SDSC Expanse Webinar:
Running Jupyter Notebooks on Expanse & SDSC HPC Systems

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

December 10, 2020
Outline

• Getting Started; Expanse Overview
• What are Jupyter Notebooks?
• Security concerns
  • HTTP vs HTTPS
  • SSH vs SSH tunneling (HTTP)
• Software Requirements for Running Notebooks on Expanse
  • Install conda, conda environments
• Methods for Running Notebooks on Expanse
  • Running notebooks on the Login node or interactive node
• SDSC Jupyter Reverse Proxy Service (JRPS)
• Live Demo
• Key Goals:
  • Learn why it is important to run Jupyter Notebooks securely.
  • Introduce the JRPS to help run them more securely
Basic Information

• Expanse User Guide:
  • https://www.sdsc.edu/support/user_guides/expanse.html

• You need to have an Expanse account in order to access the system. There are two ways to do this:
  • Submit a proposal through the XSEDE Allocation Request System
  • Request a trial account @ https://portal.xsede.org/allocations/startup.

• Online repo for this webinar:
  • https://github.com/sdsc-hpc-training/webinars/tree/master/202004_expanse_101
  • https://github.com/sdsc-hpc-training-org/notebook-examples

• Basic Skills:
  • You must be familiar with running basic Unix commands, connecting to Expanse via SSH, running notebooks, and other basic skills. See:
    • https://github.com/sdsc-hpc-training-org/basic_skills
SDSC HPC Training

• SDSC Training Events:
  • https://www.sdsc.edu/education_and_training/training.html

• HPC Training GitHub repository:
  • https://github.com/sdsc-training-org

• Interactive videos for webinars
  • https://education.sdsc.edu/training/interactive
  • download talks, code, etc.
Obtaining Notebook Examples

(base) [username@login-02:~] git clone https://github.com/sdsc-hpc-training-org/notebook_examples.git
Cloning into 'notebook_examples'...
remote: Enumerating objects: 55, done.
remote: Counting objects: 100% (55/55), done.
remote: Compressing objects: 100% (44/44), done.
remote: Total 55 (delta 6), reused 55 (delta 6), pack-reused 0
Unpacking objects: 100% (55/55), done.
(base) [username@login-02 :~] cd notebook_examples/
(base) [username@login-02 :~] /notebook_examples]
ll
 total 609
  drwxr-xr-x  7 username abc123  9 May 20 12:38 .
  drwxr-x---  58 username abc123  89 May 20 12:38 ..
  drwxr-xr-x  3 username abc123  8 May 20 12:38 Boring_Python
  drwxr-xr-x  4 username abc123  4 May 20 12:38 cuda
  drwxr-xr-x  2 username abc123  4 May 20 12:38 deep_learning
  drwxr-xr-x  8 username abc123  13 May 20 12:38 .git
-rw-r--r--  1 username abc123 432678 May 20 12:38 gnuplot.ipynb
  drwxr-xr-x  2 username abc123  6 May 21 07:34 hello-world
  drwxr-xr-x  8 username abc123 1060 May 20 12:45 hello_world.ipynb
  drwxr-xr-x  2 username abc123  10 May 20 12:38 Pandas
-rw-r--r--  1 username abc123  322 May 20 12:38 README.md
(base) [username@ login-02 :]~/notebook_examples]
WARNING!!!!
Jupyter Notebooks should not be run on the login nodes. Those jobs will be deleted.
EXPA NSE
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

REMOTE CI INTEGRATION

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

SDSC
SAN DIEGO
SUPERCOMPUTER CENTER

UC San Diego
Expanse Heterogeneous Architecture

**System Summary**

- 13 SDSC Scalable Compute Units (SSCU)
- 728 x 2s Standard Compute Nodes
- 93,184 Compute Cores
- 200 TB DDR4 Memory
- 52x 4-way GPU Nodes w/NVLINK
- 208 V100s
- 4x 2TB Large Memory Nodes
- HDR 100 non-blocking Fabric
- 12 PB Lustre High Performance Storage
- 7 PB Ceph Object Storage
- 1.2 PB on-node NVMe
- Dell EMC PowerEdge Direct Liquid Cooled

---

**Hardware Details**

- **Scalable Compute Unit (SSCU)**
  - Non-blocking fabric
  - 56 CPU nodes
  - 4 GPU nodes

- **System Layout**
  - 1 row 7 SSCU
  - 1 row 6 SSCU + Core Mgmt. rack

- **Performance Storage**
  - 12PB Lustre
  - 7 HA OSS pairs
  - 4 NVMe HA Metadata Servers

- **Object Storage**
  - 7 PB Ceph
  - 32 storage servers

---

**Software Features**

- **HDR 100 non-blocking Fabric**
- **12 PB Lustre High Performance Storage**
- **7 PB Ceph Object Storage**
- **1.2 PB on-node NVMe**

---

**Site Information**

- UC San Diego
- San Diego Supercomputer Center
The SSCU is Designed for the Long Tail Job Mix, Maximum Performance, Efficient Systems Support, and Efficient Power and Cooling

**Standard Compute Nodes**
- 2x AMD EPYC 7742 @2.25 GHz
- 128 Zen2 CPU cores
- PCIe Gen4
- 256 GB DDR4
- 1.6 TB NVMe

**GPU Nodes**
- 4x NVIDIA V100/follow-on
- 10,240 Tensor Cores
- 32 GB GDDR
- 1.6 TB NVMe
- Intel CPUs

**SSCU Components**
- 56x CPU nodes
- 7,168 Compute Cores
- 4x GPU nodes
- 1x HDR Switch
- 1x 10GbE Switch
- HDR 100 non-blocking fabric
- Wide rack for serviceability
- Direct Liquid Cooling to CPU nodes

**Non-blocking Interconnect**
1 HDR Switch/SSCU
10x (200 Gbps)  HDR
56x
4x
Compute Nodes
GPU Nodes

5 Level 2 switches
Performance Storage
Cloud Storage
26x (200 Gbps)
Expanse Connectivity Fabric

Facilitates Compute and Data Workflows
AMD EPYC 7742 Processor Architecture

- 8 Core Complex Dies (CCDs).
- CCDs connect to memory, I/O, and each other through the I/O Die.
- 8 memory channels per socket.
- DDR4 memory at 3200MHz.
- PCI Gen4, up to 128 lanes of high speed I/O.
- Memory and I/O can be abstracted into separate quadrants each with 2 DIMM channels and 32 I/O lanes.
- 2 Core Complexes (CCXs) per CCD
- 4 Zen2 cores in each CCX share a 16MB L3 cache. Total of 16x16 = 256MB L3 cache.
- Each core includes a private 512KB L2 cache.

EPYC Architecture has impact on compiling and batch script configuration.
Composable Systems will support complex, distributed, workflows – making Expanse part of a larger CI ecosystem

- Bright Cluster Manager + Kubernetes
- Core components developed via NSF-funded CHASE-CI (NSF Award # 1730158), and the Pacific Research Platform (NSF Award # 1541349)
- Requests for a composable system will be part of an XRAC request
- Advanced User Support resources available to assist with projects - this is part of our operations funding.
Integration with Public Cloud *

• Supports projects that share data, need access to novel technologies, and integrate cloud resources into workflows

• Slurm + in-house developed software + Terraform (Hashicorp)

• Early work funded internally and via NSF E-CAS/Internet2 project for CIPRES (Exploring Cloud for the Acceleration of Science, Award #1904444).

• Approach is cloud-agnostic and will support the major cloud providers.

• Users submit directly via Slurm, or as part of a composed system.

• Options for data movement: data in the cloud; remote mounting of file systems; cached filesystems (e.g., StashCache), and data transfer during the job.

* Funding for user cloud resources is not part of the Expanse award. Researcher must have access to these via other NSF awards and funding.
Expanse Status

• Expanse is in production effective 12/7/2020
• Comet will conclude operations in March 2021.
• You can transfer Comet allocations to Expanse
• New accounts -- Warning
  • When you get your account, you will not inherit any of your Comet environment, directories, scripts, etc.
• Need help? Checkout the transition workshop:
  • https://education.sdsc.edu/training/interactive/202010_comet_to_expanse/index.html
  • Will repeat the workshop in March, 2021
Software Requirements for Running Notebooks on Expanse

Anaconda: desktop application
OS X – Launch Apps with click of a Button

---

**Trapezoid Rule**

First, we define a simple function and sample it between 0 and 10 at 200 points.

```python
import numpy as np
import matplotlib.pyplot as plt

def f(x):
    return (x-3) * (x-5) * (x-7) / 85

x = np.linspace(0, 10, 200)
y = f(x)
```

Choose a region to integrate over and take only a few points in that region.

```python
a, b = 1, 8 # the left and right boundaries
N = 50 # the number of points
xint = np.linspace(a, b, N)
yint = f(xint)
```

Plot both the function and the area below it in the trapezoid approximation.

```python
plt.plot(x, y, lw=2)
plt.axis([0, 9, 0, 140])
plt.fill_between(xint, 0, yint, facecolor='gray', alpha=0.4)
plt.text(0.5 * (a + b), 38, r'''
\int_a^b f(x)dx''', horizontalalignment='center', fontsize=30);
```

Compute the integral both at high accuracy and with the trapezoid approximation.

```python
from __future__ import print_function
from scipy.integrate import quad
```
Software Requirements on HPC Systems

- Not so easy to run notebooks on HPC system/Unix
- Important and convenient to have customized, virtual Python environments,
  - install packages that aren’t installed with the system’s Python installation
  - You need different sets of Python packages for different purposes.
- We recommend that you setup your own local environment:
  - This gives you control over libraries used by your notebooks
  - Use miniconda:
- Optionally: use singularity
  - Install locally
  - Advantage of using containers: everything is built for you to use
  - Disadvantage: not easy to modify
C**onda**

- Conda is an open-source package management system and environment management system (like pip)
- Created for Python programs
  - can package and distribute software for any language.
- Conda Cheat Sheet:
  - [https://kapeli.com/cheat_sheets/Conda.docset/Contents/Resources/Documents/index](https://kapeli.com/cheat_sheets/Conda.docset/Contents/Resources/Documents/index)
- See Notebooks 101 tutorial for installation information
  - Note: installation of conda on HPC systems is known to be very slow, so plan some time for this activity
Use Conda to create a virtual environment

• Use conda to create a virtual environment
  • Choose whatever name you want
  • $ 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 JupyterLab and JupyterNotebooks
A caveat about file systems

- Be aware of where you launch your notebook service:
- Login nodes and the nodes that run batch scripts have access to the user’s home directory, but the compute nodes do not.
- The home directory is where the files that make up the virtual environment are stored by default.
- So if you want to use the virtual environment from a batch script, it either has to run on the batch node (e.g., don’t try to run it via a jsrun command) or you will have to figure out how to force conda to store virtual environments in your $MEMBERWORK directory.
- If you launch the notebook from your home dir, you will not be able to run notebooks from your projects directory
Overview of Jupyter Notebooks
What are Jupyter Notebooks?

- Why do we use them?

https://jupyter.org/
Overview of Jupyter Notebooks

- Community of open-source developers, scientists, educators, and data scientists.
- Goal: build open-source tools and create community that facilitates scientific research, reproducible and open workflows, education, computational narratives, and data analytics.
- Jupyter supports over 100 programming languages, and connects data analytics tools across a range of disciplines and communities.
Jupyter Notebooks

- Web-based interactive computing platform
- Allows users to author computational apps
- Code, equations, narrative text, interactive user interfaces, and other rich media.

- Enables collaborative creation of notebooks
- Can be used across a wide range of disciplines

https://towardsdatascience.com/the-jupyterlab-credential-store-9cc3a0b9356
JupyterLab

- Jupyter’s next-generation interface, JupyterLab facilitates data scientists to compose the interface that suits their needs.
- Flexible, extensible user interface - supports diversity of workflows in data science.
- Runs using same Jupyter server as Notebook interface → allows it to be accessed remotely on shared infrastructure (for example, via a JupyterHub)
Jupyter Env
**JupyterHub**

- Provides remote access to Jupyter servers
- Make high-powered computational environments and resources more accessible to students, researchers, and collaborators.
- Runs in the cloud or on your own hardware
- Makes it possible to serve a pre-configured data science environment to any user in the world.
- Used in education and large-scale courses as well as in collaborative and massively-open data analytics projects.

Jupyter Notebook Security
Warning: The Jupyter Notebook URL is essentially a password

Treat it very carefully
Do NOT Share
Methods for Running Notebooks

JRPS Project Motivation: By default, Notebooks run over HTTPS ➔ insecure
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
- No applications can run on login nodes
  - Jupyter Lab or Notebooks, the jobs will be killed.
- **SDSC recommendation:**
  - Use secure connections: when you choose unsecure connections your account is vulnerable to hacking
Methods for Running Notebooks on Expanse

- **Security concerns**
  - HTTP vs HTTPS
  - SSH vs SSH tunneling (HTTP)
  - Browser cookies

- **Connection scenarios:**
  - Connection to Notebook over HTTP (insecure)
  - Connection to Notebook over SSH tunneling (secure)
  - Connection to Notebook over HTTPS using the Reverse Proxy Service (very secure) [Beta testing]
  - Coming Soon: Galyleo remote notebook launcher

- **Notebooks can be run on the following nodes:**
  - Interactive nodes
  - Compute nodes
  - GPU nodes
Connection over HTTP (unsecure)
Secure but complicated: SSH Tunneling

- Port forwarding via **SSH tunneling** creates a secure connection between a local computer and a remote machine through which services can be relayed.
- Connections are encrypted
- Useful for transmitting information that uses an unencrypted protocol (IMAP, VNC, HTTP server).
- 3 Types:
  - **Local port forwarding** *(will use for notebook servers)*: connections from *SSH client* are forwarded via the *SSH server*, then to a *destination server*.
  - **Remote port forwarding**: connections from the *SSH server* are forwarded via the *SSH client*, then to a *destination server*.
  - **Dynamic port forwarding**: connections from *programs* forwarded via the *SSH client*, then via the *SSH server*, and finally to *destination servers*.

Source: [https://help.ubuntu.com/community/SSH/OpenSSH/PortForwarding](https://help.ubuntu.com/community/SSH/OpenSSH/PortForwarding)
Secure Connection over SSH Tunneling
Using Local Port Forwarding to Connect to a Jupyter Notebook Server

(base) quantum:Docs username$ ssh -L 8888:127.0.0.1:8888 username@expanse.sdsc.edu

(base) [username@expanse-ln2:~] jupyter notebook --no-browser --ip=’/bin/hostname’
[I 12:03:54.005 NotebookApp] JupyterLab extension loaded from 
/home/username/miniconda3/lib/python3.7/site-packages/jupyterlab
[I 12:03:54.005 NotebookApp] JupyterLab application directory is 
/home/username/miniconda3/share/jupyter/lab
[I 12:03:54.497 NotebookApp] Serving notebooks from local directory: /home/username
[I 12:03:54.497 NotebookApp] The Jupyter Notebook is running at:
[I 12:03:54.498 NotebookApp] http://expanse-ln2.sdsc.edu:8888/?token=bc1a7238d7dd6d401cd099a7e863d5bfb6db8a6a7f19a243
[I 12:03:54.498 NotebookApp] or
http://127.0.0.1:8888/?token=bc1a7238d7dd6d401cd099a7e863d5bfb6db8a6a7f19a243
[I 12:03:54.498 NotebookApp] Use Control-C to stop this server and shut down all kernels (twice to skip
confirmation).
[C 12:03:54.505 NotebookApp]

To access the notebook, open this file in a browser:
   file:///home/username/.local/share/jupyter/runtime/nbserver-650-open.html
Or copy and paste one of these URLs:
   http://expanse-ln2.sdsc.edu:8888/?token=bc1a7238d7dd6d401cd099a7e863d5bfb6db8a6a7f19a243
or http://127.0.0.1:8888/?token=bc1a7238d7dd6d401cd099a7e863d5bfb6db8a6a7f19a243
SDSC Reverse Proxy Service (RPS) (beta testing)


GitHub Repo: https://github.com/sdsc-hpc-training-org/reverse-proxy
What is a Reverse Proxy?

- A reverse proxy takes requests from the Internet and forwards them to servers in an internal network. Those making requests to the proxy may not be aware of the internal network.
- Img Source: [Wikipedia reverse proxy](https://en.wikipedia.org/wiki/Reverse_proxy)
RPS Architecture

- **Comet Compute Nodes**
- **Slurm: Batch Srun**
- **Comet Login**
- **Reverse Proxy Service**
- **Users**

Connections:
- HTTP from Comet Compute Nodes to Slurm: Batch Srun
- HTTP from Slurm: Batch Srun to Comet Compute Nodes
- HTTP from Comet Compute Nodes to Reverse Proxy Service
- HTTPS from Reverse Proxy Service to Users
- SSH from Users to Comet Login
- GetJupyterNotebookURL from Comet Compute Nodes to Reverse Proxy Service
SDSC Jupyter Reverse Proxy Service Overview

- RPS is a prototype system that will allow users to launch standard Jupyter Notebooks on any Expanse compute node using a reverse proxy server.
- The notebooks will be hosted on the internal cluster network as an HTTP service using standard jupyter commands.
- The service will then be made available to the user outside of the cluster firewall as an HTTPS connection between the external users' web browser and the reverse proxy server.
- The goal is to minimize software changes for our users while improving the security of user notebooks running on our HPC systems.
- The RP service is capable of running on any HPC system capable of supporting the RP server (needs Apache)
SDSC Reverse Proxy Service Overview

• Using RPS is very simple and requires no tunneling and is secure (produces HTTPS URLs).

• To use RPS:
  • SSH to an expanse login node.
  • Activate your conda environment
    • From .bashrc script or command line
  • Clone the Repo:
    ```
    git clone https://github.com/sdsc-hpc-training-org/reverse-proxy.git
    ```
  • Check software environment on the login node:
    • conda, Jupyter (notebooks, lab), and other Python packages needed for you application.
Activate your Conda environment

- Miniconda installation recommends that you put these lines into your .bashrc script, however this can take a long time and slow down login process

```bash
### Activate Conda environment
# . /home/username/miniconda3/etc/profile.d/conda.sh
# conda activate  # commented out by conda initialize
### move conda install info to separate file
```

- If you don’t need conda every time you login, run the activation from the command line

```bash
[username@login02 ~]$ which conda
/usr/bin/which: no jupyter in
(/home/username/miniconda3/bin/conda:/usr/local/bin:/usr/bin:/usr/local/sbin:
/usr/sbin:/opt/dell/srvadmin/bin:/home/username/.local/bin:/home/username/bin
)
[username@login001 reverse-proxy]$ conda activate
-bash: conda: command not found
[username@login001 reverse-proxy]$ .
/home/username/miniconda3/etc/profile.d/conda.sh
[username@login001 reverse-proxy]$ conda activate
(base) [username@login001 reverse-proxy]$ which Jupyter
~/miniconda3/bin/jupyter
```
Clone the JRPS repository

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

[username@login02 rev-pxy]$ git clone git@github.com:sdsc-hpc-training-org/reverse-proxy.git
Cloning into 'reverse-proxy'...
Warning: untrusted X11 forwarding setup failed: xauth key data not generated
remote: Enumerating objects: 180, done.
remote: Counting objects: 100% (180/180), done.
remote: Compressing objects: 100% (118/118), done.
remote: Total 884 (delta 108), reused 132 (delta 61), pack-reused 704
Receiving objects: 100% (884/884), 7.58 MiB | 16.62 MiB/s, done.
Resolving deltas: 100% (498/498), done.
[username@login02 rev-pxy]$ cd reverse-proxy
[username@login02 reverse-proxy]$ ls -al
total 127
drwxr-xr-x  9 username abc123 15 Dec 10 09:27 .
drwxr-xr-x  6 username abc123  7 Dec 10 09:27 ..
drwxr-xr-x  2 username abc123  4 Dec 10 09:27 batch
-rw-r--r--  1 username abc123  583 Dec 10 09:27 .config
-rw-r--r--  1 username abc123  6148 Dec 10 09:27 .DS_Store
drwxr-xr-x  2 username abc123  5 Dec 10 09:27 .examples_images
drwxr-xr-x  8 username abc123 13 Dec 10 09:27 .git
-rw-r--r--  1 username abc123  12 Dec 10 09:27 .gitignore
drwxr-xr-x  2 username abc123   8 Dec 10 09:27 lib
-rw-r--r--  1 username abc123  5049 Dec 10 09:27 README.md
drwxr-xr-x  2 username abc123   4 Dec 10 09:27 slurm
drwxr-xr-x  2 username abc123   4 Dec 10 09:27 slurm_expanse
-rwxr-xr-x  1 username abc123  9011 Dec 10 09:27 start-jupyter
-rw-r--r--  1 username abc123  7723 Dec 10 09:27 start_notebook
drwxr-xr-x  2 username abc123   4 Dec 10 09:27 torque
Run the script: start-notebook

- For help and information on how to run:

```
[username@login02 reverse-proxy]$ ./start-jupyter -help

-p: the partition to use, debug or compute. Default is compute

-d: the top-level directory of your jupyter notebook. Default is /home/username

-A: the project allocation to be used for this notebook. Default is system default (also called project or group)

-b: the batch script you want to submit. Only those in the batch folder are supported. Default is ./batch/batch_notebook.sh

-t: the time to run the notebook. Default is 30 minutes

-i: Get extra information about the job you submitted using the script

-s: Choose between 'notebook' and 'jupyterlab'
```

- Default configuration: requires at least your account number
Run the script: start-notebook

- Remember to load the slurm module

```bash
(base) [username@login02 reverse-proxy-branch-james-dev]$ module load slurm
(base) [username@login02 reverse-proxy-branch-james-dev]$ ./start-jupyter -A abc123
```

Your notebook is here:

```bash
https://annuity-headphone-aptitude.expanse-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-expanse/notebook.sh

Your job id is 670505

You may occasionally run the command 'squeue -j 670505' to check the status of your job

```bash
(base) [username@login02 reverse-proxy-branch-james-dev]$ squeue -u username -u username
```

<table>
<thead>
<tr>
<th>JOBID</th>
<th>PARTITION</th>
<th>NAME</th>
<th>USER</th>
<th>ST</th>
<th>TIME</th>
<th>NODES</th>
<th>Nodelist(REASON)</th>
</tr>
</thead>
<tbody>
<tr>
<td>670505</td>
<td>compute</td>
<td>notebook</td>
<td>username</td>
<td>R</td>
<td>0:37</td>
<td>1 exp-1-17</td>
<td>exp-1-17.eth.cluster:8888/</td>
</tr>
</tbody>
</table>

You may occasionally run the command 'squeue -j 670505' to check the status of your job

```bash
(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:

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
     100   9 100 0 0 52 0 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
Warning: The Jupyter Notebook URL is essentially a password

Treat it very carefully
Do NOT Share
Load Notebook URL; wait for it to launch

Expanse Reverse Proxy Service

Phase 2 of 2: Mapped

Your job has started and checked in with the proxy. Please wait up to two minutes for the URL mapping to take effect.

Seconds to auto-reload: 7

Stop Reloading

Remember: The Jupyter Notebook URL is essentially a password
Notebook is launched

When done with the notebook be sure to shut it down by quitting the notebook.
Live Demo


https://github.com/sdsc-hpc-training-org/reverse-proxy
SDSC Reverse Proxy Service Team

• Project Team:
  • Mary Thomas (SDSC)
  • Scott Sakai (SDSC)
  • Marty Kandes (SDSC)
  • Rick Wagner (UCSD)
  • Interns: James McDougall

• Project Status:
  • RPS is in early user testing.
  • Please give it a try. If you have trouble, create a ticket so we can track issues: help@xsede.org