



Oct 02, 2023

Version 2

🌐 Steps for setup of AWS organization, S3 data storage, and EC2 computing for using Python notebooks V.2

📖 [PLOS One](#)

✓ Peer-reviewed method

DOI

dx.doi.org/10.17504/protocols.io.rm7vz3z4xgx1/v2

Daniel J. Pollak¹, Gautam Chawla¹, Andrey Andreev¹, David A. Prober¹

¹Division of Biology and Biological Engineering, Tianqiao and Chrissy Chen Institute for Neuroscience, California Institute of Technology, Pasadena, California, United States of America

Andrey Andreev: Corresponding author

Prober Microscopy Team

Spotlight series



Dan P

Create & collaborate more with a free account

Edit and publish protocols, collaborate in communities, share insights through comments, and track progress with run records.

Create free account

OPEN  ACCESS



DOI: <https://dx.doi.org/10.17504/protocols.io.rm7vz3z4xgx1/v2>



External link: <https://doi.org/10.1371/journal.pone.0278316>

Protocol Citation: Daniel J. Pollak, Gautam Chawla, Andrey Andreev, David A. Prober 2023. Steps for setup of AWS organization, S3 data storage, and EC2 computing for using Python notebooks. **protocols.io**

<https://dx.doi.org/10.17504/protocols.io.rm7vz3z4xgx1/v2> Version created by **Dan P**

Manuscript citation:

Pollak DJ, Chawla G, Andreev A, Prober DA (2023) First steps into the cloud: Using Amazon data storage and computing with Python notebooks. PLOS ONE 18(2): e0278316. <https://doi.org/10.1371/journal.pone.0278316>

License: This is an open access protocol distributed under the terms of the **Creative Commons Attribution License**, which permits unrestricted use, distribution, and reproduction in any medium, provided the original author and source are credited

Protocol status: Working

We use this protocol and it's working

Created: September 30, 2023

Last Modified: October 02, 2023

Protocol Integer ID: 88610

Keywords: Python, AWS, cloud, computing, cloud processing, ec2 computing, cloud platform, cloud environment, cloud processing on amazon web service, more use cases for cloud, infrastructure into the cloud environment, environment in the cloud, s3 data storage, aws organization, example calcium imaging data in the zebrafish brain, computing environment, based computing, coding infrastructure, data processing, zebrafish brain, biologist, big data, amazon web service, guide for experimental biologist, using python notebook, experimental biologist, julia programming language, using python, example calcium imaging data, storage, python notebooks with the oncoming age, tools for management, web

Abstract

With the oncoming age of big data, biologists are encountering more use cases for cloud-based computing to streamline data processing and storage. Unfortunately, cloud platforms are difficult to learn, and there are few resources geared towards biologists for demystifying them. We have developed a guide for experimental biologists to set up cloud processing on Amazon Web Services to cheaply outsource data processing and storage. Here we provide a guide on setting up a computing environment in the cloud and showcase examples of using Python and Julia programming languages. We present example calcium imaging data in the zebrafish brain and corresponding analysis using suite2p software. Tools for management of users and budgets are discussed in the protocol. Following this guide should help researchers even with limited programming experience to get started or move existing coding infrastructure into the cloud environment.

The **last step** in this version contains a supplemental video with extra context and tips, as part of the protocols.io Spotlight series, featuring conversations with protocol authors.)

Troubleshooting



Safety warnings

- ⚠ Using "cloud" computing can lead to budget overruns due to pay-after-use nature of AWS and other providers. Consult with your home IT department on how to best manage costs and deployment of software.



Setting organization and budget management

1 Setting up a Root account

1.1 Create "root" account for your organization, using Business account type



Free Tier offers

All AWS accounts can explore 3 different types of free offers, depending on the product used.



Always free
Never expires



12 months free
Start from initial sign-up date



Trials
Start from service activation date

Sign up for AWS

Contact Information

How do you plan to use AWS?

- ☒ Business - for your work, school, or organization
- ☐ Personal - for your own projects

Who should we contact about this account?

Full Name

Organization name

Phone Number

Enter your country code and your phone number.

Country or Region

Address

City

State, Province, or Region

Postal Code


- ☐ I have read and agree to the terms of the [AWS Customer Agreement](#).

Continue (step 2 of 5)

Screenshot from <https://portal.aws.amazon.com/billing/signup?type=enterprise#/account>



- 1.2 If you are using GMail account to manage other services in your lab, for example labname@gmail.com, you can use labname+aws@gmail.com to register account with AWS (so-called "gmail + trick")
- 1.3 You will have to enter credit card information to register root account. Consult your department if you want to avoid using personal credit card
- 2 Research credits can be applied to the account of the organization. Contact entity that issued the credits (most likely your IT department)
- 3 There are two options to allow users be part of the organization. External account can be added to organization, but it might be better to create accounts within organization interface:

 Services ▼

Search

AWS Organizations

×

▼ **AWS accounts**

Invitations

Services

Policies

Settings

Get started

Organization ID

o-ohendmmm5q

AWS Org

Add

You can

☒ Cr

Cr

Cre

AWS

San

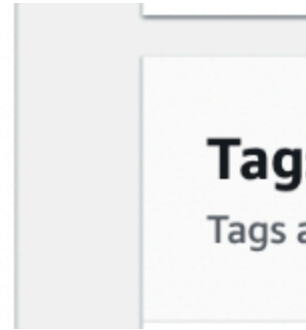
Email

acce

IAM r

The m

Org



- 4 After an account has been created, lab member should receive email and use the “reset password” function to set up new password.
- 5 Each member of the organization can have an individual budget set up. Monthly cost budget is a good starting point to manage costs on a per-account basis. Because sometimes cost can skyrocket accidentally due to misconfigured software (by the users) budget alerts and limits will provide safeguard against such overruns



Parameters

Filters [Info](#)

Dimension

Linked account

Values

Filter linked accounts by values

 *Find linked accounts*

☐ andrey andreev (568744944897)



Billing Console > Budgets > Overview

Overview Info

Budgets (4) Info

 Find a budget

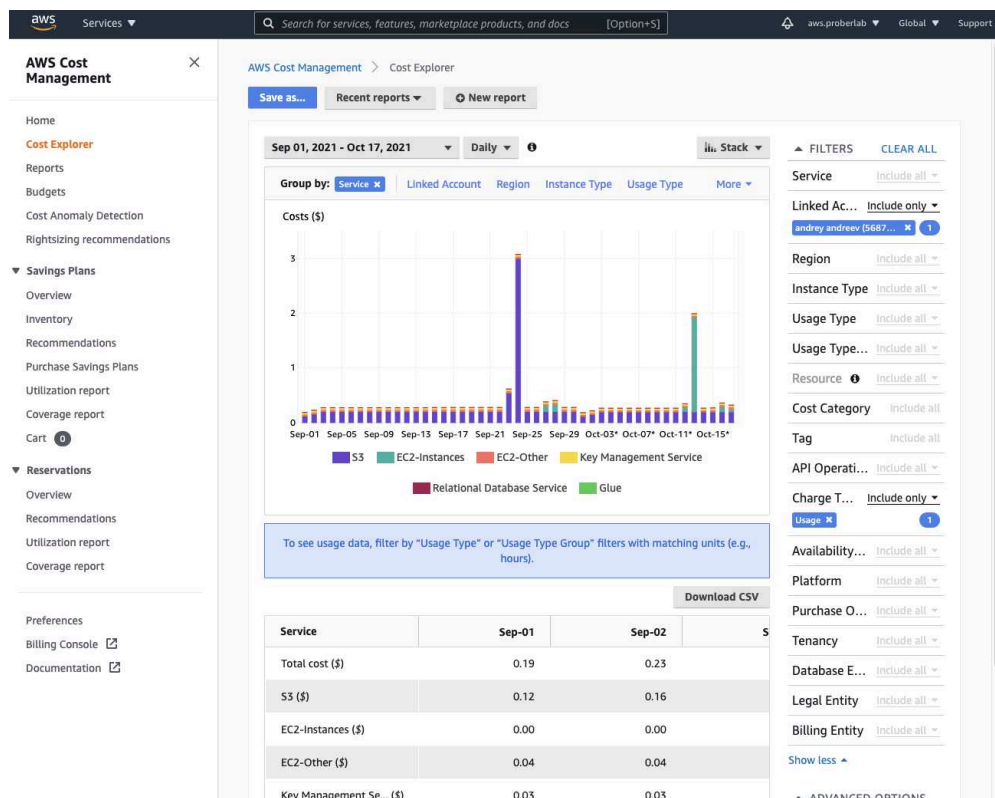


Name



aa-100 USD

- 6 It is simple to check the spending by individual account using Cost Explorer in the organization Billing Dashboard. You can select account (Linked Account). It is important to also pick correct Charge Type (most commonly you can use Credit)



EC2: Starting up a computing instance

- 7 We consulted two tutorials to set up computing instances:
 1. By Justin Bois, the AWS setup and usage lesson in his Caltech BE/Bi 103b class "Statistical Inference in the Biological Sciences" https://bebi103b.github.io/lessons/08/aws_setup.html
 2. By Chris Albon: **Run Project Jupyter Notebooks On Amazon EC2**
- 8 Find a new instance image (operating system) from the **AWS marketplace**, either Amazon Linux 2, or Ubuntu 18.04 or 20.04.
We recommend using Amazon Linux 2
- 9 On the Launch page, in the dropdown Choose Action menu, choose Launch through EC2



About ▼

Categories ▼

Delivery Methods ▼

Solutions ▼

AWS



Amazon
(64-bit x86_64)

Review your configuration and choose an action

Configuration Details

Fulfillment Option

Software Version

Region

[Usage Instructions](#)

Choose Action

[Launch through EC2](#)

[AWS Marketplace on Twitter](#)

**Solutions**

Business Applications

Data & Analytics

DevOps

Infrastructure Software

Internet of Things

Machine Learning

Bus

Blo

Col

Pro

Con

Con

CRM

- 10 This will open a Launch wizard. In the Launch Instance wizard, you will be brought through a 5 part launch sequence.

10.1 Chose AMI

Amazon Machine Image (AMI) is the blueprint of the operating system (OS). It can be Linux, FreeBSD, or even Windows. We recommend Amazon Linux 2 for this guide. Later on in this guide we discuss creating your own, personalized, private Amazon Machine Images.

10.2 Choose Instance Type.

This will configure your virtual machine "hardware" such as memory size and CPU. Some relevant parameters here are memory size (RAM), number of cores, and internet speed. For most applications, RAM should be at least 32 GB, the number of cores should be at least 8, and the internet should be very fast, especially because we will be streaming tens or even hundreds of GB of data from S3 to EC2.



Services ▼

1. Choose AMI

2. Choose Instance Type

3. Configure Instance

4. Add Tags

Step 2: Choose an Instance Type

| | | | |
|-------------------------------------|-----|-------------|--|
| <input type="checkbox"/> | | | |
| <input type="checkbox"/> | c4 | c4.4xlarge | |
| <input type="checkbox"/> | c4 | c4.8xlarge | |
| <input type="checkbox"/> | c5 | c5.large | |
| <input checked="" type="checkbox"/> | c5 | c5.xlarge | |
| <input type="checkbox"/> | c5 | c5.2xlarge | |
| <input type="checkbox"/> | c5 | c5.4xlarge | |
| <input type="checkbox"/> | c5 | c5.9xlarge | |
| <input type="checkbox"/> | c5 | c5.12xlarge | |
| <input type="checkbox"/> | c5 | c5.18xlarge | |
| <input type="checkbox"/> | c5 | c5.24xlarge | |
| <input type="checkbox"/> | c5 | c5.metal | |
| <input type="checkbox"/> | c5a | c5a.large | |



10.3 **Configure instance details**

Accept defaults

10.4 **Add Storage**

This local storage is fast but will be destroyed after you terminate instance. To start, give your instance around 3x the size of your current dataset. This provides very fast "local" storage to the instance (compared to slower S3 storage)


10.5 **Add Tags**

This is optional for better organization of instances

10.6 **Configure Security group**

Most of the time you want port 22 (ssh) to be accessible for remote connections. Jupyter Notebooks server by default runs at port 8888, so that one should also be open.

Following Justin Bois's tutorial, your security rules should look like this.


Services ▾

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add

Step 6: Configure Security Group


A security group is a set of firewall rules that control the traffic for your instance. To protect your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports.

Assign a security group: ☒ Create a **new** security group ☐ Select an **existing** security group

Security group name:

Description:

| Type ⓘ | Protocol ⓘ |
|---|----------------------------------|
| <input type="text" value="SSH"/> ▾ | <input type="text" value="TCP"/> |
| <input type="text" value="HTTPS"/> ▾ | <input type="text" value="TCP"/> |
| <input type="text" value="Custom TCP F"/> ▾ | <input type="text" value="TCP"/> |


Warning
Rules with source of 0.0.0.0/0 allow all IP addresses to access

10.7 Review Instance Launch

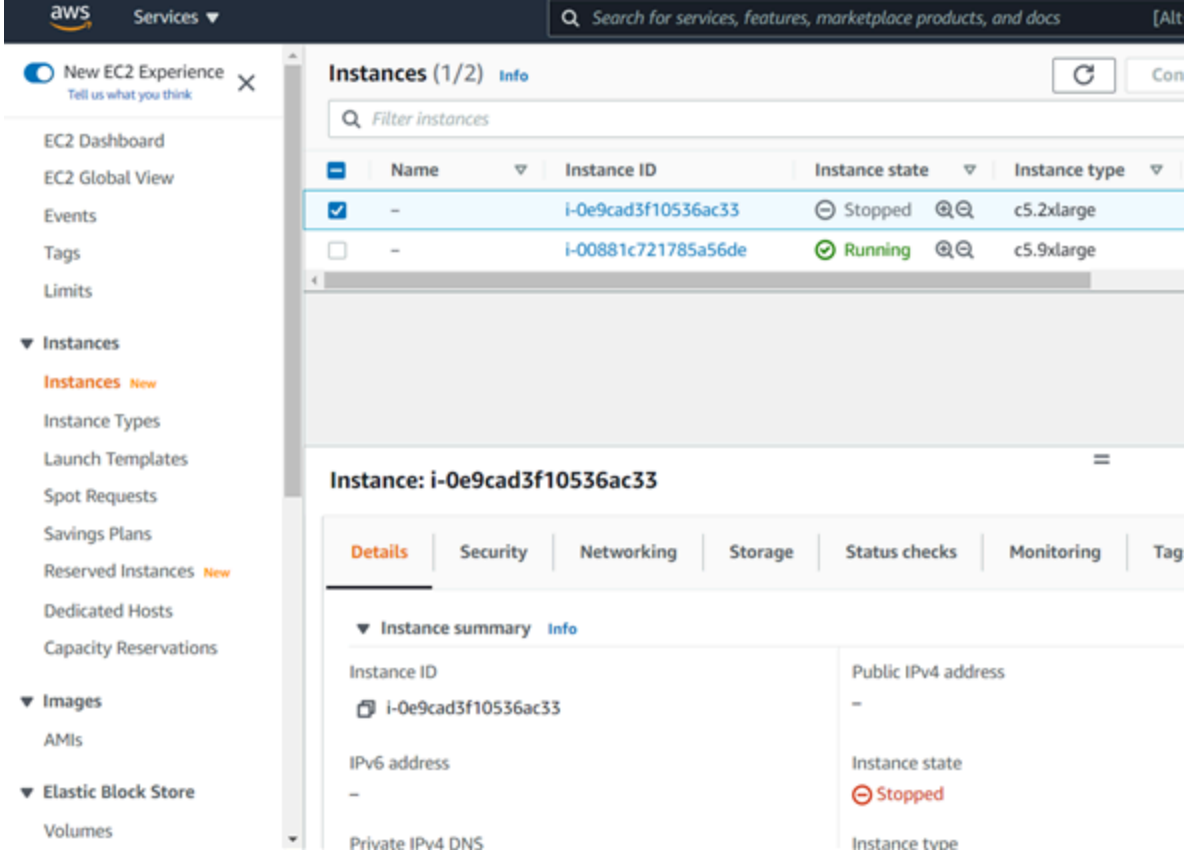
Select Launch. This will open the Key Pair popup.

1. This is going to give you a "key". Put it somewhere safe, meaning a location that is not synched to the Internet (Dropbox or Google Drive). The key will grant full access to the instance.
2. Click launch instance. It usually takes less than two minutes for instance to start.

EC2: Connecting to and using your instance

- 11 If you have just created the instance, it will start automatically.
If the instance has been "Stopped", start your instance. Process should take less than few minutes.

NB: at the end of the work with the instance, you can either Stop or Terminate instance. Stopped instance will have all its memory saved in S3 storage (and you will be billed for every GB of stored data according to S3 prices). Terminated instance will have its memory completely destroyed, and you will lose any information saved in the instance. If you don't stop the instance, you will be billed for the time it is Running, regardless of whether you are actually using it for any computations or not. We recommend stopping instances at the end of work. Check with AWS pricing and your budget to estimate cost of storing instance data.






The screenshot displays the AWS Management Console interface for EC2 instances. On the left, a navigation sidebar lists various services and tools. The main panel shows a table of instances with columns for Name, Instance ID, Instance state, and Instance type. Two instances are listed: one in a 'Stopped' state and one in a 'Running' state. Below the table, the details for the 'Stopped' instance (i-0e9cad3f10536ac33) are expanded, showing its configuration details such as Public IPv4 address, IPv6 address, Instance state (Stopped), and Instance type (c5.9xlarge).



- 12 Click Instance ID to view the instance summary page and select connect.
- 13 Using web-based terminal: Select connect, which will open the terminal in another tab



 Services 



EC2 > Instances > i-00881c721785a56de > Connect to

Connect to instance [Info](#)


Connect to your instance i-00881c721785a56de using any of these c

EC2 Instance Connect


Session Manager

SS

Instance ID

 i-00881c721785a56de


Public IP address

 54.177.139.194

User name

ec2-user

Connect using a custom user name, or use the default user name ec2

 **Note:** In most cases, the guessed user name is corre
the AMI owner has changed the default AMI user na



14 Using terminal / ssh:

```
chmod 600 /path/to/key.pem  
ssh -i /path/to/key.pem ec2-user@[public DNS]
```

Installing dependencies

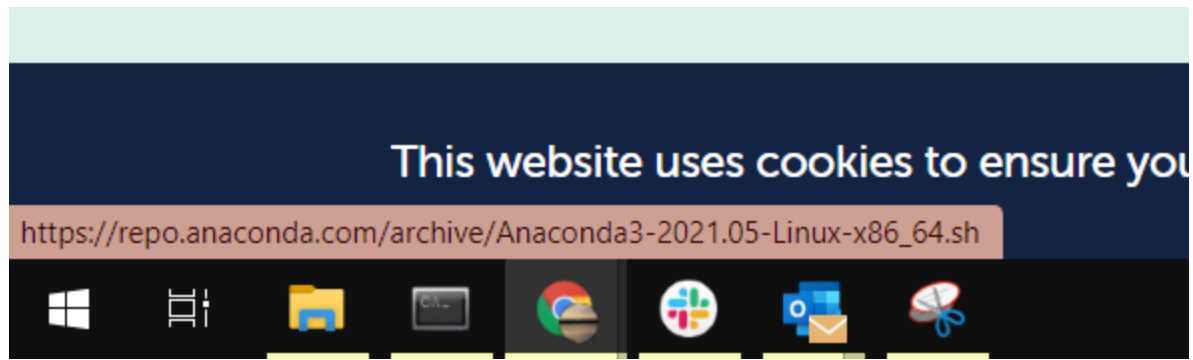
- 15 Instance is a virtual computer, and you should keep it organized like any other computer. For example, you might need to get code using Git or download from the internet. Cloned or downloaded git repositories go in the *git* folder, and all other downloads go in the *Downloads* folder. To create these folders use commands:

```
mkdir Downloads  
mkdir git
```

- 16 Per Chris Albon's tutorial, install conda as follows:

- 16.1 On the downloads page for <https://anaconda.com>, right click on the download link for Linux, and select "Copy link address".





If your local machine computer runs Windows or Mac instead of Linux, you will likely be prompted with a link for your particular operating system (OS). Take care not to copy that link. **The link should end in .sh, not .exe or .pkg and look something like this:**

https://repo.anaconda.com/archive/Anaconda3-2021.05-Linux-x86_64.sh

To install use commands:

```
wget <Linux Anaconda installer link>.sh  
bash <Linux Anaconda installer link>.sh
```

Follow the prompts on the screen to complete the installation

- 16.2 Follow the suite2p install instructions here: <https://github.com/MouseLand/suite2p>. The first step is to download a YAML file, which is not necessarily clear on Linux. To download it, we will use wget again:

```
wget  
https://raw.githubusercontent.com/MouseLand/suite2p/main/environment.yml
```

- 16.3 Activate your new suite2p environment:



```
conda activate suite2p
```

16.4 Install jupyter lab:

```
conda install -c conda-forge jupyter lab
```

16.5 If you skipped Anaconda installation, use

```
pip3 install jupyter
```

and

```
pip3 install boto3
```

16.6 Install smart_open for s3:

```
pip install smart_open[s3].
```

Make sure you install this library for s3 specifically, as specified here.

17 If your EC2 instance's OS is not Amazon Linux 2, install aws cli:

```
sudo apt install awscli
```

18 Install Julia

18.1 From your Downloads directory:



```
wget https://julialang-s3.julialang.org/bin/linux/x64/1.6/julia-1.6.2-linux-x86_64.tar.gz
```

```
tar zxvf julia-1.6.2-linux-x86_64.tar.gz
```

18.2 Install IJulia. This will allow you to run interactive julia environment inside Jupyter notebooks.

First, start Julia

```
./julia-1.6.2/bin/julia
```

And then install IJulia

```
] add IJulia
```

18.3 Within Julia interactive shell install necessary packages to work with TIFF images:

```
] import Pkg; Pkg.add("Images")  
] import Pkg; Pkg.add("TiffImages")  
] import Pkg; Pkg.add("FileIO")
```

Note that we are installing dependencies in a slightly different way than how we installed IJulia. These methods are equivalent.

Starting Jupyter lab

19

You are mostly finished with dependencies! You'll add more as necessary, but these are the most important. We will treat this section as though you have freshly started your EC2 instance.

20 Start conda environment:

```
conda activate suite2p
```


21 1. Open a persistent jupyter lab session:

```
nohup jupyter lab --ip 0.0.0.0 --  
NotebookApp.max_buffer_size=75368709120 &
```

nohup command starts the jupyter server in the background, not tied to your connection to EC2 instance via ssh. The log created by jupyter server will be written to *nohup.out* file

Several parameters are specified here so that:

1. Jupyter lab will be accessible from your browser window (--ip=0.0.0.0)
2. The memory size limit is increased (--NotebookApp.max_buffer_size=75368709120).

For applications that use a lot of memory like suite2p, Kilosort, Caiman, etc Python needs to make really big arrays, and the default memory size for jupyter lab (but not for the AWS instance) is too small. For work with 20GB datasets we use ~75 GB RAM

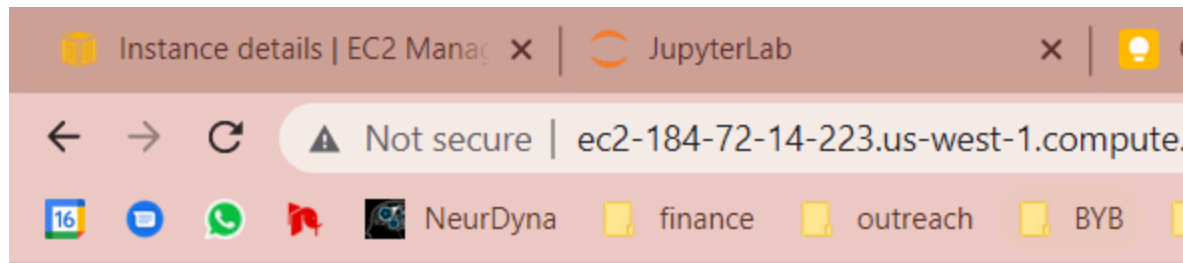
22 Open the nohup.out file to get the token.

```
cat nohup.out
```

Copy the string following "lab?token="

```
[base] [ec2-user@ip-172-31-11-39 ~]$ conda activate suite2p  
(suite2p) [ec2-user@ip-172-31-11-39 ~]$ nohup jupyter lab --ip 0.0.0.0 &  
[1] 4278  
(suite2p) [ec2-user@ip-172-31-11-39 ~]$ nohup: ignoring input and appending output to 'nohup.out'  
  
(suite2p) [ec2-user@ip-172-31-11-39 ~]$  
(suite2p) [ec2-user@ip-172-31-11-39 ~]$ ls  
anaconda3  code  data  Downloads  git  nohup.out  
(suite2p) [ec2-user@ip-172-31-11-39 ~]$ cat nohup.out  
[I 2021-09-28 18:43:07.264 ServerApp] jupyterlab | extension was successfully linked.  
[W 2021-09-28 18:43:07.269 NotebookApp] 'max_buffer_size' has moved from NotebookApp to ServerApp. This config will be passed to ServerApp. Be sure to update  
your config before our next release.  
[I 2021-09-28 18:43:07.739 ServerApp] nbclassic | extension was successfully linked.  
[I 2021-09-28 18:43:07.851 ServerApp] nbclassic | extension was successfully loaded.  
[I 2021-09-28 18:43:07.852 LabApp] JupyterLab extension loaded from /home/ec2-user/anaconda3/envs/suite2p/lib/python3.8/site-packages/jupyterlab  
[I 2021-09-28 18:43:07.852 LabApp] JupyterLab application directory is /home/ec2-user/anaconda3/envs/suite2p/share/jupyter/lab  
[I 2021-09-28 18:43:07.855 ServerApp] jupyterlab | extension was successfully loaded.  
[I 2021-09-28 18:43:07.856 ServerApp] Serving notebooks from local directory: /home/ec2-user  
[I 2021-09-28 18:43:07.856 ServerApp] Jupyter Server 1.11.0 is running at:  
[I 2021-09-28 18:43:07.856 ServerApp] http://ip-172-31-11-39.us-west-1.compute.internal:8888/lab?token=1d7bb26690646a92635d792f5f531c699f1ecb17705a976  
[I 2021-09-28 18:43:07.856 ServerApp] or http://127.0.0.1:8888/lab?token=1d7bb26690646a92635d792f5f531c699f1ecb17705a976  
[I 2021-09-28 18:43:07.856 ServerApp] Use Control-C to stop this server and shut down all kernels (twice to skip confirmation).  
[W 2021-09-28 18:43:07.860 ServerApp] No web browser found: could not locate runnable browser.
```

Use this token to login into the Python notebook:



Password or token:

Log in

Token authentication is enabled

If no password has been configured, you need to open the server

The command:

```
jupyter server list
```

will show you the URLs of running servers with their tokens, wh

Currently running servers:

```
http://localhost:8888/?token=c8de56fa... :: /Users/you/not
```

or you can paste just the token value into the password field on t

See [the documentation on how to enable a password](#) in place

Cookies are required for authenticated access to notebooks.

Setup a Password

You can also setup a password by entering your token and a new

Token

New Password

Log in and set new password

Alternatively, you might need to start up a persistent server. A persistent server will not depend on the connection between your personal computer and the EC2 instance.

22.1 The server will start a process. To find its ID, run in terminal:

1. To stop Jupyter notebook server (or lab) copy the process number and kill it:

```
ps -aux | grep python
```

```
(suite2p) [ec2-user@ip-172-31-11-39
ec2-user 6064 1.6 0.1 685628 1145
/jupyter-lab --ip 0.0.0.0
ec2-user 6074 772 26.5 29453020 19
al/share/jupyter/runtime/kernel-700b
ec2-user 6090 0.4 0.0 886772 5736
al/share/jupyter/runtime/kernel-0442
ec2-user 6170 0.0 0.0 119420 96
```

To stop Jupyter notebook server (or lab) copy the process number and kill it:

```
kill <process number>
```



S3: Uploading data *via* web-interface

- 23 To start, create a Bucket for your experimental data. It is important to block public access to the bucket

The screenshot shows the 'Create bucket' page in the Amazon S3 console. The breadcrumb trail is 'Amazon S3 > Create bucket'. The main heading is 'Create bucket' with an 'Info' link. Below it, a note states: 'Buckets are containers for data stored in S3. [Learn more](#)'. The 'General configuration' section contains a 'Bucket name' input field with the text 'experiment-name-data'. A note below the field says: 'Bucket name must be unique and must not contain spaces or uppercase letters. [See rules for bucket naming](#)'. The 'AWS Region' is set to 'US West (Oregon) us-west-2'. There is a section for 'Copy settings from existing bucket - optional' with a note: 'Only the bucket settings in the following configuration are copied.' and a 'Choose bucket' button.

- 24 You can use web interface to drag-and-drop data to upload it into the bucket or into a folder within bucket

The screenshot shows the 'Upload' page in the Amazon S3 console. The breadcrumb trail is 'Amazon S3 > experiment-name-data > Upload'. The main heading is 'Upload' with an 'Info' link. Below it, a note states: 'Add the files and folders you want to upload to S3. To upload a file larger than 160GB, use the AWS CLI, AWS SDK or Amazon S3 REST API. [Learn more](#)'. A dashed box contains the instruction: 'Drag and drop files and folders you want to upload here, or choose **Add files**, or **Add folders**.' Below this is a section titled 'Files and folders (0)' with a 'Remove' button, an 'Add files' button, and an 'Add folder' button. A note says: 'All files and folders in this table will be uploaded.' There is a search bar with the placeholder 'Find by name'. Below the search bar is a table header with columns: 'Name', 'Folder', 'Type', and 'Size'. The table is currently empty.

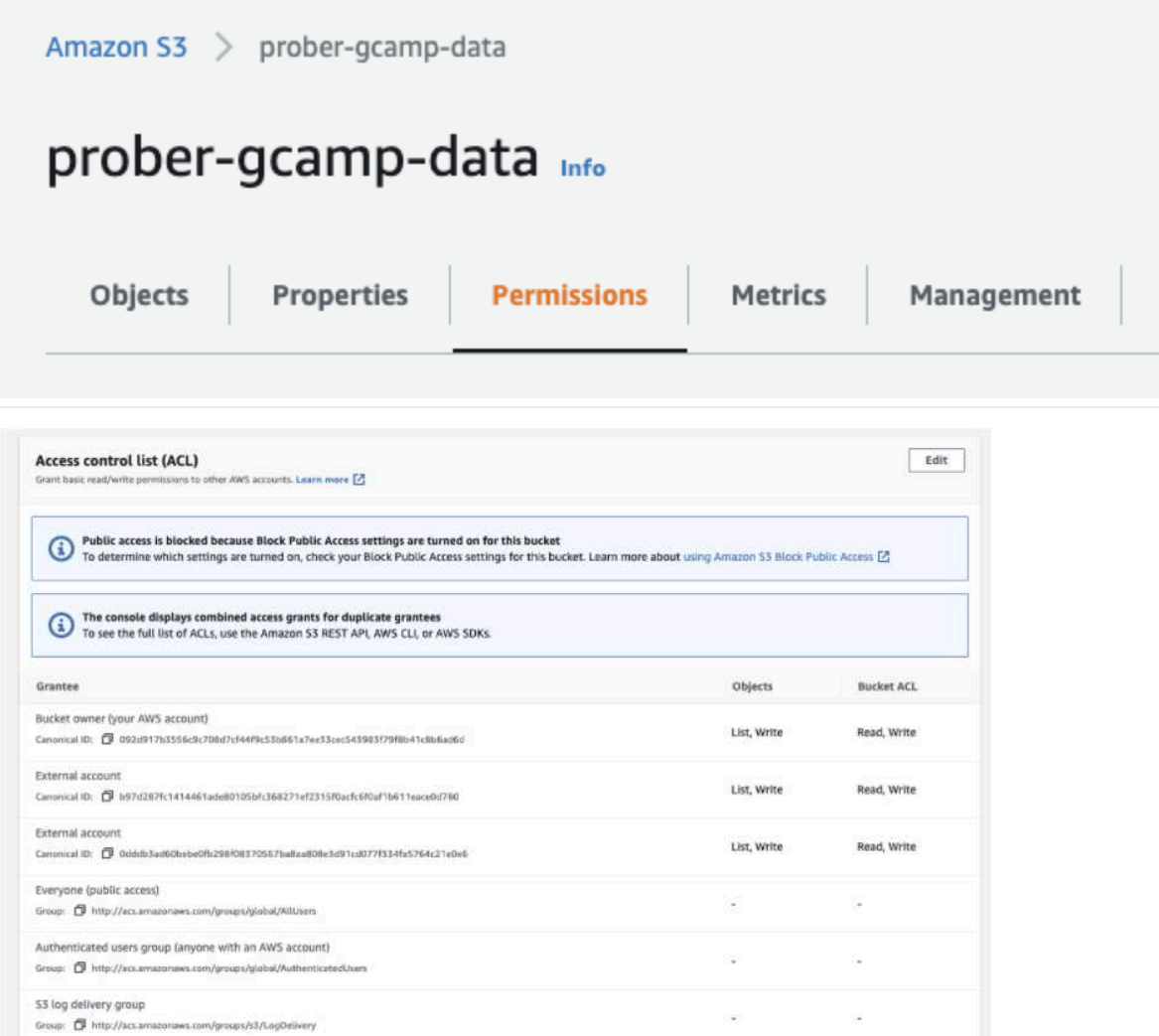
Expected speed of upload is around 10MB/s.

Largest file size allowed through web-interface is 160GB.

- 25 You can also use **command-line interface** to upload data in more automated way or upload larger files

S3: Setting up access for S3 bucket from AWS/EC2 instance

- 26 For each user, add their canonical ID to the bucket's access list (need to allow both list and write permissions). That will allow these users to access bucket from Python interface



Amazon S3 > prober-gcamp-data

prober-gcamp-data Info

Objects | Properties | **Permissions** | Metrics | Management

Access control list (ACL) Edit

Grant basic read/write permissions to other AWS accounts. [Learn more](#)

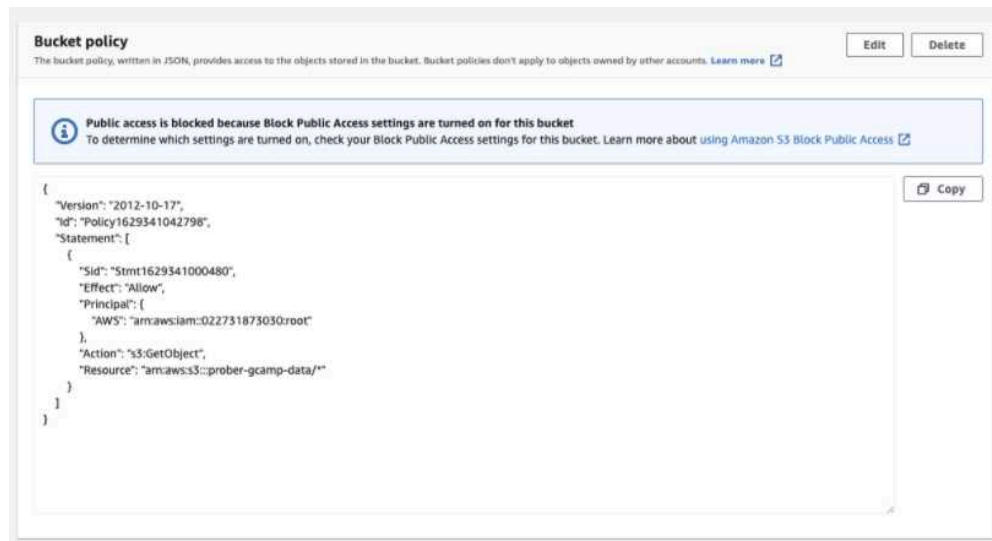
Public access is blocked because Block Public Access settings are turned on for this bucket
To determine which settings are turned on, check your Block Public Access settings for this bucket. [Learn more about using Amazon S3 Block Public Access](#)

The console displays combined access grants for duplicate grantees
To see the full list of ACLs, use the Amazon S3 REST API, AWS CLI, or AWS SDKs.

| Grantee | Objects | Bucket ACL |
|---|-------------|-------------|
| Bucket owner (your AWS account) Canonical ID: 092d917b3556c9c70bd7c1f44f9c53b861a7ec33cc543983779f8b41c8b6a06d | List, Write | Read, Write |
| External account Canonical ID: b97d287fc1414461ade80105bf368271ef2315f0acfc6f0af1b611eace0f780 | List, Write | Read, Write |
| External account Canonical ID: 0d6db3a60bebe0fb298f08370567ba8a808c3d91cd077f534fe5764c21e0e6 | List, Write | Read, Write |
| Everyone (public access) Group: http://acs.amazonaws.com/groups/global/AllUsers | - | - |
| Authenticated users group (anyone with an AWS account) Group: http://acs.amazonaws.com/groups/global/AuthenticatedUsers | - | - |
| S3 log delivery group Group: http://acs.amazonaws.com/groups/s3/LogDelivery | - | - |

- 26.1 Canonical ID can be found in AWS Web Interface → My Security Credentials → Canonical User ID (**AWS manual**)

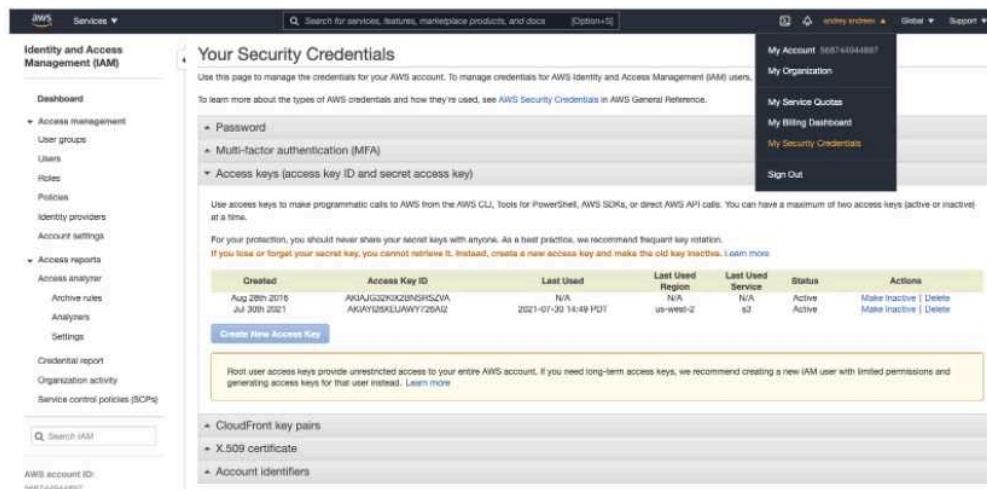
- 27 To allow downloading data from web browser interface, create a custom rule. It gives users who did not initially upload the data access to it through S3's web interface at <https://s3.console.aws.amazon.com/>



Example of policy to give user access to reading data from bucket. It has been generated using **AWS Policy Generator**

Here the "Principal" is the user that gets permission to read data from the bucket specified in "Resource" field

- 28 Each user will have to create custom key pair for your account under security Credentials → Access keys.
This will allow you to create Access Key ID and Secret Access Key.



- 29 Connect to the EC2 instance using EC2 Connect function or *ssh*. Run *aws configure* to configure the instance.
Make sure to enter the correct zone ID.

```
Access key ID: ***  
Secret access key: ***  
Default region name: us-west-2  
Default output format: json
```

- 30 While logged into the instance, run *python* to launch the Python interpreter

- 31 Test connection to bucket via the **boto3** library:

```
import boto3  
s3 = boto3.resource('s3')  
bucket = s3.Bucket('')  
for obj in bucket.objects.all():  
    print(obj.key)
```

This should print all items in the bucket

Accessing files from Python using boto3

- 32 We have just shown how to confirm that the S3 bucket is visible to your EC2 instance. To run full data processing pipelines with suite2p and other such programs, we need to go one step further, and download files into our local filesystem. Let's go through some Python code to see how it works.

[1]:

```
import os
import boto3
import matplotlib.pyplot as plt
import numpy as np
import io
import tqdm

# smart_open allows us to open large data files
from smart_open import open, register_compressor
from suite2p import run_s2p, default_ops
```

[2]:

```
# As a sanity check, list filenames from your bucket.
# We have replaced the real Bucket name with '{bucket-name}', but
# we have left the output in place.
s3 = boto3.resource('s3')
bucket = s3.Bucket('{bucket-name}')
for obj in bucket.objects.all():
    print(obj.key)
```

[Out]:

```
210610_920/300ms_bin1x_fullpower_1/300ms_bin1x_fullpower_1_MMStack
_Pos0.ome.tif
210610_920/300ms_bin1x_fullpower_1/300ms_bin1x_fullpower_1_MMStack
_Pos0_1.ome.tif
210610_920/300ms_bin1x_fullpower_1/300ms_bin1x_fullpower_1_MMStack
_Pos0_2.ome.tif
210610_920/300ms_bin1x_fullpower_1/300ms_bin1x_fullpower_1_MMStack
_Pos0_3.ome.tif
```

[3]:

```
# Next, we are going to download our files into the local
# filesystem. This will be very fast because we chose an instance
# type with fast internet.
# Make data directory
data_dir = "../data/210610_920/300ms_bin1x_fullpower_1/"
ifnot os.path.isdir(data_dir):
    os.makedirs(data_dir)
```


[4]:

```
# Write all datafiles locally
# ImageJ acquires large data by splitting it into 4GB files

for fname in
tqdm.tqdm(["300ms_bin1x_fullpower_1_MMStack_Pos0.ome.tif",
"300ms_bin1x_fullpower_1_MMStack_Pos0_1.ome.tif",
"300ms_bin1x_fullpower_1_MMStack_Pos0_2.ome.tif",
"300ms_bin1x_fullpower_1_MMStack_Pos0_3.ome.tif"]):

    # Combine path and name
    data_file = os.path.join(data_dir, fname)

    # If not a file, write it. Expected speed is 10GBps between EC2
    and S3, so 4GB file should be downloaded in 3 sec
    ifnot os.path.isfile(data_file):
    # Get object from S3
        s3.meta.client.download_file(
'prober-gcamp-data',
        os.path.join("210610_920/300ms_bin1x_fullpower_1/",
fname),
        data_file
        )

        print("wrote file successfully")
    else:
        print("file already exists")
```

[5]:

```
# Populates ops with the default options, except for batch_size,
which we will make smaller to ensure the kernel does not shut
down.
ops = default_ops()
ops['batch_size'] = 100
# Only run on specified tiffs
db = {
    'h5py': [], # a single h5 file path
    'h5py_key': 'data',
    'look_one_level_down': False, # Whether to look in ALL subfolders
    when searching for tiffs
    'data_path': [data_dir],
    'tiff_list': [elem for elem in os.listdir(data_dir) if
    elem.endswith(".tif")]
}

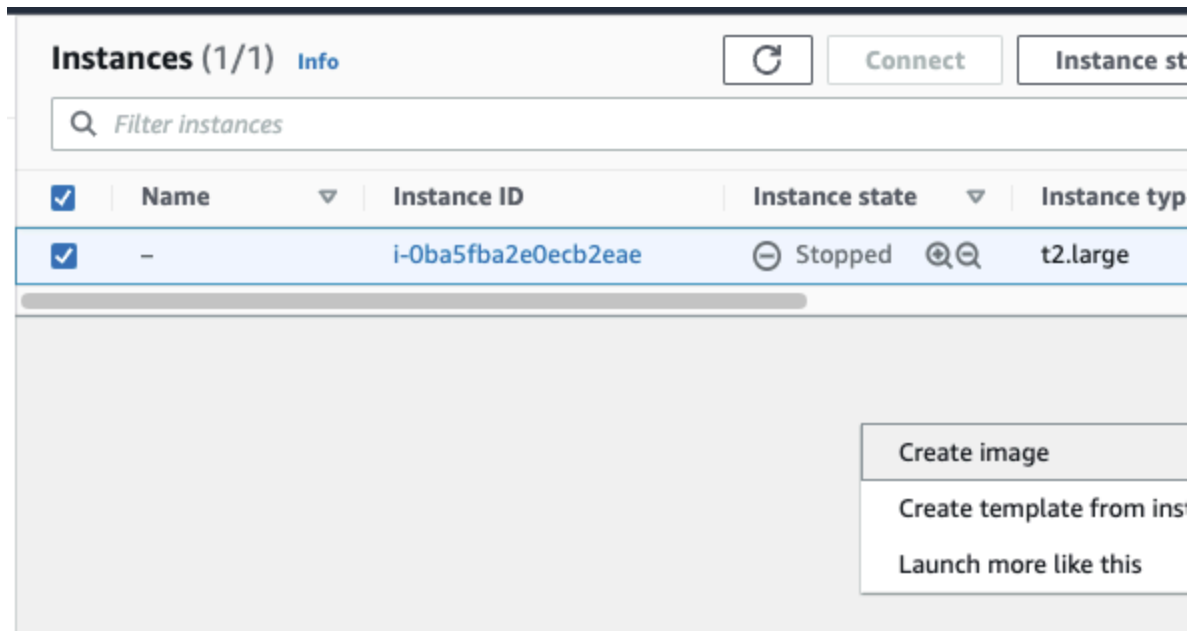
# Run suite2p
opsEnd = run_s2p(ops=ops, db=db)
```

[Out]:

```
{'h5py': [], 'h5py_key': 'data', 'look_one_level_down': False,  
    'data_path': ['./data/210610_920/300ms_bin1x_fullpower_1/'],  
    'tiff_list': ['300ms_bin1x_fullpower_1_MMStack_Pos0.ome.tif',  
        '300ms_bin1x_fullpower_1_MMStack_Pos0_1.ome.tif',  
        '300ms_bin1x_fullpower_1_MMStack_Pos0_2.ome.tif',  
        '300ms_bin1x_fullpower_1_MMStack_Pos0_3.ome.tif']}]  
  
FOUND BINARIES AND OPS IN  
['../data/210610_920/300ms_bin1x_fullpower_1/suite2p/plane0/ops.npy']  
  
>>>>>>>>>>>>>>>>>>>> PLANE 0 <<<<<<<<<<<<<<<<<<<  
NOTE: not registered / registration forced with  
ops['do_registration']>1  
      (no previous offsets to delete)  
----- REGISTRATION  
  
registering 2000 frames  
Reference frame, 103.47 sec.  
Registered 400/2000 in 188.83s  
Registered 800/2000 in 376.80s  
Registered 1200/2000 in 564.32s  
Registered 1600/2000 in 751.31s  
Registered 2000/2000 in 939.14s  
  
/home/ec2-user/anaconda3/envs/suite2p/lib/python3.8/site-  
packages/suite2p/registration/register.py:44: RuntimeWarning:  
invalid value encountered in true_divide  
    dxy = dxy / dxy.mean()  
  
added enhanced mean image  
----- Total 1269.39 sec
```

Creating AMI (Amazon Machine Image) from configured instance


- 33 It is useful to save your configured machine image for later. **Creating Image** will save all packages installed, all data, and all private/sensitive information too.
- 33.1 Go to Actions → Image and Templates → Create Image



- 33.2
1. Pick an appropriate name (for example "conda-image" for version with full installation of Python and Conda) and click Create Image
 2. NB: the image will be created with all the data saved on the "local" disk. So if you have instance with 100GB of data saved, data will be added to the image as well

Note that images are stored on S3 storage service, and you will be charged for storing every image

Now you can launch an identical instance of your customized AMI by selecting your image from "My AMIs" when spinning up a new EC2 instance.



Services ▼

1. Choose AMI
2. Choose Instance Type
3. Configure Instance
4. Add Storage
5. Add Tags
6.


Step 1: Choose an Amazon Machine Image (AMI)

An AMI is a template that contains the software configuration (operating system, application server, and AMIs.


Quick Start
My AMIs
AWS Marketplace
Community AMIs
▼ Ownership


nupack_at_scale - ami-08710ee19e5820d9b

Root device type: ebs
Virtualization type: hvm
Owner: 5


test-image - ami-0af78bffa1fdc5b22

Root device type: ebs
Virtualization type: hvm
Owner: 5


nupack-11 - ami-18710660

Spotlight video

34

https://www.youtube.com/embed/A-Se6176L0E?si=nyVPuOp_JB-ibT-F