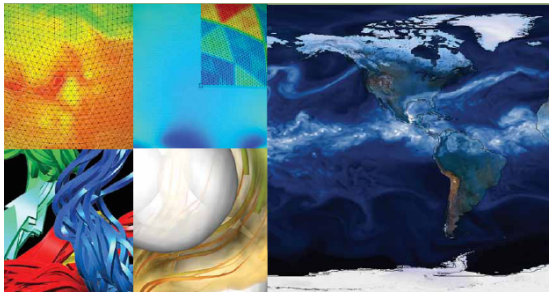
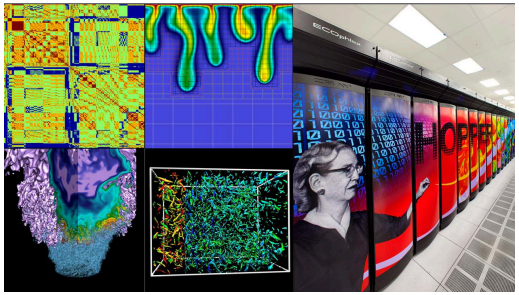
NERSC is the Production HPC & Data Facility for DOE Office of Science Research

U.S. DEPARTMENT OF ENERGY | Office of Science

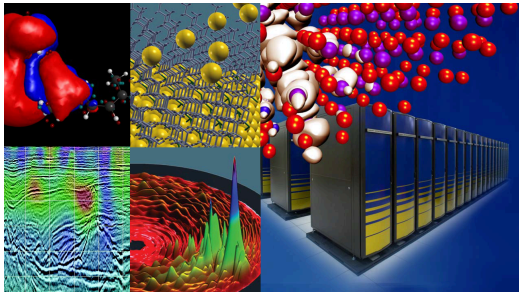
Largest funder of physical science research in U.S.



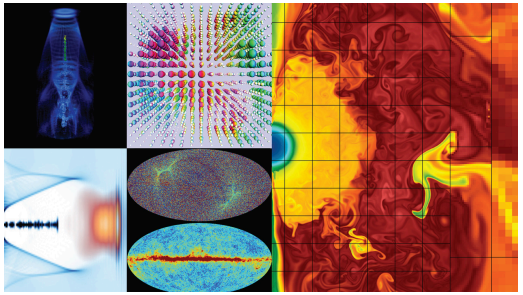
Bio Energy, Environment



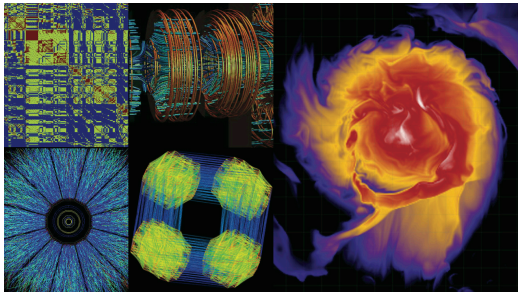
Computing



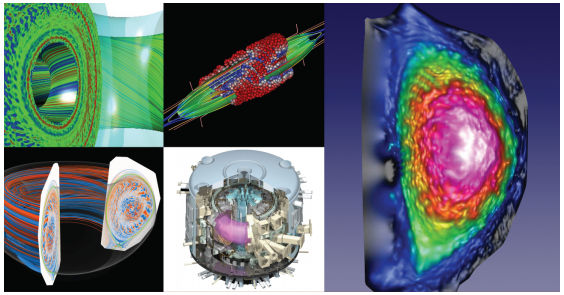
Materials, Chemistry, Geophysics



Particle Physics, Astrophysics



Nuclear Physics



Fusion Energy, Plasma Physics

The Cori System

- **Cori will transition HPC and data-centric workloads to energy efficient architectures**
- **Includes "data partition" with Haswell nodes, data friendly queues, pool of large memory interactive nodes**
- **Also has NVRAM layer, Software Defined Network**

Image source: Wikipedia

System named after Gerty Cori, Biochemist and first American woman to receive the Nobel prize in science.

Interactive Notebook Computing

- **Combine an annotated notebook with live code execution and results**
  - "literate computing" environment, where narrative and computation go hand in hand
- **Jupyter emerged from iPython and is becoming a very popular web based framework for interactive notebook computing**
- **Not just python - has over 60 language kernels**
  - Julia, R, Bash, Root, Ruby ...
- **c/f Fernando Perez keynote**

Notebook examples

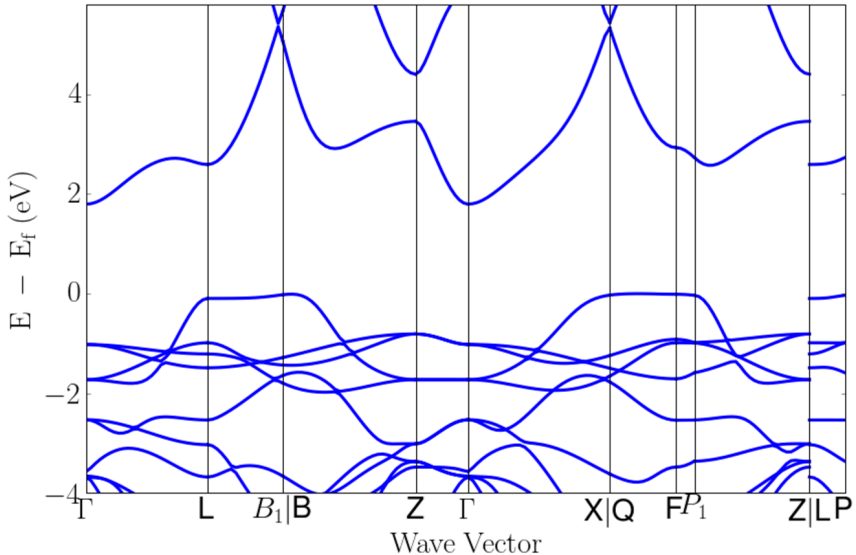
https://github.com/materialsproject/mapidoc/blob/master/example_notebooks/pymatgen_examples/Analyze%20and%20plot%20band%20structures.ipynb

In [9]:
```
#is the material a metal (i.e., the fermi level cross a band)
print bs.is_metal()
#print information on the band gap
print bs.get_band_gap()
#print the energy of the 20th band and 10th kpoint
print bs.bands[Spin.up][20][10]
#print energy of direct band gap
print bs.get_direct_band_gap()
#print information on the vbm
print bs.get_vbm()
```

```
False
{'energy': 1.7978000000000005, 'transition': u'(0.591,0.409,0.000)-\\Gamma', 'direct': False}
18.0201
2.6904
{u'kpoint': <pymatgen.electronic_structure.bandstructure.Kpoint object at 0x5192610>, u'kpoint_index': [123], u'energy': 6.1023, u'band_index': {1: [15]}, u'projections': {}}
```

Here, we plot the bs object. By default for an insulator we have en energy limit of cbm+4eV and vbm-4 eV

In [7]:
```
%matplotlib inline
from pymatgen.electronic_structure.plotter import BSPlotter
plotter=BSPlotter(bs)
plotter.get_plot().show()
```



Jupyter in HPC

HPC workflows at scale for data-intensive science => asynchronous, batch execution.

But …

Scientific insight => interactive, iterative exploration and analysis

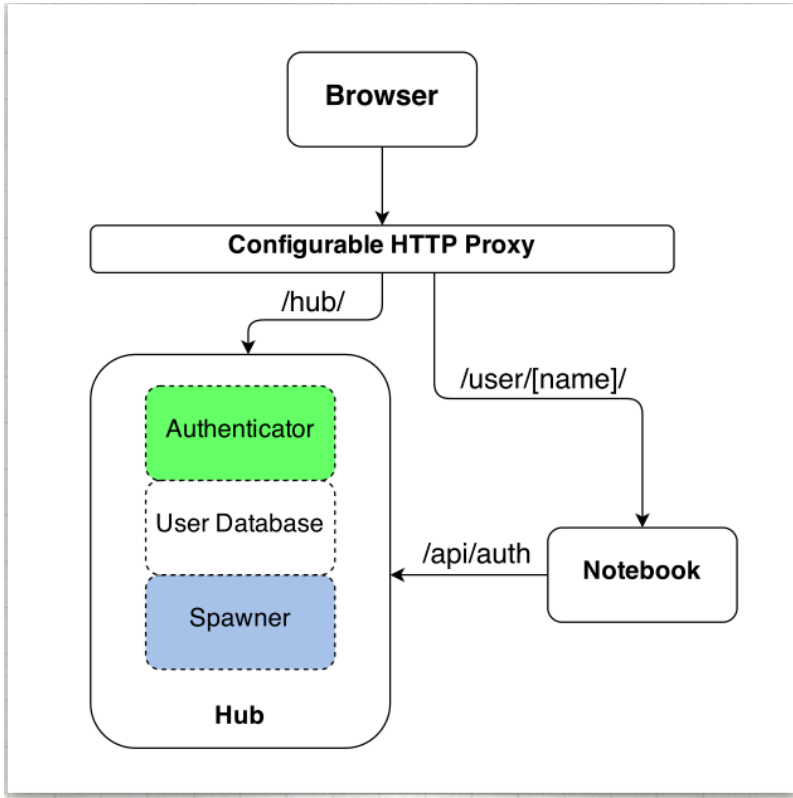
How can Jupyter bridge this gap?

Motivation

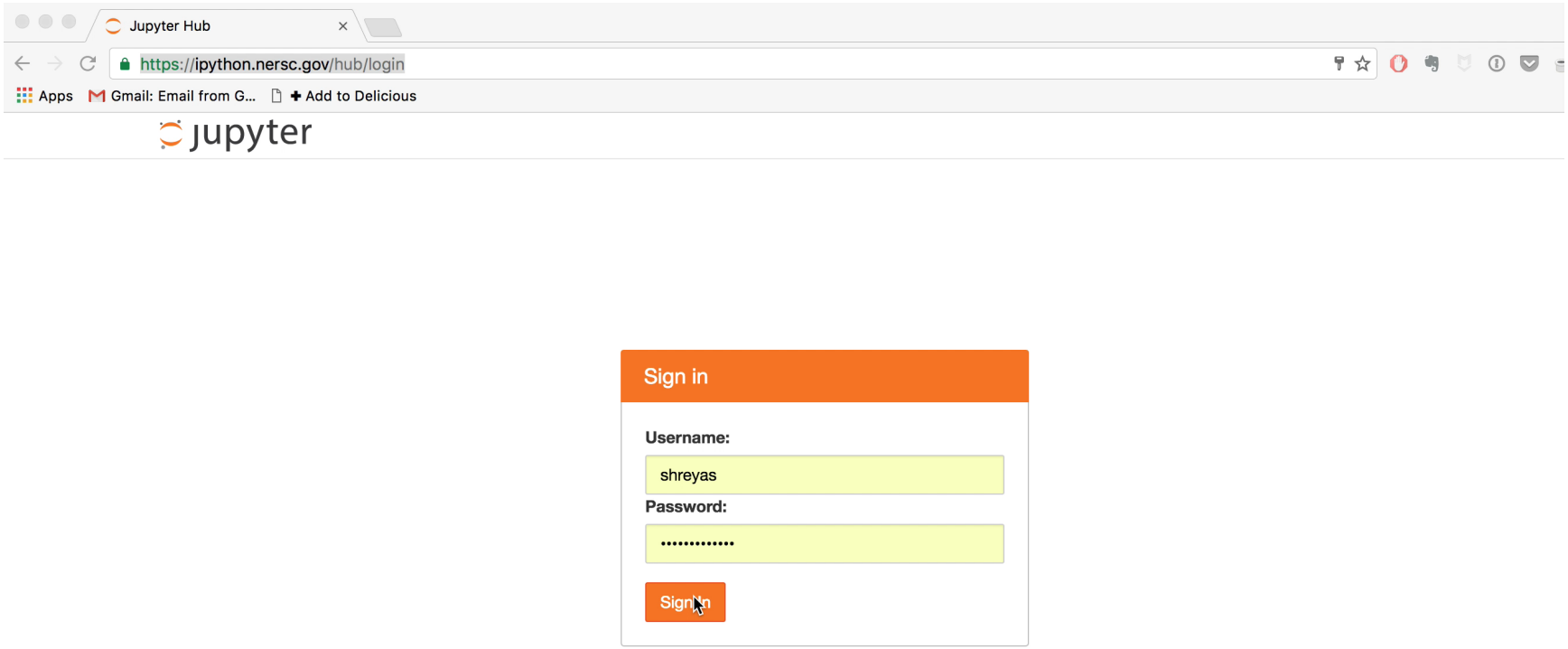
- **Python is the most popular language/tool at NERSC**
- **Users want to be able to use Jupyter notebooks to drive their jobs, run post-job analyses etc.**
- **Access to NERSC resources through these interfaces**
  - Filesystems
  - Batch Queue
  - Network, DBs
- **Support for custom libraries and kernels**
- **Centralized service to deploy notebooks in a standard authenticated manner**

Jupyterhub

- Service to deploy notebooks in a multi-user environment
- Manages user authentication, notebook deployment and web proxies
- Node.js + Python Tornado

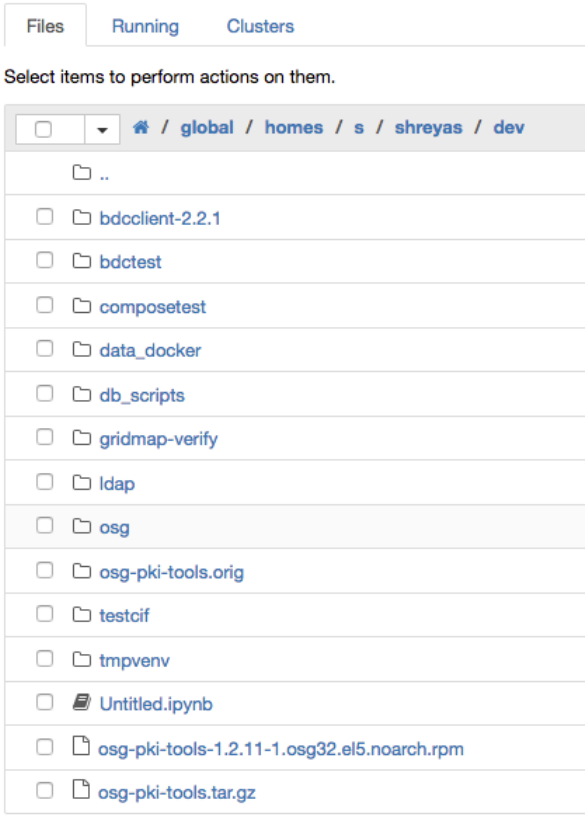


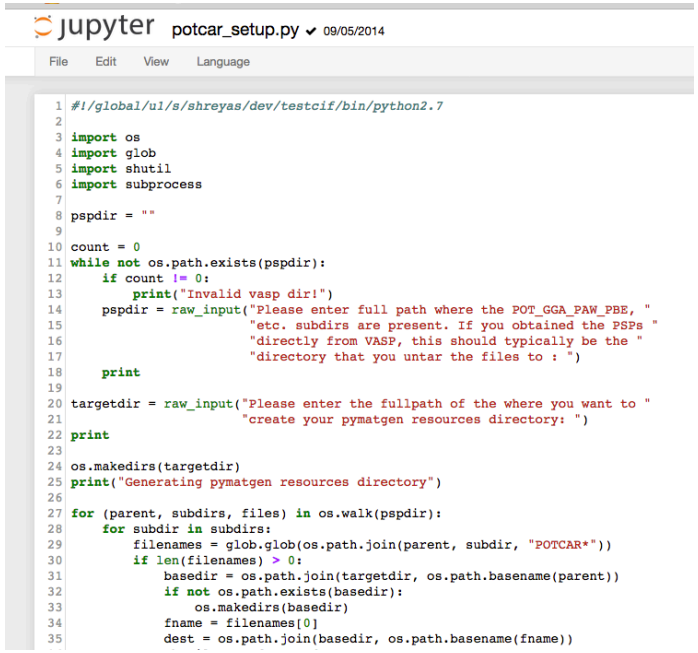
Jupyterhub Example

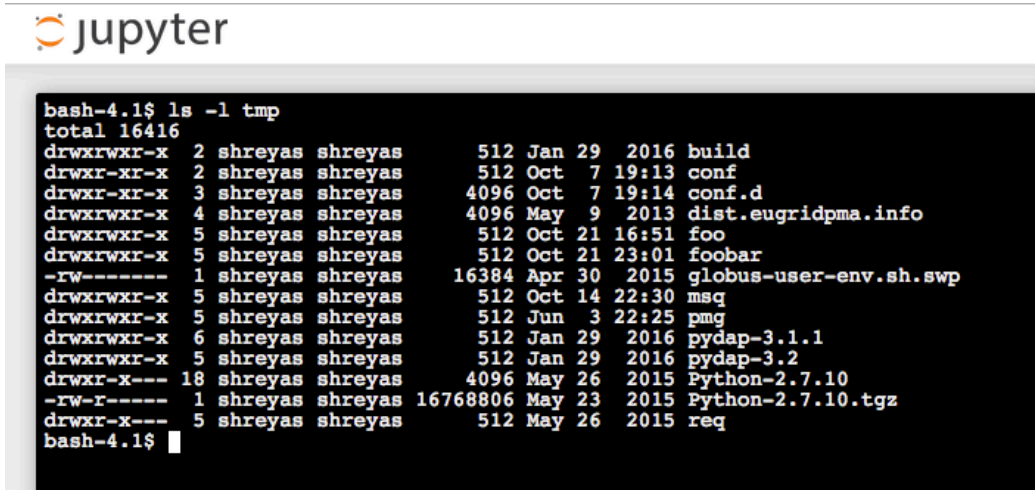


But Wait .. There's More

- **Jupyter also includes**
  - File Browser
  - Text Editor (with full syntax highlighting)
  - Terminal



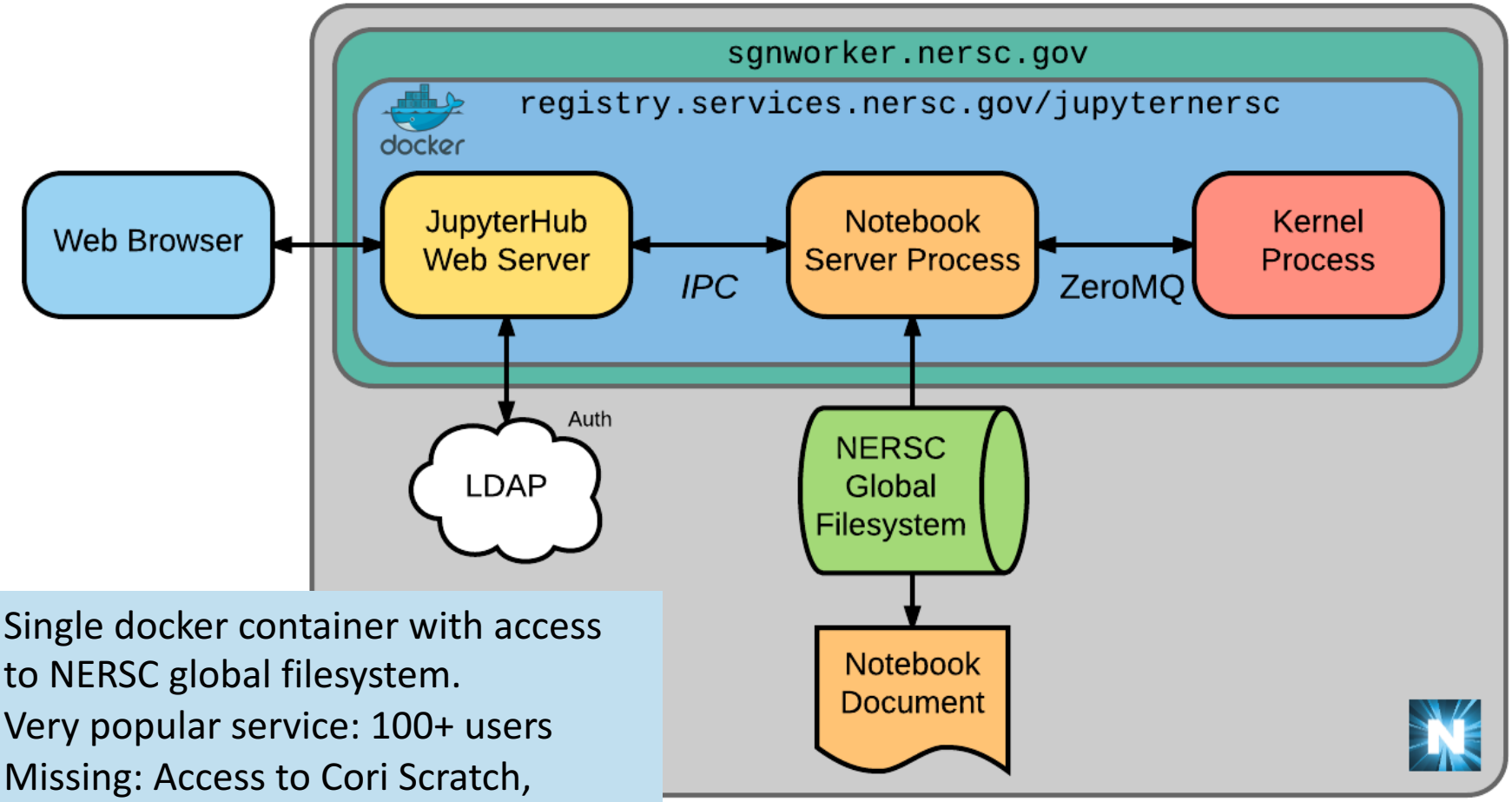




Jupyterhub Prototype

- Jupyterhub on a single node
- Service runs as root and spins up notebooks locally
- Jupyter has access to global filesystems and lives within the NERSC network

Jupyterhub in a Single Container

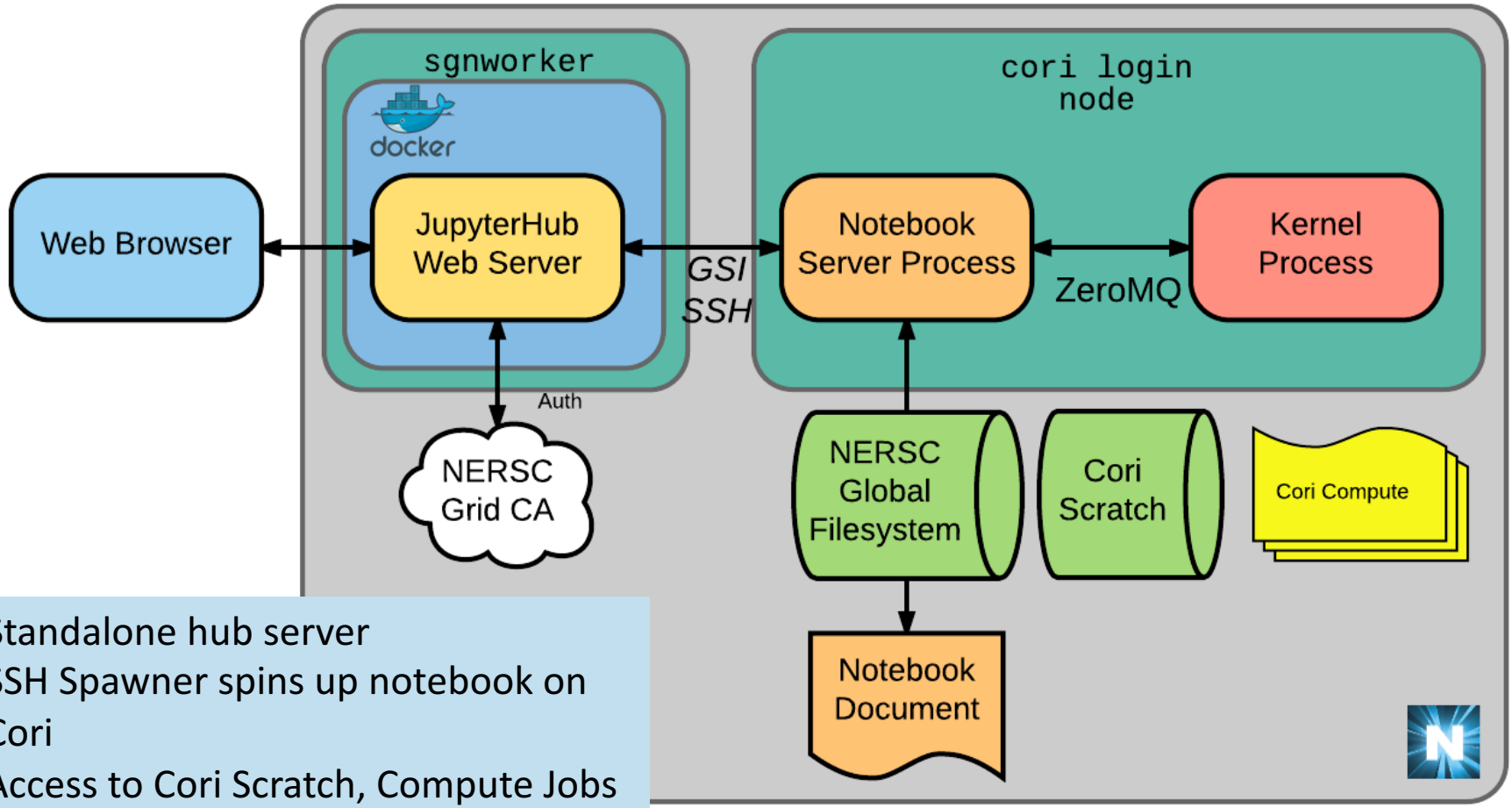


- Single docker container with access to NERSC global filesystem.
- Very popular service: 100+ users
- Missing: Access to Cori Scratch, Compute Jobs

Jupyterhub Cori Service

- **Users would like access to big compute jobs!!!**
- **Access to Cori Scratch Filesystem**
- **Existing Software on Cori (including kernels)**
- **Take advantage of large memory interactive nodes**
- **Security considerations**
  - Don’t want to run as root
  - Don’t want to run public facing web service inside a Cori node
- **New service uses a distributed model where Jupyterhub runs on a science gateway node but spins up notebooks remotely**

New Jupyterhub Implementation For Cori



- Standalone hub server
- SSH Spawner spins up notebook on Cori
- Access to Cori Scratch, Compute Jobs via SLURM magic

What we built

Jupyter architecture is completely pluggable

So we developed

- **Custom Authenticator that allows user to login and get a GSI X509 certificate**
- **Custom Spawner that creates a remote notebook via SSH (using public key or GSI auth)**
- **Ipython Magic Commands to launch and query jobs via SLURM**

GSI Authenticator

- User logs in with username and password. Authenticator uses myproxy to login to NERSC CA server with username/password and retrieves credentials (X509 certificate)
- Jupyterhub runs as a standalone service and doesn't need root access. In fact, no root access needed across this architecture.
- https://github.com/NERSC/gsiauthenticator

Spawner: SSH

- **We wrote an SSH Spawner that will will SSH into the Cori node with users credential**
  - Supports GSISSH (use with certificates from GSI authenticator)
  - Supports SSH key based auth
- **SSH Spawner starts up notebook server process and goes away; Notebook server communicates directly with hub**
  - No tunnels or persistent connections needed
- **Keep track of the PID for poll and shutdown functions (also via SSH)**
- **Inspired by Andrea Zonca's RemoteSpawner (SDSC)**
- **https://github.com/NERSC/SSHSpawner**

Ipython Magic for SLURM

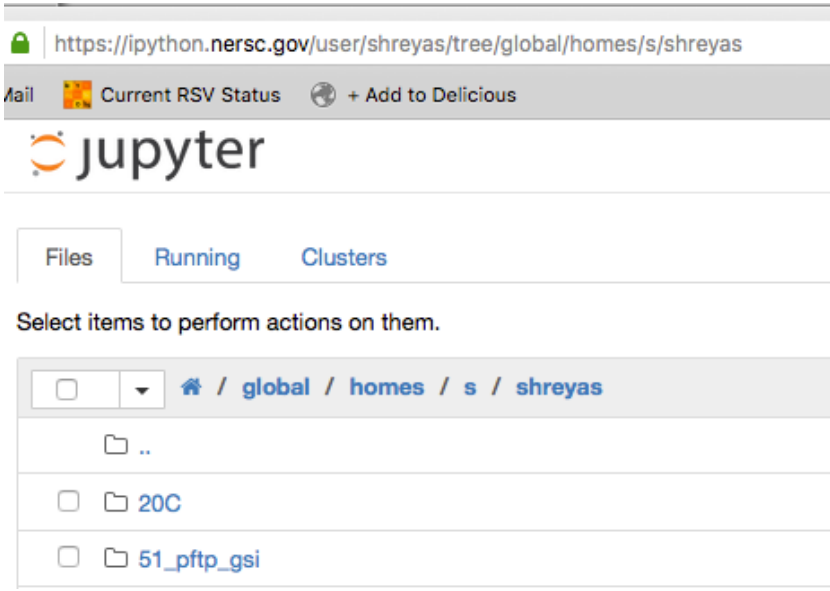
- Created some simple magics to facilitate interaction with SLURM batch queuing system
- Implemented %%sbatch and %squeue
- Pandas mode to slurp the results
- https://github.com/NERSC/slurm-magic

```
In [2]: %%sbatch -p debug -t 10 -N 1
#!/bin/bash
srun -n 32 hostname
...:
Out[2]: u'Submitted batch job 2754280\n'

In [3]: !cat slurm-2754280.out
nid00044
nid00044
...
```

```
In [3]: %squeue -u rthomas
Out[3]:
     JOBID     USER ACCOUNT NAME PARTITION    QOS  NODES TIME_LIMIT  TIME ST  \
0  2764292  rthomas    mpccc   sh     debug  debug      1      10:00  0:11  R

   PRIORITY          SUBMIT_TIME           START_TIME
0     69060  2016-07-21T21:12:34  2016-07-21T21:13:21
```

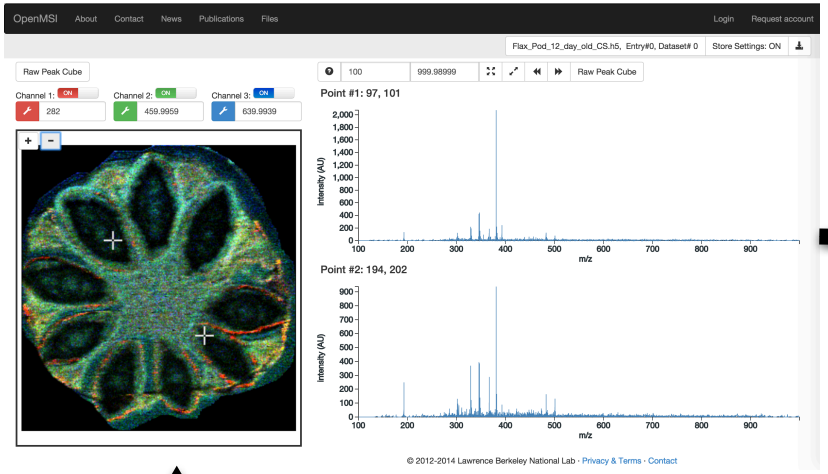
Other Features

- Allow users to browse other filesystems while defaulting to homedir
- Users can drop in custom kernels in their .ipython dir

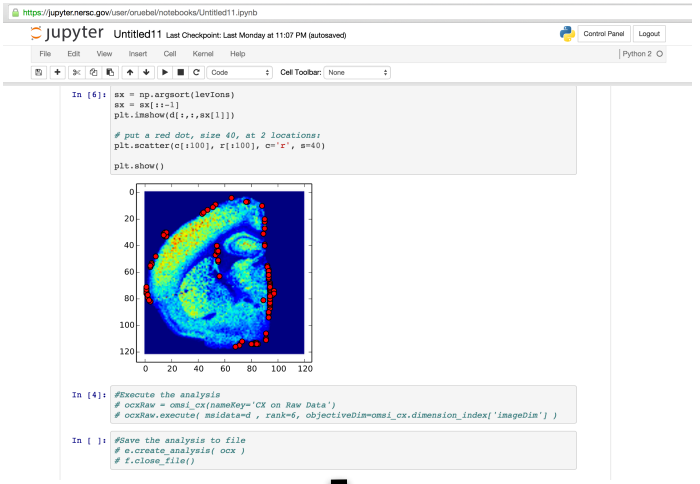


Use Case: OpenMSI + Jupytherhub

openmsi.nersc.gov



jupyter.nersc.gov

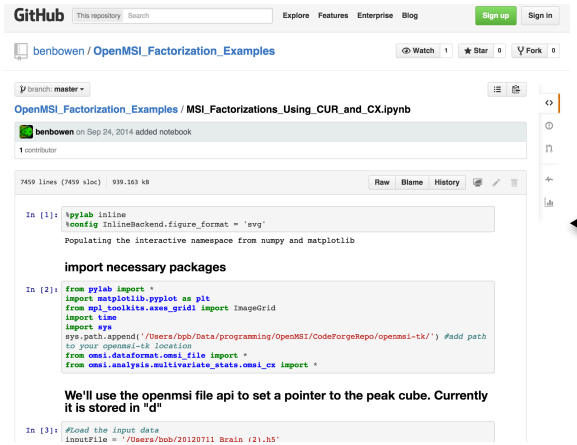


Launch, run, and integrate notebooks

- Manage data, analyses, users and resources
- Customize templates

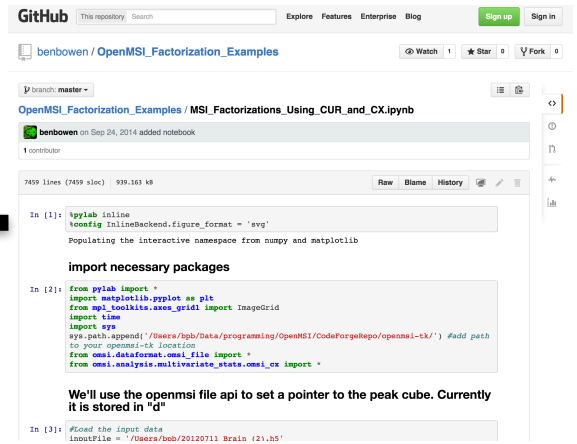
Share analysis results

Template analysis notebook repository



Contribute new analysis templates

User notebook repository



Future Directions

- Driving a Spark/Dask job from a notebook on Cori
- Spawning notebook kernels on Cori compute nodes
- Tighter interaction with MPI jobs
- Interactive HPC workflows (LDRD)

What Our Users Say

“I’ll never have to leave a notebook again that’s like the ultimate dream”

We're Hiring

- **Postdoc to work on Jupyter for human-in-the-loop interactive HPC workflows**
- **Various CSE positions**
- **http://jobs.lbl.gov**

Contact: scholia@lbl.gov

More info

Contact us:

- scholia@lbl.gov
- scanon@lbl.gov
- rcthomas@lbl.gov

All our code is on github

https://github.com/NERSC/gsiauthenticator

https://github.com/NERSC/SSHSpawner

https://github.com/NERSC/slurm-magic