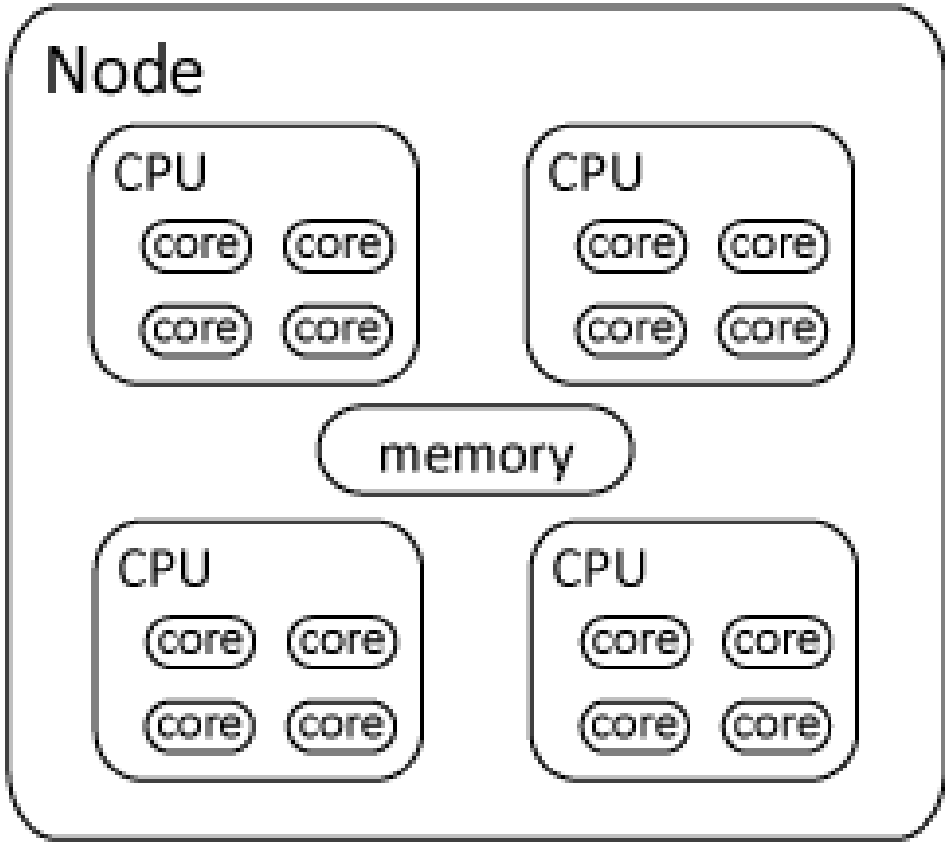
Why Use a Cluster?

- “larger” computing systems
  - More cores, more memory tp use on a single job
  - Run larger jobs/models/analyses
- Access to more resources
  - Run many jobs simultaneously
  - Run a single job on multiple computers
- Many programs need linux environment to run

CONTINUUM ANALYTICS

Cluster Architecture Diagram




Node
CPU
core
core
core
core
CPU
core
core
core
core
memory
CPU
core
core
core
core
CPU
core
core
core
core

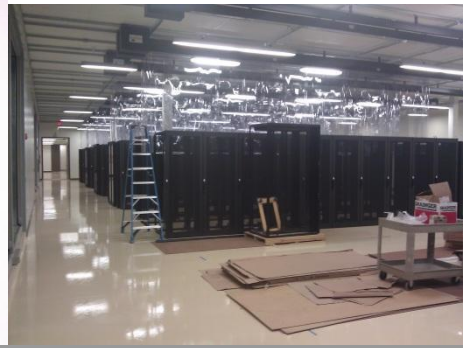
Sources of Useful Information

- Getting Started Guide
  - https://www.chpc.utah.edu/documentation/gettingstarted.php
- CHPC policies
  - https://www.chpc.utah.edu/documentation/policies/index.php
- Cluster Usage Guides
  - https://www.chpc.utah.edu/documentation/guides/index.php
- Application Documentation
  - https://www.chpc.utah.edu/documentation/software/index.php
- Programming Guide
  - https://www.chpc.utah.edu/documentation/ProgrammingGuide.php
- How to Videos
  - https://www.chpc.utah.edu/documentation/videos/index.php

Downtown Data Center

- Came online Spring 2012
- CHPC completed move to DDC Spring 2013
- Shared with enterprise (academic/hospital) groups
- 92 racks and 1.2MW of power with upgrade path to add capacity for research computing
- Metro optical ring connecting campus, data center, & internet2
- 24/7/365 facility



CHPC Resources & Services

- ***Computational Clusters*** – Notchpeak, Kingspeak, Lonepeak, Ember, Ash, Tangent
- ***Storage*** – home, group, and scratch storage along with tape backup and archive storage options
- ***Windows Servers*** – mainly statistics usage and windows only applications
- ***Virtual Machines*** – for needs not met with cluster and windows server
- ***Protected Environment*** –computational cluster Redwood, storage, VMs, and Windows Server
- ***Networking Support*** – support compute environment; work with researchers on data movement
- ***User Support*** – assistance with use of resources; installation of applications; training sessions

Windows Statistics Server

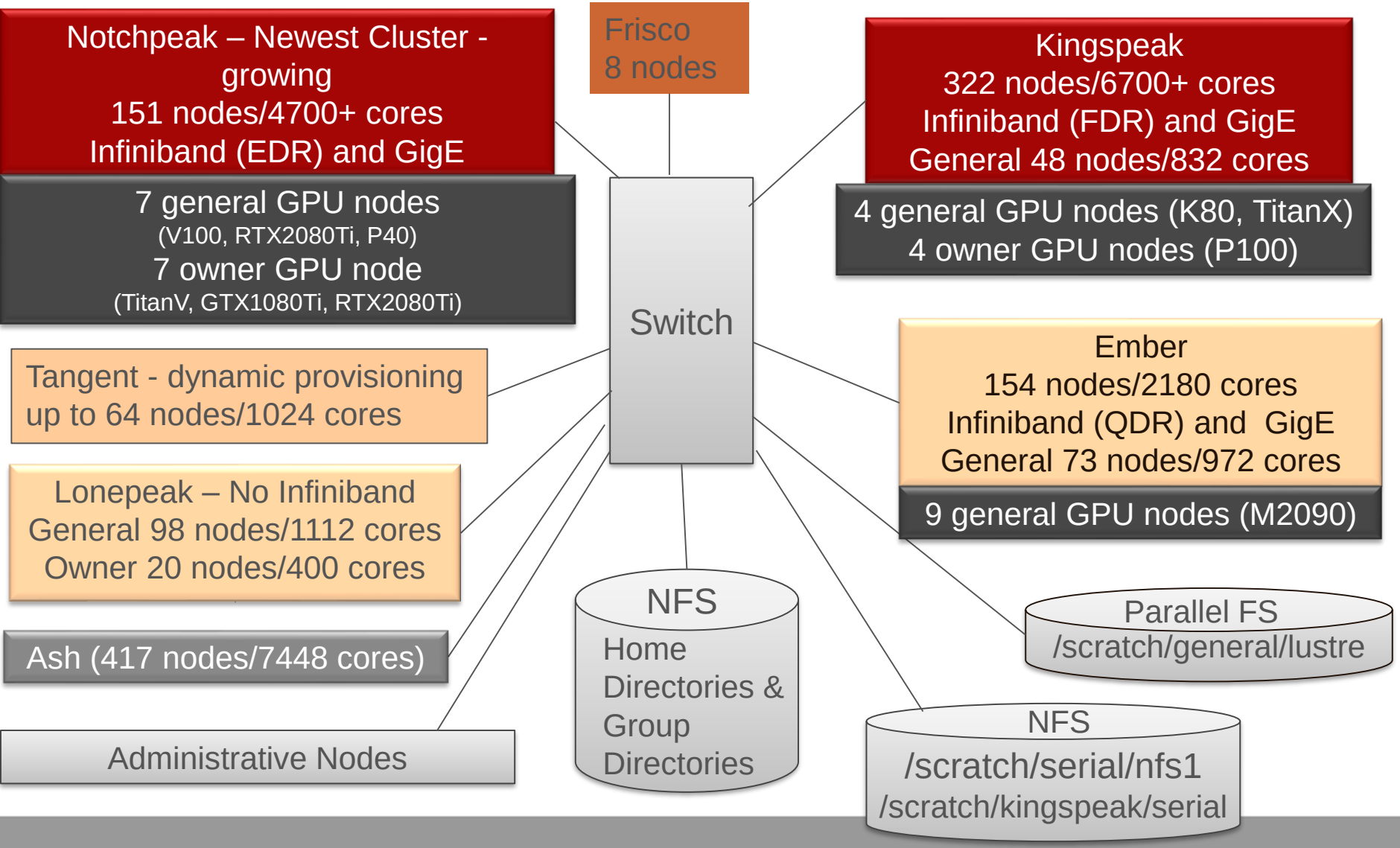
- Beehive – refreshed 2019
  - 48 physical cores, 512TB memory
- Presently has the following software installed
  - SAS 9.4 with text miner
  - SPSS
  - R
  - STATA
  - Mathematica
  - Matlab
- ***If you need other software, please contact us to discuss***

Virtual Machine Farm

- For needs and applications that do not fit in compute cluster or Windows server
- Multiple VM servers with failover – hardware refreshed 2019
- VM storage
- Have community mysql/mssql VMs, git repositories, web servers, etc
- New – user VMs (not use of community ones) will have a cost, both for the VM and for any customization needed.

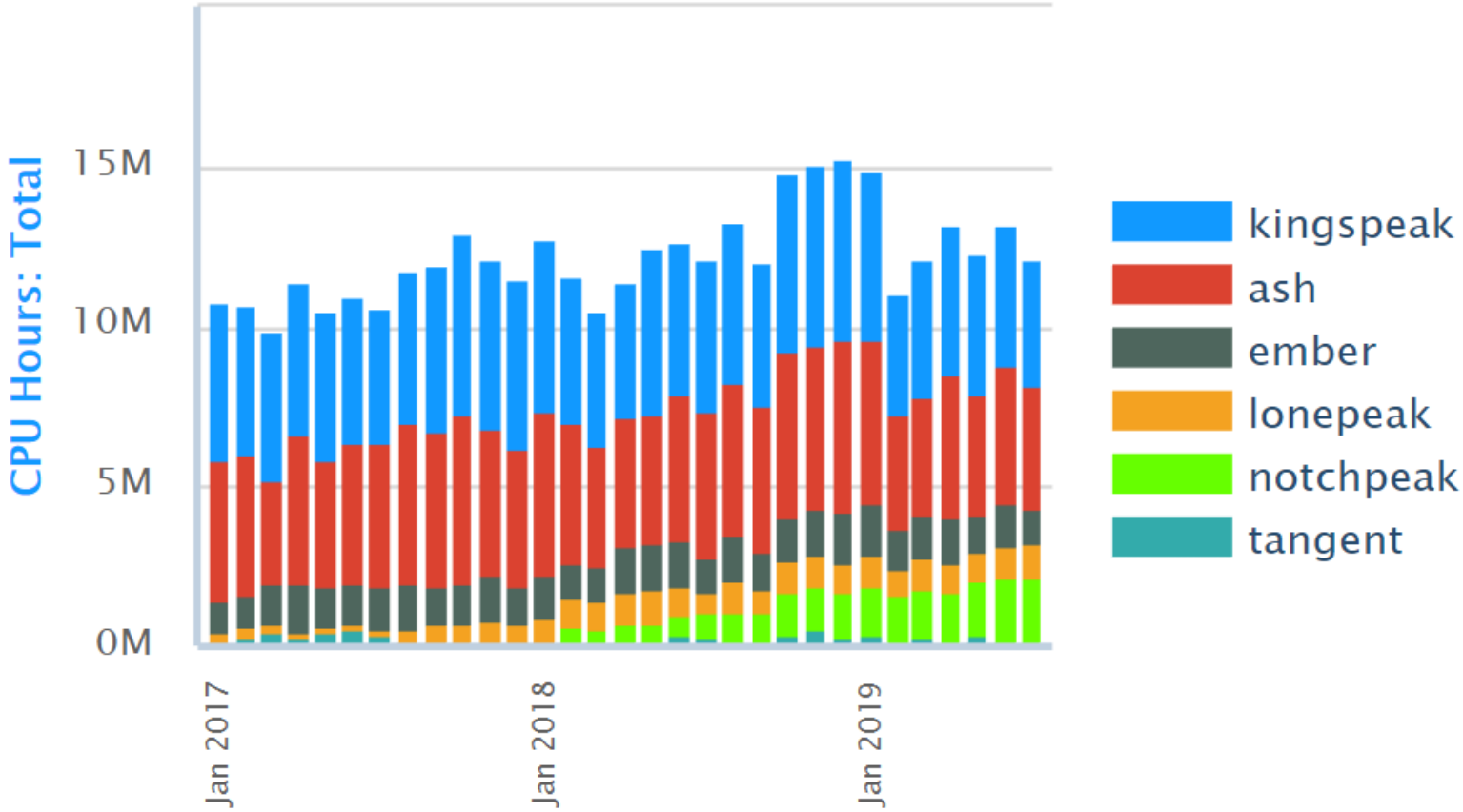
Blocks	RAM (GB)	Cores	Storage (GB)	Price
1	4	2	50	$475
2	8	2	100	$705
4	16	4	200	$1175
8	32	8	400	$2115
16	64	8	800	$3995

Additional VM storage available, in 100GB increments, at a cost of $850/TB.


Notchpeak – Newest Cluster - growing
151 nodes/4700+ cores
Infiniband (EDR) and GigE
7 general GPU nodes
(V100, RTX2080Ti, P40)
7 owner GPU node
(TitanV, GTX1080Ti, RTX2080Ti)
Frisco
8 nodes
Kingspeak
322 nodes/6700+ cores
Infiniband (FDR) and GigE
General 48 nodes/832 cores
4 general GPU nodes (K80, TitanX)
4 owner GPU nodes (P100)
Switch
Tangent - dynamic provisioning
up to 64 nodes/1024 cores
Ember
154 nodes/2180 cores
Infiniband (QDR) and GigE
General 73 nodes/972 cores
9 general GPU nodes (M2090)
Lonepeak – No Infiniband
General 98 nodes/1112 cores
Owner 20 nodes/400 cores
NFS
Home Directories & Group Directories
Parallel FS
/scratch/general/lustre
Ash (417 nodes/7448 cores)
Administrative Nodes
NFS
/scratch/serial/nfs1
/scratch/kingspeak/serial

Core Hour Usage

Over 134M core hours provided in 2017; 160M in 2018; 84M first half 2019
Over this time period – over 1100 users from over 315 groups ran 10 M jobs



CHPC Clusters - Condominium Model

- Notchpeak, Kingspeak
  - General resources – Allocation process for cycles
  - Out of allocation – freecycle allowed (with preemption)
- Owner nodes – new purchased added to notchpeak
  - Users from all groups allowed on as owner-guest when not in use (preemption)
  - Found on notchpeak, kingspeak, ember, lonepeak
  - 40 core Intel XeonSP(Cascade) @ 2.1 GHz cpu speed, 192 GB RAM, 2TB local hard drive, 5 year warranty, EDR IB connectivity @ ~$6000/node; with 384 GB RAM ~$7675/node
- Ember, Lonepeak
  - no allocation and no preemption
- Tangent
  - dynamically provisioned cluster resource; no allocation and no preemption
- Ash – Owner cluster
  - All users have guest access as smithp-guest (preemption)

CHPC Provides Core Infrastructure

- Physical needs (racks, power, cooling)
- Core ethernet and IB fabric; IB to ethernet bridging
- Login/head/management nodes
- Scheduling, batch and allocation systems
- HPC scratch space
- Some application and licensing costs
- High speed data transfer nodes (DTNs)
- 'Science DMZ' network
- CHPC Staff

Getting a CHPC Account

- CHPC uses the UU campus ID (uNID) and password
- For USU faculty members:
  - Open a ticket (helpdesk@chpc.utah.edu) requesting an UU affiliate ID, providing full name, date of birth, email address
  - Do the password reset as instructed
  - Complete the CHPC account application
    - https://www.chpc.utah.edu/role/user/account_request.php
  - CHPC will enable you to be able to do requests for your group members via UU HR (will need full name, date of birth, email address)
    - https://www.hr.utah.edu/forms/affiliate.php
- Students – work with your research advisor

uNIDs can be requested for up to 3 years – will need to renew if needed for longer

Login Nodes

- Login or interactive nodes with each cluster
- Interactive nodes only used for short compiles, editing and very short test runs
- ***No more than 15 minutes and no jobs of any length that make heavy use of cpu or memory!***
- Have script which watches running processes and notifies users when in violation of the acceptable usage policy

Arbiter Script

- Script that monitors usage of login nodes (including frisco nodes)
- Sets two levels
  - "hard limit" – uses cgroups to set maximum usage limits
  - "threshold limit" – limit above which user tracked and possibly penalized
- Penalized (max usage limit lowered) after 15 cpu-minutes usage on cluster logins or 120 cluster-minutes on frisco nodes
- Emails user, with graph of usage over time, when entering penalty state

If you get an email from arbiter and want to understand how to change your usage to avoid penalty, contact us

Accessing Login Nodes

- Use FastX from Mac, Windows, or Linux desktops -- preferred
  - https://www.chpc.utah.edu/documentation/software/fastx2.php
- Alternatively:
  - From windows need ssh client
    - PuTTY http://www.chiark.greenend.org.uk/~sgtatham/putty/
    - Xshell http://www.netsarang.com/products/xsh_overview.html
    - Mobaxterm https://mobaxterm.mobatek.net/
  - With putty also need X forwarding application
    - Xming http://www.straightrunning.com/XmingNotes/
      - Look for “mesa” version
  - From mac/linux – use terminal ssh (with –X for X-forwarding)

FastX – Tool for Remote X

- https://www.starnet.com/fastx
- Used to interact with remote linux systems graphically in much more efficient and effective way then simple X forwarding
- Persistence – you can disconnect without closing session, lets you resume sessions from other devices
- Server on all interactive nodes as well as the frisco nodes; some servers have graphics cards and support OpenGL
- Clients for windows, mac and linux; can be installed on both university and personal desktops
- Has web based client option – which we will use today

FastX

- For FastX – see “To Use” section of documentation at https://www.chpc.utah.edu/documentation/software/fastx2.php
- Download client following directions on page
- Do install
- Start program
- Set host to kingspeak1.chpc.utah.edu OR kingspeak2.chpc.utah.edu OR other interactive node OR one of the frisco nodes (frisco1-frisco8.chpc.utah.edu)

Ex 1: Login to CHPC via FastX Web Client

- Open web browser to: http://notchpeak1.chpc.utah.edu:3000
- Enter your username (uNID) & password and hit “Log In” button
- Hit the “Launch Session” button
- Click on “xterm”, then hit the “Launch” button

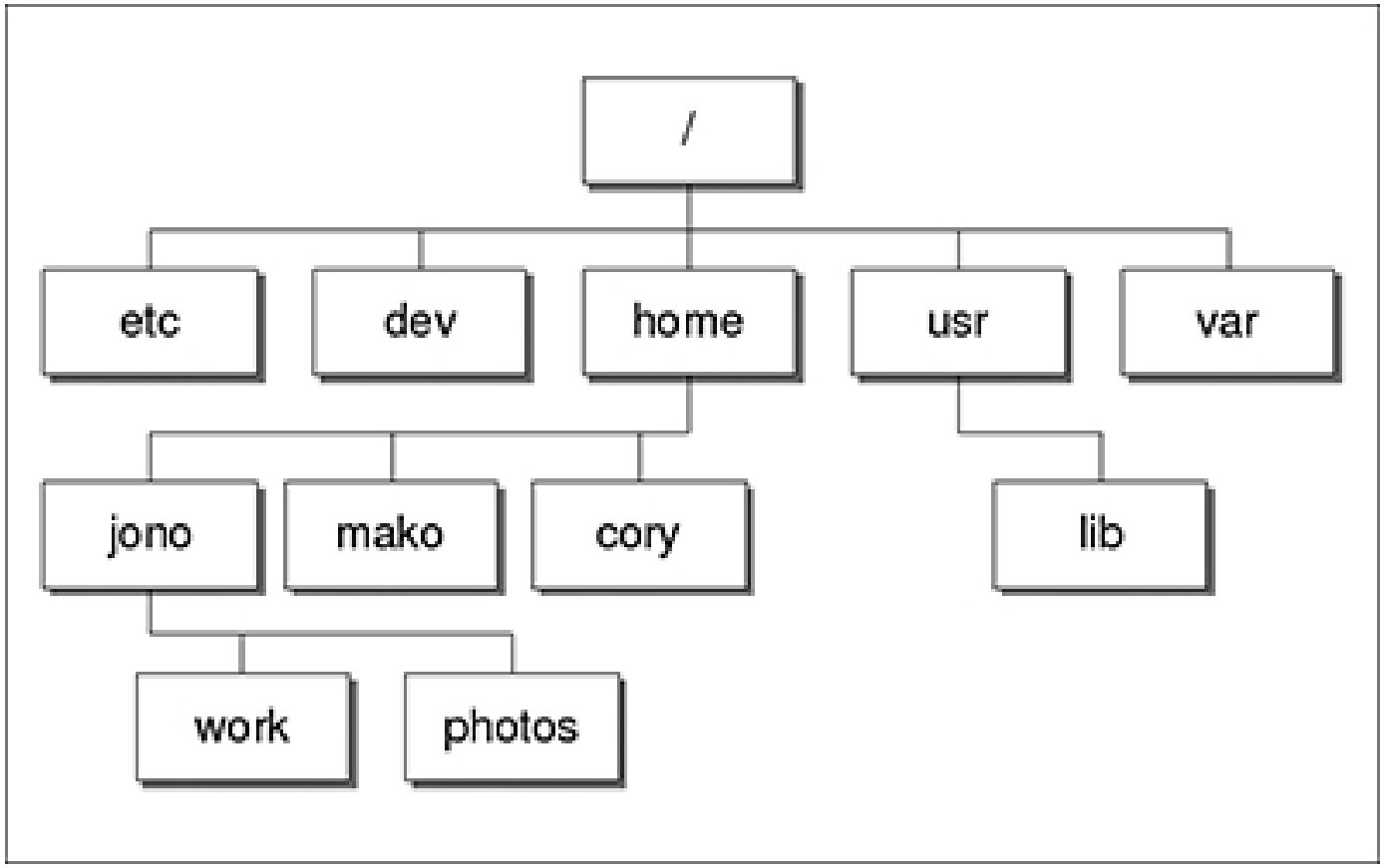
Learning Linux

- CHPC linux presentations
  - https://www.chpc.utah.edu/presentations/IntroLinux3parts.php
- Other sources
  - Cornell Virtual worshops
    https://cvw.cac.cornell.edu/Linux
  - Carpenties
    - Software, https://software-carpentry.org/
    - HPC, https://hpc-carpentry.github.io/

Storage Options

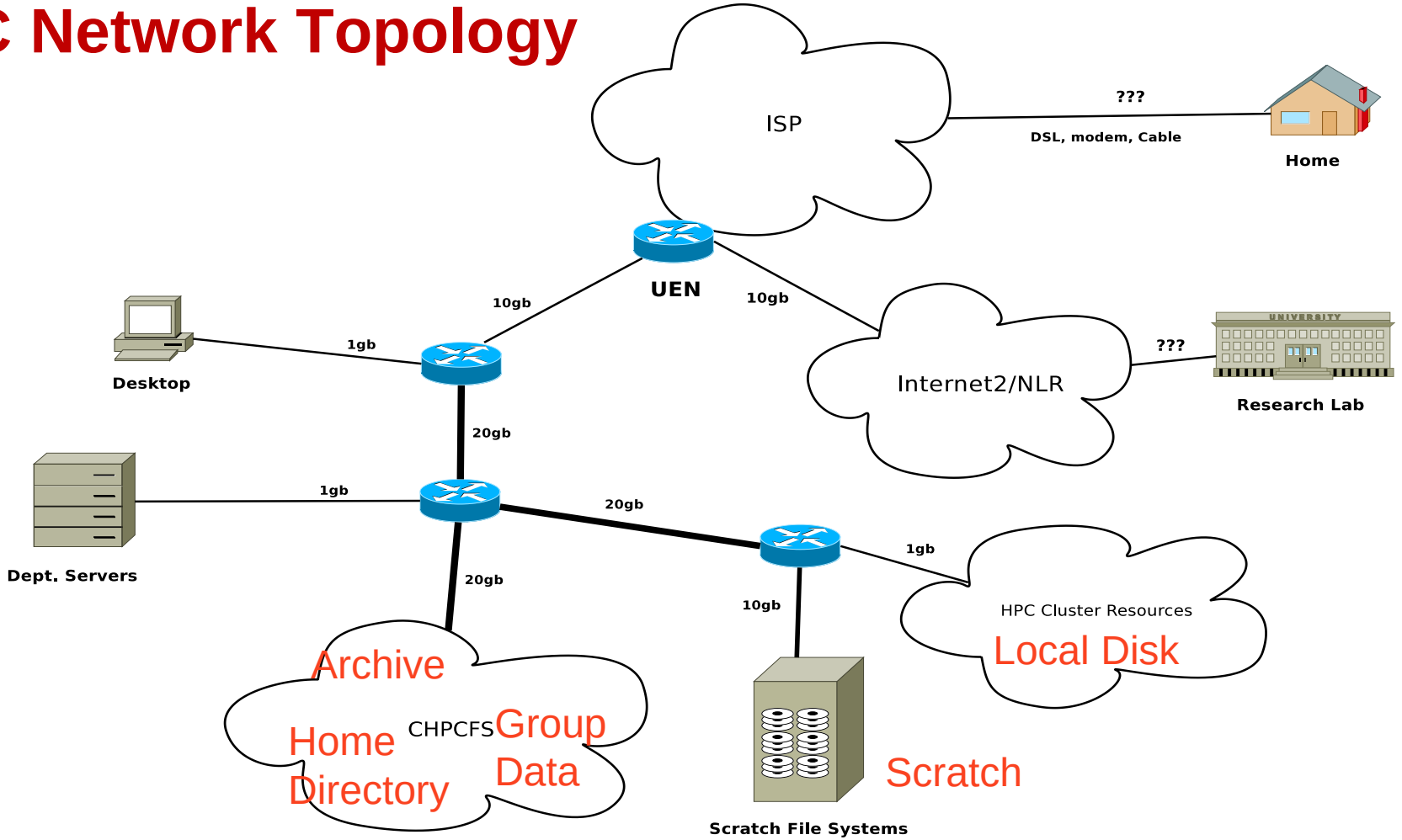
- *Home Directories* -- /uufs/chpc.utah.edu/common/home/<uNID>
  - Home directories 50 GB, not backed up
  - Groups can larger home directory per group at $750/TB for remaining ~3 year warranty; this comes with backup (nightly incremental, weekly full, 2 week retention)
  - Compellent solution - two disk based copies mirrored for HA
- *Group Level File Systems*
  - Group space @$150/TB
- *Network Mounted Scratch File Systems*
  - For use by all users; scrubbed of files older than 60 days
  - 700 TB Lustre Parallel file system (/scratch/general/lustre)
  - 175 TB NFS mounted file system (/scratch/kingspeak/serial)
  - 600 TB NFS mounted file system (/scratch/serial/nfs1)
- *Disk Based Archive Storage*
  - Archive at$150/TB

Directory Structure



At CHPC: Top level for user directories is /uufs/chpc.utah.edu/common/home/$USER

CHPC Network Topology



File Systems

- Access speed based on connectivity
- Local disk fastest – local to each node; varies in size
  - /scratch/local
- Network mounted scratch file systems
  - /scratch/general/lustre
  - /scratch/kingspeak/serial
  - /scratch/serial/nfs1
  - home directories and group spaces (***don't use for large i/o!***)

Remember NFS mounted spaces – including file systems for group spaces – are a shared resource!

Storage Resources for USU

- All users have a 50 GB home directory unless group has purchased additional home
  - /uufs/chpc.utah.edu/common/home/uNID
  - If default 50 GB NOT BACKED UP
  - If purchased by group – backed up nightly incremental, weekly full, 2 week retention window
- Some groups have purchased group space
  - /uufs/chpc.utah.edu/common/home/pi_grp
  - Generally NOT BACKED UP
- Scratch storage – open to all users
  - No quotas
  - NOT BACKED UP
  - All files not accessed in 60 days are AUTOMATICALLY deleted

Querying disk usage

- **mydiskquota** -- Tool to see what storage you have and how much space you are using
- **du** – Linux command to get disk usage
  - du –h
  - du -hs
- **baobab** – GUI disk usage analyzer

Moving Files to/from CHPC

- Multiple options
  - https://www.chpc.utah.edu/documentation/data_services.php
- Depends on source and size of data

- Smaller files to/from personal computers
  - Command line
    - scp (-C)
    - rsync – Linux, persistent (`rsync -azv from to`)
  - Mount CHPC space on local machine
    - Must first connect to UU VPN – vpnaccess.utah.edu
  - GUI application to install on local machine
    - WinSCP (Windows), Cyberduck (Mac) – not persistent
- Large files
  - Use data transfer nodes (bypass UU campus firewall)
    `airplane0[1-4]-dmz.chpc.utah.edu`
    `dtn0[1,4]-dmz.chpc.utah.edu`
  - Linux/mac can use rsync
  - Set up Globus end point on your personal machine
    https://www.chpc.utah.edu/documentation/software/globus.php

Ex 2: Getting Files for today

- For files for today's training – on github
  - git clone https://github.com/CHPC-UofU/USU-lectures.git

Login scripts

- CHPC provides login scripts ("dot" files) when creating account for both tcsh and bash shells
- These files set the environment so that applications are found, batch commands work – ***Do not remove***
- Choose shell at account creation – can change at www.chpc.utah.edu (sign in, select edit profile)
- Four files: .bashrc, .tcshrc, .custom.sh, .custom.csh
  - **The first two should not be edited**
  - **The second two is where to add custom module loads**
- Will automatically execute an .aliases file if it exists

CHPC Uses Modules for Setting Environment

- Modules are a way of managing the user environment in an interactive session or a batch job
- CHPC provides login scripts ("dot" files) when creating account for both tcsh and bash shells
- These files set the environment so that applications are found, batch commands work – ***Do not remove or edit!***
- **https://www.chpc.utah.edu/documentation/software/modules.php** for information

Why Modules

- Modules lets users dynamically change the environment – including easily adding and removing directories needed for a given task from $PATH etc – without needing to log out and back in
- Easy to switch between version of a package or application – again without having to start a new session
- Useful when packages have conflicts in their environment settings

All accounts automatically use modules –

- This is done via the login scripts CHPC provides all accounts, even if you have older dot files
- CHPC uses modules to set up environments upon login: chpc/1.0

Module Documentation at CHPC

- https://www.chpc.utah.edu/documentation/software/modules.php
- https://www.chpc.utah.edu/documentation/software/modules-advanced.php
- Video -- https://www.youtube.com/watch?v=Cu6C5lNLDAY

We make use of TACC's LMOD

- https://www.tacc.utexas.edu/research-development/tacc-projects/lmod
- LUA based

Recommendations & Helpful Hints

- Keep both the cshell and bash versions of provided login scripts in your home directory
- DO NOT make changes in the .tcshrc and .bashrc
- Use the .custom.csh/.custom.sh to load modules for programs regularly used in ssh sessions
- Use .aliases file to create aliases – but do not set other environment variables in this file; if this file exists it will be sourced during login
- The software database mentions which installations have modules – if there is one you would like us to create, let us know!

Basic Module commands

- **module** - shows the list of module commands
- **module load <name>** - loads module name (shortcut: **ml <name>**)
- **module unload <name>** - unloads module name (**ml -<name>**)
- **module avail** - shows a list of "available" modules (**ml av**)
- **module list** - shows a list of loaded modules (**ml**)
- **module help** - prints help for the module command
- **module help <name>** - prints help for module
- **module show <name>** - prints the module file
- **module purge** - unload all modules
- **module reset system** – resets to system default (only chpc module loaded)
- **module swap <name1> <name2>** - swaps between two modules

CHPC Module Organization

- Core
  - Contains modules for applications independent of both the compiler and MPI implementation
- Compiler
  - Contains modules for applications dependent on a compiler (& version) but not on a MPI implementation
- MPI
  - Contains modules for applications dependent on both a compiler and a MPI implementation

Modules themselves are named by application name/version

Other Information

- Cannot have multiple compilers loaded
  - If you have intel loaded, and load any gcc it will unload intel
  - As a result – we no longer have compiler tag as part of the module name in libraries and applications that are compiler dependent (ex – mpich, openmpi, netcdf, fftw)
- Parallel versions of boost, HDF5 have separate modules
  - hdf5 for module for serial build, phdf5 for module for parallel build
  - boost for serial, pboost for parallel

Default, aliases, and hidden modules

- For some applications have a default module – one that is installed if you do not provide a specific version
  - typically the latest version is specified to be the default
- For some modules, especially those with long version names, there is also an alias defined
  - **ml intel/18** loads the default 2018 intel version (2018.1.193)
  - **ml intel/18.0** loads the 2018.1.163 version
- We have depreciated older installations and their modules so some have been hidden
  - **module --show_hidden avail**

Module avail command

- **module avail** shows all modules available based on already loaded module
  - This also marks default (D), already loaded (L), gpu specific (g) and aliases
- Some modules are dependent on other modules based on organization
  - these modules are not listed unless the modules they depend on are loaded

Module spider command

- **module spider** shows all modules, including modules that aren't available
- Use **module spider <string>** to see a subset of modules with **string** in name, and how to either load the module or how to get more detailed information on how to load

Module show command

- Format **module show modulename/version**
- Shows you the content of the module file
- This is useful if there is information on running the program included in the module

Software on Clusters

- Have a variety of compliers, mpi packages, math libraries and applications installed
- Some licensing restrictions may apply
- If you need a package we do not currently have installed – ask us!
- Currently we place installations at:
  - /uufs/chpc.utah.edu/sys/installdir
- Have a searchable application database
  - https://www.chpc.utah.edu/software/chpc/

Ex 3: Exploring modules

Batch System Information

- Used to access compute nodes which must be used for any extensive use
- Use SLURM – Simple Linux Utility for Resource Management
- https://www.chpc.utah.edu/documentation/software/slurm.php

Basic SLURM commands

- **sinfo** - shows partition/node state
- **sbatch <scriptname>** - launches a batch script
- **squeue** - shows all jobs in the queue
  - **squeue -u <username>** - shows only your jobs
- **scancel <jobid>** - cancels a job

Notes:

For **sinfo, squeue** – can add **–M all** to see all clusters using given slurm installation (notchpeak, kingspeak, lonepeak, ember, ash)

Can also add **–M cluster** OR use full path /uufs/<cluster>.peaks/sys/pkg/slurm/std/bin/<command> to look at the queue, or submit or cancel jobs for a different cluster (ember is .arches)

Some Useful Aliases

- Bash to add to .aliases file:

```
alias si="sinfo -o \"%20P %5D %14F %8z %10m %10d %11l %16f %N\""
alias si2="sinfo -o \"%20P %5D %6t %8z %10m %10d %11l %16f %N\""
alias sq="squeue -o \"%8i %12j %4t %10u %20q %20a %10g %20P %10Q %5D %11l %11L %R\""
```

- Tcsh to add to .aliases file:

```
alias si 'sinfo -o "%20P %5D %14F %8z %10m %11l %16f %N"'
alias si2 'sinfo -o "%20P %5D %6t %8z %10m %10d %11l %N"'
alias sq 'squeue -o "%8i %12j %4t %10u %20q %20a %10g %20P %10Q %5D %11l %11L %R"‘
```

Can add **–M** to **si** and **sq** also

You can find these on the CHPC Slurm page
https://www.chpc.utah.edu/documentation/software/slurm.php#aliases

SLURM Batch Directives

#SBATCH --time 1:00:00 ← wall time of a job (or -t)

#SBATCH --partition=name ← partition to use (or -p)

#SBATCH --account=name ← account to use (or -A)

#SBATCH --nodes=2 ← number of nodes (or -N)

#SBATCH --ntasks 12 ← total number of tasks (or -n)

#SBATCH --mail-type=FAIL,BEGIN,END ← events on which to send email

#SBATCH --mail-user=name@example.com ← email address to use

#SBATCH -o slurm-%j.out-%N ← name for stdout; %j is job#, %N node

#SBATCH -e slurm-%j.err-%N ← name for stderr; %j is job#, %N node

#SBATCH --constraint “C20” ← can use features given for nodes (or -C)

Accounts and Partitions

- You need to specify an **account** and a **partition** to run jobs
- You can see a list of partitions using the sinfo command
- For general allocation usage the partition is the cluster name
- If no allocation (or out of allocation) use *clustername*-freecycle for partition
- Your account is typically your PI's name (e.g., if your PI is Baggins, use the "baggins" account) – there are a few exceptions!
- Private node accounts and partition have the same name – PI last name with cluster abbreviation, e.g., baggins-kp, baggins-em, etc
- Private nodes can be used as a guest using the "owner-guest" account and the *cluster*-guest partition
- Remember general nodes on notchpeak and kingspeak need allocation; general nodes on ember, lonepeak and tangent are open to all users without allocation

More on Accounts

Allocations and node ownership status	What resource(s) are available
No general allocation, no owner nodes	Unallocated general nodes Allocated general nodes in freecycle mode - not recommended Guest access on owner nodes
General allocation, no owner nodes	Unallocated general nodes Allocated general nodes Guest access on owner nodes
Group owner nodes, no general allocation	Unallocated general nodes Allocated general nodes in freecycle mode - not recommended Group owned nodes Guest access on owner nodes of other groups
Group owner node, general allocation	Unallocated general nodes Allocated general nodes Group owned nodes Guest access on owner nodes of other groups

See https://www.chpc.utah.edu/documentation/guides/index.php#parts

Query your accounts/partitions

~]$ **sacctmgr -p show assoc user=u0028729 format=cluster,account,partition,qos**

Cluster|Account|Partition|QOS|

notchpeak|chpc||notchpeak|
kingspeak|kingspeak-gpu||kingspeak-gpu|
ember|ember-gpu||ember-gpu|
ash|smithp-guest||ash-guest,ash-guest-res|
lonepeak|chpc||lonepeak|
kingspeak|chpc||kingspeak|
lonepeak|owner-guest||lonepeak-guest|
ember|owner-guest||ember-guest|
kingspeak|owner-guest||kingspeak-guest|
kingspeak|owner-gpu-guest||kingspeak-gpu-guest|
ember|chpc||ember|

Note that partition field is empty – for the most part partition and qos are paired

Query your allocation

~]$ myallocation

You have a general allocation on kingspeak. Account: chpc, Partition: kingspeak
You can use preemptable mode on kingspeak. Account: owner-guest, Partition: kingspeak-guest
You can use preemptable GPU mode on kingspeak. Account: owner-gpu-guest, Partition: kingspeak-gpu-guest
You have a GPU allocation on kingspeak. Account: kingspeak-gpu, Partition: kingspeak-gpu
You have a general allocation on notchpeak. Account: chpc, Partition: notchpeak
You can use preemptable GPU mode on notchpeak. Account: owner-gpu-guest, Partition: notchpeak-gpu-guest
You can use preemptable mode on notchpeak. Account: owner-guest, Partition: notchpeak-guest
You have a GPU allocation on notchpeak. Account: notchpeak-gpu, Partition: notchpeak-gpu
You have a general allocation on ember. Account: chpc, Partition: ember
You can use preemptable mode on ember. Account: owner-guest, Partition: ember-guest
You have a GPU allocation on ember. Account: ember-gpu, Partition: ember-gpu
You have a general allocation on lonepeak. Account: chpc, Partition: lonepeak
You can use preemptable mode on lonepeak. Account: owner-guest, Partition: lonepeak-guest

Ex 4: Exploring slurm, accounts, partitions

Compute Resources for USU

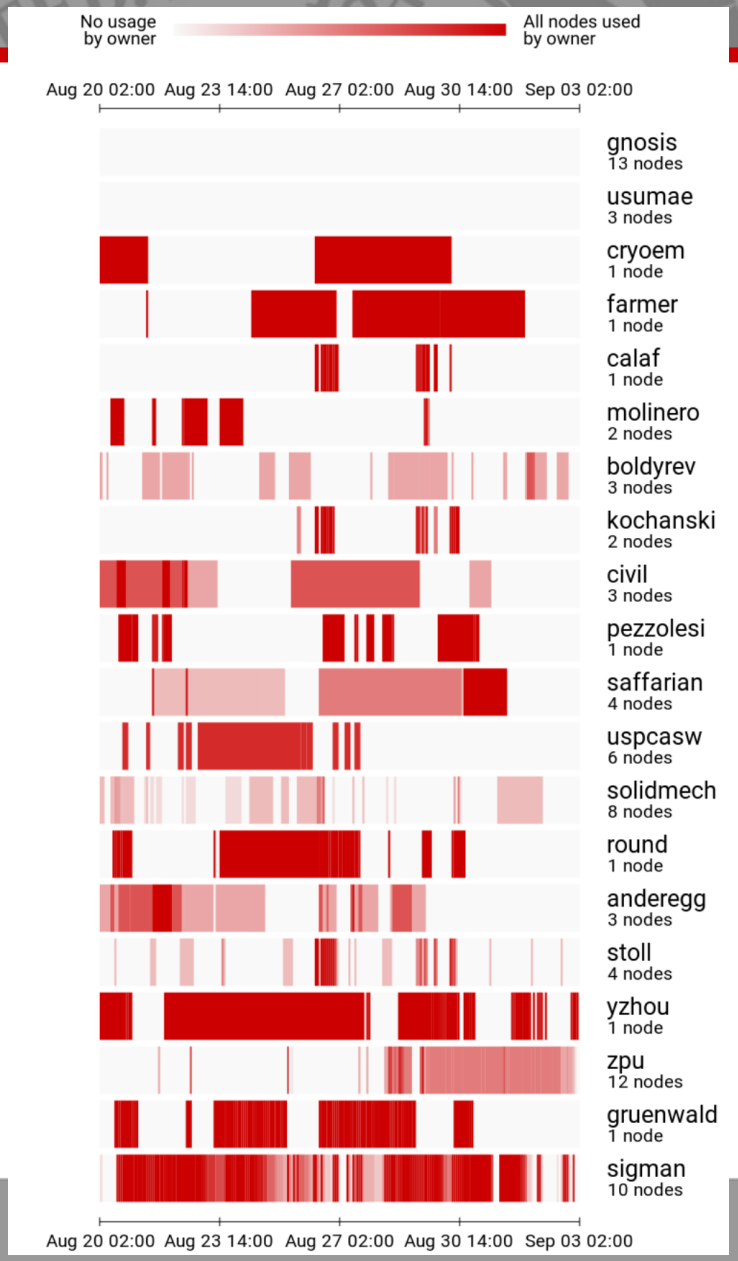
- On Ember – USU owns 18 nodes – open to all USU users
  - Nodes are old, probably not your first choice
- All users can use the general nodes of lonepeak, ember, and tangent
- All groups can apply for allocation to use the general (non-owner) nodes of notchpeak and kingspeak
- All users can use of idle owner nodes in guest mode
- All users can use of the non-owner GPU nodes (request access)
- All users can use the notchpeak-shared-short nodes
- Any group can purchase nodes at hardware cost
  - Some USU groups have done this
- On www.chpc.utah.edu – can see current usage

Allocation

- General Allocation Process Information
  - https://www.chpc.utah.edu/documentation/policies/1.4AllocationPolicies.php
- Regular allocation form
  - https://www.chpc.utah.edu/apps/profile/allocation_form.php
  - Requests due Sept 1, Dec 1, Mar 1, and Jun 1
  - Allocation in core hours – max is 200,000 core hours per quarter
- Quick allocation
  - https://www.chpc.utah.edu/apps/profile/allocation_quick_form.php
- Check usage -- https://www.chpc.utah.edu/usage/cluster/current-project-general.php
- Simplified quick allocation requests & general allocation requests for up to 20,000 core-hours per quarter

Owner/Owner-guest

- CHPC provides heat maps of usage of owner nodes by the owner over last two weeks
- https://www.chpc.utah.edu/usage/constraints/
- Use information provided to target specific owner partitions with use of constraints (more later)



Node Sharing

- Use the shared partition for a given set of nodes (using normal account for that partition)
  - notchpeak* 4 4/0/0/4 2:16:2 95000 1800000 3-00:00:00 chpc,c32,m96 notch[005-008]
  - notchpeak-shared 4 4/0/0/4 2:16:2 95000 1800000 3-00:00:00 chpc,c32,m96 notch[005-008]
  - molinero-np 2 2/0/0/2 2:16:2 95000 1800000 3-00:00:00 molinero,c32,m96 notch[024-025]
  - molinero-shared-np 2 2/0/0/2 2:16:2 95000 1800000 3-00:00:00 molinero,c32,m96 notch[024-025]
- In script:
  - #SBATCH --partition=cluster-shared
  - #SBATCH --ntasks=2
  - #SBATCH --mem=32G
- If there is no memory directive used the default is that 2G/core will be allocated to the job.
- Allocation usage of a shared job is based on the percentage of the cores and the memory used, whichever is higher

https://www.chpc.utah.edu/documentation/software/node-sharing.php

notchpeak-shared-short partition

- Account/partition devoted to interactive jobs
- Two 64 core, 256 GB AMD Zen CPU based nodes
- Max walltime 8 hours
- Max 32 tasks, 64 GB RAM per user
- Instant job allocation = interactivity of the job
- Good for testing, debugging, etc
- `srun -n 1 -N 1 -A notchpeak-shared-short -p notchpeak-shared-short -t 8:00:00 --pty /bin/bash -l`

SLURM Environment Variables

- Depends on SLURM Batch Directives used
- Can get them for a given set of directives by using the env command inside a script (or in a srun session).
- Some useful environment variables:
  - $SLURM_JOBID
  - $SLURM_SUBMIT_DIR
  - $SLURM_NNODES
  - $SLURM_NTASKS

Basic SLURM script flow

1. Set up the #SBATCH directives for the scheduler to request resources for job
2. Set up the working environment, by loading appropriate modules
3. If necessary, add any additional libraries or programs to $PATH and $LD_LIBRARY_PATH, or set other environment needs
4. Set up temporary/scratch directories if needed
5. Switch to the working directory (often group/scratch)
6. Run the program
7. Copy over any results files needed
8. Clean up any temporary files or directories

Basic SLURM script - bash

```
#!/bin/bash
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-guest
#Set up whatever package we need to run with
module load somemodule
#set up the temporary directory
SCRDIR=/scratch/general/lustre/$USER/$SLURM_JOBID
mkdir -p $SCRDIR
#copy over input files
cp file.input $SCRDIR/.
cd $SCRDIR
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

Basic SLURM script - tcsh

```
#!/bin/tcsh
#SBATCH --time=02:00:00
#SBATCH --nodes=1
#SBATCH -o slurmjob-%j.out-%N
#SBATCH -e slurmjob-%j.err-%N
#SBATCH --account=owner-guest
#SBATCH --partition=kingspeak-guest
#Set up whatever package we need to run with
module load somemodule
#set up the scratch directory
set SCRDIR /scratch/local/$USER/$SLURM_JOBID
mkdir -p $SCRDIR
#move input files into scratch directory
cp file.input $SCRDIR/.
cd $SCRDIR
#Run the program with our input
myprogram < file.input > file.output
#Move files out of working directory and clean up
cp file.output $HOME/.
cd $HOME
rm -rf $SCRDIR
```

Checking job efficiency

~]$ pestat -u $USER

Hostname	Partition	Node State	Num_CPU Use/Tot		CPUload	Memsize (MB)	Freemem (MB)	Joblist JobId User ...
kp016	kingspeak*	alloc	16	16	1.00*	64000	55494	7430561 u0123456
kp378	schmidt-kp	alloc	28	28	.00*	256000	250656	7430496 u0123456

Checking Job Performance

- With an active job – can ssh to node
  - Useful commands, top, ps, sar
- Also from interactive node can query job
  - /uufs/chpc.utah.edu/sys/installdir/pestat/pestat
  - Working on adding documentation
- Can query node status
  - scontrol show node notch024

Ex 5: Job Submission via command line

- Test job – a single process matlab job
- To start job
  - sbatch run_matlab_serial.slr
- To check queue
  - squeue –u $USER
- To check performance
  - pestat –u $USER
- Output
  - matlab_serial.log

Running interactive batch jobs

- An interactive command is launched through the srun command

```
srun --time=1:00:00 --nodes=1 --account=chpc
        --partition=ember --pty /bin/tcsh -l
```

- Launching an interactive job automatically forwards environment information, including X11 forwarding
- "`--pty`" must be set to shell preferred for the session (either /bin/tcsh or /bin/bash
- `-l` (lower case “L”) at the end required

Parallel Execution

- MPI installations at CHPC are SLURM aware, so **mpirun** will usually work without a machinefile (unless you are manipulating the machinefile in your scripts)
- If machinefile or host list needed, create the node list:
  - `srun hostname | sort -u > nodefile.$SLURM_JOBID`
  - `srun hostname | sort > nodefile.$SLURM_JOBID`
- Alternatively, you can use the srun command instead, but you need to compile with a more recently compiled MPI
- Mileage may vary, and for different MPI distributions, srun or mpirun may be preferred (check our slurm page on the CHPC website for more info or email us)

Slurm for use of GPU Nodes

- Ember – 8 GPU nodes
  - All are general nodes, each with M2090 cards
- Kingspeak – 8 GPU nodes
  - 4 general nodes, two with 4 Tesla K80 cards (8 GPUs) each, two with 8 GeForce TitanX cards each
  - 4 owner nodes each with 2 Tesla P100 cards (owned by School of Computing)
- Notchpeak – 13 GPU nodes
  - 3 general nodes each with 3 Tesla V100 cards; 3 general nodes each with 4 RTX2080Ti cards; 1 general node with 2 RTX2080Ti plus 1 P40 card
  - 1 owner node with 4 TitanV cards
  - 1 owner node with 8 GTX1080Ti cards
  - 4 owner nodes each with 4 RTX2080Ti cards
- Use partition and account set to ***cluster***-gpu (for general) or ***cluster***-gpu-guest for guest jobs on owner
- Must get added to the gpu accounts – request via helpdesk@chpc.utah.edu
- Use only if you are making use of the GPU for the calculation
- Most codes do not yet make efficient use of multiple GPUs so we have enabled node sharing
- See https://www.chpc.utah.edu/documentation/guides/gpus-accelerators.php

Node Sharing on GPU nodes

- Need to specify number of CPU cores, amount of memory, and number of GPU
- Core hours used based on highest % requested among cores, memory and GPUs

Option	Explanation
#SBATCH --gres=gpu:k80:1	request one K80 GPU (others types names are titanx, m2090, p100, v100, titanv, 1080ti, 2080ti, p40)
#SBATCH --mem=4G	request 4 GB of RAM (default is 2GB/core if not specified)
#SBATCH --mem=0	request all memory of the node; use this if you do not want to share the node as this will give you all the memory
#SBATCH --tasks=1	requests 1 core

Strategies for Serial Applications

- https://www.chpc.utah.edu/documentation/software/serial-jobs.php
- When running serial applications (no MPI, no threads) unless memory constraint, you should look to options to bundle jobs together so using all cores on nodes
- There are multiple ways to do so
  - srun --multi-prog
  - submit script
- Also consider OpenScienceGrid as an option (especially if you have a large number of single core, short jobs)

Strategies for Job Arrays

- https://www.chpc.utah.edu/documentation/software/slurm.php#jobarr
- Useful if you have many similar jobs when each use all cores on a node or multiple nodes to run where only difference is input file
- sbatch --array=1-30%n myscript.sh – where n is maximum number of jobs to run at same time
- In script: use $SLURM_ARRAY_TASK_ID to specify input file:
  - ./myprogram input$SLURM_ARRAY_TASK_ID.dat

Job Priorities

- https://www.chpc.utah.edu/documentation/software/slurm.php#priority
- **sprio** give job priority for all jobs
  - sprio –j JOBID for a given job
  - sprio –u UNID for all a given user's jobs
- Combination of three factors added to base priority
  - Time in queue
  - Fairshare
  - Job size

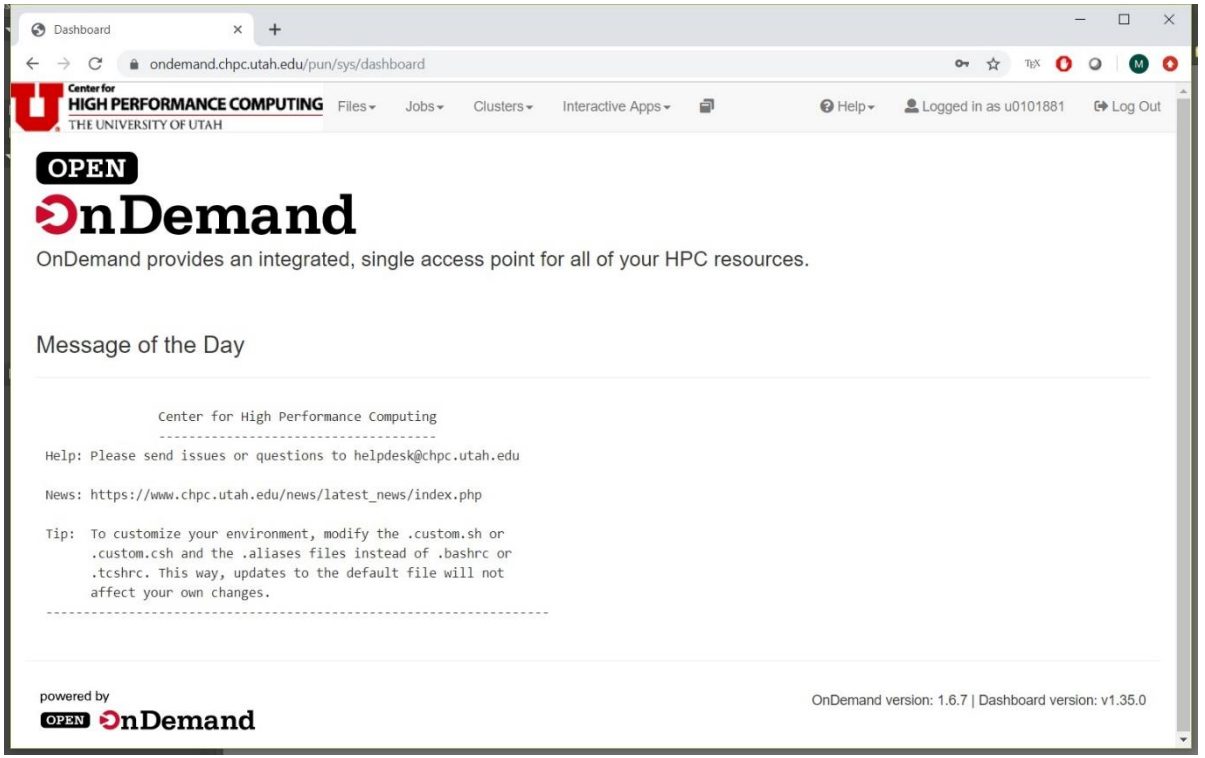
What is Open OnDemand?

- Web portal to HPC resources - openondemand.org
- Easier, command line free, use of HPC resources
- File management module
- Job submission and monitoring module
- Interactive desktop and applications
  - e.g. MATLAB, ANSYS, Jupyter Notebook, R Studio Server
- Actively developed and supported by NSF
- Need internet access, web browser and CHPC account

CHPC's Open OnDemand

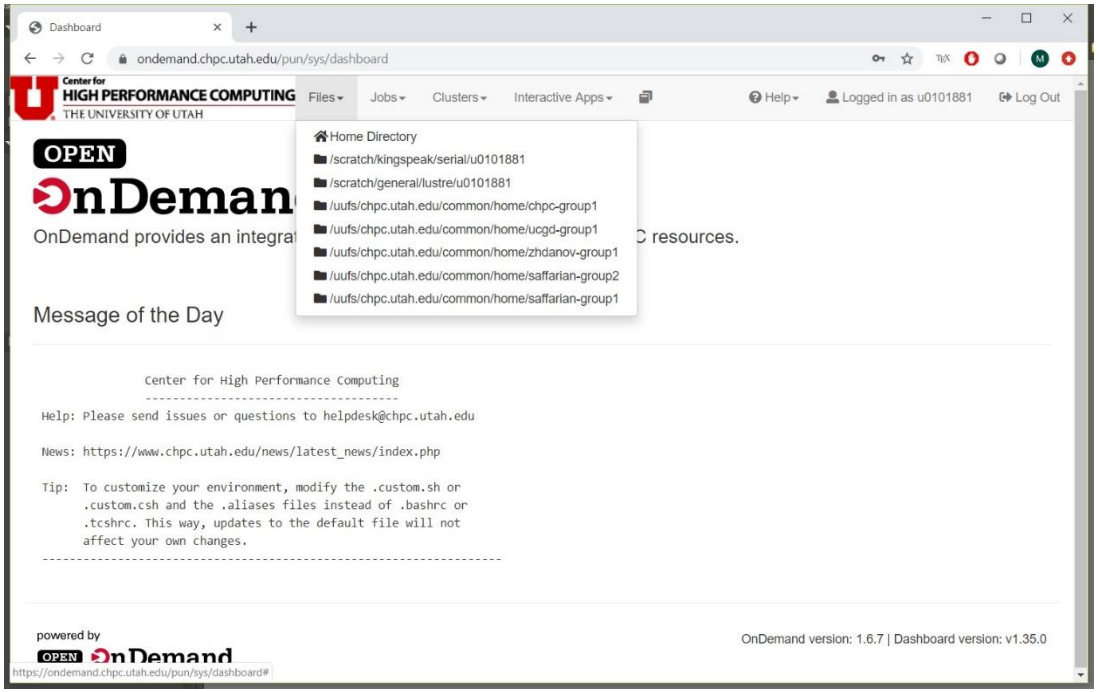
- ondemand.chpc.utah.edu

- Log in with your uNID and password



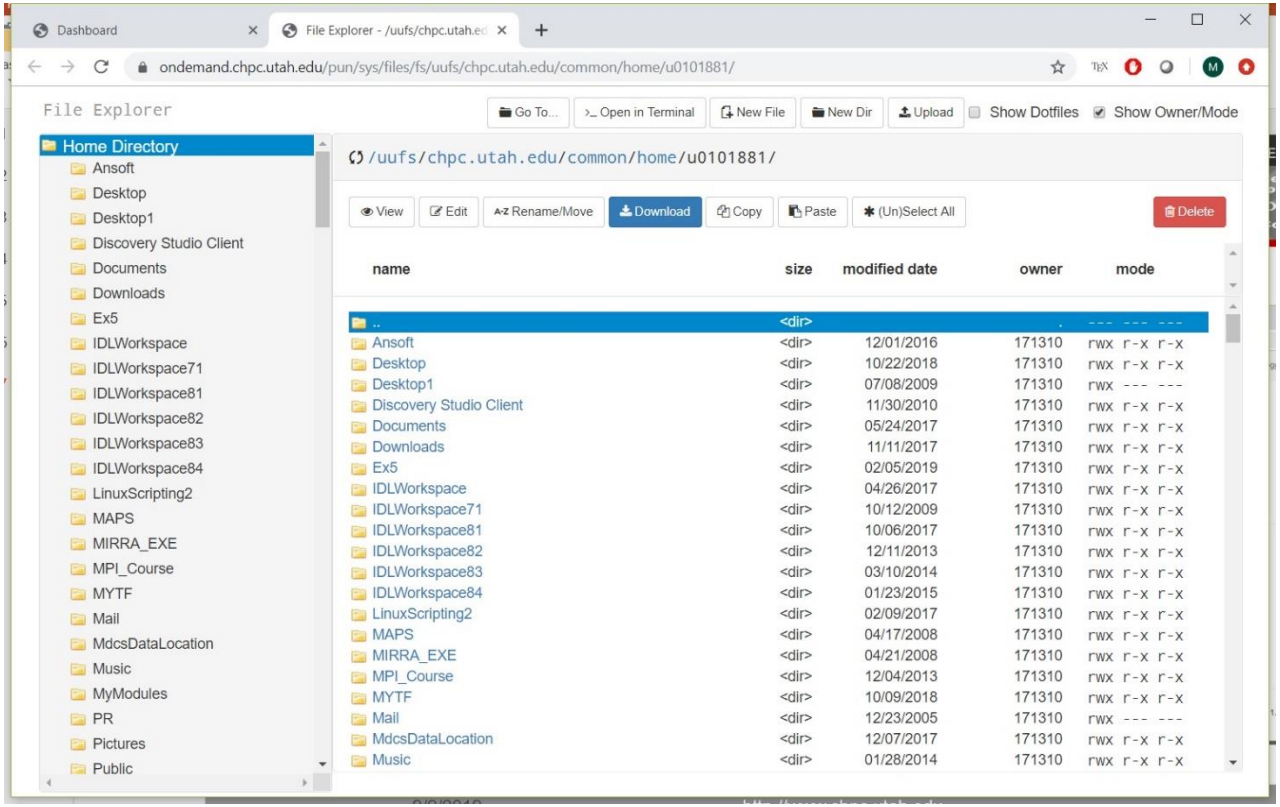
File explorer

- Sees all file systems where user has access
- Allows various file operations, including editing



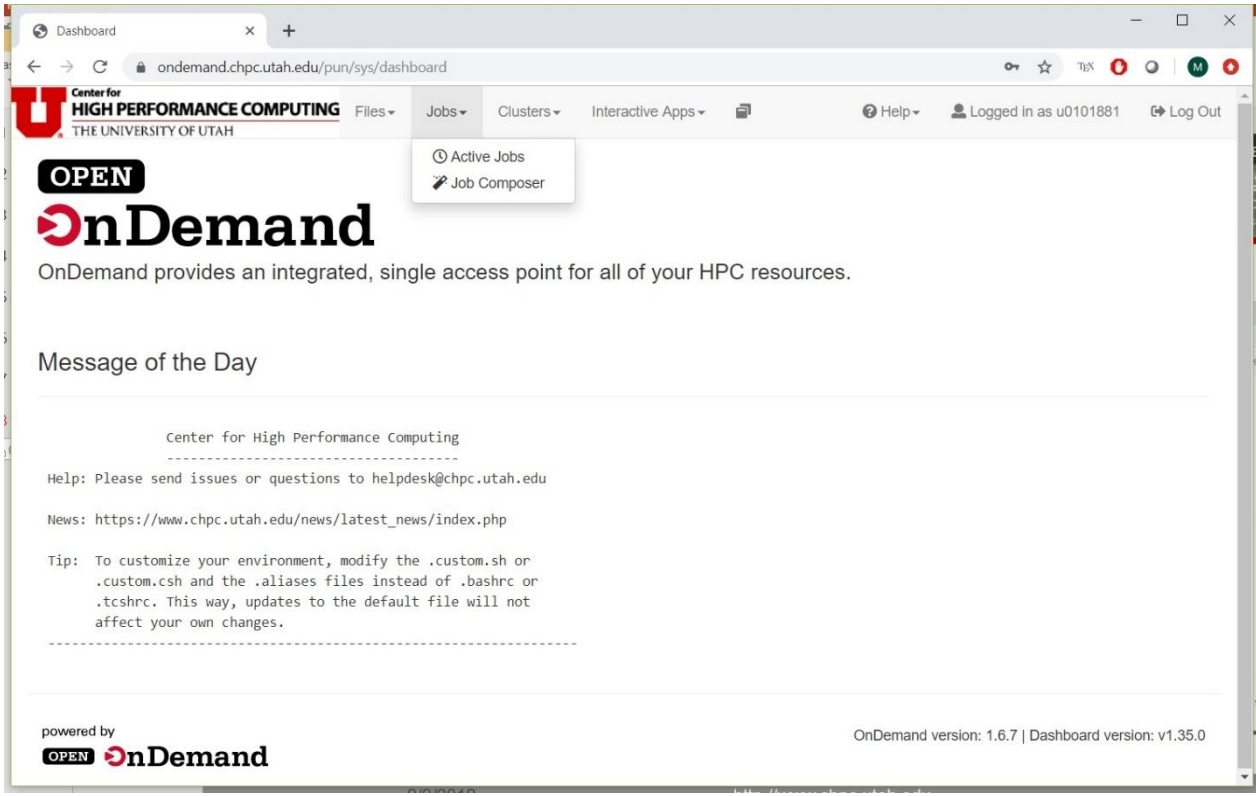
File explorer

- Drag and drop copying, renaming
- File viewing and editing
- Open in terminal
- Upload and Download



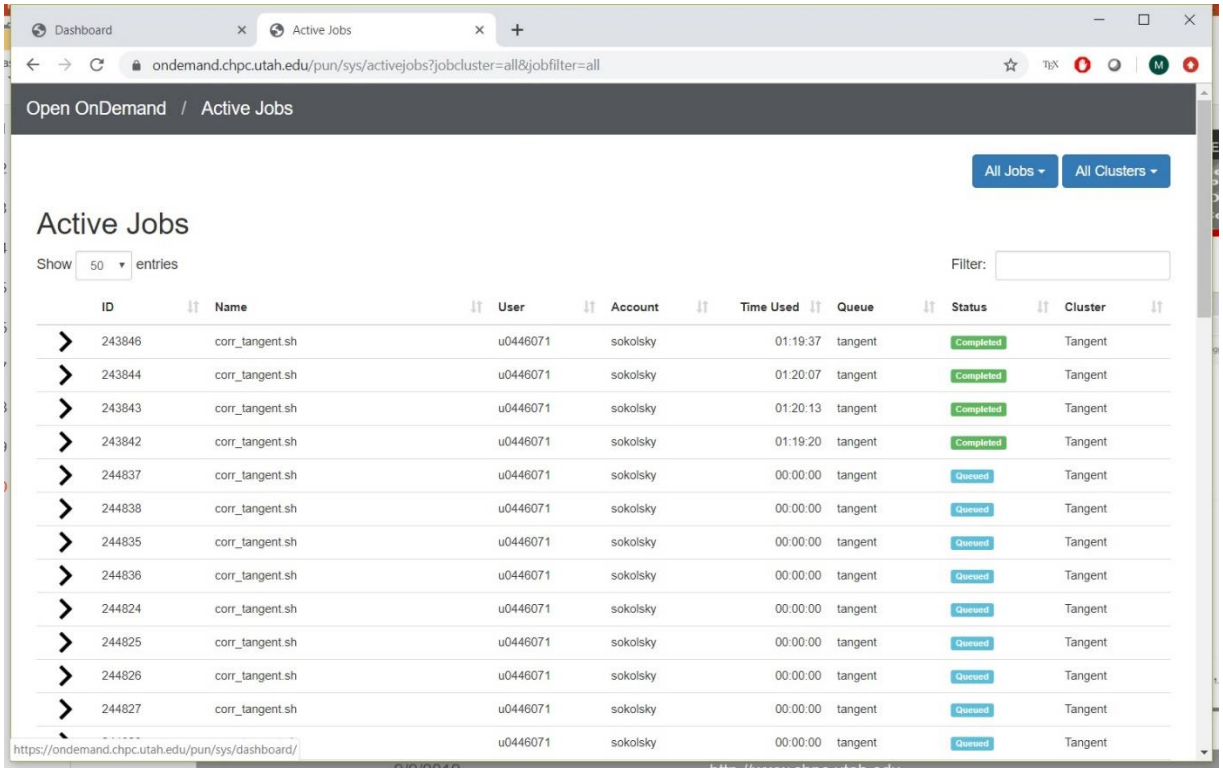
Jobs

- Listing of active jobs
- Creating and submitting new jobs



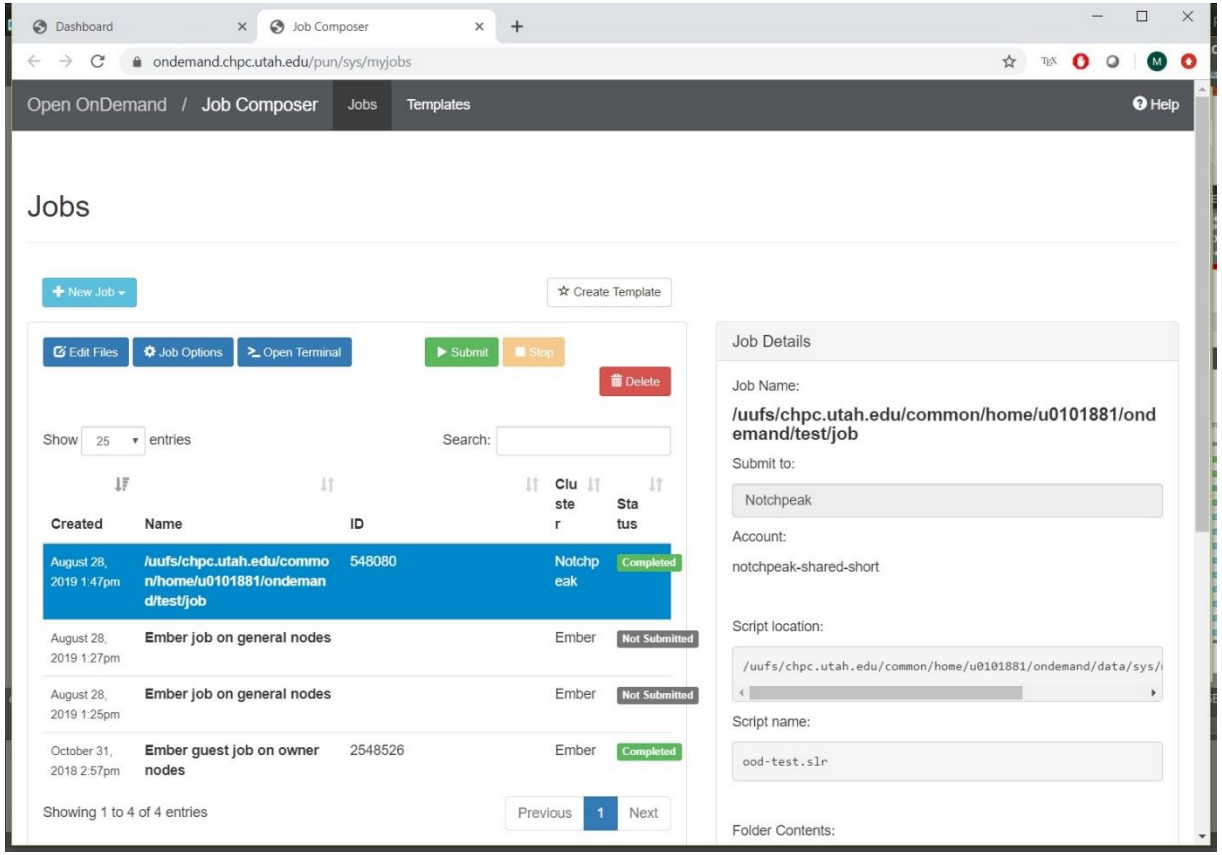
Active jobs

- Filter by all or user only jobs
- Filter by all clusters or specific cluster
- Expanding shows job details
- Use filter to search for jobs



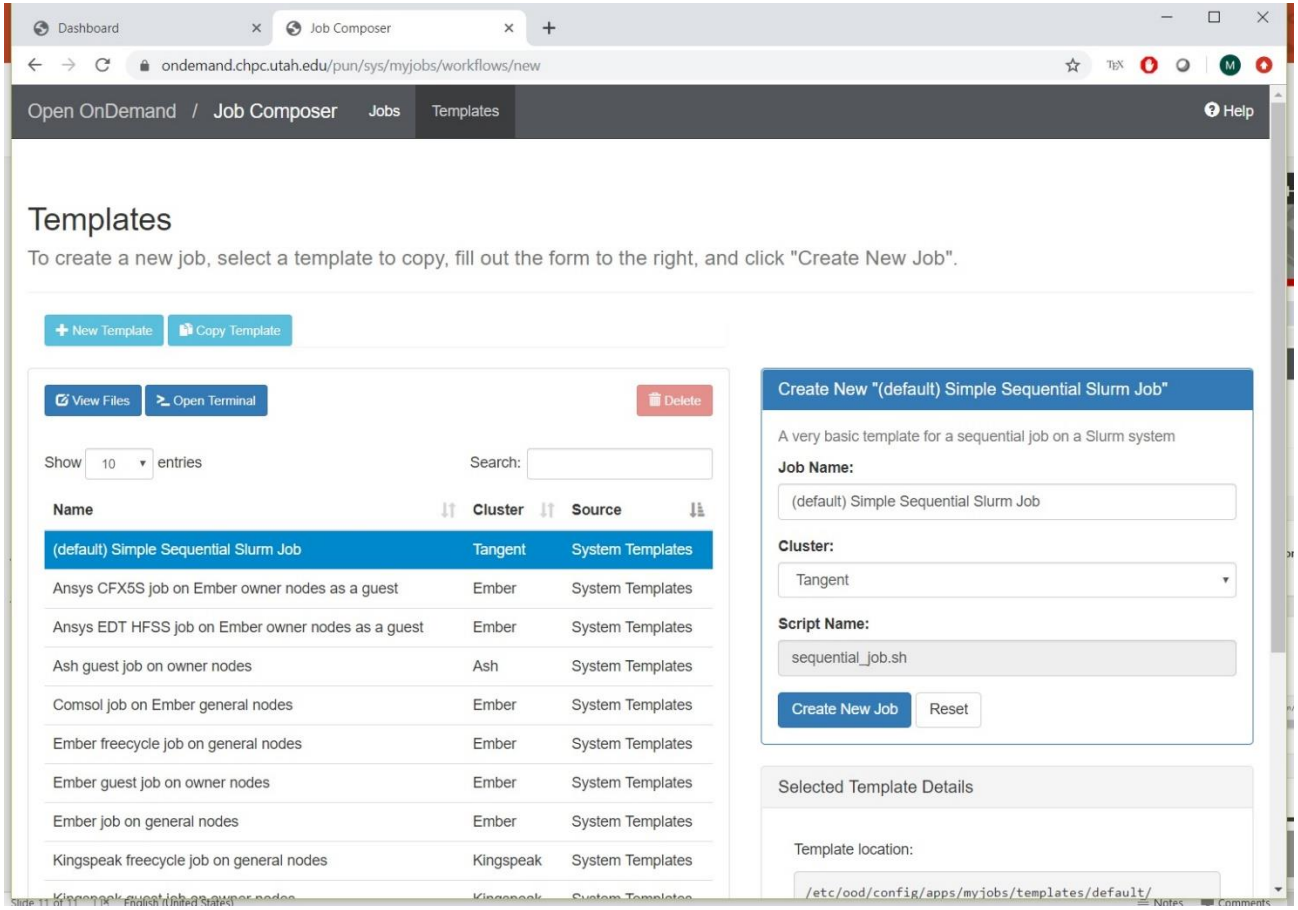
Job composer - jobs

- Create and edit job scripts
- Edit job input files (in File Explorer)
- Submit/cancel jobs
- See job status
- Caveat - OOD copies all job files to ~/ondemand/data/sys/myjobs/projects/default



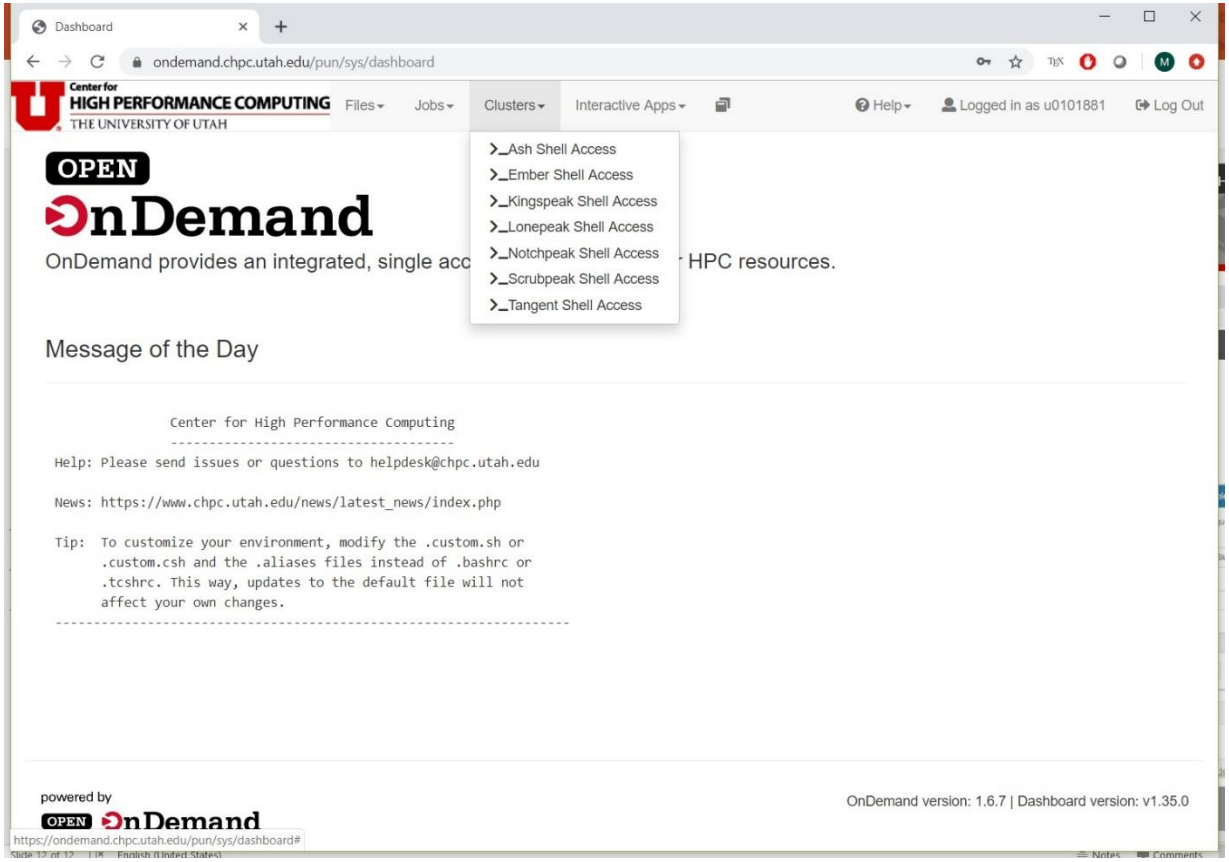
Job composer - templates

- SLURM job script templates
- Create new jobs based on these templates
- Modify these jobs based on specific needs
- https://github.com/CHPC-UofU/chpc-myjobs-templates



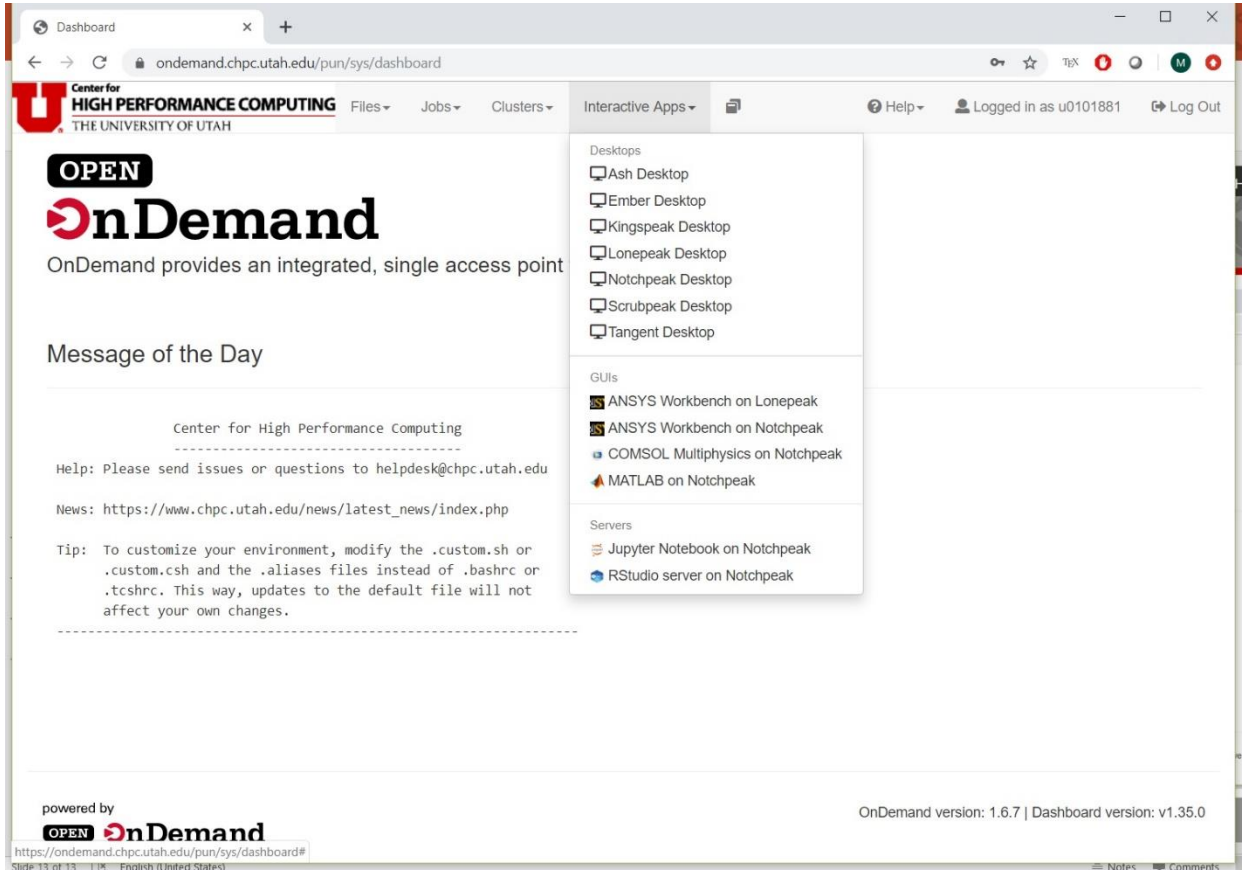
Clusters terminal access

- Shell terminal access to each cluster
- Opens a new browser tab with terminal



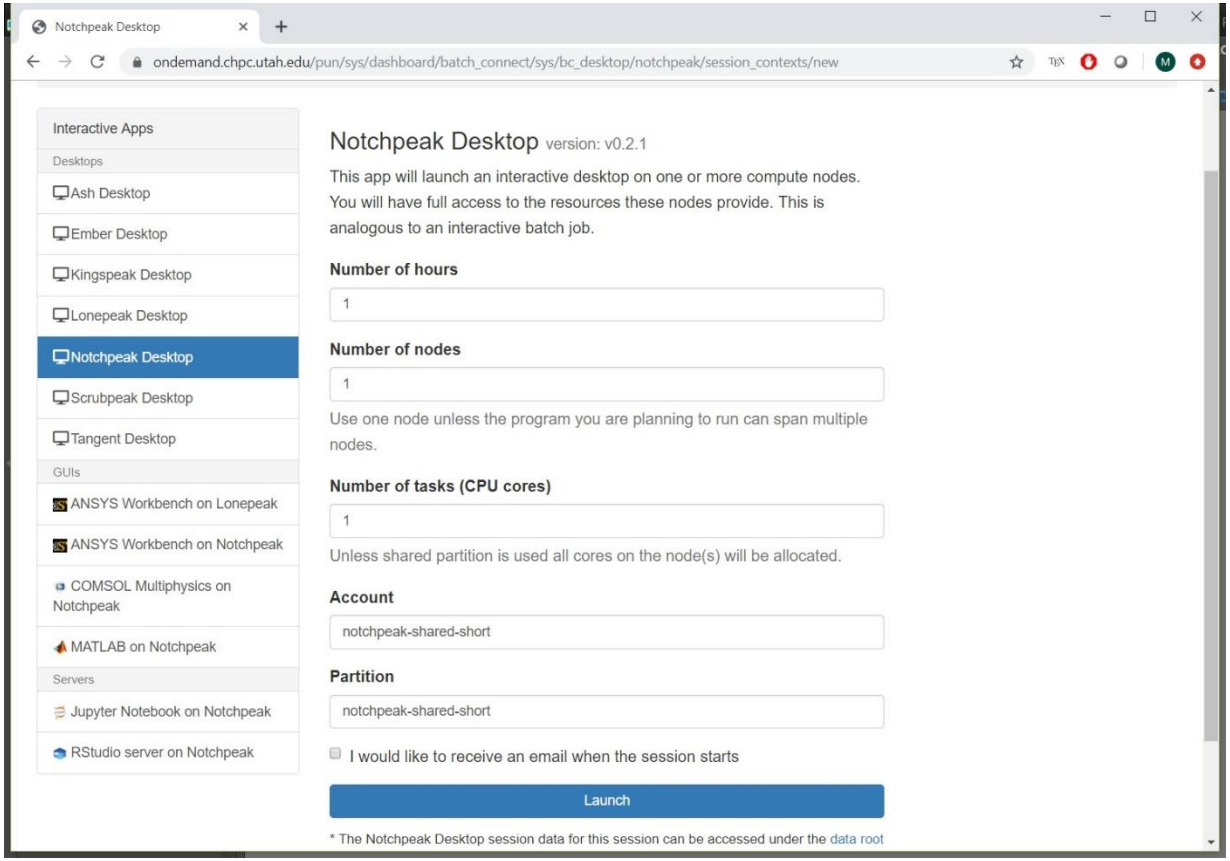
Interactive apps

- Interactive jobs
- The most unique feature of OOD
- Session on a compute node inside interactive SLURM job
- Either remote desktop or application



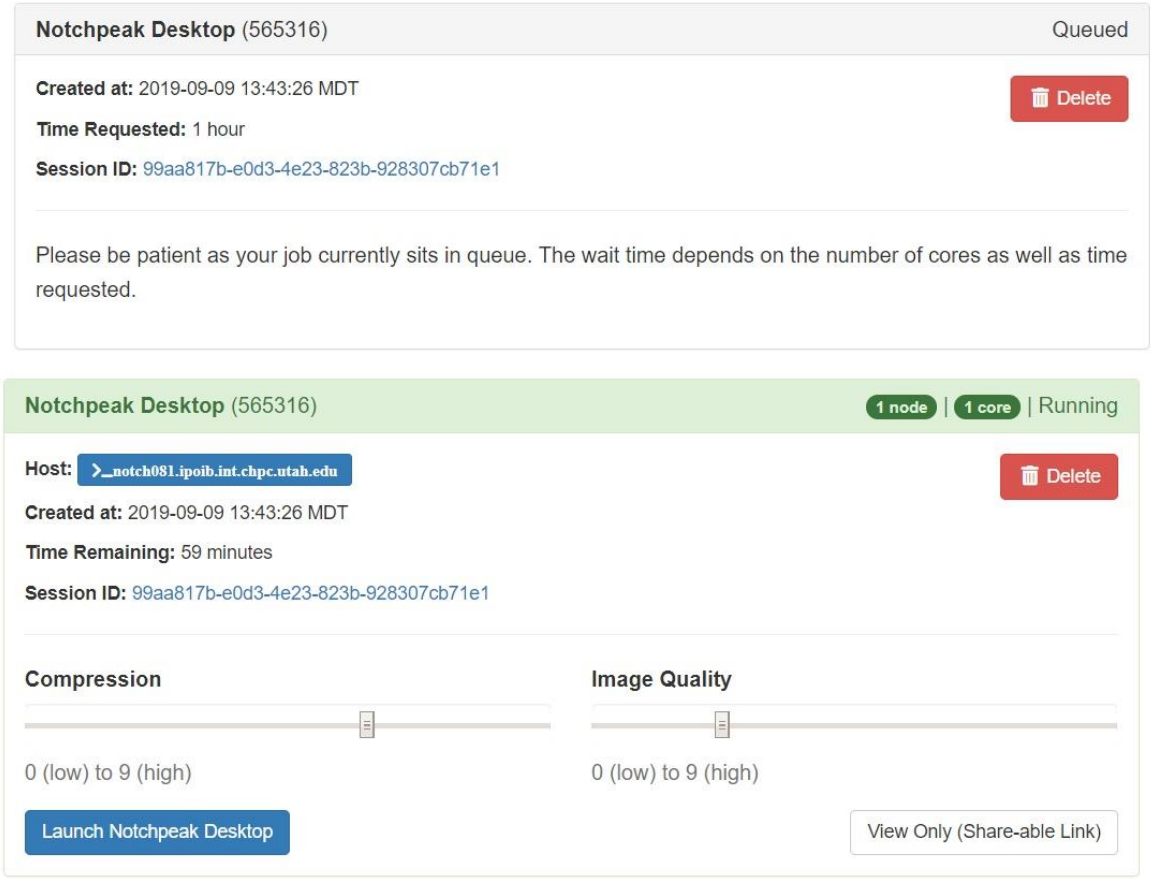
Interactive apps - desktop

- Specific for each cluster
- To start the desktop job ASAP use notchpeak-shared-short
- Wait time may be longer on other clusters unless group has owner nodes



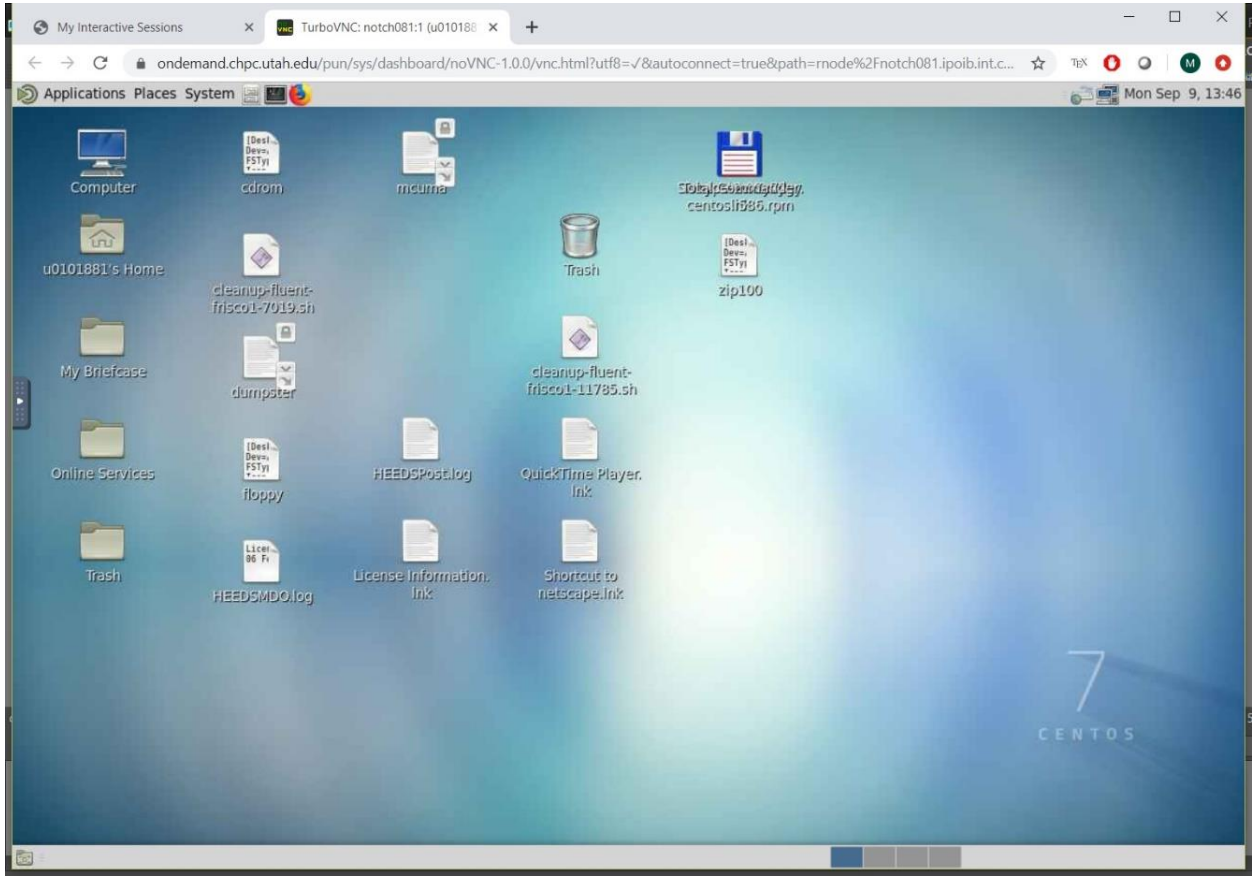
Interactive desktop launch

- First job is queued
- Once job starts, Launch button appears
- Can modify the viewing quality
- Also can share the link for others to view (but not do anything else)



Interactive desktop launch

- Interactive job's remote desktop is launched in a separate browser tab
- Closing the tab does not delete the job (persistent connection)
- Must hit Delete to delete the job



Ex 6: Job submission via OOD

Getting Help

- CHPC website
  - www.chpc.utah.edu
    - Getting started guide, cluster usage guides, software manual pages, CHPC policies
- Ticketing System
  - Email: helpdesk@chpc.utah.edu
- Help Desk: 405 INSCC, 581-6440 (9-5 M-F)
- Mailing Lists:

  chpc-hpc-users@lists.utah.edu used to send messages to users

RMACC Computing Resources

http://rmacc.org/accessingsummit
https://www.colorado.edu/rc/

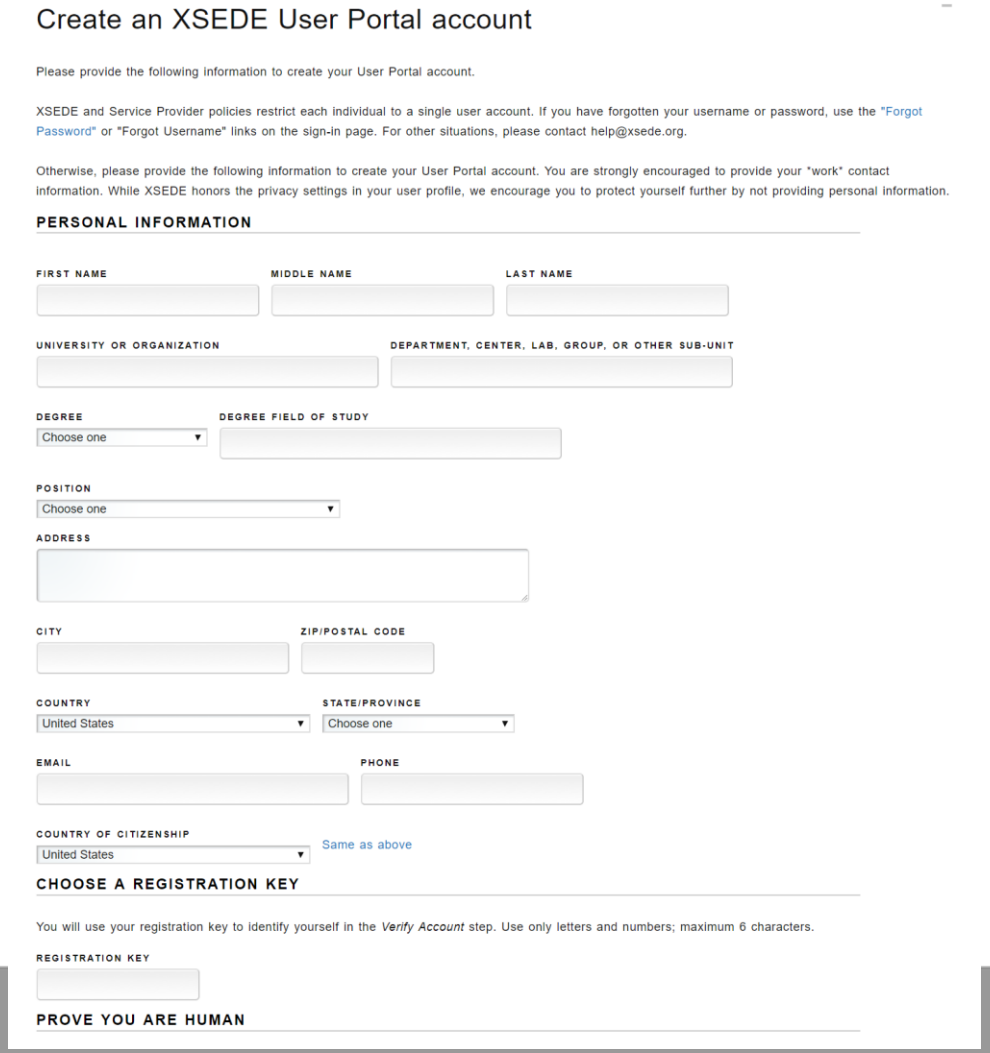
RMACC-Summit funded by a MRI grant by CU Boulder and CSU -- 10% cycles for institutions in RMACC region, especially institutions without own compute resources

- General compute
  - Haswell 24 cores/node, 128GB RAM
- High memory
  - 48 cores/node 2TB
- GPU nodes
  - 24 cores, 2 K80s/node
- KNL Xeon Phi

- Now can access with XSEDE login credentials via SSOHub

Creating an XSEDE portal account (XUP)

- portal.xsede.org
- Now requires DUO 2Factor authentication
- Fill in personal information
- Choose a registration key
- System will send you email with a confirmation number
- Use confirmation number together with passkey to verify your account



Create an XSEDE User Portal account

Please provide the following information to create your User Portal account.

XSEDE and Service Provider policies restrict each individual to a single user account. If you have forgotten your username or password, use the "Forgot Password" or "Forgot Username" links on the sign-in page. For other situations, please contact help@xsede.org.

Otherwise, please provide the following information to create your User Portal account. You are strongly encouraged to provide your "work" contact information. While XSEDE honors the privacy settings in your user profile, we encourage you to protect yourself further by not providing personal information.

PERSONAL INFORMATION

FIRST NAME | MIDDLE NAME | LAST NAME

UNIVERSITY OR ORGANIZATION | DEPARTMENT, CENTER, LAB, GROUP, OR OTHER SUB-UNIT

DEGREE: Choose one | DEGREE FIELD OF STUDY

POSITION: Choose one

ADDRESS

CITY | ZIP/POSTAL CODE

COUNTRY: United States | STATE/PROVINCE: Choose one

EMAIL | PHONE

COUNTRY OF CITIZENSHIP: United States — Same as above

CHOOSE A REGISTRATION KEY

You will use your registration key to identify yourself in the *Verify Account* step. Use only letters and numbers; maximum 6 characters.

REGISTRATION KEY

PROVE YOU ARE HUMAN

RMACC-Summit Access

After you have XSEDE login:

- send request from your institutional email address to rc-help@colorado.edu
  - https://github.com/ResearchComputing/Research-Computing-User-Tutorials/wiki/RMACC-Access-to-Summit
- Allocations
  - Can run without allocation for smaller needs
  - https://www.colorado.edu/rc/userservices/allocations
- For training
  - https://www.colorado.edu/rc/userservices/training

Single Sign On (SSO) Login Hub

- ssh <XUPlogin>@login.xsede.org
- >gsissh <machine-name>
- Easy to setup host alias file
- https://portal.xsede.org/web/xup/single-sign-on-hub