# Interactive High-Performance Computing on Expanse

**October 29, 2020**

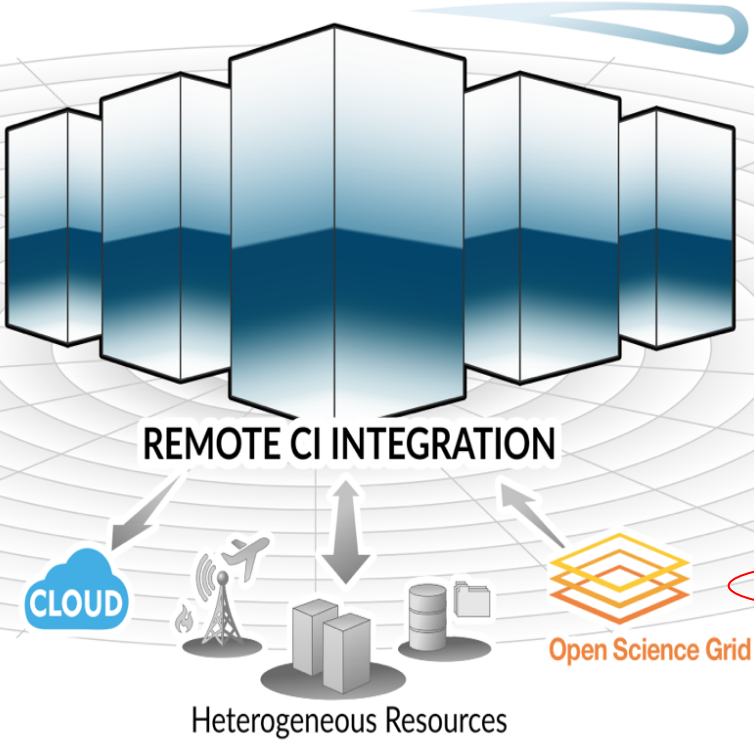

Mary Thomas
(mthomas@sdsc.edu)

# EXPANSE

## COMPUTING WITHOUT BOUNDARIES
5 PETAFLOP/S HPC and DATA RESOURCE

**HPC RESOURCE**
13 Scalable Compute Units
728 Standard Compute Nodes
52 GPU Nodes: 208 GPUs
4 Large Memory Nodes

**DATA CENTRIC ARCHITECTURE**
12PB Perf. Storage: 140GB/s, 200k IOPS
Fast I/O Node-Local NVMe Storage
7PB Ceph Object Storage
High-Performance R&E Networking




**LONG-TAIL SCIENCE**
Multi-Messenger Astronomy
Genomics
Earth Science
Social Science

**INNOVATIVE OPERATIONS**
Composable Systems
High-Throughput Computing
Science Gateways
Interactive Computing
Containerized Computing
Cloud Bursting

For more details see the Expanse user guide @ https://www.sdsc.edu/support/user_guides/expanse.html
and the "Introduction to Expanse" webinar @ https://www.sdsc.edu/event_items/202006_Introduction_to_Expanse.html

# Outline

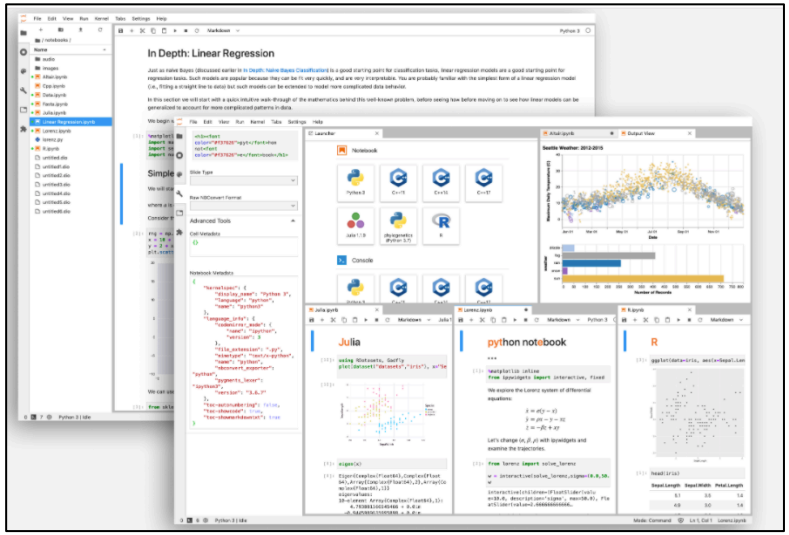
- What is Interactive High-Performance Computing?
- Types of Interactive Computing
- How to Access an Interactive Compute Node
- How to Run Applications
- How to Run Jupyter Notebooks

# What is Interactive HPC-Computing

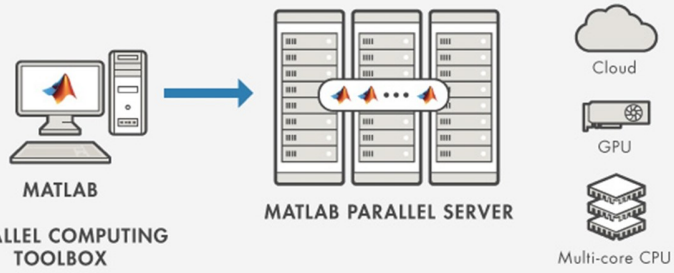
- In **computer** science, **interactive computing** refers to software which accepts input from the user as it runs.
  - **Interactive** software includes commonly used programs, such as word processors or spreadsheet applications.
- **Interactive HPC systems** involve real-time user inputs to perform tasks on a set of compute node(s) such as:
  - Code development, real-time data exploration, and visualizations.
  - Used when application has large data sets/is too large to download to local device, software is difficult install, etc.
  - User inputs come vai command line interface or application GUI (Matlab, R-studio).
  - Actions performed on remoted compute nodes.

# Interactive HPC Scenarios

https://jupyter.org/




```
>> parpool(parcluster('HPC1'),100);
>> parfor i=1:3000
>>     c(i,:) = eig(rand(1000));
>> end
```

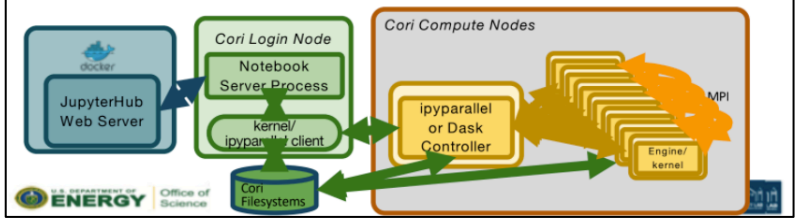



https://azuremarketplace.microsoft.com/en-us/marketplace/apps/mathworks-inc.matlab-parallel-server-listing?tab=Overview

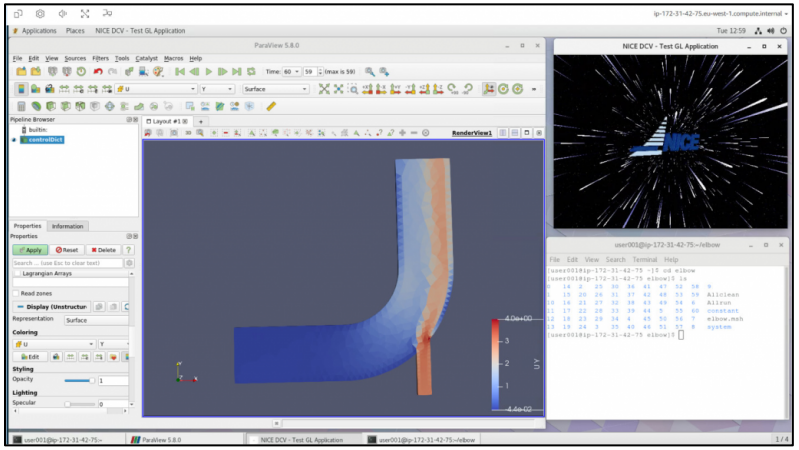
## Jupyter architecture

NERSC

- **Allocate nodes on Cori interactive queue and start ipyparallel or Dask cluster**
  - Developed %ipcluster magic to setup within notebook
- **Compute nodes traditionally do not have external address**
  - Required network configuration / policy decisions
- **Distributed training communication is via MPI Horovod or Cray ML Plugin**




https://drive.google.com/file/d/1-OFjrk1q3L1d3uakr2xkozrPn2c2VZpZ/view



https://aws.amazon.com/blogs/compute/how-to-run-3d-interactive-applications-with-nice-dcv-in-aws-batch/

# Accessing Interactive Compute Nodes

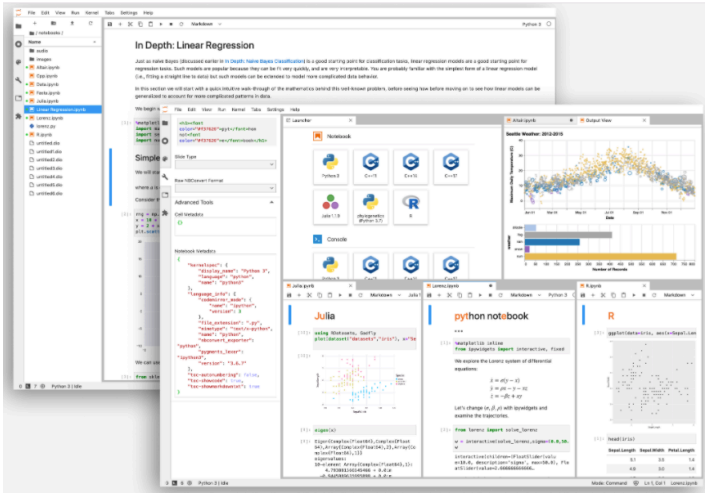
- Load *slurm* module
- New: account info is required
- Use the *srun* command to obtain nodes for 'live,' command line interactive access:

| | |
|---|---|
| **CPU** | `srun --partition=debug --pty --account=use300 \`<br>`--nodes=1 --ntasks-per-node=128 --mem=248 \`<br>`-t 00:30:00 --wait=0 --export=ALL /bin/bash` |
| **GPU** | `srun --partition=gpu-debug --pty --account=abc123\`<br>`--nodes=1 --ntasks-per-node=40 --mem=374 --gpus=4\`<br>`-t 00:30:00 --wait=0 --export=ALL /bin/bash` |

# Running Juptyer Services on HPC Systems at SDSC

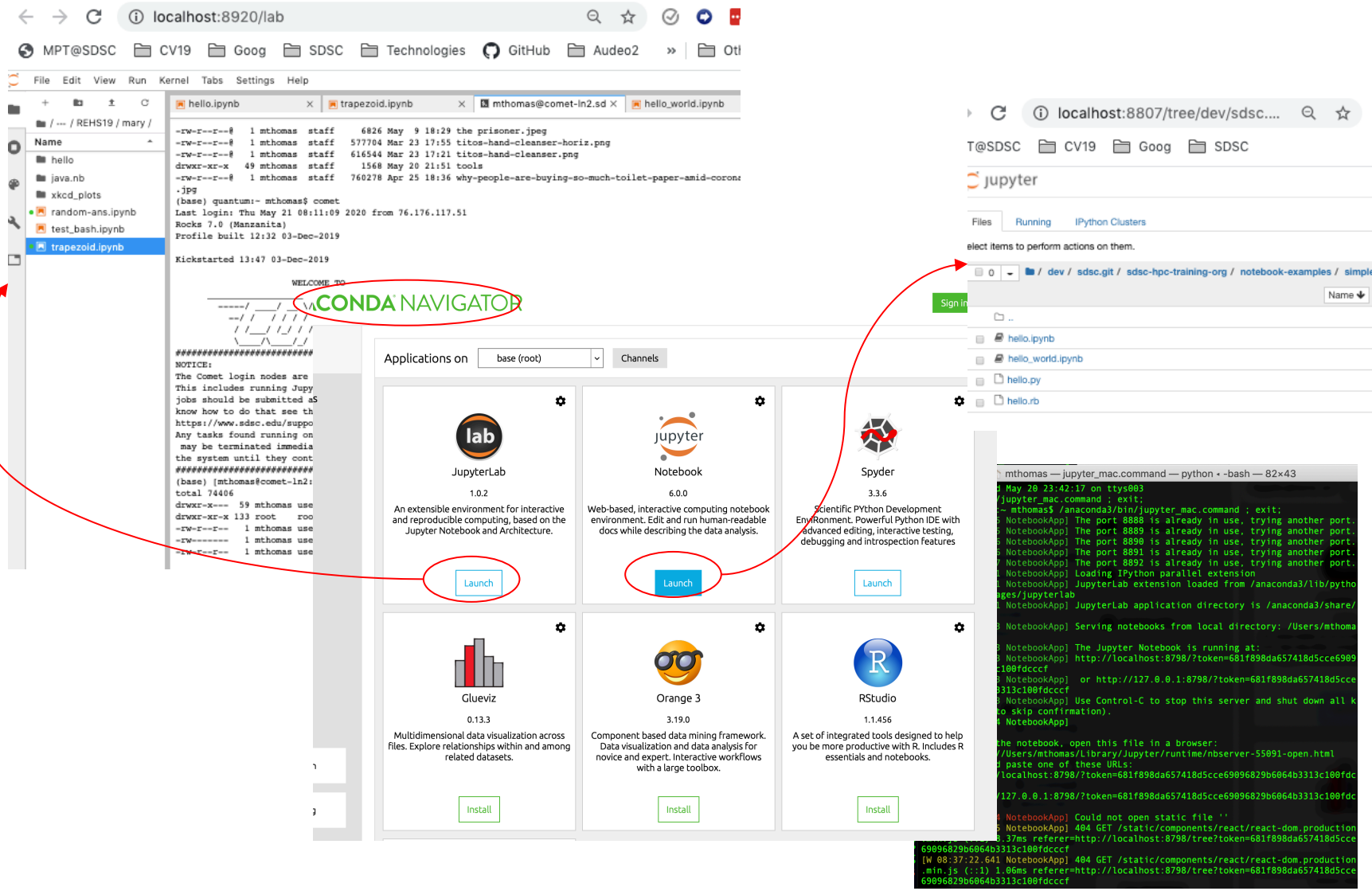
- What is Jupyter?

  ***Jupyter*** *is a free, open-source, interactive web tool known as a computational notebook, which researchers can* ***use*** *to combine software code, computational output, explanatory text and multimedia resources in a single document.* *(J. Perkel, https://www.nature.com/articles/d41586-018-07196-1)*




- Typically use Jupyter Notebooks, or JupyterLab (advanced version of a notebook)
- To run Juptyer services:
  - Laptop: install Anaconda
  - HPC systems, you need to manually install
- Related Tutorial:
  - https://comet-notebooks-101.readthedocs.io/en/master/

# Laptop: Install Anaconda



# HPC Systems: Use Conda

- Conda is an open-source package management system and environment management system that runs on Windows, macOS, and Linux.
  - Setup your own local environment so you have control
- Created for Python programs
  - can package and distribute software for any language.
- Download:
  - https://docs.conda.io/projects/conda/en/latest/
- To Install Conda on Comet or Expanse, see related tutorial:
  - https://comet-notebooks-101.readthedocs.io/en/master/prerequisites.html#install-miniconda
- Conda Cheat Sheet:
  - https://kapeli.com/cheat_sheets/Conda.docset/Contents/Resources/Documents/index

# Custom Conda Virtual Environments

- Use conda to create a virtual environment
  - $ conda create --name example_env
- To see which virtual environments you've created:
  - $ conda env list
- To use a particular virtual environment (e.g., one named 'example_env'):
- $ source activate example_env # Note: don't use 'conda activate'
- Install `Jupyter Notebooks and JupyterLab`

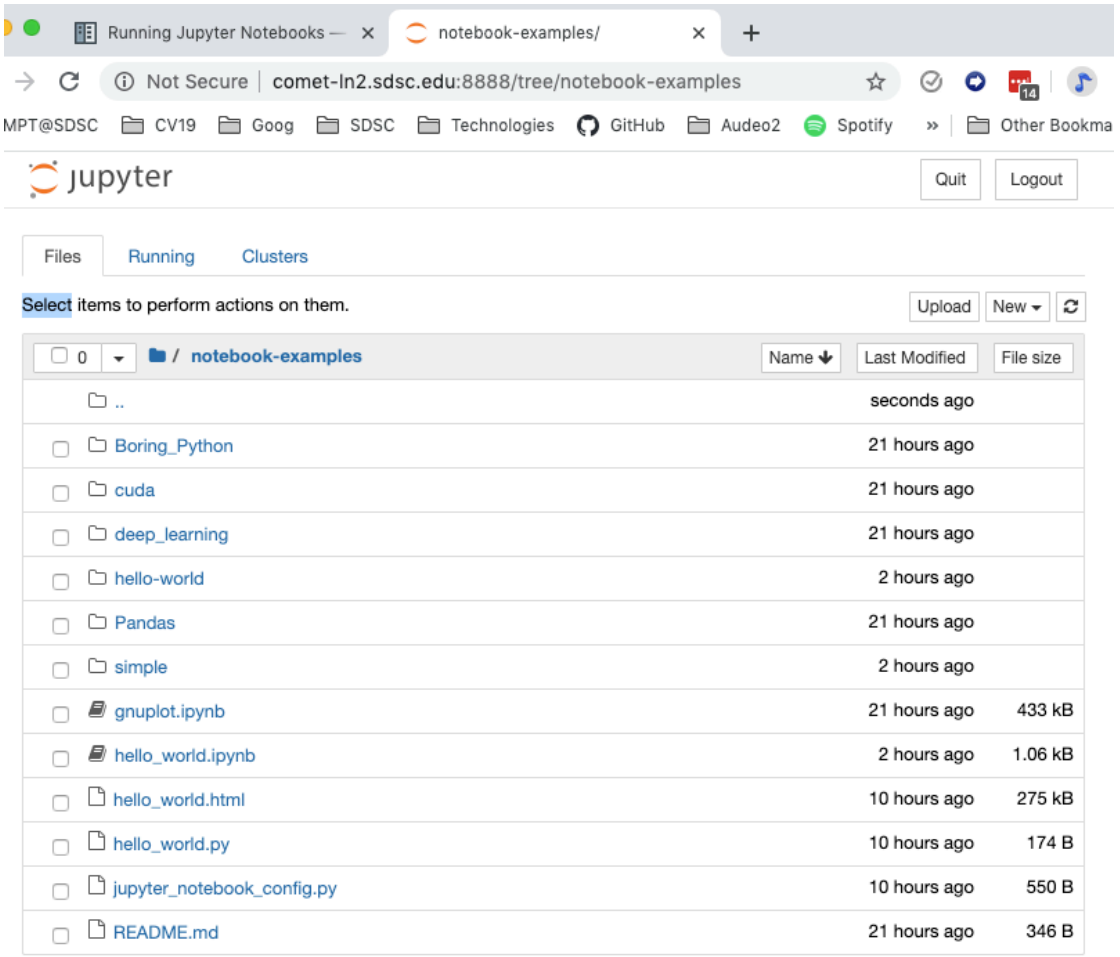
# Installing Jupyter Servers

- To run Jupyter servers, you need to install them using *conda*:
- To Install Jupyter Notebooks, see:
  - https://comet-notebooks-101.readthedocs.io/en/master/prerequisites.html#install-jupyter-notebook
- To Install JupyterLab, see:
  - https://comet-notebooks-101.readthedocs.io/en/master/prerequisites.html#install-jupyterlab
- Note: installing conda, and Jupyter can take a very long time (5-10 minutes)

# Key Vulnerability:
# Notebooks Provide Access to HPC File Systems

SDSC Jupyter Services Policy:

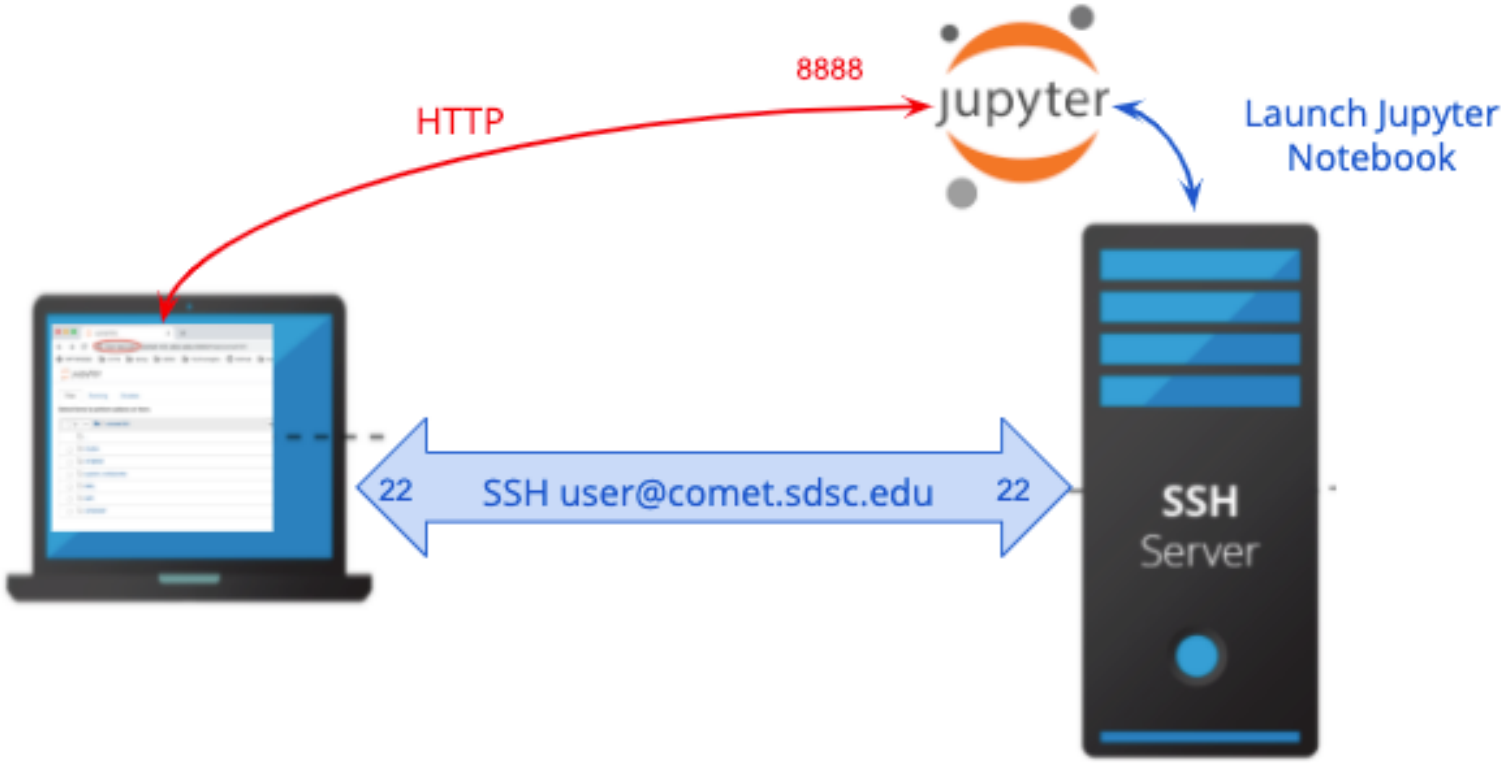
- Portals, JupyterHub, and other services cannot be mounted directly to disk (must be on VM)
  - Many use root in vulnerable ways
  - If a user launches Jupyter Lab or Notebooks, the jobs will be killed.
- No applications can run on login nodes
- SDSC recommendation:
  - use secure connections: when you choose unsecure connections your account is vulnerable to hacking



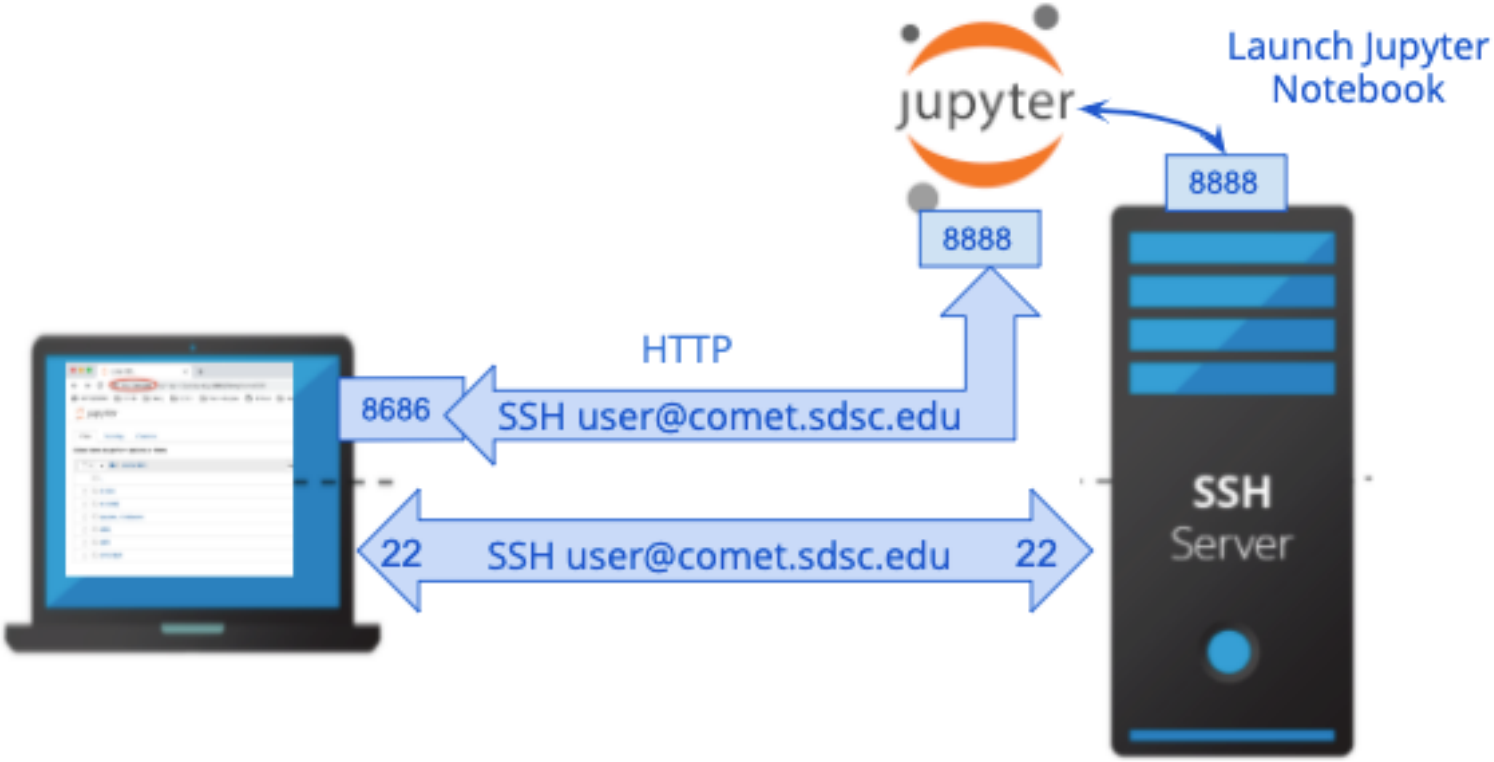
# Methods for Running Notebooks on Comet/Expanse

- Connection scenarios:
  - Connection to Notebook over HTTP (insecure)
  - Connection to Notebook over SSH tunneling (secure, but inconvenient)
  - Connection to Notebook over HTTPS using the Reverse Proxy Service (very secure) [Beta testing]
  - Coming Soon: Galyleo remote notebook launcher
- SDSC allows Jupyter Services to be run on the following nodes:
  - Interactive node
  - Compute node
  - GPU node

# Connection over HTTP
## (unsecure)

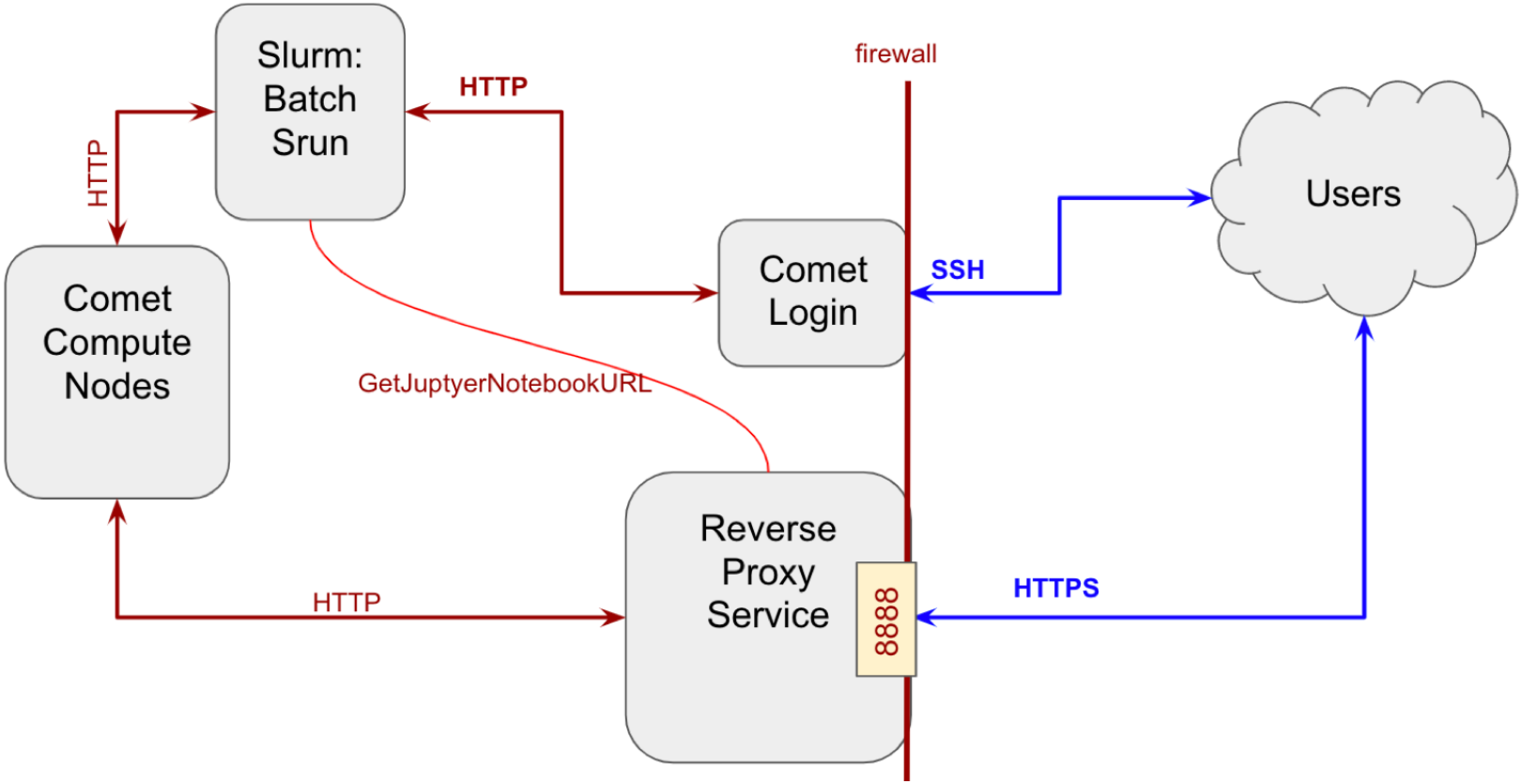


# Secure Connection over SSH Tunneling
## (secure but inconvenient and unstable)




**For instructions on how to set up SSH Tunneling, see:**
**https://comet-notebooks-101.readthedocs.io/en/master/methods/tunneling.html**

# SDSC Jupyter Reverse Proxy Service (JRPS)




https://comet-notebooks-101.readthedocs.io/en/latest/methods/reverseProxy.html

https://github.com/sdsc-hpc-training-org/reverse-proxy

# SDSC Reverse Proxy Service Usage

- Using RPS is very simple and requires no tunneling and is secure (produces HTTPS URLs).
- To use JRPS:
  - SSH to a comet login node.
  - Clone the Repo:

    git clone https://github.com/sdsc-hpc-training-org/reverse-proxy.git
  - Check your software environment on the login node: \conda, Jupyter (notebooks, lab), and other Python packages needed for you application.
- See related tutorial:
  - https://comet-notebooks-101.readthedocs.io/en/master/methods/reverseProxy.html

# Usage

The `start-jupyter` script performs the following tasks:

- Sends request to the reverse proxy server (RPS) to get a one-time token and a port number
- Launches the `jupyter notebook` command using the token and port number.
- Prints a secure URL containing the token to the terminal, so that the user can copy/paste the URL into a local browser:

```
./start-jupyter [-p <string>] [-d <string>] [-A <string>] [-b <string>] [-t time] [-i]
```

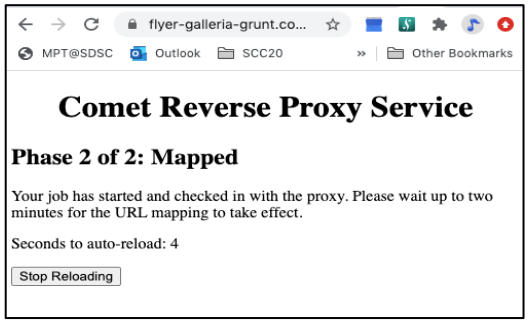
- -p: the partition to wait for (debug or compute). Default Partition is "compute"
- -d: the top-level directory of your jupyter notebook Default Dir is /home/$USER
- -A: the project allocation to be used for this notebook Default Allocation is your sbatch system default allocation (also called project or group)
- -b: the batch script you want to submit with your notebook. Only those in the `batch` folder are supported. Default batch script is ./batch/batch_notebook.sh
- -t: the time to run the notebook. Your account will be charged for the time you put here so be careful. Default time is 30 minutes
- -i: Get extra information about the job you submitted using the script

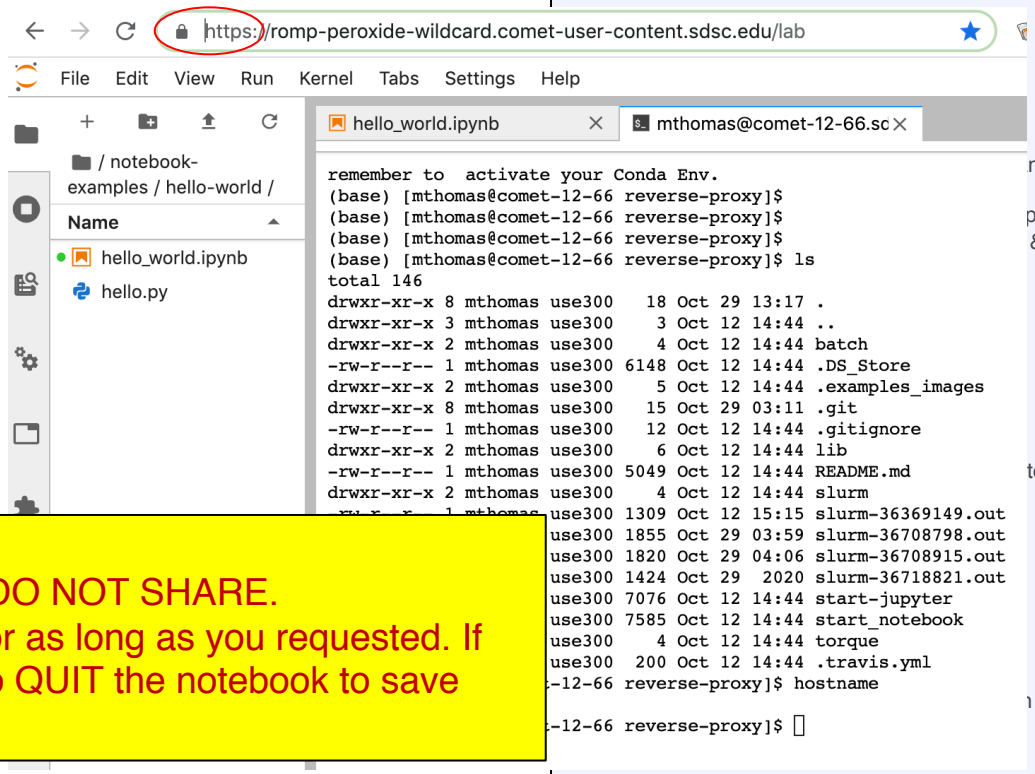
# Install Reverse Proxy Scripts

```
(base) [mthomas@login01 ~]$ mkdir rev-pxy
(base) [mthomas@login01 ~]$ cd rev-pxy/
(base) [mthomas@login01 rev-pxy]$ git clone https://github.com/sdsc-hpc-training-org/reverse-proxy.git
Cloning into 'reverse-proxy'...
remote: Enumerating objects: 14, done.
remote: Counting objects: 100% (14/14), done.
remote: Compressing objects: 100% (9/9), done.
remote: Total 718 (delta 7), reused 12 (delta 5), pack-reused 704
Receiving objects: 100% (718/718), 7.56 MiB | 17.47 MiB/s, done.
Resolving deltas: 100% (397/397), done.
(base) [mthomas@login01 rev-pxy]$ ll
total 19
drwxr-xr-x  3 mthomas use300  3 Oct 29 03:04 .
drwxr-x--- 18 mthomas use300 33 Oct 29 03:04 ..
drwxr-xr-x  8 mthomas use300 14 Oct 29 03:04 reverse-proxy
(base) [mthomas@login01 rev-pxy]$ ll reverse-proxy/
total 24
drwxr-xr-x 8 mthomas use300   14 Oct 29 03:04 .
drwxr-xr-x 3 mthomas use300    3 Oct 29 03:04 ..
drwxr-xr-x 2 mthomas use300    4 Oct 29 03:04 batch
-rw-r--r-- 1 mthomas use300 6148 Oct 29 03:04 .DS_Store
drwxr-xr-x 2 mthomas use300    5 Oct 29 03:04 .examples_images
drwxr-xr-x 8 mthomas use300   13 Oct 29 03:04 .git
-rw-r--r-- 1 mthomas use300   12 Oct 29 03:04 .gitignore
drwxr-xr-x 2 mthomas use300    6 Oct 29 03:04 lib
-rw-r--r-- 1 mthomas use300 5049 Oct 29 03:04 README.md
drwxr-xr-x 2 mthomas use300    4 Oct 29 03:04 slurm
-rwxr-xr-x 1 mthomas use300 7076 Oct 29 03:04 start-jupyter
-rwxr-xr-x 1 mthomas use300 7585 Oct 29 03:04 start_notebook
drwxr-xr-x 2 mthomas use300    4 Oct 29 03:04 torque
-rw-r--r-- 1 mthomas use300  200 Oct 29 03:04 .travis.yml
```

# Using start-jupyter

```
(base) [mthomas@comet-ln3 reverse-proxy]$ ./start-jupyter -b
slurm/jupyterlab.sh
Your notebook is here:
https://headscarf-barber-unframed.comet-user-
content.sdsc.edu?token=3b587a19a8c4ba79f3af754e9fa2a8f5
Using default partition: compute
No time given. Default is 30 mins
```



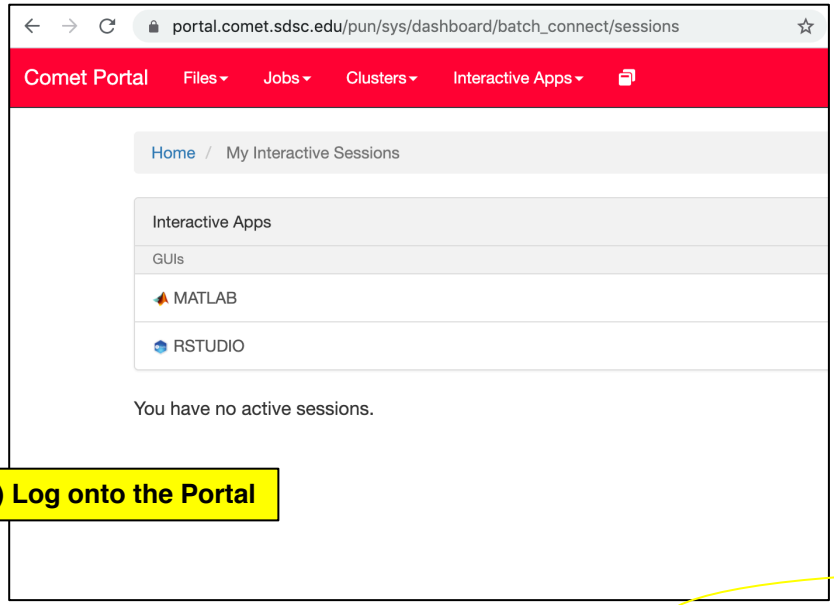




```
(base) [mthomas@comet-ln3 reverse-proxy]$ cat slurm/jupyterlab.sh
#!/bin/bash
## ================================================
## This is an example batch script which can be submitted as part of a
## reverse proxy jupyter notebook. This batch script creates the jupyter
## notebook on a compute node, while the start notebook script is used to
ubmit this batch script. You should never submit this batch script on
s own, e.g. `sbatch batch_notebook.sh`. Don't do that :). You can
pecify this particluar batch script by using the -b flag, e.g.
start_notebook.sh -b batch/batch_notebook.sh
================================================

ou can add your own slurm directives here, but they will override
nything you gave to the start_notebook script like the time, partition, etc
ATCH --nodes=1
ATCH --ntasks-per-node=24

##### Define modules here. #####
me)
pyterlab'" ll exit 1
&
token&port=$PORT"'
means it waits for the jupyter
```

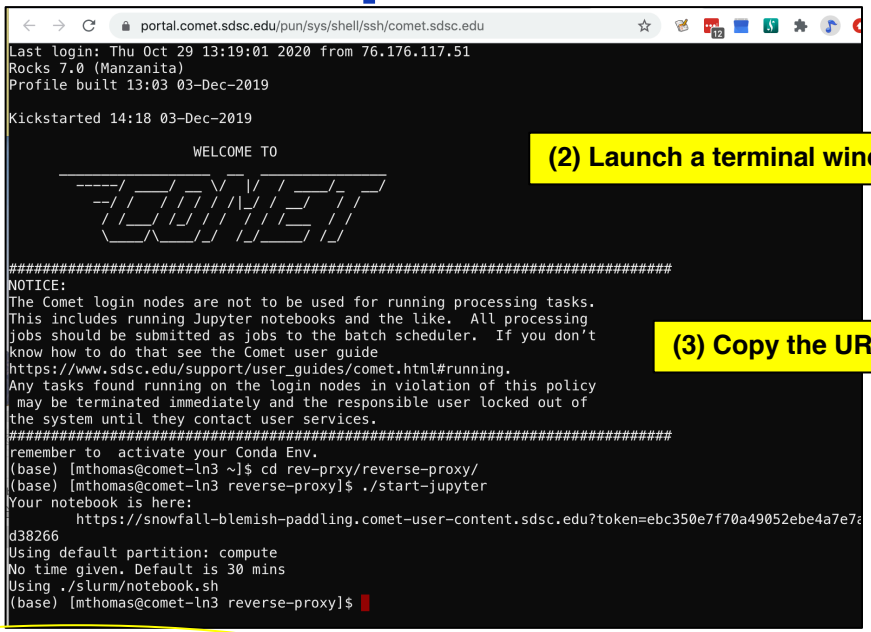
NOTES:

- The URL is like a password: DO NOT SHARE.
- The Jupyter service will run for as long as you requested. If you are done early, be sure to QUIT the notebook to save SU's

# Using Reverse Proxy from Expanse Portal

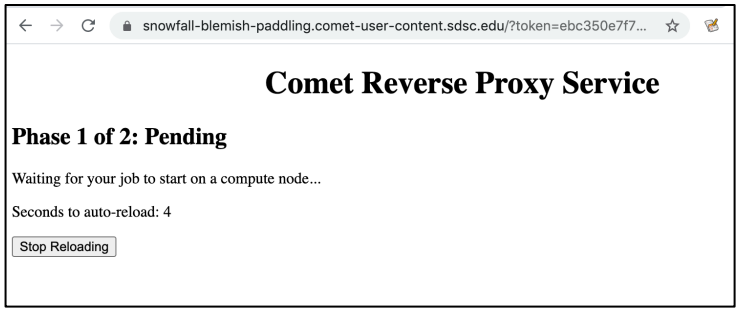



(1) Log onto the Portal

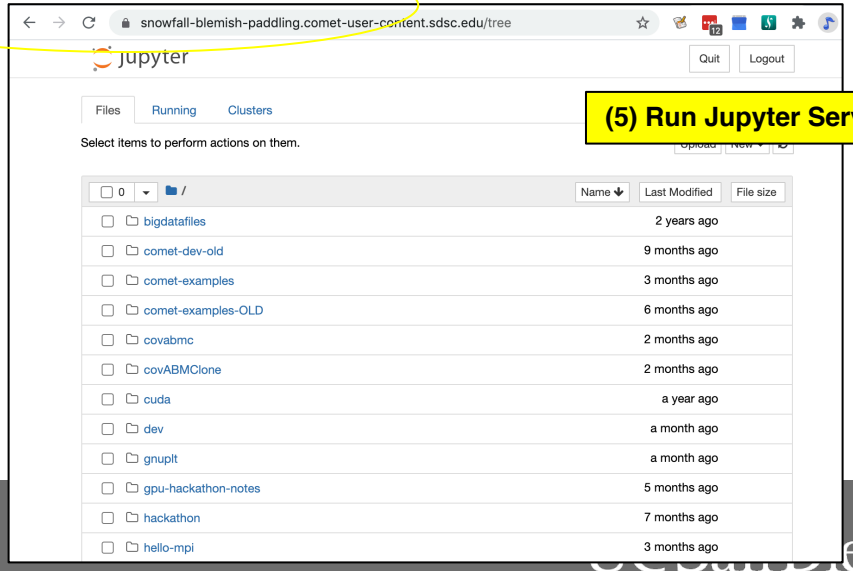



(2) Launch a terminal window

(3) Copy the URL




(4) Monitor status window:
If the queue is busy this may take a long time




(5) Run Jupyter Service

# Thank You

# Resources

- Expanse User Guide
  - https://www.sdsc.edu/support/user_guides/expanse.html
- GitHub Repo for this webinar: clone code examples for this tutorial – clone example code:
  - https://github.com/sdsc-hpc-training-org/expanse-101
- SDSC Training Resources
  - https://www.sdsc.edu/education_and_training/training
  - https://github.com/sdsc-hpc-training/webinars
- XSEDE Training Resources
  - https://www.xsede.org/for-users/training
  - https://cvw.cac.cornell.edu/expanse/