

# Running Secure Jupyter Notebooks Using the Satellite Proxy Service

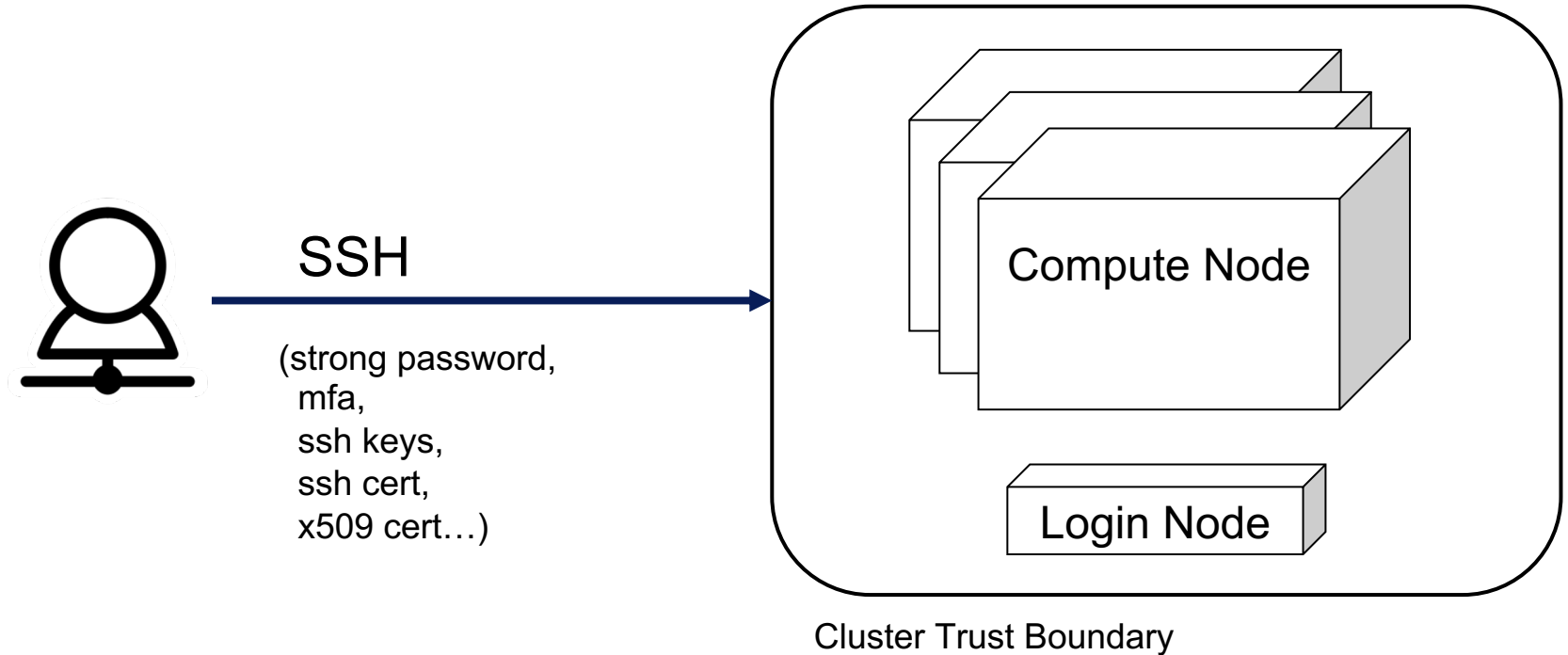
Presented by:  
Mary Thomas (mpthomas at ucsd.edu)

CIML Summer Institute  
6/18/2021

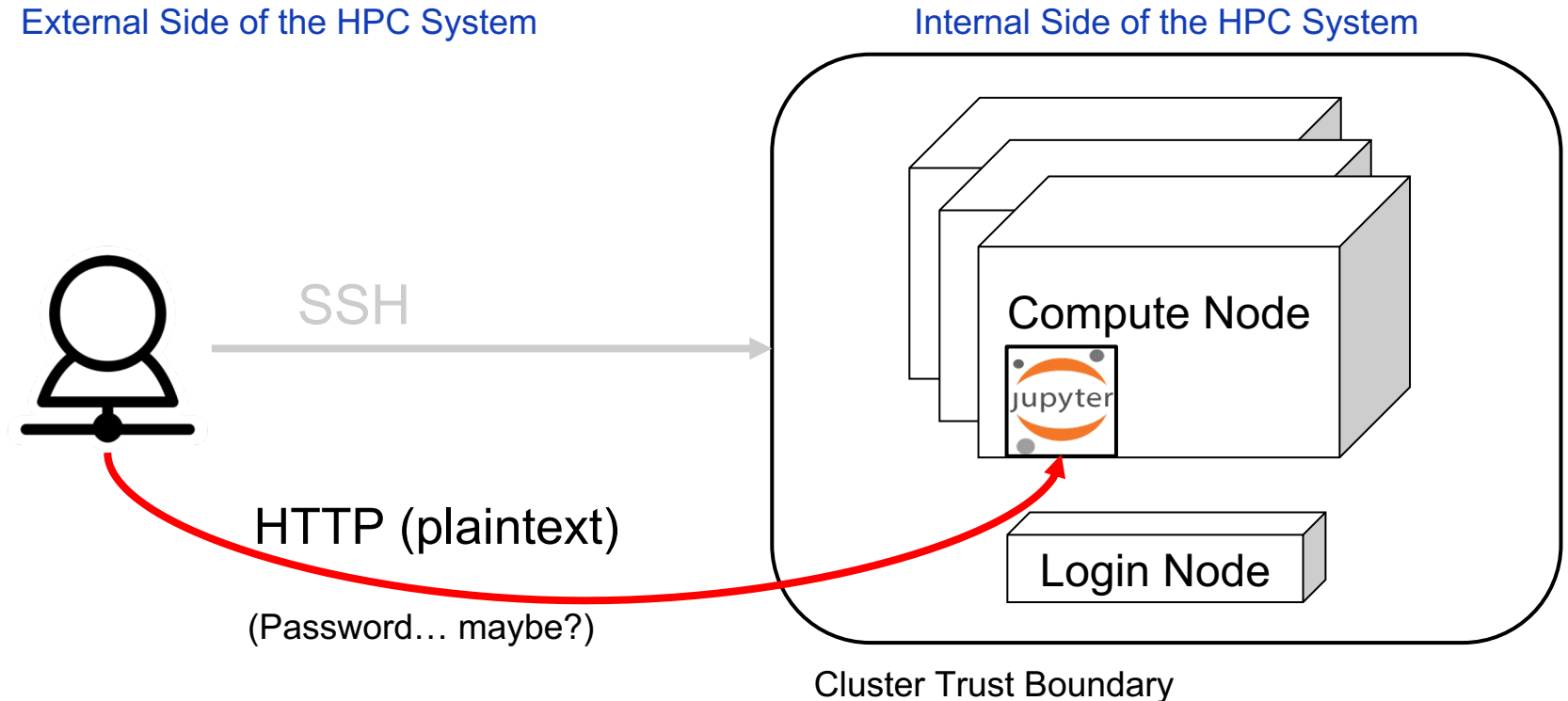
# Introduction: Secure Connections to HPC Systems are Important

External Side of the HPC System

Internal Side of the HPC System



# Introduction: Jupyter Notebooks are Popular



But provide a plaintext back-door to the system

# Motivation: Make Doing A Right Thing Easier than The Wrong Things

## A Wrong Thing: Plaintext to Compute Node

- Submit batch job.
- Wait till job runs.
- Figure out what node it's on.
- Point web browser at node.

## A Right Thing: Improve secure access:

- Invoke the Satellite Reverse Proxy Service
- Point browser at secure, encrypted URL (HTTPS).
- (Wait until Jupyter Notebook shows up.)

# SDSC Satellite Reverse Proxy Service

Just Two Components!

- **Satellite**: a self-service HTTP(s) reverse-proxy.
- **Satellite Client**: a shell-based utility to orchestrate a user's interaction with both Satellite and Slurm to start a Jupyter session within a batch job.

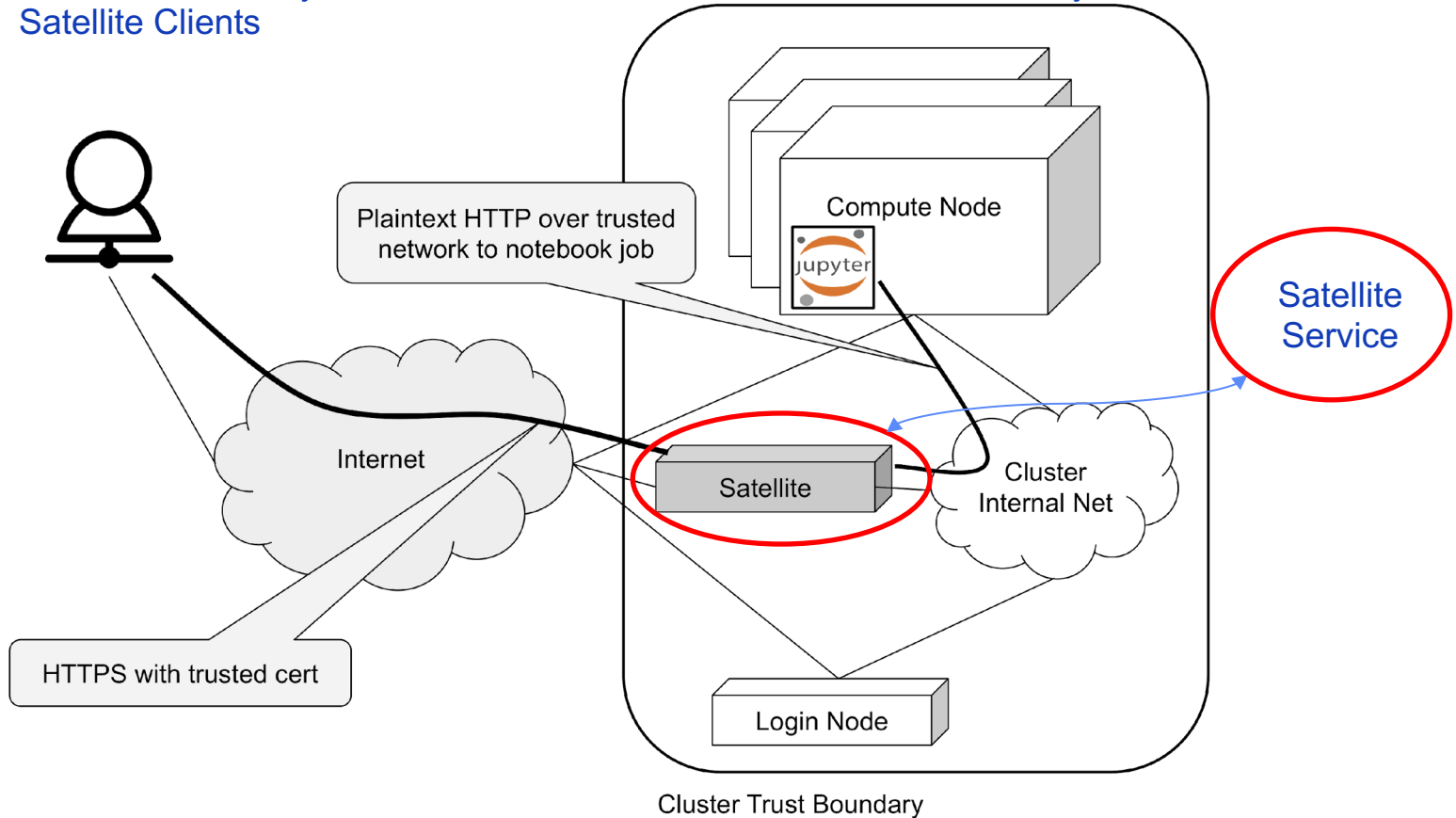
# SDSC Satellite Proxy Service

GitHub Repo: <https://github.com/sdsc-hpc-training-org/reverse-proxy>

# Jupyter Notebooks With Satellite

External Side of the HPC System:  
Satellite Clients

Internal Side of the HPC System



# Satellite Proxy Service Design

## Goals/Requirements/...

- Satellite must serve proxied content over HTTPS, using a certificate signed by a CA approved by CA/Browser Forum.
- Users must be able to create and destroy their own services (proxy mappings), no admin/user support involved.
- Each instance (mapping) must be served from a distinct URL (sub-domain).
- Internal side of the proxy may be over HTTP, but only on a network already trusted for plaintext. (e.g. NFS home directories)
- Satellite code should be designed to minimize unmanaged dependencies. (Dependencies should be available through system package manager.)
- Satellite API must be usable with only wget/curl.



# SDSC Satellite Clients

# start-jupyter

- 1st generation shell utility developed to orchestrate a user's interaction with both Satellite and Slurm to start a Jupyter session within a batch job.
- Key features in design:
  - User calls `start-jupyter` launch script, which requests token from Satellite, passes token to batch job script and submits the job to Slurm; token redeemed from batch job once it runs
  - Provided user with a prefabricated set of batch job scripts to choose from for certain popular applications on each system; user could modify/make their own custom batch job script
  - Small custom shell function library to make code more reusable
  - Currently runs on: Expanse, Comet, TSCC, TSCC Stratus

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

# galileo

- 2nd generation shell utility developed to orchestrate a user's interaction with both Satellite and Slurm to start a Jupyter session within a batch job.
- Developed while reviewing start-jupyter codebase to sort out how best to support Expanse (OOD) Portal and HPC User Services Group long-term; effectively recycled existing an SSH tunneling orchestration utility to use Satellite proxy service instead.
- Key features in design:
  - Recreate same interactions with Satellite service.
  - Increase flexibility for users to configure software environment; but also try to make it simpler for them to do themselves
  - Batch job script is generated completely on-the-fly.
  - Command-line argument driven.
  - Quiet mode for OOD portal

<https://github.com/mkandes/galileo>

# galileo demo examples on Expanse

# Location of galileo directory on Expanse

```
export PATH="/cm/shared/apps/sdsc/galileo:${PATH}"
```

# Example 1: Launch a Jupyter Notebook session on a single CPU core in the 'debug' partition on Expanse using the 'base' Anaconda3 software environment provided as part of Expanse's standard software modules.

```
/cm/shared/apps/sdsc/galileo/galileo.sh launch --account 'use300' --partition 'debug' --cpus-per-task 1 --memory-per-node 1 --time-limit 00:30:00 --jupyter 'notebook' --notebook-dir "/exppanse/lustre/projects/use300/${USER}" --env-modules 'cpu,gcc,anaconda3' --conda-env 'base' --quiet
```

# Example 2: Launch a JupyterLab session on a single GPU in the 'gpu-debug' partition on Expanse using the latest PyTorch Singularity container available.

```
galileo.sh launch --account 'use300' --partition 'gpu-debug' --cpus-per-task 10 --memory-per-node 93 --gpus 1 --time-limit 00:30:00 --jupyter 'lab' --notebook-dir "/exppanse/lustre/projects/use300/${USER}" --env-modules 'singularitypro' --sif '/cm/shared/apps/containers/singularity/pytorch/pytorch-gpu.sif' --bind '/exppanse,/scratch' --nv --quiet
```

# Running GPU notebook using galileo.sh

Step 1:  
Login and setup user  
environment

Step 2:  
Run command to  
launch secure  
notebook

Step 3:  
Copy URL; paste into  
browser on local  
system

Step 4:  
Monitor Slurm queue;  
wait for job to start

```
[mthomas@login01 ~]$ which galileo.sh
/usr/bin/which: no galileo.sh in (/cm/shared/apps/spack/cpu/opt/spack/linux-centos8-zen/gcc-8.3.1
[SNIP]
home/mthomas/.local/bin:/home/mthomas/bin)
[mthomas@login01 ~]$ export PATH="/cm/shared/apps/sdsc/galileo:${PATH}"
[mthomas@login01 ~]$ which galileo.sh
/cm/shared/apps/sdsc/galileo/galileo.sh
[mthomas@login01 ~]$ squeue -u mthomas
      JOBID PARTITION  NAME   USER ST  TIME  NODES NODELIST(REASON)
[mthomas@login01 ~]$ galileo.sh launch --account 'use300' --partition 'gpu-debug' --cpus-per-task 10 --memory-
per-node 93 --gpus 1 --time-limit 00:30:00 --jupyter 'lab' --notebook-dir "/exppanse/lustre/projects/use300/${USER}" -
-env-modules 'singularitypro' --sif '/cm/shared/apps/containers/singularity/pytorch/pytorch-gpu.sif' --bind
'/exppanse,/scratch' --nv --quiet

https://cytoplasm-expansive-shack.exppanse-user-content.sdsc.edu?token=ef077027b1d26dcc1c7d99d889a82b86

[mthomas@login01 ~]$ squeue -u mthomas
      JOBID PARTITION  NAME   USER ST  TIME  NODES NODELIST(REASON)
      3138746 gpu-debug galileo- mthomas PD    0:00      1 (None)
[mthomas@login01 ~]$ squeue -u mthomas
      JOBID PARTITION  NAME   USER ST  TIME  NODES NODELIST(REASON)
      3138746 gpu-debug galileo- mthomas R    0:20      1 exp-7-59
```

- Paste the **HTTPS** URL into a web browser

Satellite Reverse Proxy Service

SDSC Expanse

Job State: Unknown

```

graph LR
    InQueue((In Queue)) --- Running((Running))
    Running --- Mapped((Mapped))
    Mapped --- Proxied((Proxied))
  
```

Satellite Reverse Proxy Service

SDSC Expanse

Job State: Mapped

```

graph LR
    InQueue((In Queue)) --- Running((Running))
    Running --- Mapped((Mapped))
    Mapped --- Proxied((Proxied))
  
```

Satellite Reverse Proxy Service

SDSC Expanse

Job State: Proxied

```

graph LR
    InQueue((In Queue)) --- Running((Running))
    Running --- Mapped((Mapped))
    Mapped --- Proxied((Proxied))
  
```

**In Queue**  
Job has not yet started.

**Running**  
Job has started, but has not redeemed Satellite Token.

**Mapped**  
Job has redeemed Satellite Token, but no proxy entry exists yet.

**Proxied**  
Proxy entry created, ready to go!

**Dead**  
Job died or exited, no further progress will occur.

File Edit View Run Kernel Tabs Settings Help

/ notebook-examples / Pandas /

Name	Last Modified
dataframeexample.PNG	a month ago
graphexample2.png	a month ago
PandasCSV.ipynb	an hour ago
popupexample.PNG	a month ago
seriesexample.PNG	a month ago
sheetexample.PNG	a month ago
TrendsData.txt	a month ago
TrendsDataMissing.csv	a month ago

## Automate the Boring Stuff

### Chapter 4: Lists

Sarah Xie, Leo, Dhruv Kumar

Adapted from <https://automatetheboringstuff.com>

One more topic you'll need to understand before you can begin with tuples can contain multiple values, which makes writing programs other lists, you can use them to arrange data into hierarchical structures.

In this chapter, I'll discuss the basics of lists. I'll also teach you about I'll briefly cover the sequence data types (lists, tuples, and strings to the dictionary data type.

### The List Data Type ¶

A list is a value that contains multiple values in an ordered sequence variable or passed to a function like any other value), not the value string values are typed with quote characters to mark where the closing square bracket, []. Values inside the list are also called iter example, enter the following into the interactive shell:

```
[15]: >>> [1, 2, 3]
```

# Summary and Future Work

- Satellite:
  - Daemon / Systemd unit for more frequent mapping updates. Current cron-driven process only updates at the top of the minute.
  - Investigate internal PKI to operate over untrusted networks. Requires service support for TLS. (Jupyter Notebooks do!)
  - Develop usage metrics
- galileo:
  - Add system config function to make deployment process simple for other systems, especially when deploying upgrade of galileo
  - Add user config function to make subsequent calls to the same software environment and resource request simpler; i.e., templates
  - Complete integration with Expanse (OOD) portal; add a mechanism to allow OOD to track open notebook sessions
  - Secure other web-based tools through Satellite. e.g., TensorBoard
  - Extend to other HPC systems at SDSC/XSEDE, etc (in collaboration with Jupyter project).
- Jupyter Working Group @ SDSC:
  - Mary Thomas, Scott Sakai, Marty Kandes (SDSC); Rick Wagner (UCSD); James McDougall (intern)

**Questions?**



# Satellite Client Example: `start-jupyter`

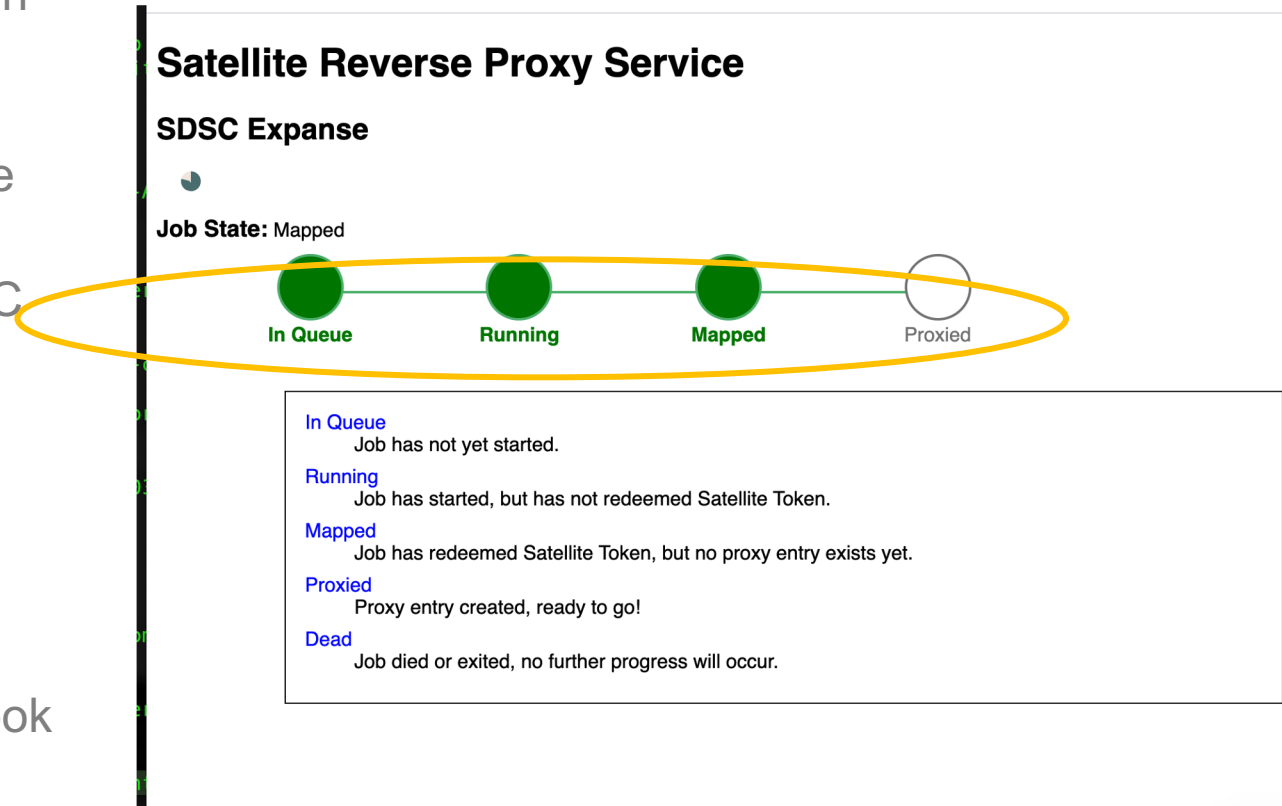
# Running start-notebook

```
(base) [username@login02 reverse-proxy-branch-james-dev]$ ./start-jupyter -A abc123
Your notebook is here:
https://annuity-headphone-aptitude.expense-user-content.sdsc.edu?token=ae0ff01b6780aa32893d6673976769cf
If you encounter any issues, please email help@xsede.org and mention the Reverse Proxy Service.
No time given. Default is 30 mins
Using ./slurm-expense/notebook.sh
Your job id is 670505
You may occasionally run the command 'squeue -j 670505' to check the status of your job
(base) [username@login02 reverse-proxy-branch-james-dev]$ squeue -u username -u username
      JOBID PARTITION  NAME  USER ST   TIME  NODES NODELIST(REASON)
      670505  compute notebook username R    0:37    1 exp-1-17
(base) [username@login02 reverse-proxy-branch-james-dev]$ cat slurm-670505.out
[I 09:36:06.377 NotebookApp] Serving notebooks from local directory: /home/username
[I 09:36:06.377 NotebookApp] Jupyter Notebook 6.1.4 is running at:
[I 09:36:06.377 NotebookApp] http://exp-1-17.eth.cluster:8888/?token=...
[I 09:36:06.377 NotebookApp] or http://127.0.0.1:8888/?token=...
[I 09:36:06.377 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).
% Total    % Received % Xferd Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed
100    9  100    9  0    0   52    0 --:--:-- --:--:-- --:--:--   52
Success!
[I 09:37:14.362 NotebookApp] 302 GET /?token=ae0ff01b6780aa32893d6673976769cf (10.21.0.30) 0.36ms
```

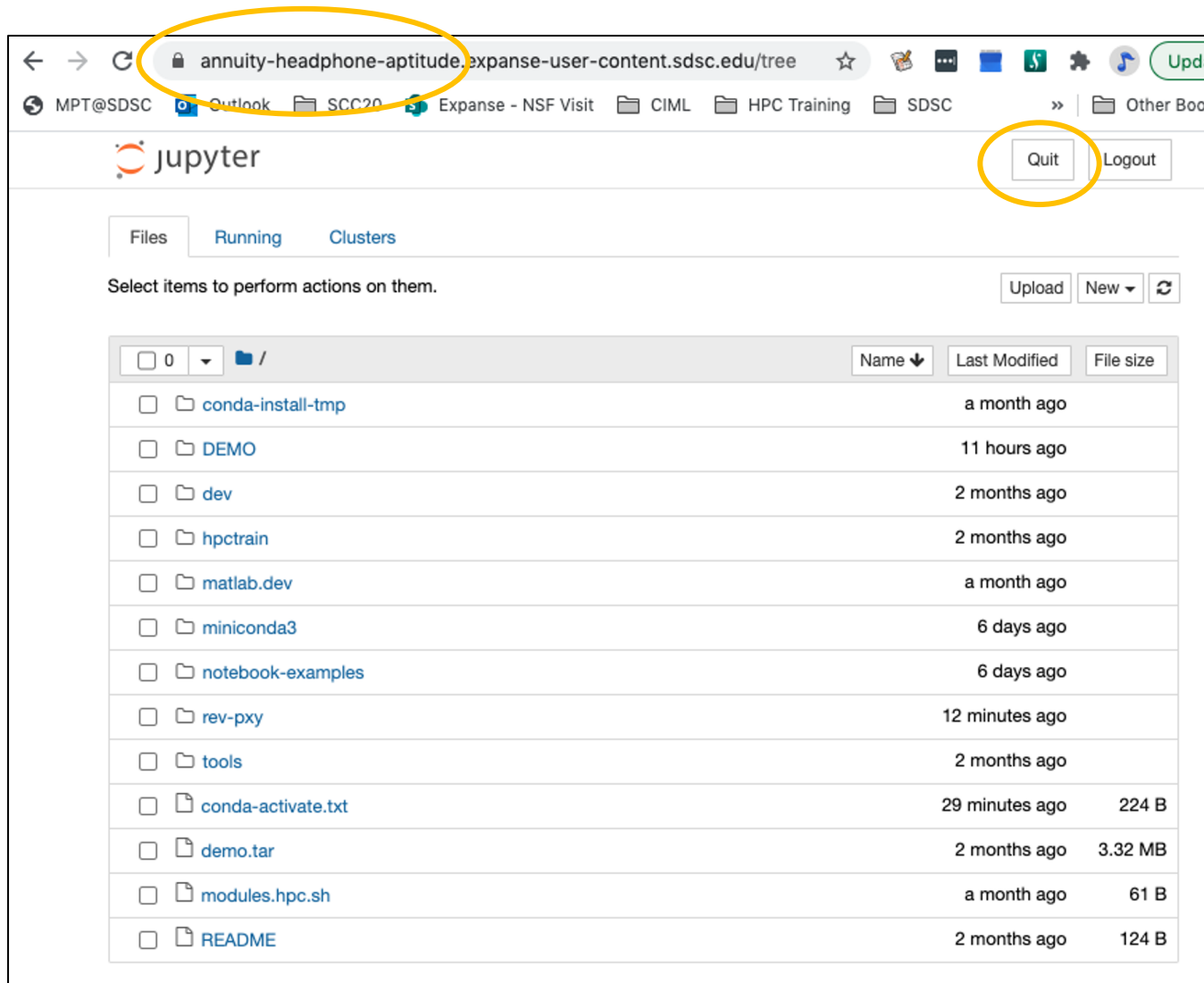
- Paste the **HTTPS** URL into a web browser

# Satellite Server Pening Page

- Load notebook URL in browser; wait for it to launch
- Monitor pending page
- Run the “queue” command on the HPC system to check job status
- If the job queue is busy, it may take a while to launch the notebook
- Treat Jupyter Notebook URL as a password



# Your notebook is launched



The screenshot shows the JupyterLab web interface. The browser address bar displays the URL `annuity-headphone-aptitude.expense-user-content.sdsc.edu/tree`, which is circled in yellow. In the top right corner of the JupyterLab interface, the 'Quit' button is also circled in yellow. Below the 'Quit' button is a 'Logout' button. The main interface shows a file browser with a list of files and folders. The 'Files' tab is selected. The list includes folders like `conda-install-tmp`, `DEMO`, `dev`, `hpctrain`, `matlab.dev`, `miniconda3`, `notebook-examples`, `rev-pxy`, and `tools`, as well as files like `conda-activate.txt`, `demo.tar`, `modules.hpc.sh`, and `README`. The 'Running' tab is also visible.

Name	Last Modified	File size
<input type="checkbox"/> <code>conda-install-tmp</code>	a month ago	
<input type="checkbox"/> <code>DEMO</code>	11 hours ago	
<input type="checkbox"/> <code>dev</code>	2 months ago	
<input type="checkbox"/> <code>hpctrain</code>	2 months ago	
<input type="checkbox"/> <code>matlab.dev</code>	a month ago	
<input type="checkbox"/> <code>miniconda3</code>	6 days ago	
<input type="checkbox"/> <code>notebook-examples</code>	6 days ago	
<input type="checkbox"/> <code>rev-pxy</code>	12 minutes ago	
<input type="checkbox"/> <code>tools</code>	2 months ago	
<input type="checkbox"/> <code>conda-activate.txt</code>	29 minutes ago	224 B
<input type="checkbox"/> <code>demo.tar</code>	2 months ago	3.32 MB
<input type="checkbox"/> <code>modules.hpc.sh</code>	a month ago	61 B
<input type="checkbox"/> <code>README</code>	2 months ago	124 B

When done with the notebook be sure to **shut it down** by quitting the notebook