Interactive Computing and Running Jupyter Notebooks

March 4, 2021

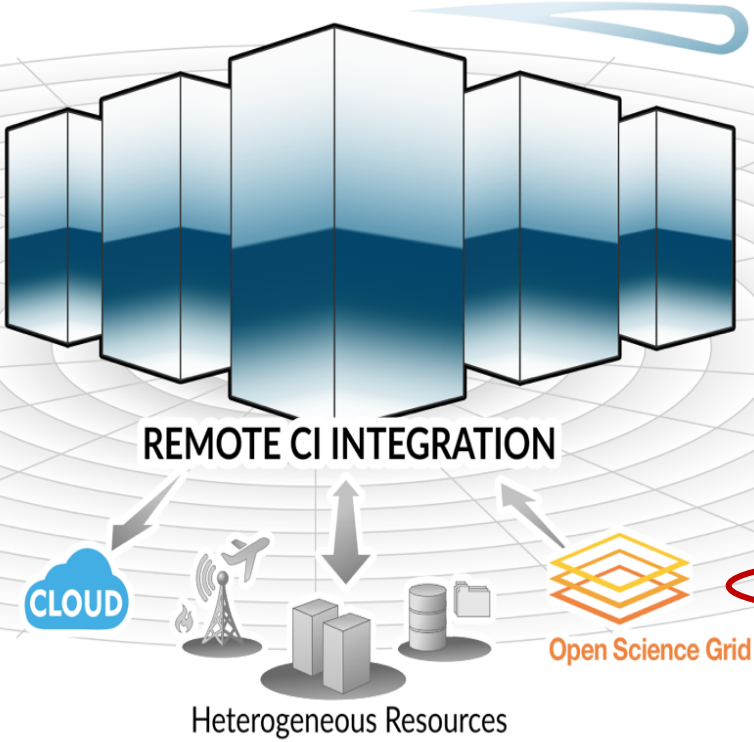
Mary Thomas
(SDSC)

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

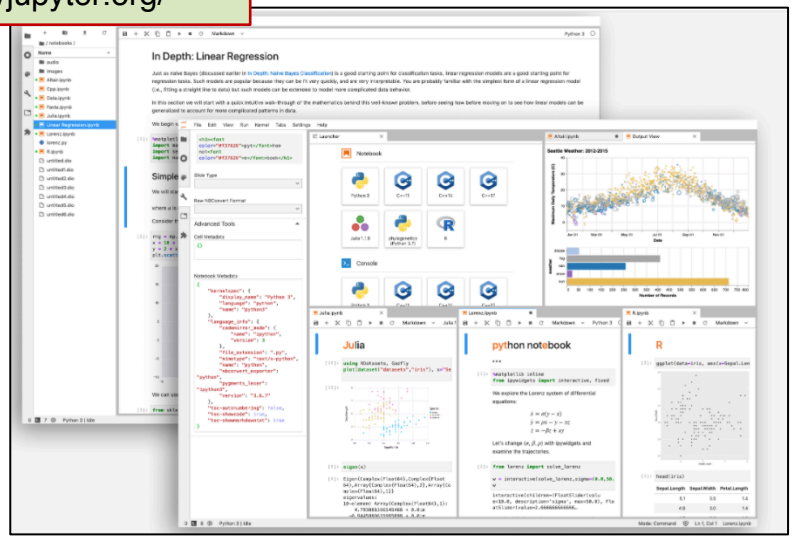
- What is Interactive High-Performance Computing?
- Types of Interactive Computing
- How to Access an Interactive Compute Node
- How to Run Applications
- How to Run Jupyter Notebooks

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 computing** involves *real-time* user inputs to perform tasks on a set of compute node(s) including:
  - Code development, real-time data exploration, and visualizations.
  - Used when applications have large data sets or are too large to download to local device, software is difficult install, etc.
  - User inputs come via command line interface or application GUI (Jupyter Notebooks, Matlab, R-studio).
  - Actions performed on remote compute nodes as a result of user input or program out.

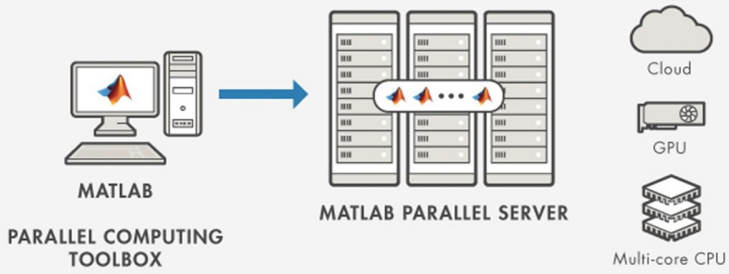
Interactive HPC Scenarios

https://jupyter.org/



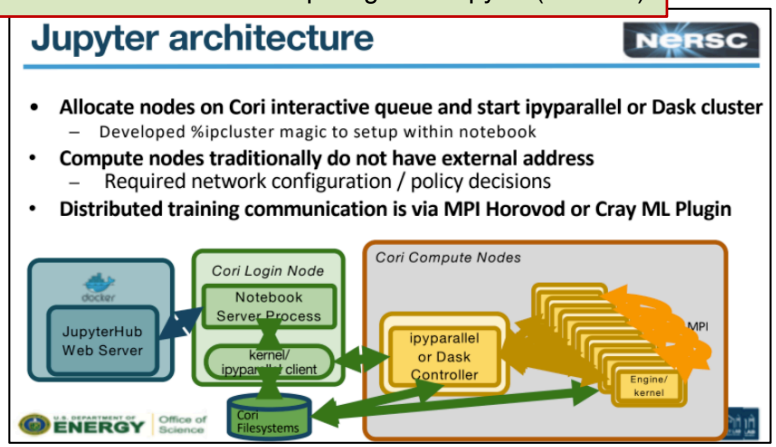
Parallel Matlab (AWS)

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



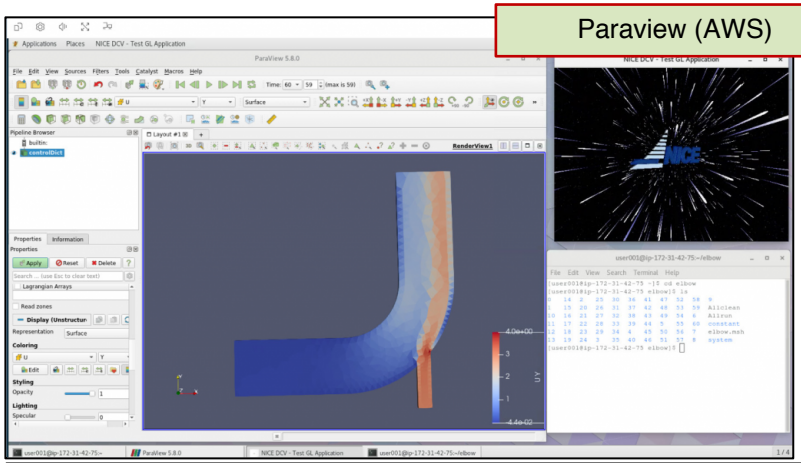
https://azuremarketplace.microsoft.com/en-us/marketplace/apps/mathworks-inc.matlab-parallel-server-listing?tab=Overview

Interactive Distributed Computing with Jupyter (NERSC)



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

Paraview (AWS)



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

Accessing Interactive Compute Nodes on Expanse

- Use the *srun* command to obtain nodes for 'live,' command line interactive access:
- Notes:
  - account info is required
  - Start your testing using shared or debug partitions

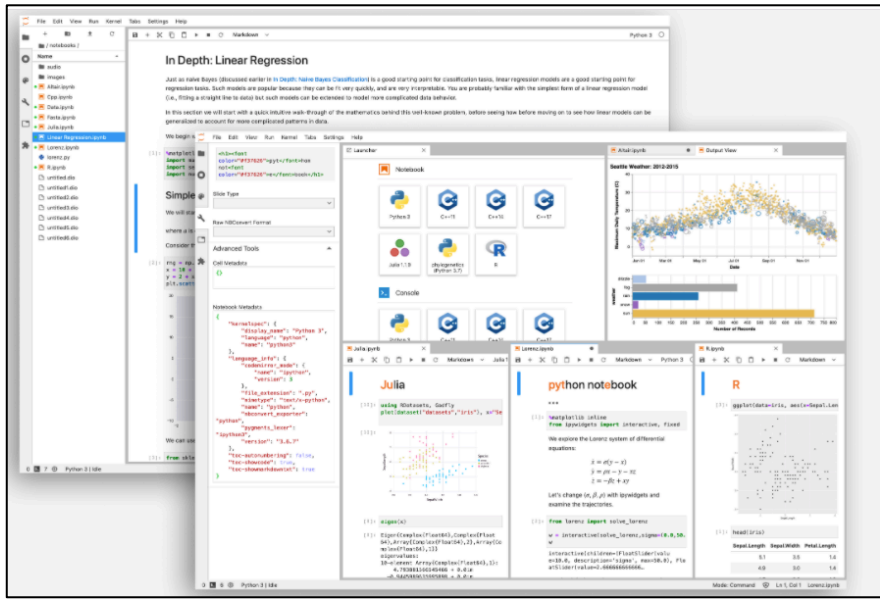
CPU	`srun --partition=shared --pty --nodes=1 --ntasks-per-node=16 --mem=248 -t 00:30:00 --wait=0 --export=ALL --account=abc123 /bin/bash`
GPU	`srun --partition=gpu-shared --pty --nodes=1 --ntasks-per-node=16 --mem=374 --gpus=1 -t 00:30:00 --wait=0 --export=ALL --account=abc123 /bin/bash`

Running Juptyer Services on HPC Systems at SDSC

- What is Jupyter?

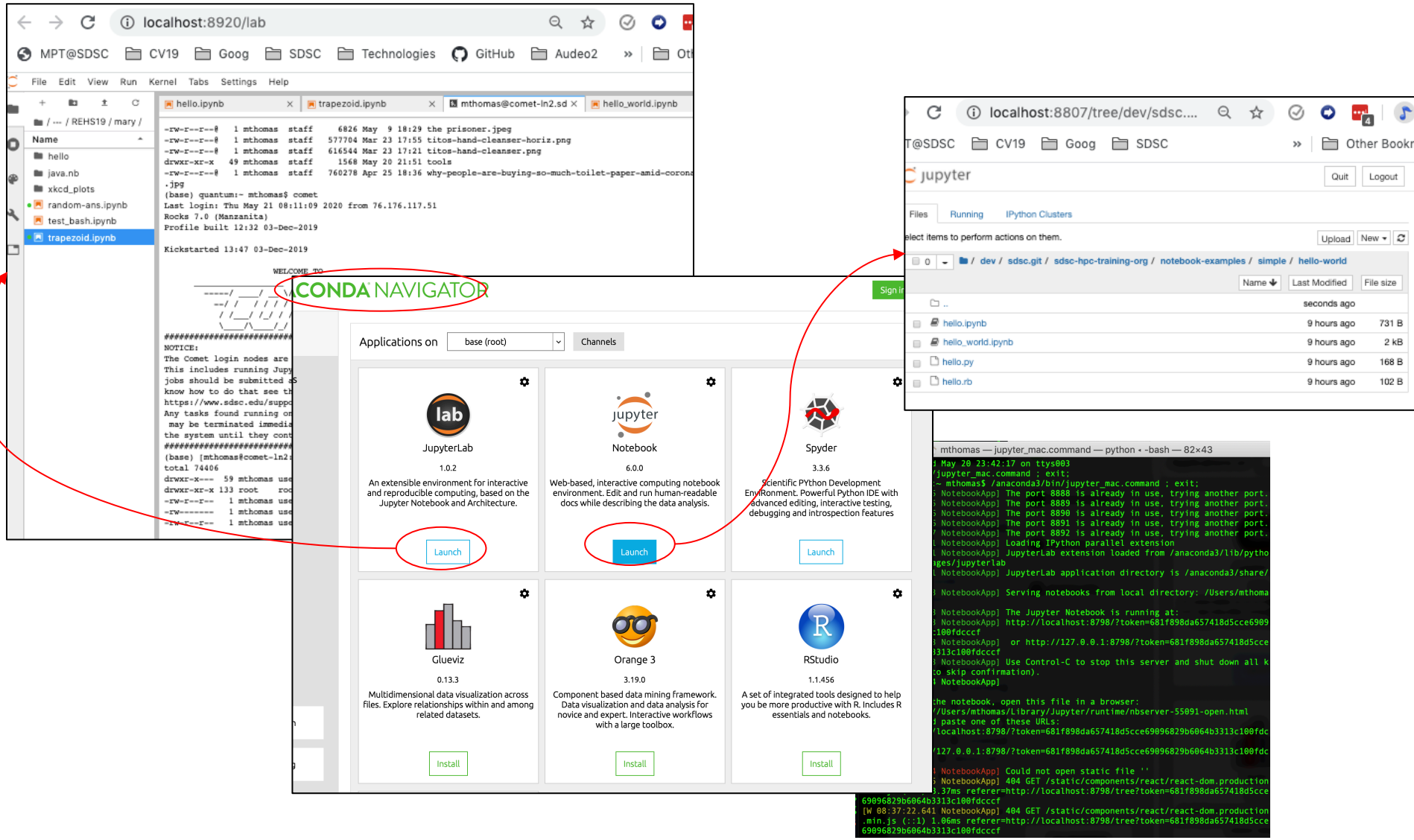
  Jupyter *is a free, open-source, interactive web tool known as a computational notebook, which researchers can use to combine software code, computational output, explanatory text and multimedia resources in a single document.* *(J. Perkel, https://www.nature.com/articles/d41586-018-07196-1)*

- Common Jupyter Services:
  - Jupyter Notebooks (single user)
  - JupyterLab: advanced version of notebook
  - JupyterHub: multiuser Jupyter service
- To run Juptyer services:
  - Laptop/local host, use Anaconda:
    - https://docs.anaconda.com/anaconda/install/
  - HPC systems: use miniconda:
    - https://docs.conda.io/en/latest/miniconda.html



Related Tutorial: https://hpc-training.sdsc.edu/notebooks-101/notebook-101.html

Using Anaconda on Local Host or Laptop



HPC Systems: Use Conda

- Conda: open-source package and environment management system:
  - Runs on Windows, macOS, and Linux.
  - Less overhead than Anaconda
- 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 Expanse, see Notebooks tutorial:
  - https://hpc-training.sdsc.edu/notebooks-101/notebook-101.html#software-prerequisites
  - Note: do not put the conda activation code into the .bashrc or .bash_profile files
    - This ends up making the login process extremely long,
- When you are ready to use notebooks, run these commands

```
[user@login01 ~]. /home/user/miniconda3/etc/profile.d/conda.sh
[user@login01 ~]conda activate
(base) [user@login01 ~]
```

- Conda Cheat Sheet:
  - https://kapeli.com/cheat_sheets/Conda.docset/Contents/Resources/Documents/index

Conda Environments

(https://docs.conda.io/projects/conda/en/latest/user-guide/concepts/environments.html)

- You can load the conda environment in 2 ways: at login (from your .bash_profile) or via command line (when you are ready to use conda, run Juptyer)
  - On installation of miniconda, the code will install the activation commands in your .bashrc file automatically. This takes time which delays your login process.
  - SDSC recommends that you remove those lines and save in a text file.
- The example below shows how to do this:

```
Last login: Thu Mar  4 06:42:34 2021 from 76.176.117.51
[user@login01 ~]$ cat conda-activate.txt
### Commands to activate Conda environment
. /home/user/miniconda3/etc/profile.d/conda.sh
conda activate
[user@login01 ~]$ . /home/user/miniconda3/etc/profile.d/conda.sh
[user@login01 ~]$ conda activate
(base) [user@login01 ~]$
```

- The (base) prefix to the command line prompt shows that we are running in the base conda environment. Note that (base) was not present until the conda activation commands were run.

Custom Conda Virtual Environments

- Use conda to create a virtual environment
  - $ conda create --name example_env
- To see which virtual environments you've created:

```
[user@login01 ~]conda env list
```

- To use a particular virtual environment (e.g., one named 'example_env'), run:

  $ source activate example_env

  (Note: don't use 'conda activate)

Conda Enviromnet Example

```
(base) [user@login01 ~]$ conda list
# packages in environment at /home/user/miniconda3:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                       main
argon2-cffi               20.1.0          py38h7b6447c_1
[SNIP 20-30 lines]
zlib                      1.2.11          h7b6447c_3
(base) [user@login01 ~]$ conda create --name demo_env
Collecting package metadata (current_repodata.json): done
Solving environment: done
## Package Plan ##
  environment location:
/home/user/miniconda3/envs/demo_env
Proceed ([y]/n)? y
Preparing transaction: done
Verifying transaction: done
Executing transaction: done
#
# To activate this environment, use
#
#     $ conda activate demo_env
#
# To deactivate an active environment, use
#
#     $ conda deactivate
```

```
(base) [user@login01 ~]$ conda activate demo_env
(base) [user@login01 ~]$ conda list
# packages in environment at /home/ user
/miniconda3/envs/demo_env:
#
# Name                    Version                   Build  Channel
(demo_env) [user@login01 ~]$ conda install numpy
Collecting package metadata (current_repodata.json): done
Solving environment: done
[SNIP][several mins later]
(demo_env) [user @login01 ~]$ conda list
# packages in environment at /home/ user
/miniconda3/envs/demo_env:
#
# Name                    Version                   Build  Channel
_libgcc_mutex             0.1                       main
blas                      1.0                       mkl
[SNIP]
xz                        5.2.5           h7b6447c_0
zlib                      1.2.11          h7b6447c_3
(demo_env) [user @login01 ~]$
(demo_env) [user @login01 ~]$conda deactivate
(base) [user @login01 ~]$ conda deactivate
 [user@login01 ~]$
```

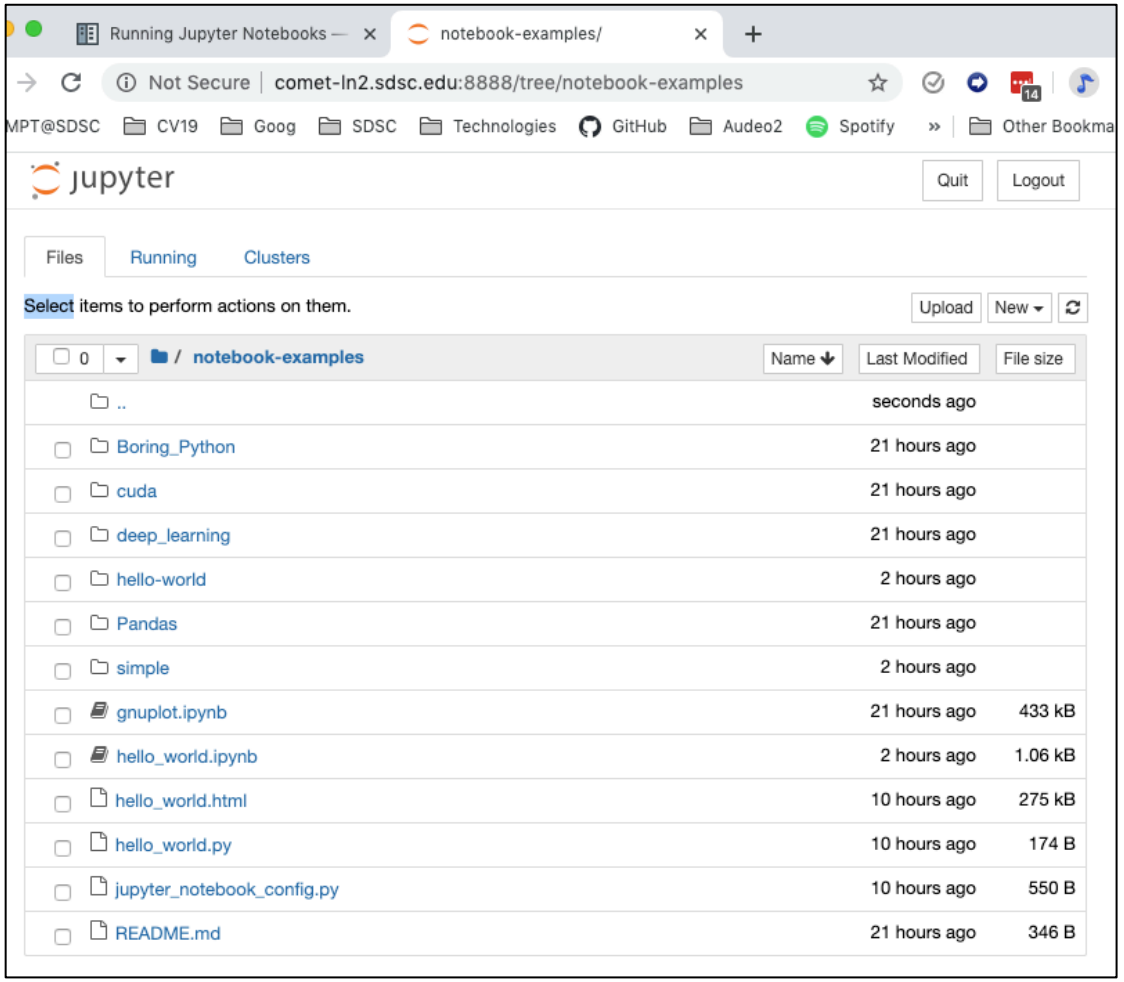
Installing Jupyter Servers

- To run Jupyter servers (notebooks, jupyterlab), 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)

Jupyter Has a Key Vulnerability: Jupyter Servers Provide Access to HPC File Systems

SDSC Jupyter Services Policy:

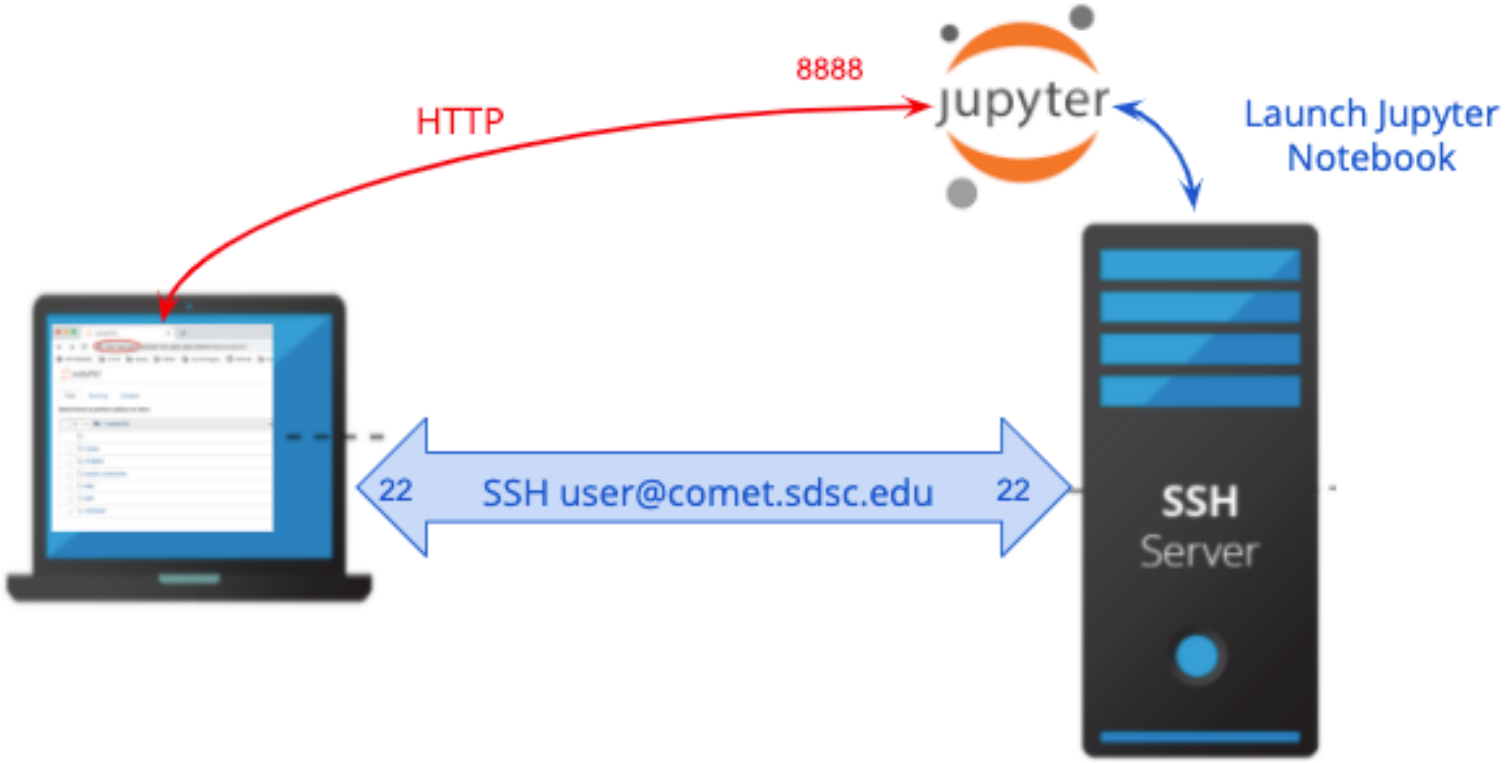
- Portals, JupyterHub, and other services cannot be mounted directly to disk (must be on VM or external)
  - Many use root in vulnerable ways
  - If a user launches Jupyter Lab or Notebooks, the jobs will be killed.
- No applications can run on login nodes
- SDSC recommendation:
  - use secure connections: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



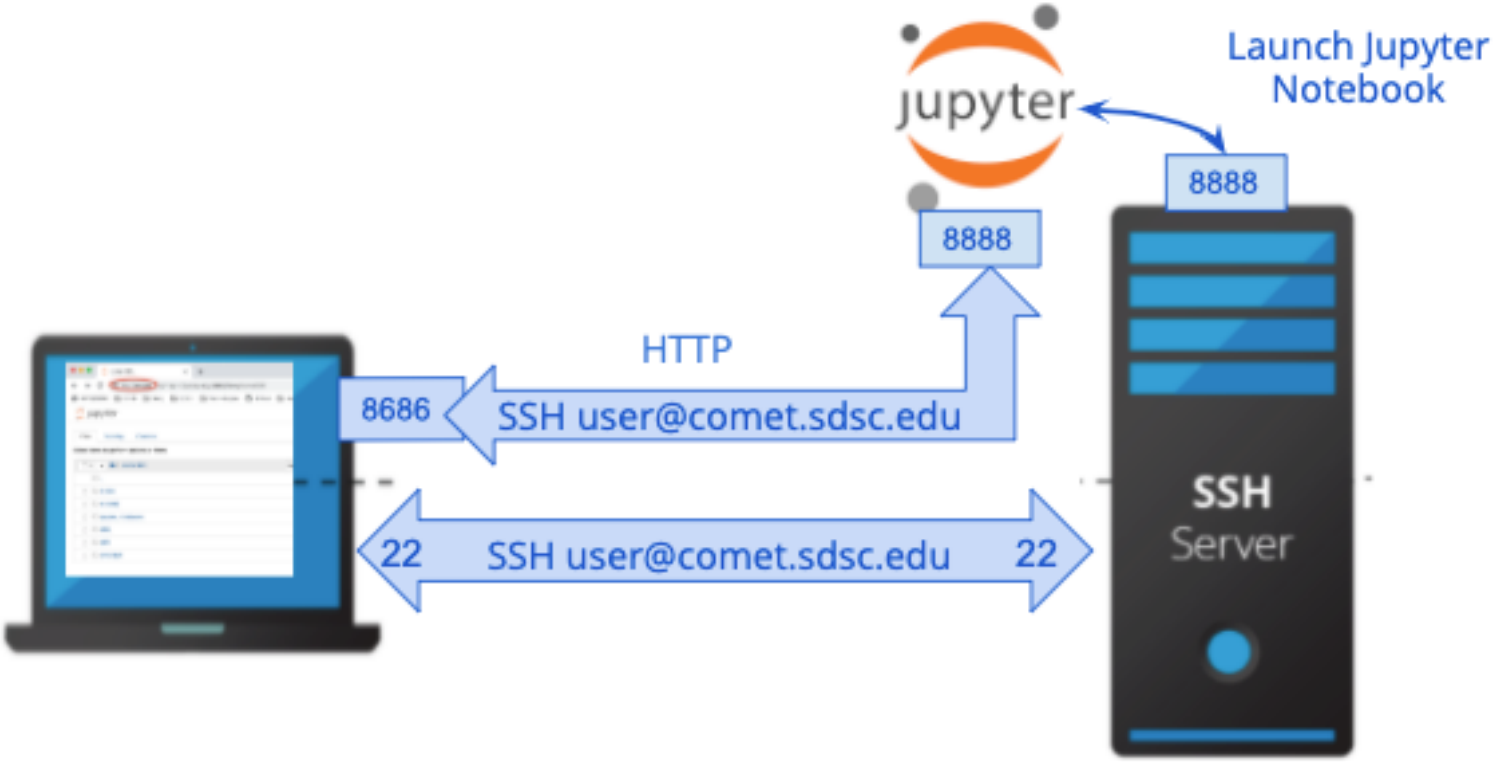
Methods for Running Notebooks on Comet/Expanse

- 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)

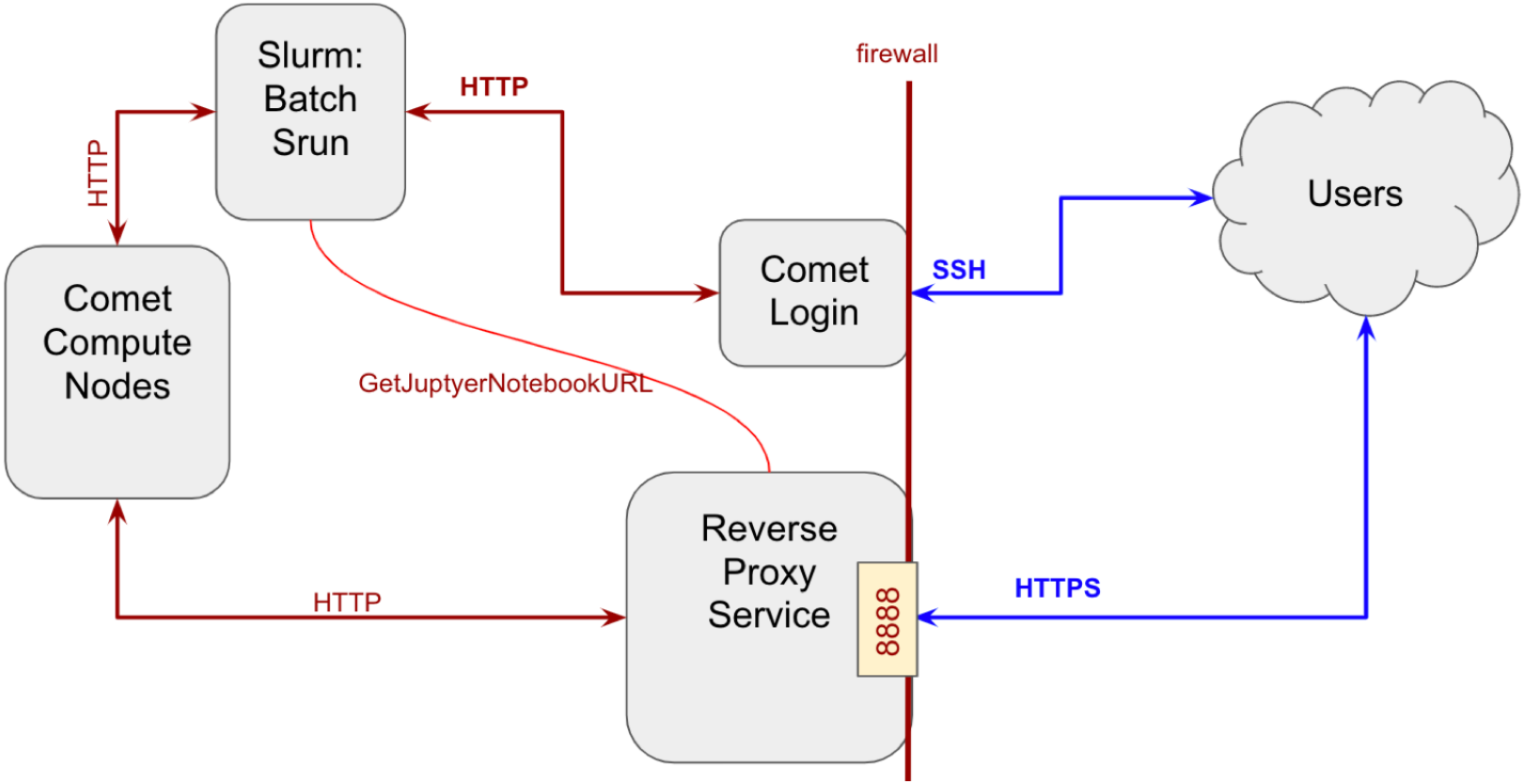


Secure Connection over SSH Tunneling
(secure but inconvenient and unstable)



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

SDSC Reverse Proxy Service (RPS)
(secure, convenient and stable)



Documentation: https://hpc-training.sdsc.edu/notebooks-101/

GitHub Repo: 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 an Expanse login node: login.expanse.sdsc.edu
  - 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 installation instructions:
  - https://hpc-training.sdsc.edu/notebooks-101/notebook-101.html#software-prerequisites

Install Reverse Proxy Scripts

```
[user@login01 ~]$ git clone https://github.com/sdsc-hpc-training-org/reverse-proxy.git
Cloning into 'reverse-proxy'...
remote: Enumerating objects: 265, done.
remote: Counting objects:  63% (167/265)
remote: Counting objects: 100% (265/265), done.
remote: Compressing objects: 100% (167/167), done.
remote: Total 969 (delta 163), reused 192 (delta 97), pack-reused 704
Receiving objects: 100% (969/969), 7.60 MiB | 16.28 MiB/s, done.
Resolving deltas: 100% (553/553), done.
[user@login01 ~]$ Cloning into 'reverse-proxy'...
-bash: Cloning: command not found
[user@login01 ~]$
[user@login01 ~]$ cd reverse-proxy/
[user@login01 reverse-proxy]$ ls -al
total 118
drwxr-xr-x  9 user abc123   14 Mar  4 06:47 .
drwxr-x--- 28 user abc123   48 Mar  4 06:47 ..
drwxr-xr-x  2 user abc123    4 Mar  4 06:47 batch
-rw-r--r--  1 user abc123  588 Mar  4 06:47 .config
-rw-r--r--  1 user abc123 6148 Mar  4 06:47 .DS_Store
drwxr-xr-x  2 user abc123    5 Mar  4 06:47 .examples_images
drwxr-xr-x  8 user abc123   13 Mar  4 06:47 .git
-rw-r--r--  1 user abc123   12 Mar  4 06:47 .gitignore
drwxr-xr-x  2 user abc123    9 Mar  4 06:47 lib
-rw-r--r--  1 user abc123 5981 Mar  4 06:47 README.md
drwxr-xr-x  2 user abc123    5 Mar  4 06:47 slurm
drwxr-xr-x  2 user abc123    4 Mar  4 06:47 slurm_expanse
-rwxr-xr-x  1 user abc123 9555 Mar  4 06:47 start-jupyter
drwxr-xr-x  2 user abc123    4 Mar  4 06:47 torque
```

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 service` command using the token and port number.
- Prints a secure URL (https) 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.

Using start-jupyter

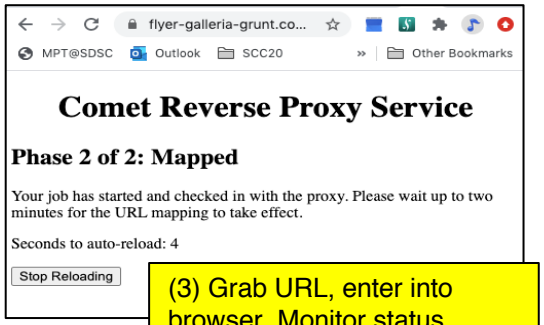
base == active conda env

(1) Launch start-notebook process

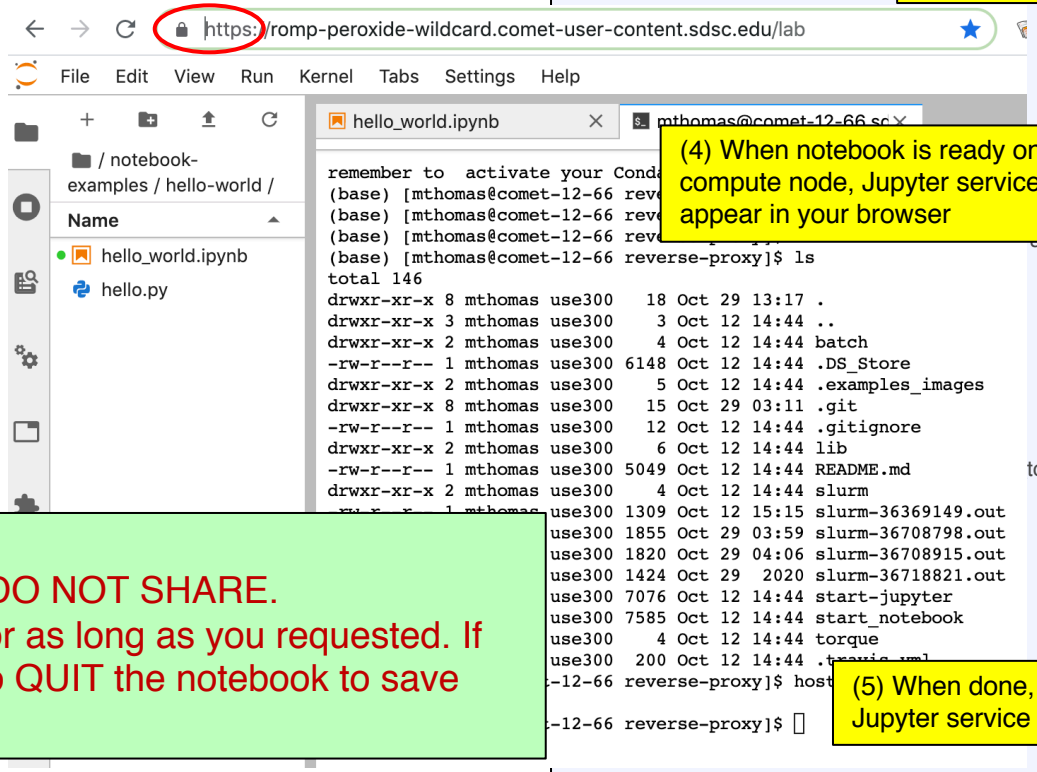
```
(base) [user@comet-ln3 reverse-proxy]$ ./start-jupyter -b slurm/jupyterlab.sh
Your notebook is here:
https://headscarf-barber-unframed.comet-user-content.sdsc.edu?token=3b587a19a8c4ba79f3af754e9fa2a8f5
Using default partition: compute
No time given. Default is 30 mins
```

```
(base) [user@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
#SBATCH --ntasks-per-node=24

##### Define modules here. ####
```

(2) start-notebook builds batch script and submits job to Slurm queue; prints out the URL.



(3) Grab URL, enter into browser. Monitor status window:
If the queue is busy this may take a long time



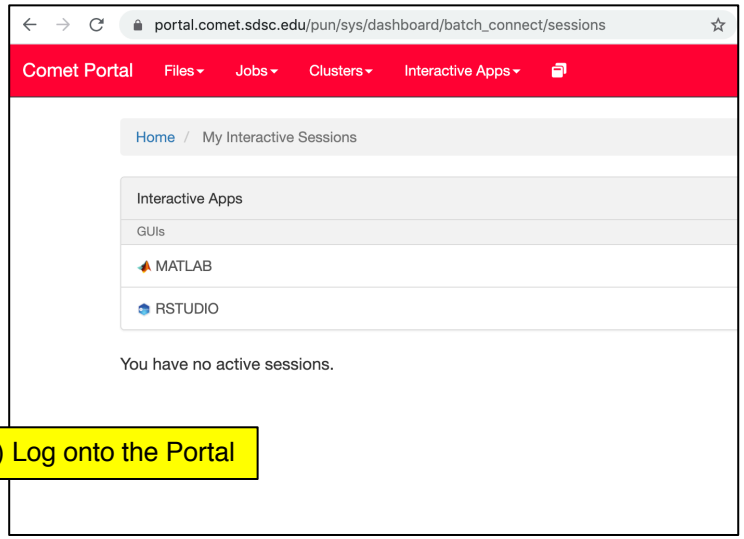
(4) When notebook is ready on compute node, Jupyter service will appear in your browser

(5) When done, be sure to QUIT the Jupyter service – this will kill the job.

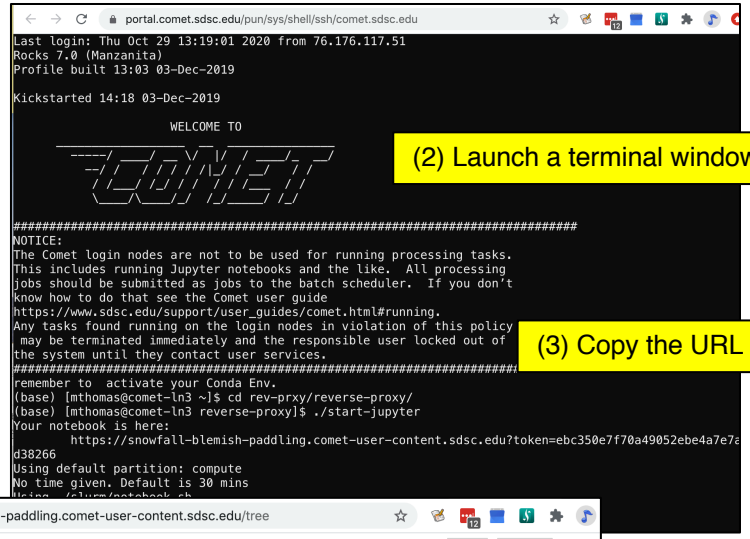
Security Note:

- 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: Terminal Window

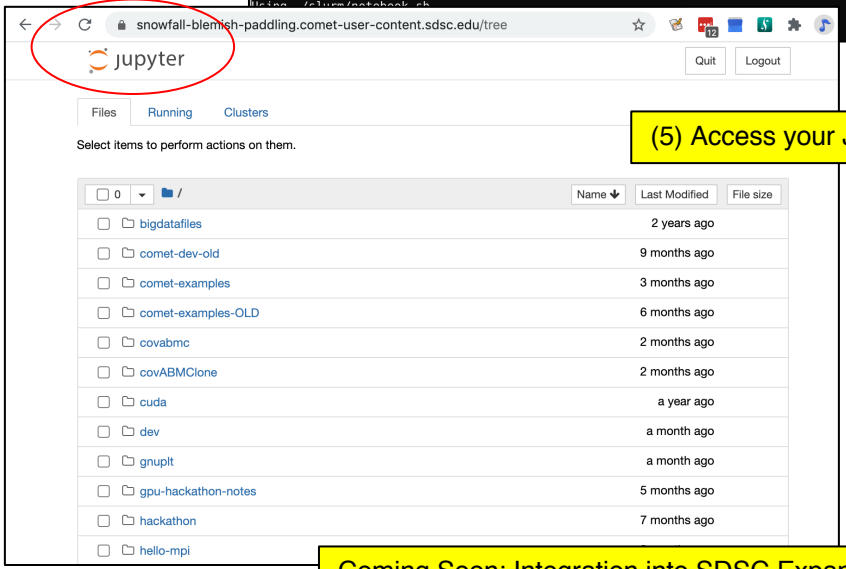


(1) Log onto the Portal

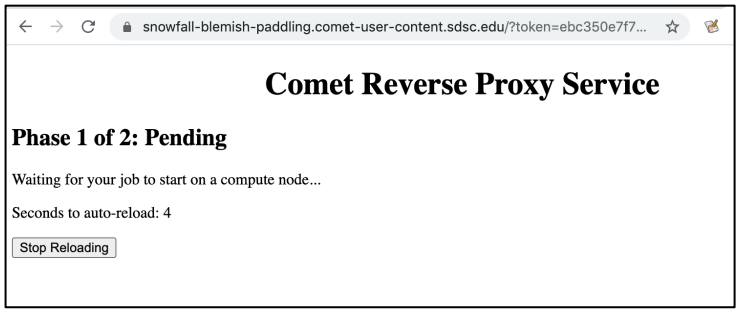


(2) Launch a terminal window

(3) Copy the URL



(5) Access your Jupyter Service



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

Coming Soon: Integration into SDSC Expanse Portal

Thank You

Resources

- Expanse User Guide
  - https://www.sdsc.edu/support/user_guides/expanse.html
- GitHub Repo for this workshop:
  - https://github.com/sdsc-hpc-training-org/comet-to-expanse-transition/tree/main/2021-Comet-to-Expanse-Transition-Tutorial
- GitHub Repo for Expanse Notebooks:
  - https://hpc-training.sdsc.edu/notebooks-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/