



Running jupyter notebook remotely in a docker swarm cluster

Jordi Deu-Pons
Barcelona 2017



jupyter



docker

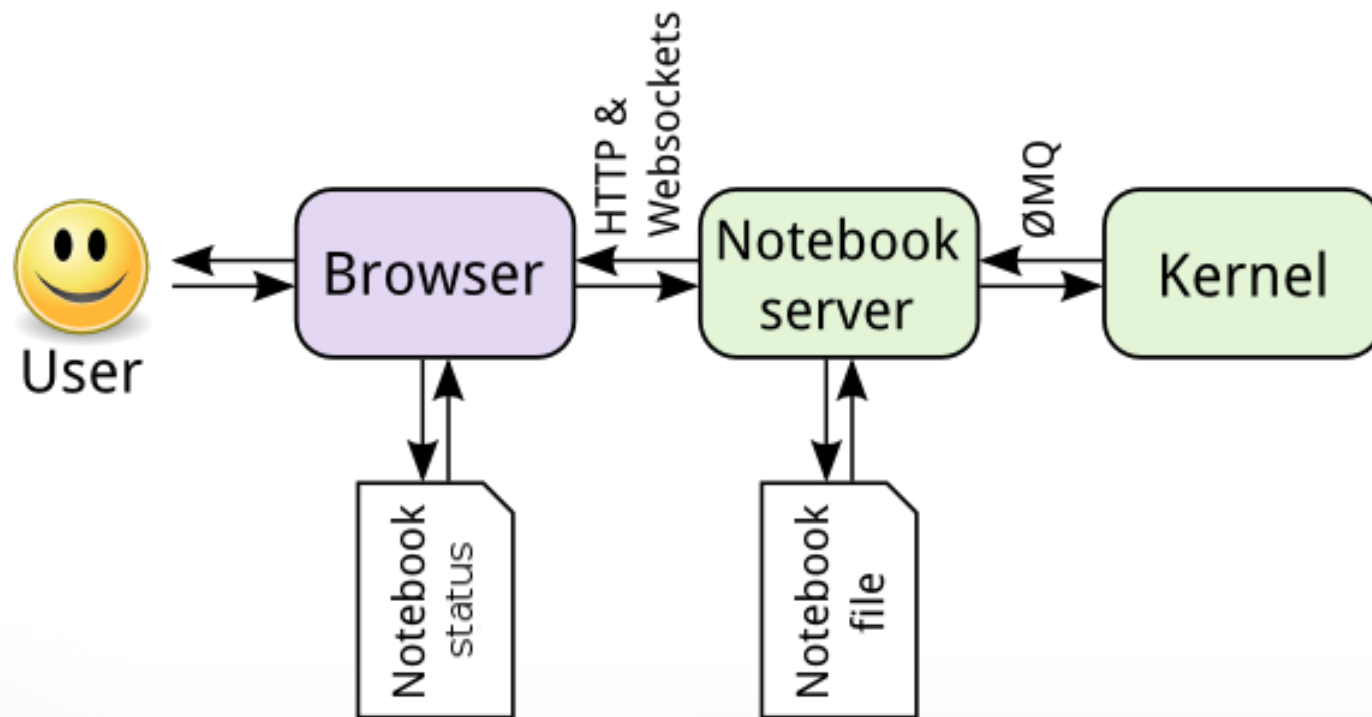


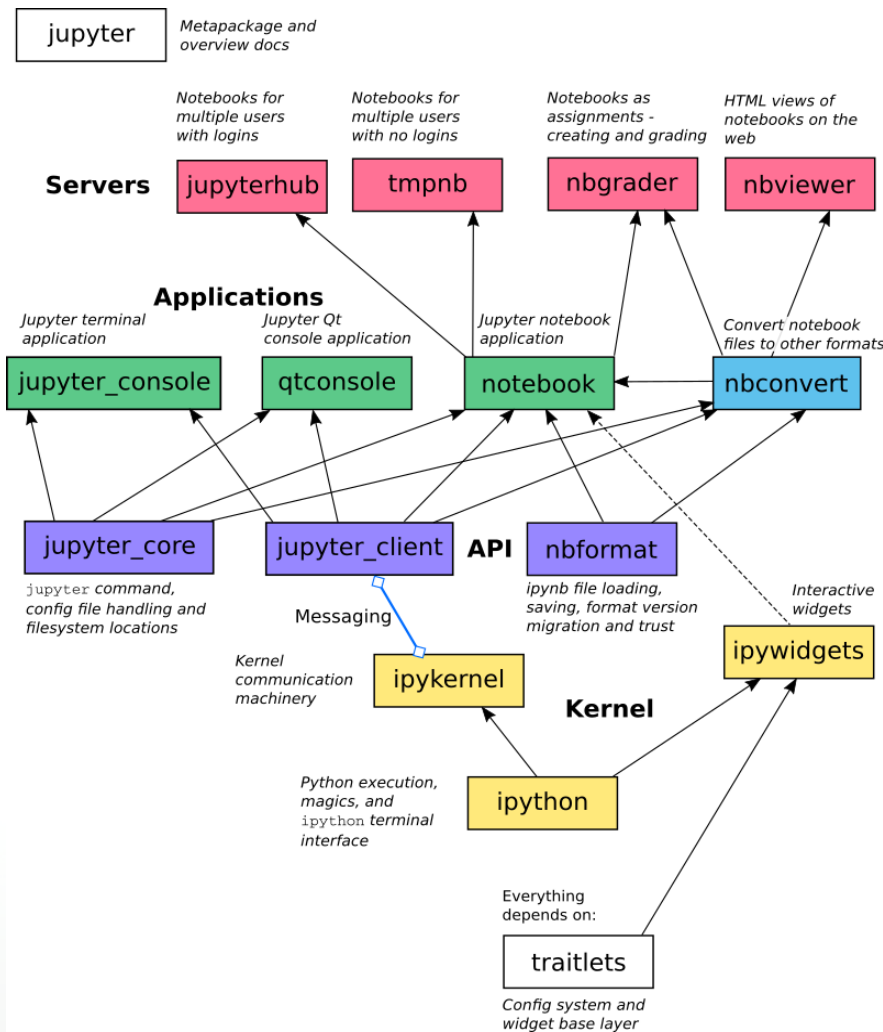
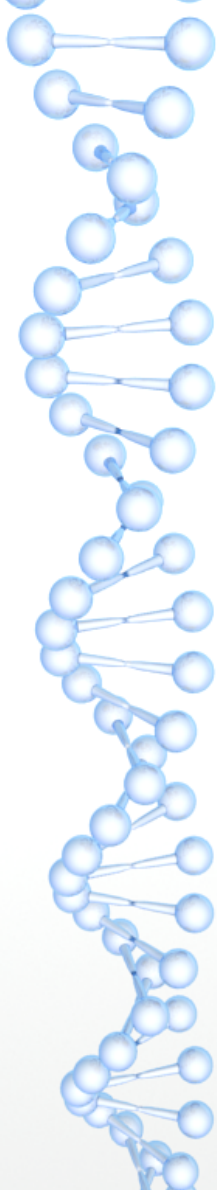


Content of this talk

- Jupyter architecture overview
- Running a notebook remotely
- JupyterHub solution
- Alternative solution using remote desktops
- Pros and cons
- Future

Jupyter architecture overview



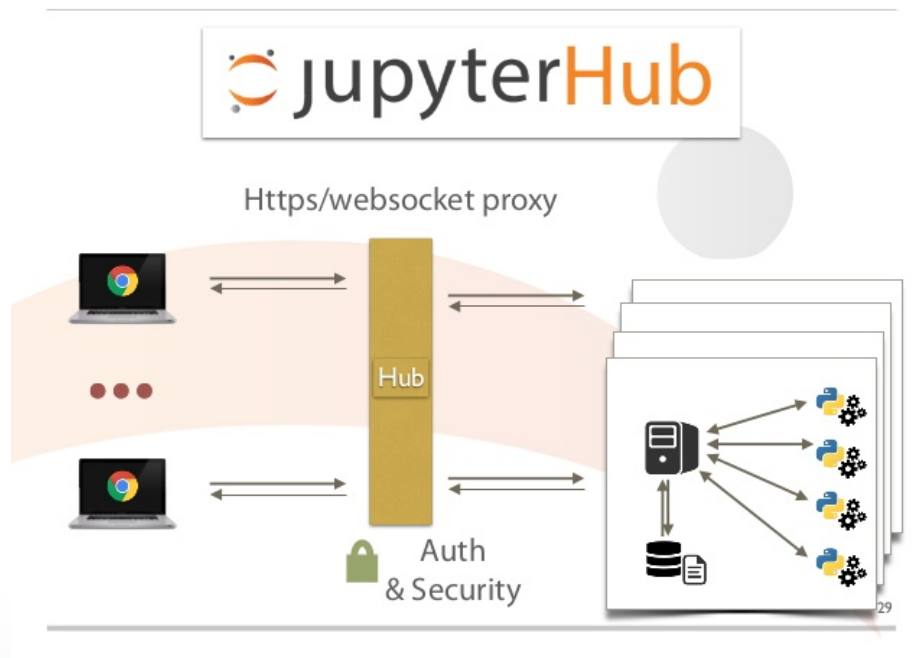




Running a notebook remotely

- Proxy requests
- Authenticate the users
- Central management
- Reconnect to a running notebook
- Share with other users
- Collaborative notebook editing

JupyterHub solution





Alternative solution

Technologies used:

- Docker swarm cluster
- VNC server + Firefox + Jupyter
- Apache Guacamole (*clientless remote desktop gateway*)

Alternative solution

```
.
├── builds
│   ├── jupyteruser
│   │   └── Dockerfile
│   ├── jupytervnc
│   │   └── Dockerfile
│   └── etc
│       ├── menu.xml
│       └── supervisord.conf
├── scripts
│   ├── cluster_create.sh
│   ├── deploy_backbone.sh
│   └── deploy_notebooks.sh
└── stack
    ├── guacamole.sql
    ├── guacamole.yml
    ├── notebooks.yml
    ├── proxy.yml
    └── registry.yml
```




Docker cluster

```
# Create 3 machines
for i in 1 2 3; do
    docker-machine create -d virtualbox node-$i
done
eval $(docker-machine env node-1)

# Initialize docker manager
docker swarm init --advertise-addr $(docker-machine ip node-1)

# Join workers to the cluster
TOKEN=$(docker swarm join-token -q worker)
for i in 2 3; do
    eval $(docker-machine env node-$i)
    docker swarm join --token $TOKEN --advertise-addr $(docker-machine ip node-$i) \
        $(docker-machine ip node-1):2377
done
```



Deploy backbone services

```
# Create proxy and notebooks networks
```

```
docker network create --driver overlay proxy
```

```
docker network create --driver overlay notebooks
```

```
# Start proxy, registry and guacamole
```

```
docker stack deploy -c ../stack/proxy.yml proxy
```

```
docker stack deploy -c ../stack/guacamole.yml guacamole
```

```
docker stack deploy -c ../stack/registry.yml registry
```

Build and push images

- Common image (builds/jupyterenv/Dockerfile)

```
$ docker build -t localhost:5000/jupyterenv:5.0 .
```

```
$ docker push localhost:5000/jupyterenv:5.0
```

```
FROM debian:jessie-slim
RUN apt-get -y update \
    && apt-get install -y --no-install-recommends x11vnc xvfb supervisor openbox firefox-esr wget bzip2 ca-certificates \
    ...
# Configure environment
ENV CONDA_DIR /opt/conda
...
COPY etc/supervisord.conf /etc/supervisor/supervisord.conf
COPY etc/menu.xml /etc/xdg/openbox/menu.xml
...
RUN wget --quiet https://repo.continuum.io/miniconda/Miniconda3-4.3.11-Linux-x86_64.sh && \
    echo "b9fe70ce7b6fa8df05abfb56995959b897d0365299f5046063bc236843474fb8 *Miniconda3-4.3.11-Linux-x86_64.sh" | sha256sum -c - && \
    ...
USER $BG_USER
EXPOSE 5900
WORKDIR /home/$BG_USER
CMD ["/usr/bin/supervisord"]
```



Build and push images

- User image (builds/jupyteruser/Dockerfile)

```
$ docker build -t localhost:5000/jupyteruser:5.0 .  
$ docker push localhost:5000/jupyteruser:5.0
```

```
FROM localhost:5000/jupyterenv:5.0  
ENV BG_NEW_USER username  
ENV BG_NEW_UID 1391  
  
# Change user UID  
USER root  
  
RUN usermod -u $BG_NEW_UID $BG_USER \  
&& usermod -l $BG_NEW_USER $BG_USER \  
&& chown -R -h $BG_NEW_USER /var/log/supervisor  
  
USER $BG_NEW_USER
```



Config and start a notebook

```
$ docker stack deploy -c ../stack/notebooks.yml nb
```

```
version: '3'

services:
  example:
    image: localhost:5000/jupyteruser:5.0
    networks:
      - notebooks
    volumes:
      - ${REPO_HOME}:/workspace
    working_dir: /workspace
    environment:
      - CONDA_ENVS_PATH=${REPO_HOME}/envs
```

Config guacamole connection

EDIT CONNECTION

Name:

Location:

Protocol:

PARAMETERS

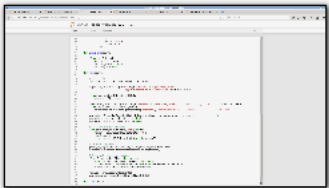
Network

Hostname:

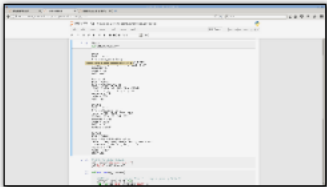
Port:

Connect to the notebook

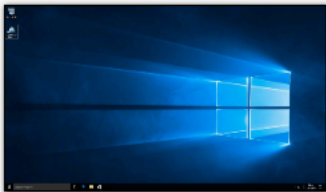
RECENT CONNECTIONS jdeu ▾



intogen



gendas

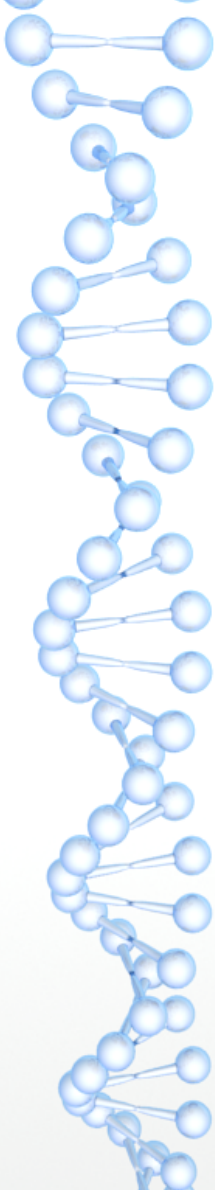


Windows 10

ALL CONNECTIONS Filter

- projects
 - gendas
 - intogen
- Windows 10

Currently in use by 1 user.



gendas x

bbglab.irbbarcelona.org/guacamole/#/client/MgBJAG15c3Fs

Rank scores - Mozilla Firefox

Rank scores x jdeu@2891e05eb7... x

localhost:8888/notebooks/src/examples/Rank scores.ipynb#

Jupyter Rank scores Last Checkpoint: Last Monday at 3:15 PM (autosaved)

File Edit View Insert Cell Kernel Help

Not Trusted Python [conda env:py35]

```
begin = BEGIN
end = END

In [4]: # Create a Gendas engine
gd = Gendas('data/gendas.conf')
gd['hg19'] = HG19Source()

In [5]: def mut_rank(gd, baserow):

    # Get scores of all the position in this gene group by tri>alt
    context = defaultdict(list)
    for r in gd['cadd'].merge(gd['hg19']):
        key = "{}>{}".format(r['hg19'][-1:1], r['cadd']['ALT'])
        context[key].append(r['cadd']['PHRED'])

    rows = []
    for m in gd['variants'].merge(gd['cadd'], on=['REF', 'ALT']).merge(gd['hg19']):

        key = "{}>{}".format(m['hg19'][-1:1], m['variants']['ALT'])
        ctx_scores = context[key]

        # Create the output row
        row = dict(baserow)
        for k, v in m['variants'].items():
            row[k] = v

        row['KEY'] = key
        row['SCORE'] = m['cadd']['PHRED']
        row['CONTEXT'] = ctx_scores

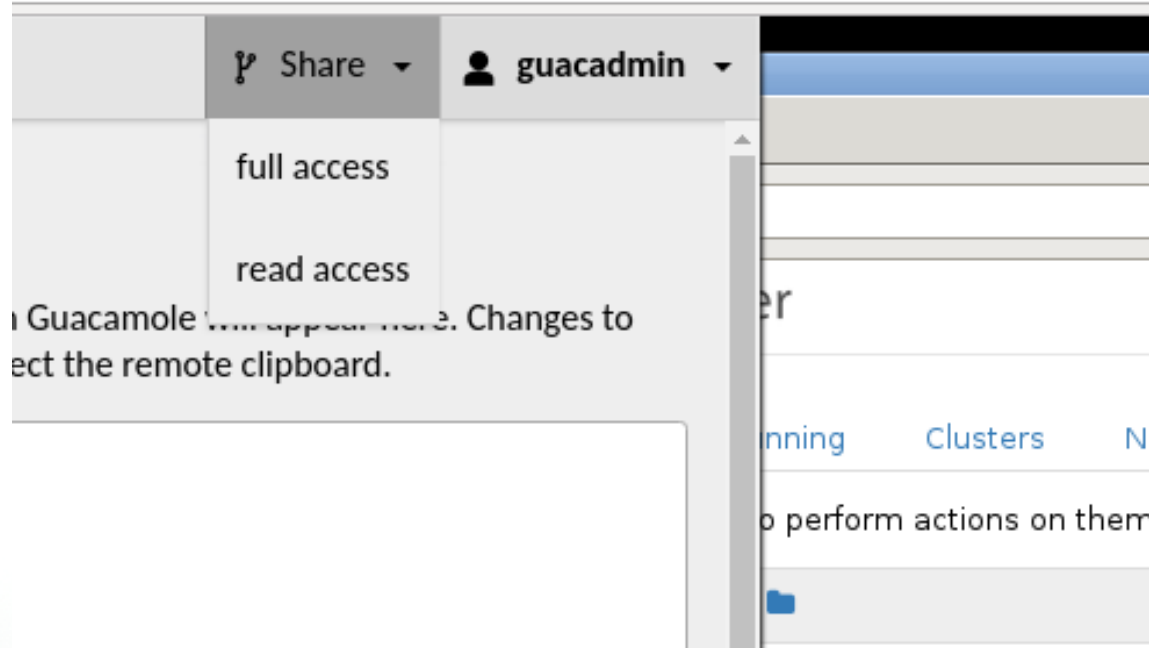
        rows.append(row)

    return rows

In [6]: %time
data = pd.DataFrame.from_dict(
    flatten(
```


Share connection

58.99.100/guacamole/#/client/MQBjAG15c3Fs





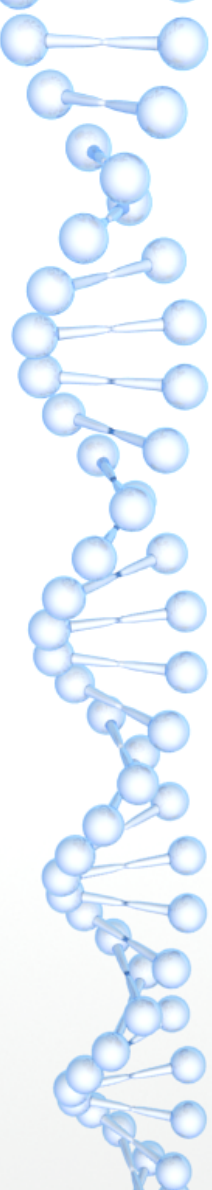
Pros and cons


- **Pros**
 - You can disconnect at any time
 - Extra security layer
 - Concurrent notebook editing
- **Cons**
 - Indirect copy&paste
 - Scrolling less responsive
 - Different look&feel

Future: notebook hibernation

docker checkpoint create ...

(already available with experimental enable)





[Main page](#)
[Recent changes](#)
[Random page](#)
[Help](#)

Tools

- [What links here](#)
- [Related changes](#)
- [Special pages](#)
- [Printable version](#)
- [Permanent link](#)
- [Page information](#)

News

- [Google+](#)
- [Twitter](#)


Page [Discussion](#)

Docker

This article describes the status of CRIU integration with Docker, and how to use it.

Contents [\[hide\]](#)

- 1 Docker Experimental
 - 1.1 checkpoint
 - 1.2 restore
 - 1.2.1 Restoring into a **new** container
 - 1.3 Synopsis
- 2 Compatibility Notes
 - 2.1 TTY
 - 2.2 Seccomp
 - 2.3 OverlayFS
 - 2.4 Async IO
- 3 External checkpoint/restore



github.com/jordeu/pytalks



Future: collaborative editing

Brian E. Granger, 6 Feb 2017

Yes, we are doing this work in JupyterLab. We are building these features in a manner that will initially support Google Drive Real Time API, but could also have other real time backends plugged into it.

We have a full time post doc at UC Berkeley, Ian Rose, working on these things. The real time stuff isn't quite ready for public usage, but it is moving pretty fast at this point.

<https://github.com/ian-r-rose/jupyterlab-google-drive>

