Practical
Cloud Computing

Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - Elastic Compute Cloud (EC2)
  - Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow

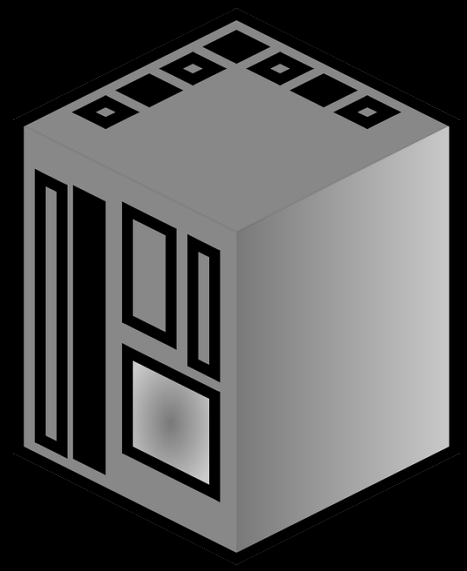
Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - Elastic Compute Cloud (EC2)
  - Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow


Running programs remotely
on distant computers
(In the cloud)

Client - Server

?



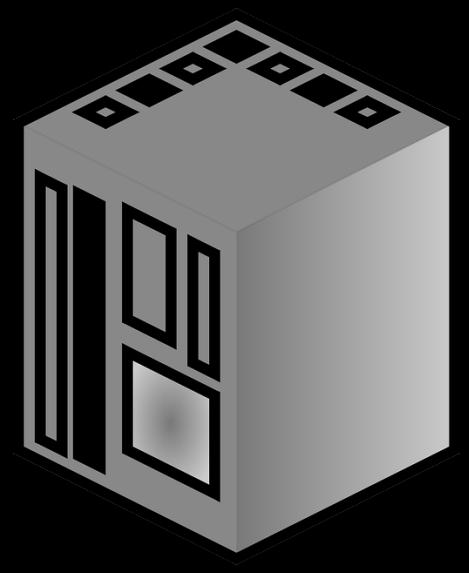
?

```
~ Asciicast recording started.
~ Hit ctrl+d or type "exit" to finish.

root@rublev:/# ls -aslh --color=auto -F
total 100K
4.0K drwxr-xr-x  23 root root 4.0K Aug 14 11:01 ./
4.0K drwxr-xr-x  23 root root 4.0K Aug 14 11:01 ../
4.0K drwxr-xr-x   2 root root 4.0K Mar 14 04:17 bin/
4.0K drwxr-xr-x   3 root root 4.0K Jun  6 13:39 boot/
   0 drwxr-xr-x  14 root root 3.1K Jul 23 11:34 dev/
 12K drwxr-xr-x 151 root root  12K Aug 21 10:08 etc/
4.0K drwxr-xr-x   3 root root 4.0K Mar 14 04:28 home/
   0 lrwxrwxrwx   1 root root   32 Mar 14 03:53 initrd.img -> /boot/initrd.
.2.0-4-686-pae
4.0K drwxr-xr-x  17 root root 4.0K Jul 23 10:57 lib/
 16K drwx------   2 root root  16K Mar 14 03:51 lost+found/
4.0K drwxr-xr-x   4 root root 4.0K Mar 14 03:51 media/
4.0K drwxr-xr-x   4 root root 4.0K May 21 17:46 mnt/
4.0K drwxr-xr-x   5 root root 4.0K Jul 23 11:31 opt/
   0 dr-xr-xr-x 327 root root    0 Jul 23 11:34 proc/
4.0K drwx------   2 root root 4.0K May 29 04:08 .pulse/
4.0K -rw-------   1 root root  256 Mar 14 04:28 .pulse-cookie
4.0K drwx------  15 root root 4.0K Aug 21 09:49 root/
   0 drwxr-xr-x  25 root root 1.1K Aug 21 10:28 run/
4.0K drwxr-xr-x   2 root root 4.0K Jul 23 11:32 sbin/
4.0K drwxr-xr-x   2 root root 4.0K Jun 10  2012 selinux/
4.0K drwxr-xr-x   2 root root 4.0K Mar 14 03:52 srv/
   0 drwxr-xr-x  13 root root    0 Jul 23 11:34 sys/
4.0K drwxrwxrwt  18 root root 4.0K Aug 21 10:34 tmp/
4.0K drwxr-xr-x  10 root root 4.0K Mar 14 03:52 usr/
4.0K drwxr-xr-x  15 root root 4.0K Jul 10 09:15 var/
   0 lrwxrwxrwx   1 root root   28 Mar 14 03:53 vmlinuz -> boot/vmlinuz-3.2
.6.0-pae
```

Client - Server

Client



Server

```
~ Asciicast recording started.
~ Hit ctrl+d or type "exit" to finish.

root@rublev:/# ls -aslh --color=auto -F
total 100K
4.0K drwxr-xr-x  23 root root 4.0K Aug 14 11:01 ./
4.0K drwxr-xr-x  23 root root 4.0K Aug 14 11:01 ../
4.0K drwxr-xr-x   2 root root 4.0K Mar 14 04:17 bin/
4.0K drwxr-xr-x   3 root root 4.0K Jun  6 13:39 boot/
   0 drwxr-xr-x  14 root root 3.1K Jul 23 11:34 dev/
 12K drwxr-xr-x 151 root root  12K Aug 21 10:08 etc/
4.0K drwxr-xr-x   3 root root 4.0K Mar 14 04:28 home/
   0 lrwxrwxrwx   1 root root   32 Mar 14 03:53 initrd.img -> /boot/initrd.
.2.0-4-686-pae
4.0K drwxr-xr-x  17 root root 4.0K Jul 23 10:57 lib/
 16K drwx------   2 root root  16K Mar 14 03:51 lost+found/
4.0K drwxr-xr-x   4 root root 4.0K Mar 14 03:51 media/
4.0K drwxr-xr-x   4 root root 4.0K May 21 17:46 mnt/
4.0K drwxr-xr-x   5 root root 4.0K Jul 23 11:31 opt/
   0 dr-xr-xr-x 327 root root    0 Jul 23 11:34 proc/
4.0K drwx------   2 root root 4.0K May 29 04:08 .pulse/
4.0K -rw-------   1 root root  256 Mar 14 04:28 .pulse-cookie
4.0K drwx------  15 root root 4.0K Aug 21 09:49 root/
   0 drwxr-xr-x  25 root root 1.1K Aug 21 10:28 run/
4.0K drwxr-xr-x   2 root root 4.0K Jul 23 11:32 sbin/
4.0K drwxr-xr-x   2 root root 4.0K Jun 10  2012 selinux/
4.0K drwxr-xr-x   2 root root 4.0K Mar 14 03:52 srv/
   0 drwxr-xr-x  13 root root    0 Jul 23 11:34 sys/
4.0K drwxrwxrwt  18 root root 4.0K Aug 21 10:34 tmp/
4.0K drwxr-xr-x  10 root root 4.0K Mar 14 03:52 usr/
4.0K drwxr-xr-x  15 root root 4.0K Jul 10 09:15 var/
   0 lrwxrwxrwx   1 root root   28 Mar 14 03:53 vmlinuz -> boot/vmlinuz-3.2
.6.0-pae
```

Advantages

- Accessibility (from anywhere)
- Dynamic Scaling
- Storage
- Maintenance

Disadvantages

- **Security**
  - A third party owns the server
- **Cost**
  - In the long term, cloud service can cost a lot

Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - Elastic Compute Cloud (EC2)
  - Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow

Why Amazon

- Flexibility of many instance types
- A lot of traction over the years
- Libraries and tools built around AWS

Amazon Web Services

Compute

EC2
Virtual Servers in the Cloud

Lambda
Run Code in Response to Events

EC2 Container Service
Run and Manage Docker Containers

Storage & Content Delivery

S3
Scalable Storage in the Cloud

Storage Gateway
Integrates On-Premises IT Environments with Cloud Storage

Glacier
Archive Storage in the Cloud

CloudFront
Global Content Delivery Network

Database

RDS
MySQL, Postgres, Oracle, SQL Server, and Amazon Aurora

Administration & Security

Directory Service
Managed Directories in the Cloud

Identity & Access Management
Access Control and Key Management

Trusted Advisor
AWS Cloud Optimization Expert

CloudTrail
User Activity and Change Tracking

Config
Resource Configurations and Inventory

CloudWatch
Resource and Application Monitoring

Deployment & Management

Elastic Beanstalk
AWS Application Container

OpsWorks
DevOps Application Management Service

CloudFormation
Templated AWS Resource Creation

CodeDeploy
Automated Deployments

Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - ★ Elastic Compute Cloud (EC2)
  - ★ Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow

Elastic Compute Cloud (EC2)

- Virtual Machine in the cloud
- Can choose what Machine to launch
  - ★ `RedHat (Linux)`
  - ★ `Ubuntu (Linux)`
- Use local Terminal as the client to access remote machine

Step 1:
Machine Image

Step 1: Choose an Amazon Machine Image (AMI)

Cancel and Exit

My AMIs

AWS Marketplace

Community AMIs

Free tier only

Amazon Linux AMI 2015.03 (HVM), SSD Volume Type - ami-1ecae776

Amazon Linux

Free tier eligible

The Amazon Linux AMI is an EBS-backed, AWS-supported image. The default image includes AWS command line tools, Python, Ruby, Perl, and Java. The repositories include Docker, PHP, MySQL, PostgreSQL, and other packages.

Root device type: ebs Virtualization type: hvm

Select

64-bit

Username: ec2-user

Red Hat Enterprise Linux 7.1 (HVM), SSD Volume Type - ami-12663b7a

Red Hat

Free tier eligible

Red Hat Enterprise Linux version 7.1 (HVM), EBS General Purpose (SSD) Volume Type

Root device type: ebs Virtualization type: hvm

Select

64-bit

Username: ec2-user

SUSE Linux Enterprise Server 12 (HVM), SSD Volume Type - ami-aeb532c6

SUSE Linux

Free tier eligible

SUSE Linux Enterprise Server 12 (HVM), EBS General Purpose (SSD) Volume Type. Public Cloud, Advanced Systems Management, Web and Scripting, and Legacy modules enabled.

Root device type: ebs Virtualization type: hvm

Select

64-bit

Username: root

Ubuntu Server 14.04 LTS (HVM), SSD Volume Type - ami-d05e75b8

Ubuntu

Free tier eligible

Ubuntu Server 14.04 LTS (HVM), EBS General Purpose (SSD) Volume Type. Support available from Canonical (http://www.ubuntu.com/cloud/services).

Select

64-bit

Username: ubuntu

Step 2:
Specifications

Step 2: Choose an Instance Type

	Family	Type	vCPUs	Memory (GiB)	Instance Storage (GB)
☑	General purpose	t2.micro Free tier eligible	1 $0.013 per Hour	1	EBS only
☐	General purpose	t2.small	1 $0.026 per Hour	2	EBS only
☐	General purpose	t2.medium	2 $0.052 per Hour	4	EBS only
☐	General purpose	m3.medium	1 $0.07 per Hour	3.75	1 x 4 (SSD)
☐	General purpose	m3.large	2 $0.14 per Hour	7.5	1 x 32 (SSD)
☐	General purpose	m3.xlarge	4 $0.28 per Hour	15	2 x 40 (SSD)
☐	General purpose	m3.2xlarge	8 $0.56 per Hour	30	2 x 80 (SSD)
☐	Compute optimized	c4.large	2 $0.116 per Hour	3.75	EBS only

Step 3: Configuration

Step 3: Configure Instance Details

Configure the instance to suit your requirements. You can launch multiple instances from the same AMI, request Spot Instances to take a
role to the instance, and more.

Number of instances 1

Purchasing option ☑ Request Spot Instances

Current price

us-east-1b	0.050
us-east-1c	0.800
us-east-1d	1.000
us-east-1e	0.600

Maximum price $ (e.g. 0.045 = 4.5 cents/hour)

Launch group (Optional)

Request valid from Any time Edit

Request valid to Any time Edit

Persistent request ☐ Persistent request

Spot instances are available to reduce price starting from m3.medium

Spot Instance

- Bid the machine with the price you set (as max)
- Takes longer to start
- Cannot stop and restart instance
- Much cheaper in general
- Otherwise expensive to use larger instances

Step 3: Configuration

Step 3: Configure Instance Details

Auto-assign Public IP — Use subnet setting (Enable)

IAM role — None — Create new IAM role

Shutdown behavior — Stop

Enable termination protection — Protect against accidental termination

Monitoring — Enable CloudWatch detailed monitoring
Additional charges apply.

Tenancy — Shared tenancy (multi-tenant hardware)
Additional charges will apply for dedicated tenancy.

Advanced Details

Scripts to run at launch (Installations)

User data — As text / As file / Input is already base64 encoded

Choose File — No file chosen

Step 4:
Storage

Step 4: Add Storage

Your instance will be launched with the following storage device settings. You can attach additional EBS volumes and instance store volumes to your instance edit the settings of the root volume. You can also attach additional EBS volumes after launching an instance, but not instance store volumes. Learn more abo storage options in Amazon EC2.

Type	Device	Snapshot	Size (GiB)	Volume Type	IOPS	Delete on Termination
Root	/dev/xvda	snap-b772aec8	16	General Purpose (SSD)	48 / 3000	☑

Add New Volume

Add more storage as you see fit

Maybe you want to keep the results of your job

Free tier eligible customers can get up to 30 GB of EBS General Purpose (SSD) or Magnetic storage. Learn more about free usage tier eligibility and usage restrictions.

Step 5:
Tagging

Step 5: Tag Instance

A tag consists of a case-sensitive key-value pair. For example, you could define a tag with key = Name and value = Webserver. Learn more about tag

Key (127 characters maximum)	Value (255 characters maximum)
Name	example_1
Purpose	demostration

Create Tag (Up to 10 tags maximum)

- First Key is by default "Name"
- This controls the name that shows up later

- Second Key onwards is whatever you want
- To help you identify your instance

Step 6:
Security Group

Step 6: Configure Security Group

A security group is a set of firewall rules that control the traffic for your instance. On this page, you can add rules to allow specific traffic to reach your instance. For example, if you want t
web server and allow Internet traffic to reach your instance, add rules that allow unrestricted access to the HTTP and HTTPS ports. You can create a new security group or select from ar
below. Learn more about Amazon EC2 security groups.

Assign a security group: Create a **new** security group

Select an **existing** security group

Use previous settings
if you have already defined it

Security group name: launch-wizard-5

Description: launch-wizard-5 created 2015-05-28T18:14:16.592-07:00

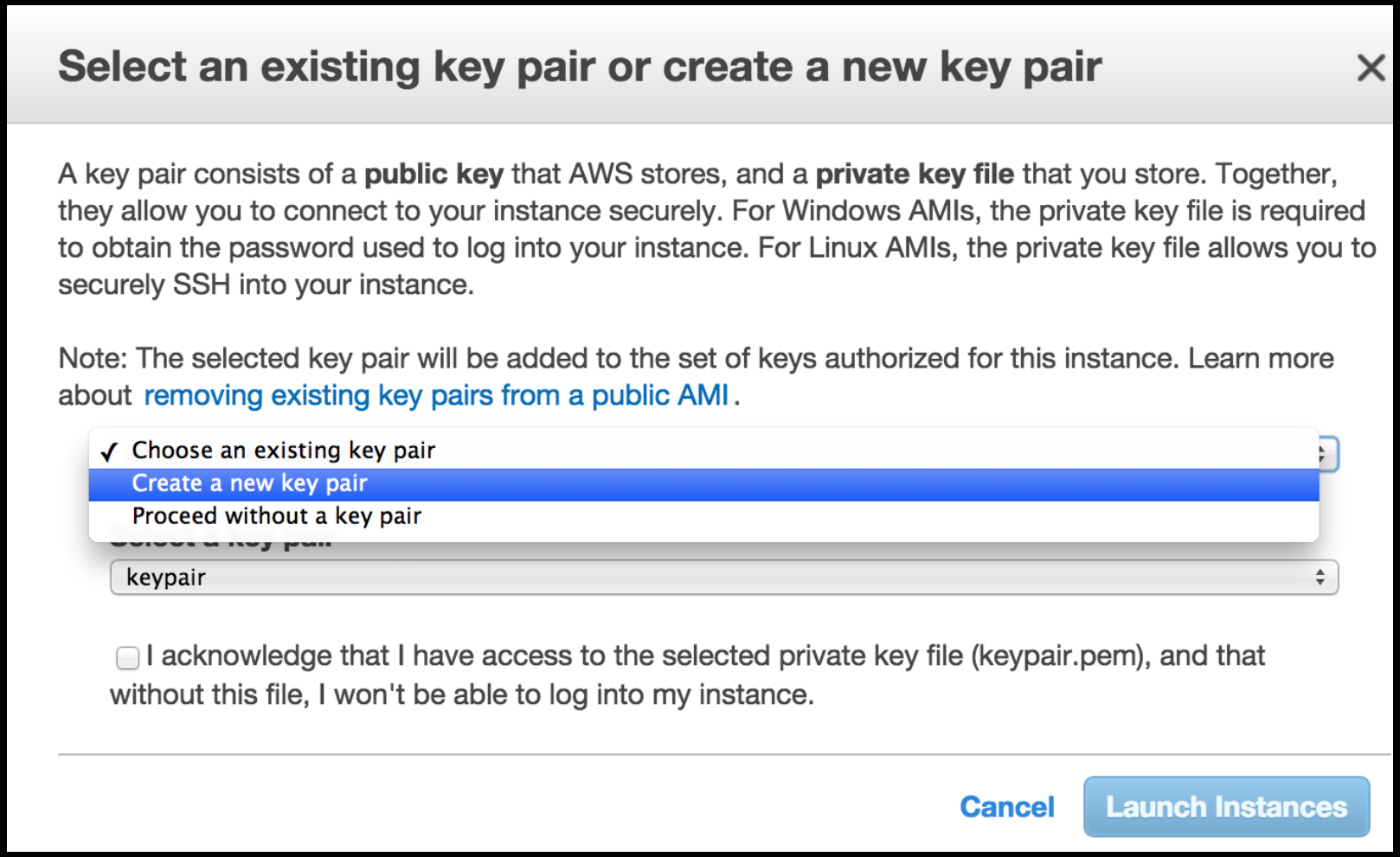
Type	Protocol	Port Range	Source	
SSH	TCP	22	Anywhere	0.0.0.0/0
Custom TCP Rule	TCP	8888	Anywhere	0.0.0.0/0

Add Rule

Otherwise if you want to open up other ports …
(For running your web app mainly)

Caution:

Make sure you have the .pem file locally, otherwise create a new key pair



Change permission of the `.pem` file

- After you have downloaded the `.pem`
- Run this line in the terminal

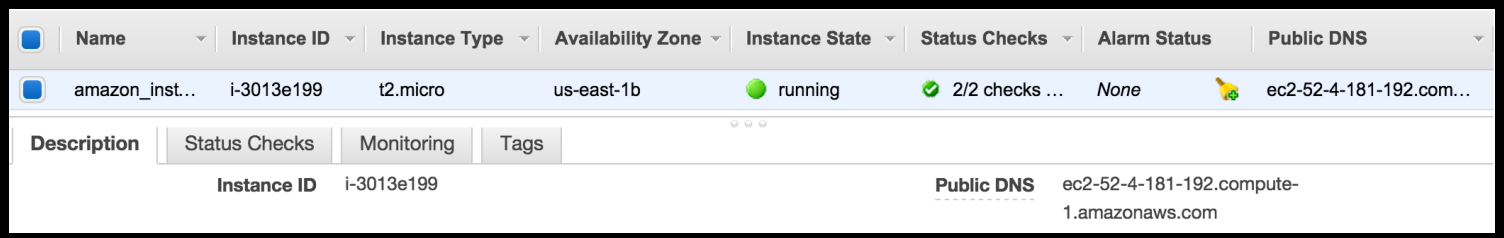
```
chmod 600 my-key-pair.pem
```

Logging into EC2

```
ssh -X -i keypair.pem User@Domain
```

User: `ec2-user` (**See Step 1: Machine Type**)

Domain: `ec2-52-4-181-192.compute-1.amazonaws.com`



Copying files to EC2

```
scp -i keypair.pem SampleFile.txt User@Domain:~

scp -i keypair.pem -r SampleFolder User@Domain:~
```

User: ec2-user (**See Step 1: Machine Type**)

Domain: ec2-52-4-181-192.compute-1.amazonaws.com

Stop / Terminate your instance if you are not running tasks

You can restart later

Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - ⋆ Elastic Compute Cloud (EC2)
  - ⋆ Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow

Simple Storage Service (S3)

- Storage space for big data
- S3 storage —> Permanent and big data
- EC2 storage —> Temporary and small data

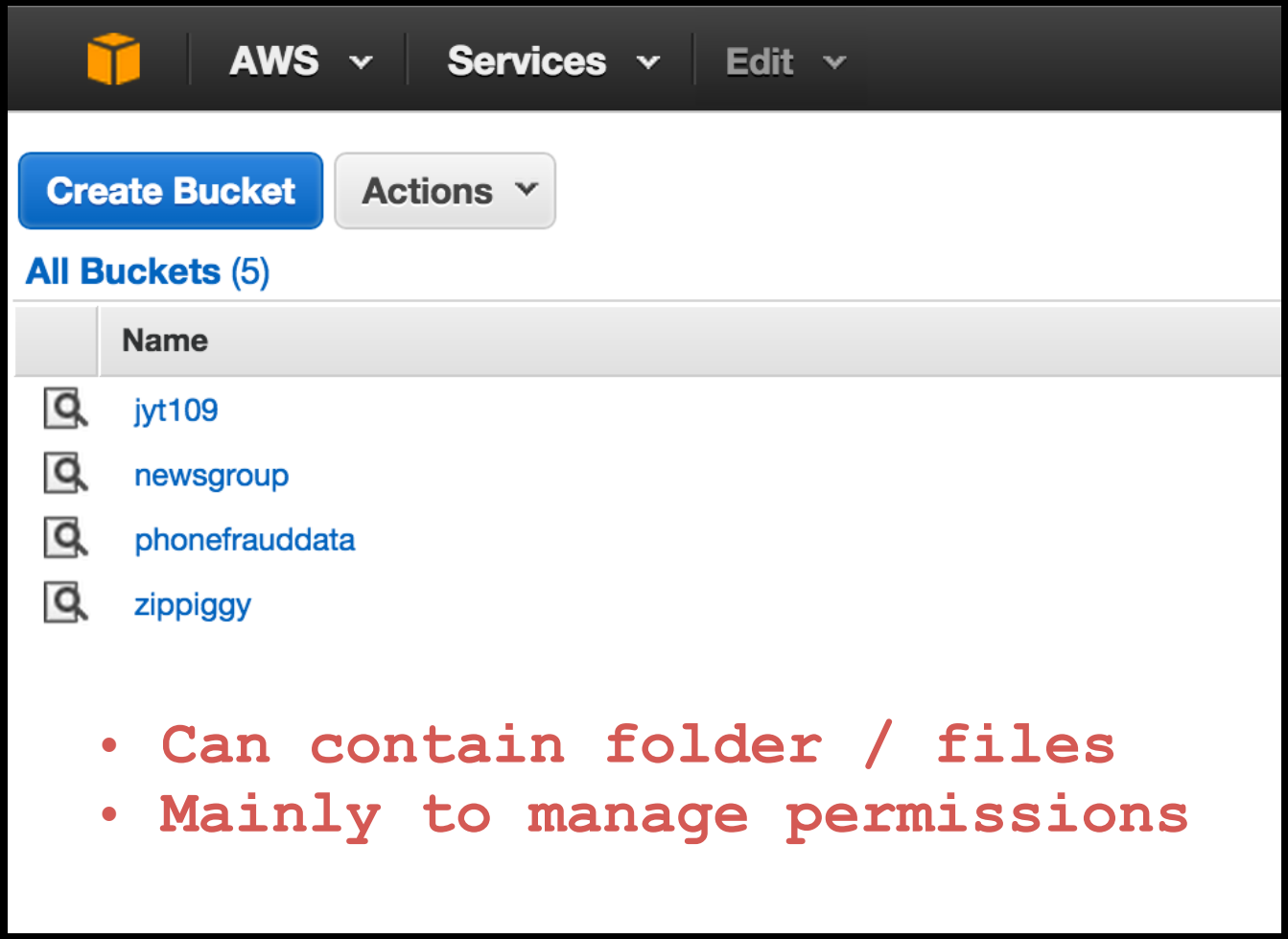
S3 Operations

- Upload
- Download
- Read
- Write

S3 Cost

	Standard Storage
First 1 TB / month	$0.0300 per GB
Next 49 TB / month	$0.0295 per GB
Next 450 TB / month	$0.0290 per GB
Next 500 TB / month	$0.0285 per GB
Next 4000 TB / month	$0.0280 per GB
Over 5000 TB / month	$0.0275 per GB

Buckets



- Can contain folder / files
- Mainly to manage permissions

Create Bucket

AWS Services Edit Jef

that Amazon uses to run its own global network of web sites. The service aims to maximize benefits of scale and to pass those benefits on to developers.

You can read, write, and delete objects ranging
unlimited. Each object is stored in a bucket wit

Get started by simply creating a bucket and up

Create Bucket

S3 at a glance

Create

Create a bucket in one of several

Upload objects to your bucket. Amazon

Manage your data with Amazon S3's

Create a Bucket - Select a Bucket Name and Region

Cancel

A bucket is a container for objects stored in Amazon S3. When creating a bucket, you can choose a Region to optimize for latency, minimize costs, or address regulatory requirements. For more information regarding bucket naming conventions, please visit the Amazon S3 documentation.

Bucket Name: samplebucket

- Bucket name must be unique
  i.e. no one has ever used it before
- Must be all lower case
- Must not have underscore

eate Cancel

Bucket Permissions

None | Properties | Transfers

Bucket: jyt109

Bucket: jyt109
Region: US Standard
Creation Date: Thu Sep 04 19:02:10 GMT-700 2014
Owner: jeffrey.tang09

Permissions

You can control access to the bucket and its contents using access policies. For more information, see Managing Access Permissions in the Amazon S3 Developer Guide.

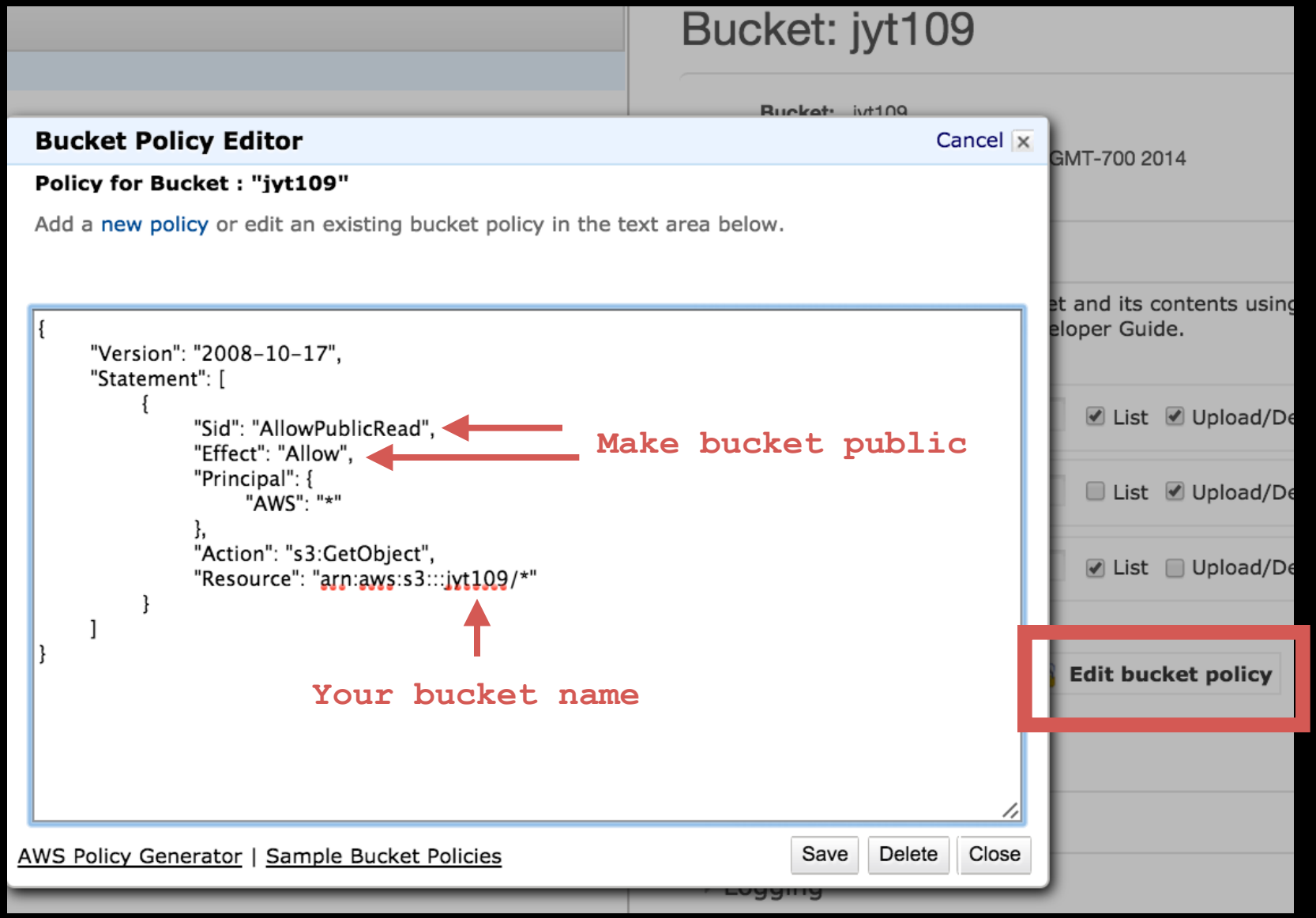
Grantee	List	Upload/Delete	View Permissions	Edit Permissions
jeffrey.tang09	☑	☑	☑	☑
Log Delivery	☐	☑	☑	☐
Everyone	☑	☐	☐	☐

To make the bucket public

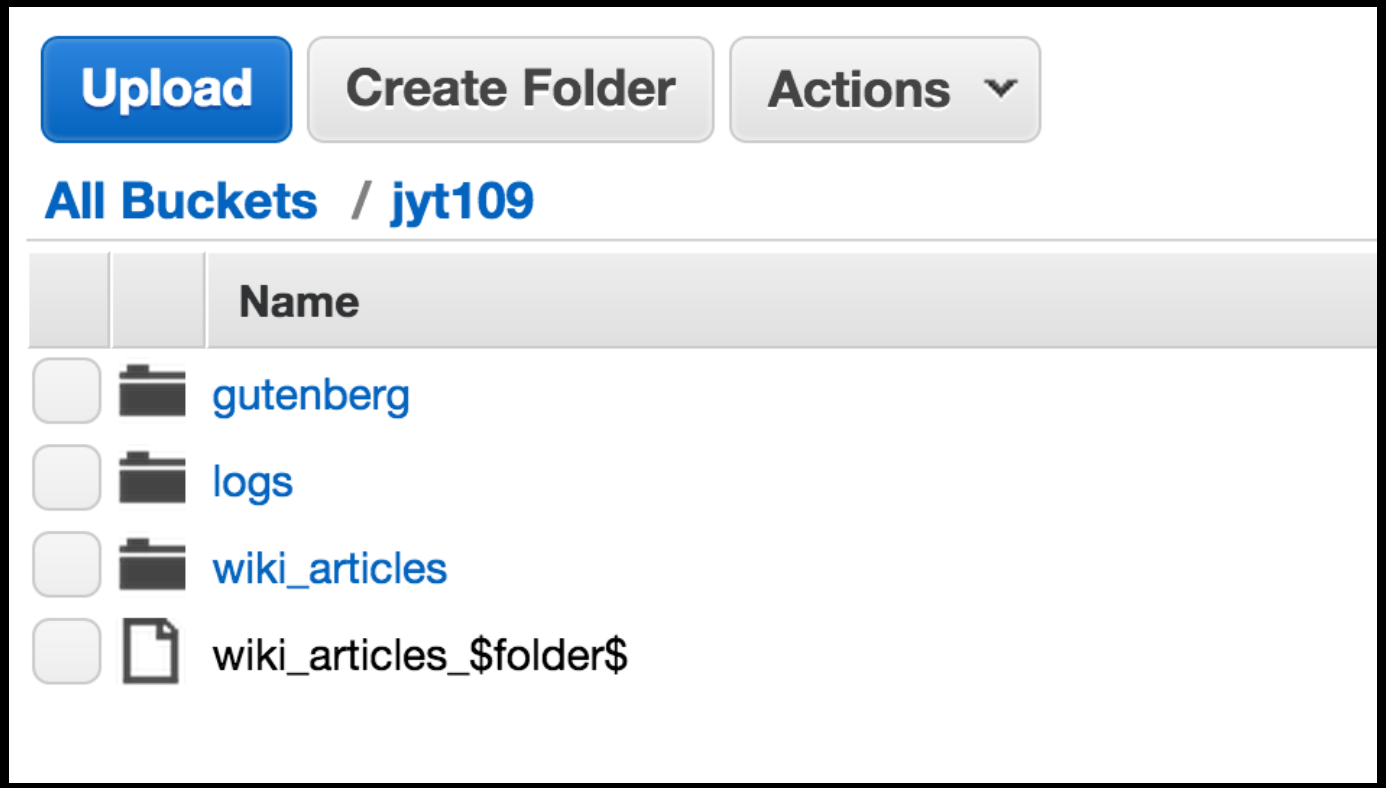
Add more permissions | Edit bucket policy | Add CORS Configuration

Save | Cancel

Bucket Policy



Upload Data


Upload
Create Folder
Actions
All Buckets / jyt109
Name
gutenberg
logs
wiki_articles
wiki_articles_$folder$

I need code to interact with S3 !

i.e. GUI sucks

Pandas S3

Pandas has an easy API to read from S3

```
import pandas as pd

data = pd.read_csv('https://s3.amazonaws.com/galvanizebucket/sample.txt',
                   header=None)
```

Or if you have very big files, use `chunksize`

(Chunk your file into specified pieces by line)

```
data_chunks = pd.read_csv(url, header=None, chunksize=6)
chunk_df = data_chunks.get_chunk()
```

But I also want to write and delete and manage buckets

i.e. pandas is good but not enough

Boto: AWS in Python

```
import boto

# Connect to S3
conn = boto.connect_s3(access_key, access_secret_key)

# List all the buckets
all_buckets = [b.name for b in conn.get_all_buckets()]
print all_buckets
```

http://boto.readthedocs.org/en/latest/s3_tut.html

Create a new bucket

```
# Check if bucket exist. If exist get bucket, else create one
bucket_name = 'galvanizebucket'

if conn.lookup(bucket_name) is None:
    b = conn.create_bucket(bucket_name, policy='public-read')
else:
    b = conn.get_bucket(bucket_name)
```

Make the bucket

Read / Write Files

```
# Write new file
file_object = b.new_key('sample.txt')
file_object.set_contents_from_string('HAHAHAH',
                                     policy='public-read')

# Read from file
print file_object.get_contents_as_string()
```

List / Delete Files

```
# Print all the files in the bucket
filenames = [f.name for f in b.list()]
print filenames

# Delete a file
a = b.new_key('somefilename')
a.delete()
```

Delete Bucket

```
# Delete Bucket
# Must delete all files in bucket
# Before deleting bucket
conn.delete_bucket('galvanizebucket')
```

Usefulness of Boto

- Can **read / write anything** to S3 from **anywhere**
- Either from EC2 or locally
- Almost unlimited storage (Cheap storage)

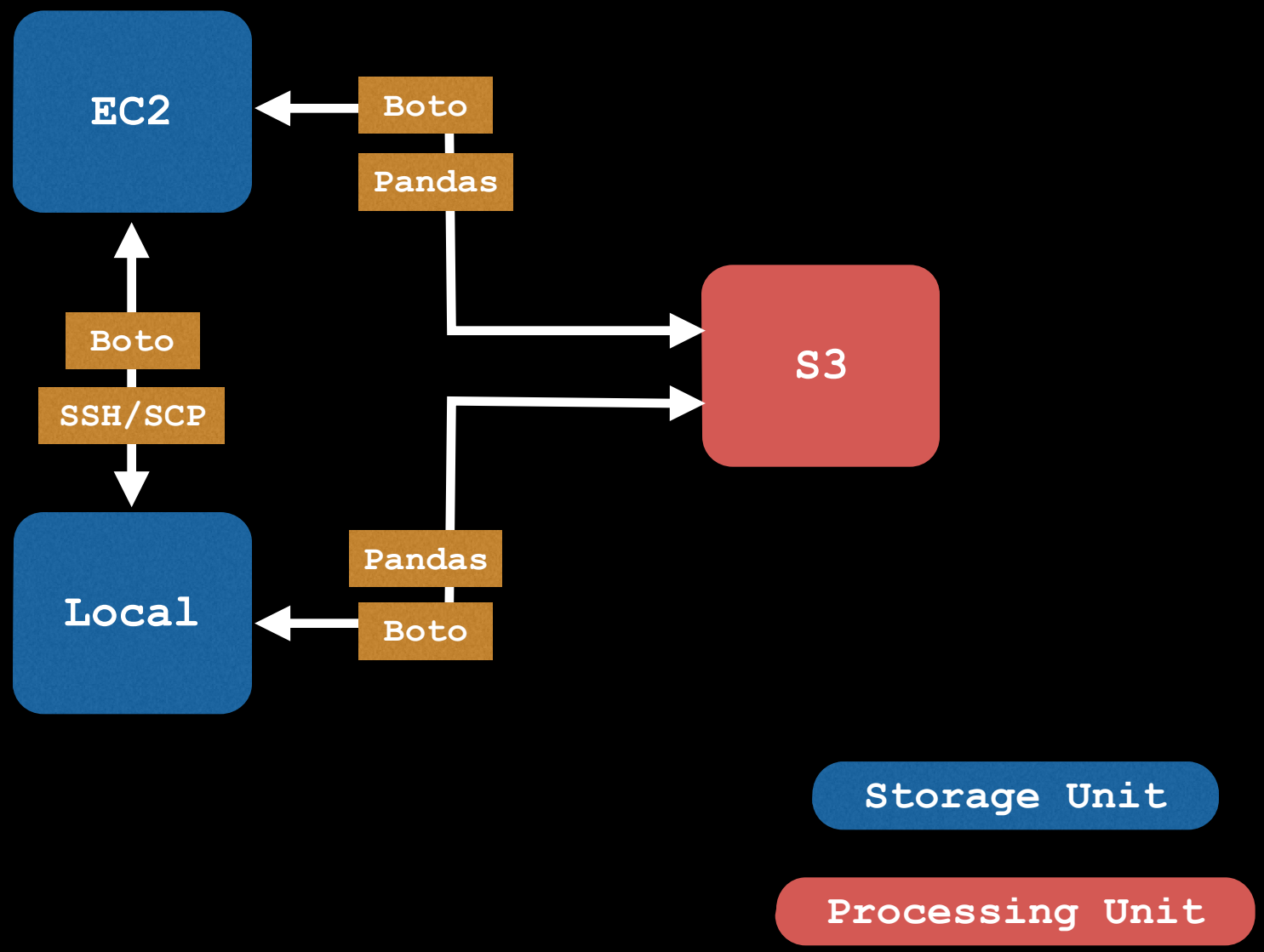
Boto EC2 ?

- Boto also support EC2 (Similar API with S3)
  - ★ http://boto.readthedocs.org/en/latest/ec2_tut.html
- More advanced automation with Fabric
  - ★ http://www.fabfile.org/

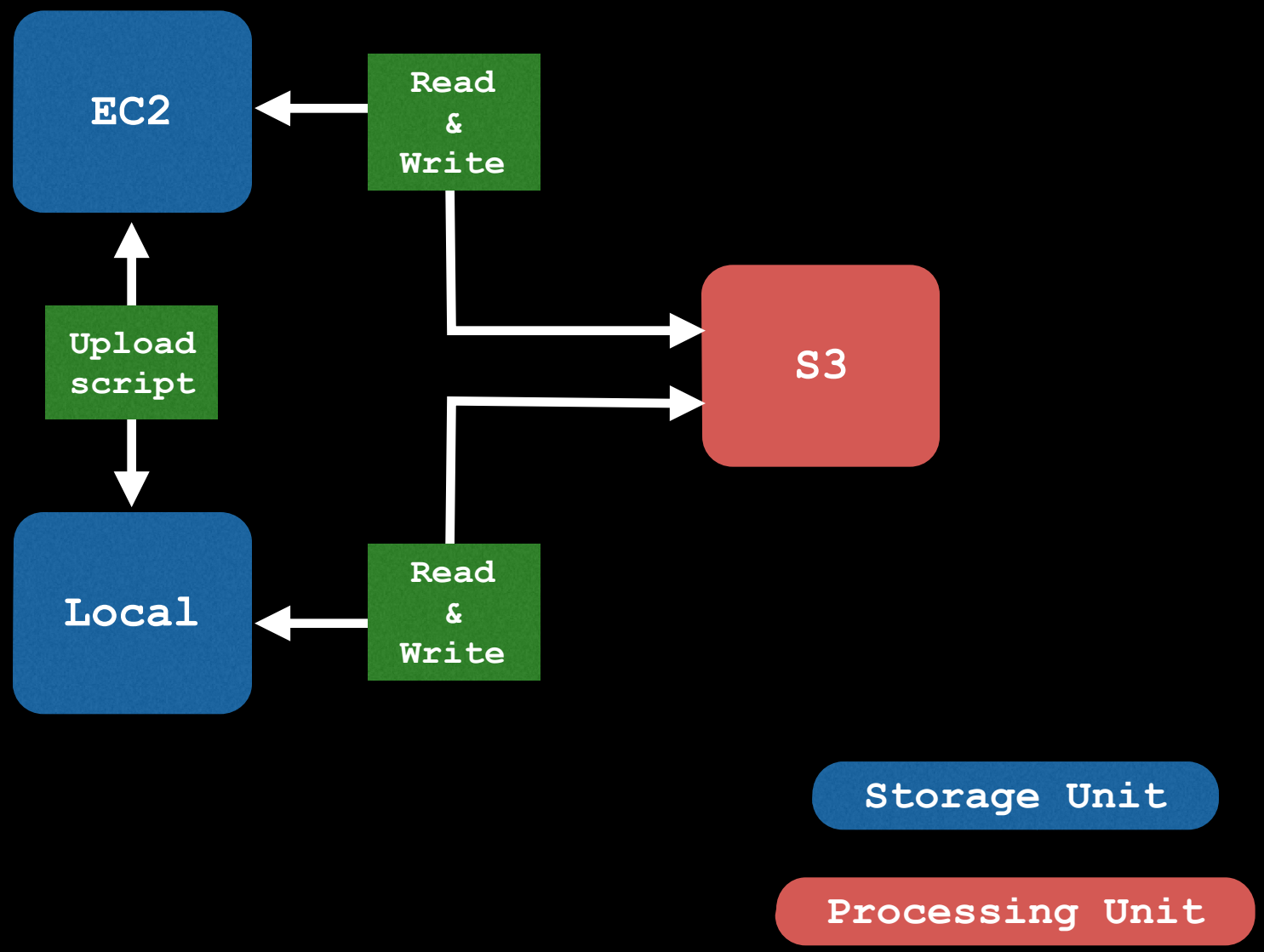
Goals

- Definition and Motivation
- Amazon Web Services (AWS)
  - ⋆ Elastic Compute Cloud (EC2)
  - ⋆ Simple Storage Service (S3)
- EC2 / S3 Overview and Work Flow

Tools Overview



Functions Overview



EC2 / S3 Usage

- When data is too big to fit locally
- Take a long time to run
- Have to run continuously (Hosting Web Apps)

Work Flow

1. Upload / Access data on S3
2. Use pandas / Boto to pull a subset down to local
3. Develop script locally
4. Upload script to EC2
5. Run script on EC2 on full set
6. Write results to S3

Summary

- Understand the difference between S3 and EC2
- Interact with S3 / EC2 with GUI / command line
- Interact with S3 / EC2 with Python
- Work flow with S3 / EC2 for heavy processing