Scalable ML and AWS Practical 2024

By Xianyuan Liu [Adapted from Michael Smith 2021-2023]

You will have been sent an email with a login link to the Amazon console.
Please **skip** the steps marked with **(SKIP)**, as they do not apply to our lab.

Logging in and the console

Once logged in, you will be placed in the AWS Console Home.

1. Locations

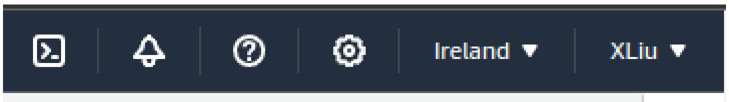
Please note that AWS is hosted in multiple locations worldwide.

Regions
Coming Soon

From Amazon:

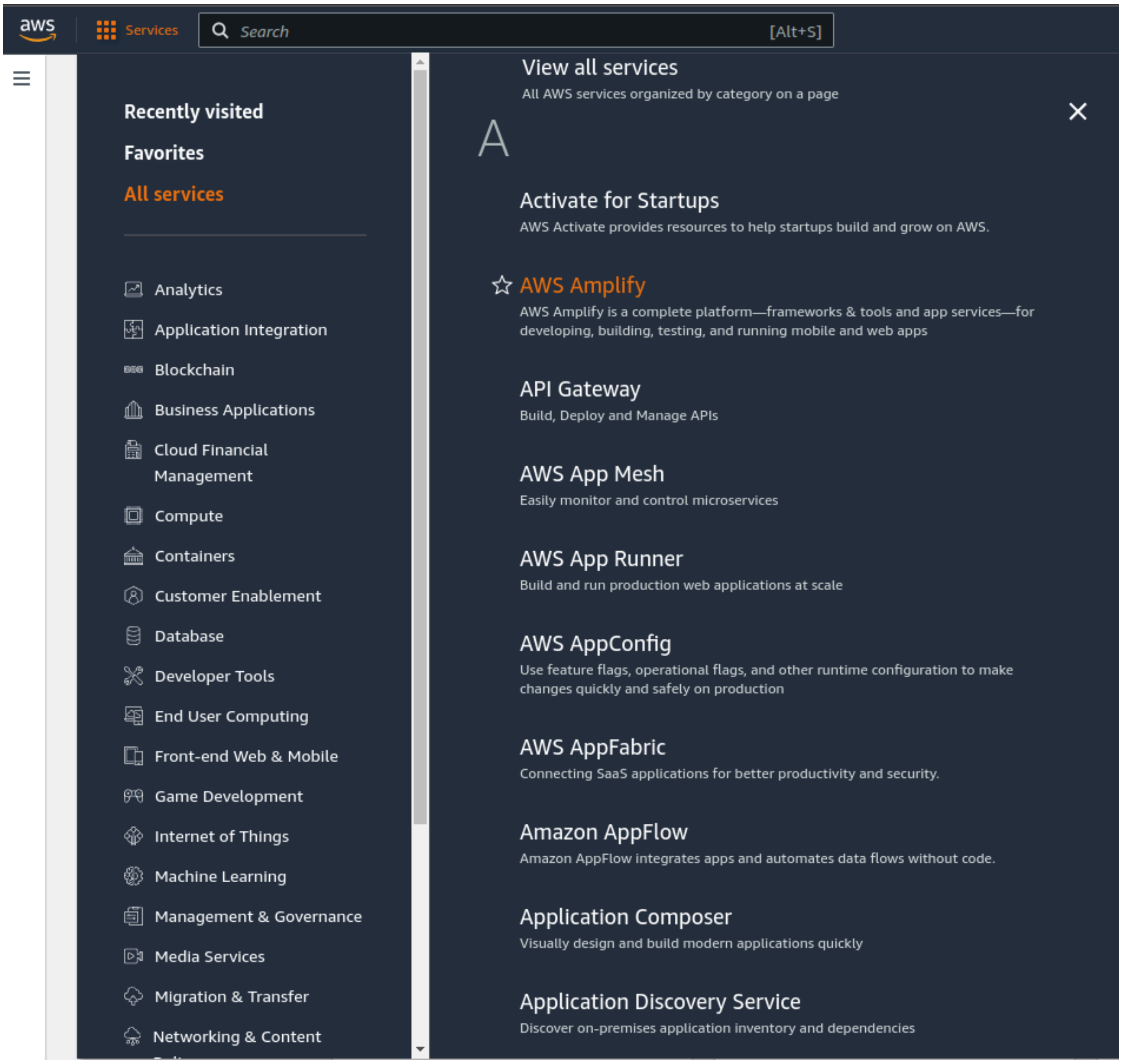
> *"These locations are composed of Regions and Availability Zones. Each Region is a separate geographic area. Each Region has multiple, isolated locations known as Availability Zones. Amazon EC2 provides you with the ability to place resources, such as instances, and data in multiple locations. Although rare, failures can occur that affect the availability of instances that are in the same location. If you host all your instances in a single location that is affected by such a failure, none of your instances would be available."*

We will use one region today, **Europe (Ireland) eu-west-1**. You can see (and select) the region from the drop-down list in the top right of the console:



2. Services

You can click “Services” in the top left of the console to see the list of services available…



Recently visited

Favorites

All services

- Analytics
- Application Integration
- Blockchain
- Business Applications
- Cloud Financial Management
- Compute
- Containers
- Customer Enablement
- Database
- Developer Tools
- End User Computing
- Front-end Web & Mobile
- Game Development
- Internet of Things
- Machine Learning
- Management & Governance
- Media Services
- Migration & Transfer
- Networking & Content

View all services
All AWS services organized by category on a page

A

Activate for Startups
AWS Activate provides resources to help startups build and grow on AWS.

☆ AWS Amplify
AWS Amplify is a complete platform—frameworks & tools and app services—for developing, building, testing, and running mobile and web apps

API Gateway
Build, Deploy and Manage APIs

AWS App Mesh
Easily monitor and control microservices

AWS App Runner
Build and run production web applications at scale

AWS AppConfig
Use feature flags, operational flags, and other runtime configuration to make changes quickly and safely on production

AWS AppFabric
Connecting SaaS applications for better productivity and security.

Amazon AppFlow
Amazon AppFlow integrates apps and automates data flows without code.

Application Composer
Visually design and build modern applications quickly

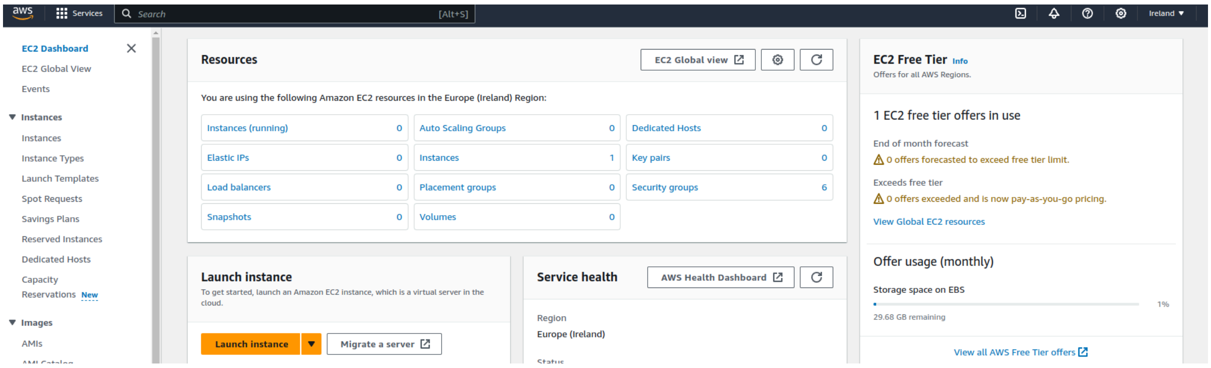
Application Discovery Service
Discover on-premises application inventory and dependencies

Under 'all services' one can select which tool of AWS one wishes to use. We will restrict ourselves for today to Elastic Compute Cloud (**EC2**), Simple Storage Service (**S3**), Elastic Map Reduce (**EMR**) and **Lambda**.

Elastic Compute Cloud (**EC2**)

First, we'll explore EC2, set up an 'instance' (virtual machine) and connect to it.

- Click on **EC2** in services (or search for it in the top search bar), opening the EC2 console:



Resources

You are using the following Amazon EC2 resources in the Europe (Ireland) Region:

Instances (running)	0	Auto Scaling Groups	0	Dedicated Hosts	0
Elastic IPs	0	Instances	1	Key pairs	0
Load balancers	0	Placement groups	0	Security groups	6
Snapshots	0	Volumes	0		

Launch instance

To get started, launch an Amazon EC2 instance, which is a virtual server in the cloud.

Service health

Region

Europe (Ireland)

Status

EC2 Free Tier

Offers for all AWS Regions.

1 EC2 free tier offers in use

End of month forecast

⚠ 0 offers forecasted to exceed free tier limit.

Exceeds free tier

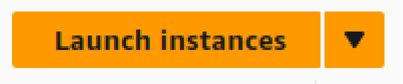
⚠ 0 offers exceeded and is now pay-as-you-go pricing.

Offer usage (monthly)

Storage space on EBS 1%

29.68 GB remaining

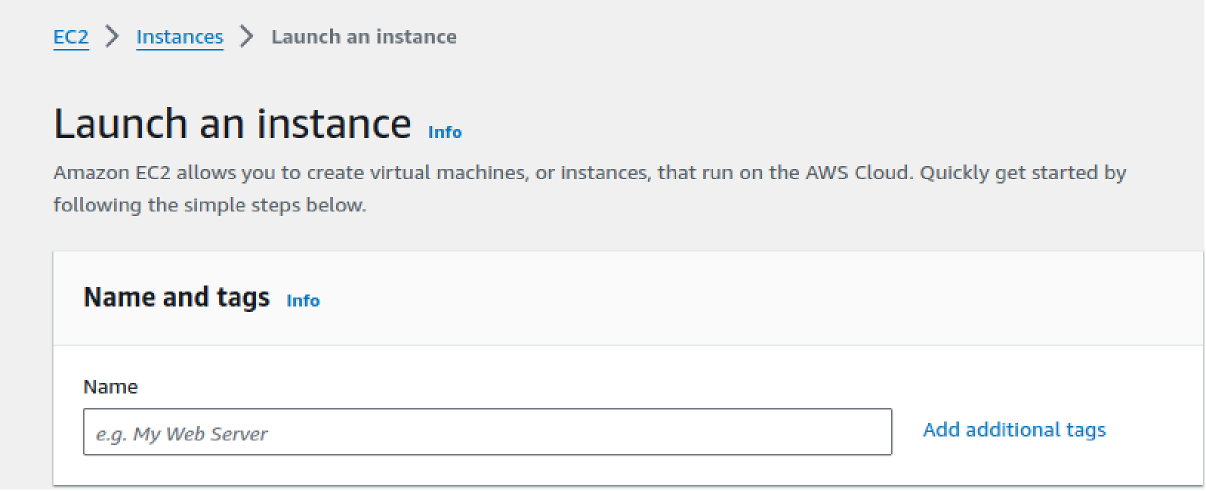
- Click on 'Instances' from the left pane. You will see an empty table and a big '**Launch Instance**' button - click it!



1. Launching instance

Step 1: Name instance

As everyone is using the same account, it is necessary to **label** your **instance**. It might be useful to enter a **unique** name by **adding your ID**, e.g. the first part of your email address ('xliu'). A good and unique name can help you find your instance easily.



Launch an instance

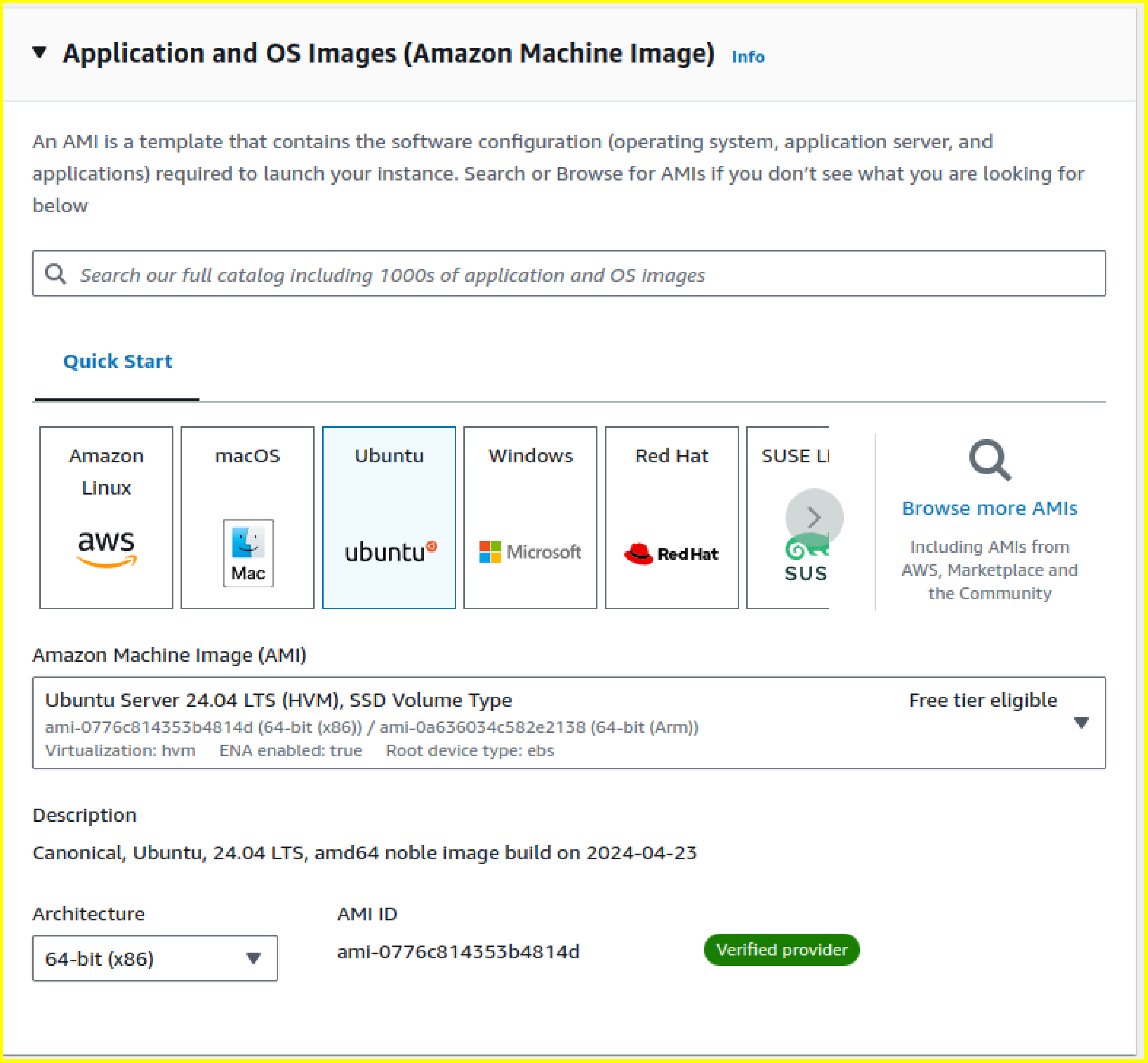
Amazon EC2 allows you to create virtual machines, or instances, that run on the AWS Cloud. Quickly get started by following the simple steps below.

Name and tags

Name: e.g. My Web Server

Step 2: Application and OS images

The second step is deciding which machine image to use for your new instance. For an easy start, consider using one of the images created by Amazon. **Select the Ubuntu Server 24.04 LTS (HVM) machine image.**



▼ Application and OS Images (Amazon Machine Image)

An AMI is a template that contains the software configuration (operating system, application server, and applications) required to launch your instance. Search or Browse for AMIs if you don't see what you are looking for below

Search our full catalog including 1000s of application and OS images

Amazon Machine Image (AMI)

Ubuntu Server 24.04 LTS (HVM), SSD Volume Type — Free tier eligible
ami-0776c814353b4814d (64-bit (x86)) / ami-0a636034c582e2138 (64-bit (Arm))
Virtualization: hvm ENA enabled: true Root device type: ebs

Description

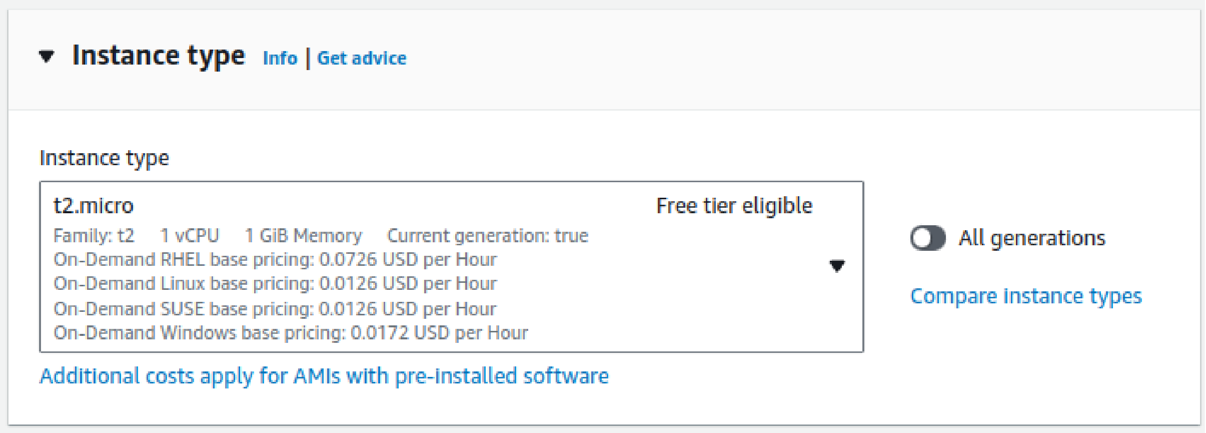
Canonical, Ubuntu, 24.04 LTS, amd64 noble image build on 2024-04-23

Architecture: 64-bit (x86)

AMI ID: ami-0776c814353b4814d

Step 3: Choose the Instance type

The third step is selecting the (virtual) hardware the instance will run on. For this practical, we ask that you **use t2.micro (preferred type), t2.nano, or t2.small instance type** (as we have increased the limit on this account to allow you all to start one). (The current limit is set to 512 vCPUs can be instantiated by the class).



▾ Instance type Info | Get advice

Instance type

t2.micro — Free tier eligible
Family: t2 1 vCPU 1 GiB Memory Current generation: true
On-Demand RHEL base pricing: 0.0726 USD per Hour
On-Demand Linux base pricing: 0.0126 USD per Hour
On-Demand SUSE base pricing: 0.0126 USD per Hour
On-Demand Windows base pricing: 0.0172 USD per Hour

Additional costs apply for AMIs with pre-installed software

Step 4: Key pair

A quick tour, to explain key pairs and ssh… (quoting from ssh.com):

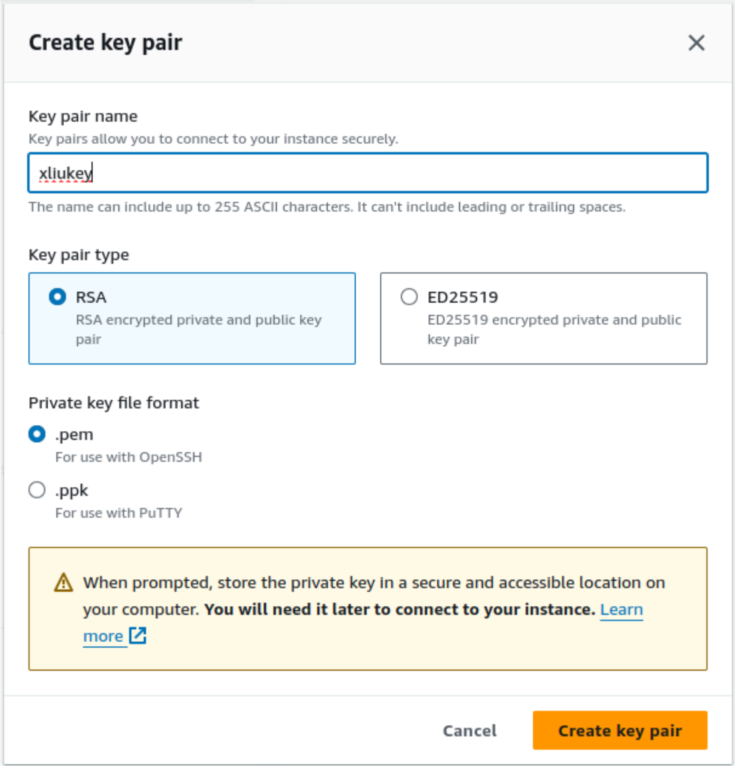
> *"Each SSH key pair includes two keys:*
>
> *A **public key** that is copied to the SSH server(s). Anyone with a copy of the public key can encrypt data which can then only be read by the person who holds the corresponding private key. Once an SSH server receives a public key from a user and considers the key trustworthy, the server marks the key as authorized in its authorized_keys file. Such keys are called authorized keys.*
>
> *A **private key** that remains (only) with the user. The possession of this key is proof of the user's identity. Only a user in possession of a private key that corresponds to the public key on the server will be able to authenticate successfully. The private keys need to be stored and handled carefully, and no copies of the private key should be distributed. The private keys used for user authentication are called identity keys."*

AWS uses a key pair because it is typically more secure, allows for automation, and is generally the standard method for secure communication.

- Click '**create a new key pair**' and enter a ****unique** key pair name**, e.g. your instance name + key ('xliukey'). [Use the defaults: **RSA**]

  Please note: in **Private key file format**, **Linux** users should choose .**pem** file, while **Windows** users should choose a .**ppk** file for use with PuTTY in EMR (The following example uses .pem. Windows users should replace .pem with .ppk).

- Click '**Create key pair**'. You'll receive a file called 'xliukey.pem' (or 'xliukey.ppk' or whatever). You'll need this to SSH into this new instance. Keep it safe and secret.



Create key pair

Key pair name

Key pairs allow you to connect to your instance securely.

xliukey

The name can include up to 255 ASCII characters. It can't include leading or trailing spaces.

Key pair type

● RSA
RSA encrypted private and public key pair

○ ED25519
ED25519 encrypted private and public key pair

Private key file format

● .pem
For use with OpenSSH

○ .ppk
For use with PuTTY

⚠ When prompted, store the private key in a secure and accessible location on your computer. **You will need it later to connect to your instance.** Learn more

*Step 5: Other options (**SKIP**)*

There is no need to modify any of this. But a couple of items of interest:

- *Allow SSH traffic from anywhere.*
- *Security groups are how EC2 organises access to the instances you create. I've already created one called '**justssh**' which gives access to the SSH port from anywhere. Typically one would restrict this to be from just your IP address, for example. Feel free to either use a security group that already exists or create a new one. You'll need to be able to SSH into the server later.*

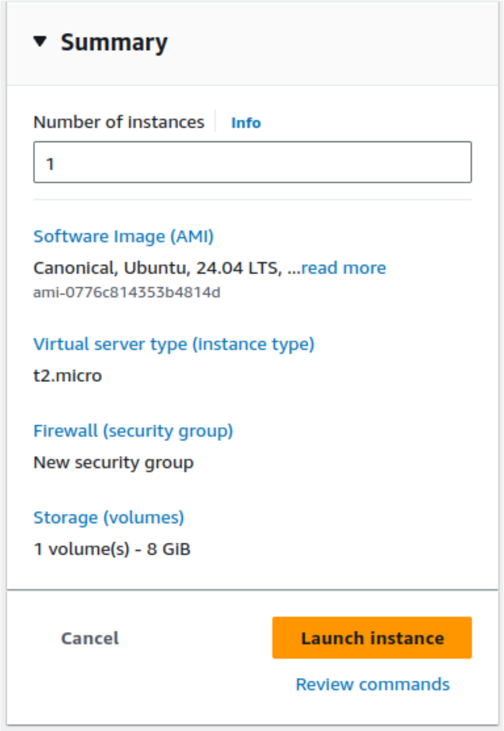
***Spot pricing** Typically AWS will not be using all its computational resources. To make use of this 'spare' hardware, AWS offer a service called 'spot pricing' which is typically considerably cheaper than the on-demand price but comes at the cost of an instance that might be terminated with two minute warning.*

***Storage** Simply leave it an 8GB general-purpose SSD.*
Please note:
There are many types of storage provided by AWS:

- *Low cost, slow access: Amazon Glacier*
- *Elastic Block Store: This is the type of storage you need in your EC2 instance. (More info) This comes in four flavours,*
    - *slowest/cheapest: sc1 (cold HDD, solid state)*
    - *still cheap: st1 (throughput-optimised HDD)*
    - *solid-state: gp2,gp3 (general purpose SSD)*
    - *fast/expensive: io1,io2.*

Step 6: Launch

Finally, click ‘**Launch instance**’ on the right-hand side:



Number of instances Info

1

Software Image (AMI)

Canonical, Ubuntu, 24.04 LTS, ...read more

ami-0776c814353b4814d

Virtual server type (instance type)

t2.micro

Firewall (security group)

New security group

Storage (volumes)

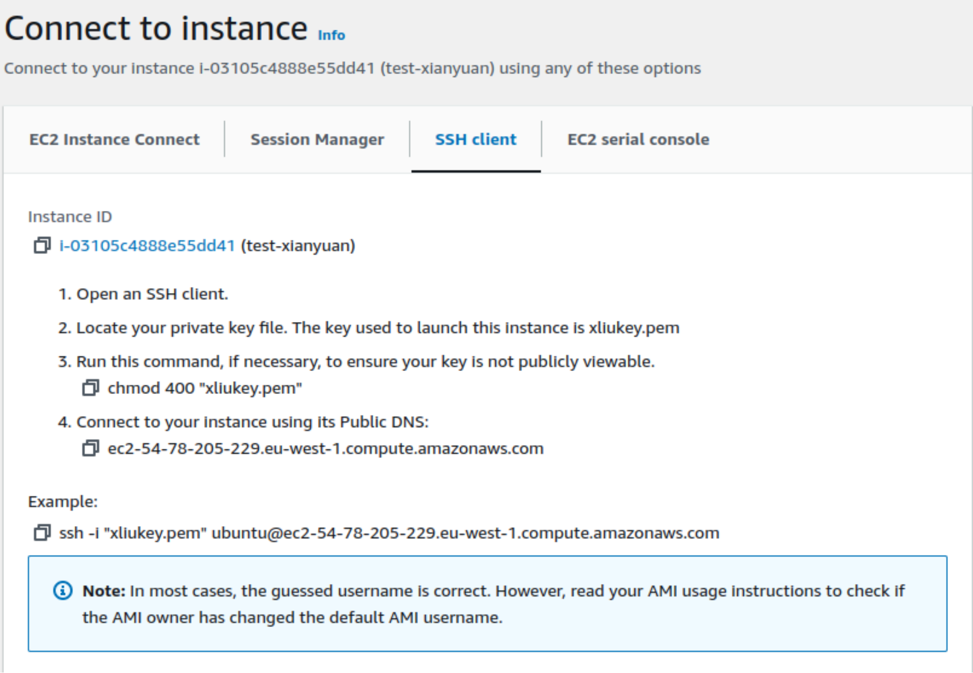
1 volume(s) - 8 GiB

2. SSHing into your new instance

You'll be shown a summary stating your instances are launching. Either **click** the link to the instance or return to the list of instances and filter by the tag your username or email address you entered.

You might need to wait a few seconds while the instance starts... Also, you might need to click on the refresh button:

Click on the instance and then right-click (or use the button at the top) select ‘Connect’, then click ‘SSH client’. This will give instructions on how to SSH in. The example is as below. Please locate your .pem/.ppk file.



Connect to instance Info

Connect to your instance i-03105c4888e55dd41 (test-xianyuan) using any of these options

Instance ID

i-03105c4888e55dd41 (test-xianyuan)

1. Open an SSH client.
2. Locate your private key file. The key used to launch this instance is xliukey.pem
3. Run this command, if necessary, to ensure your key is not publicly viewable.
   chmod 400 "xliukey.pem"
4. Connect to your instance using its Public DNS:
   ec2-54-78-205-229.eu-west-1.compute.amazonaws.com

Example:

```
ssh -i "xliukey.pem" ubuntu@ec2-54-78-205-229.eu-west-1.compute.amazonaws.com
```

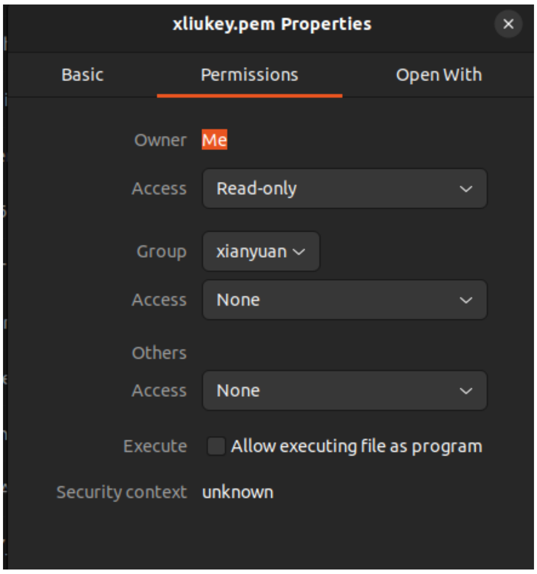
Note: In most cases, the guessed username is correct. However, read your AMI usage instructions to check if the AMI owner has changed the default AMI username.

For Linux users...

In Linux and iOS, the key file's permissions should be set to read-only using the command line:

chmod 400 filename.pem

or the file browser (right-click the key file, and choose properties->permissions):



xliukey.pem Properties

Owner Me

Access Read-only

Group xianyuan

Access None

Others

Access None

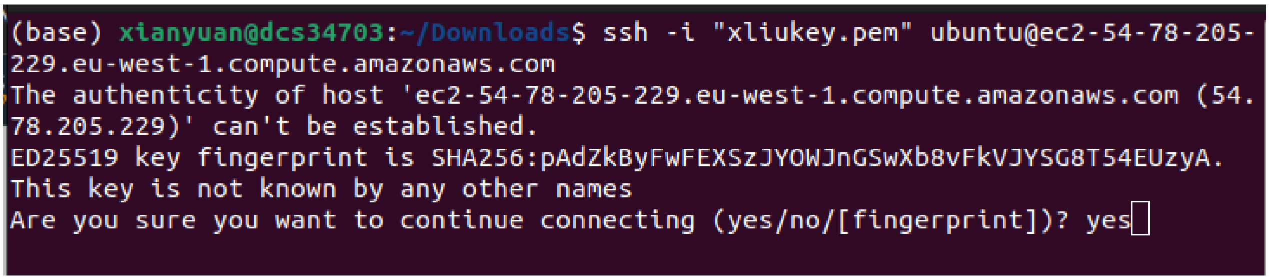
Execute □ Allow executing file as program

Security context unknown

then ssh:

ssh -i "filename.pem" ubuntu@ec2-54-78-205-229.eu-west-1.compute.amazonaws.com

It will ask if you’re sure (as it can’t guarantee that there’s not a man-in-the-middle-attack):



```
(base) xianyuan@dcs34703:~/Downloads$ ssh -i "xliukey.pem" ubuntu@ec2-54-78-205-229.eu-west-1.compute.amazonaws.com
The authenticity of host 'ec2-54-78-205-229.eu-west-1.compute.amazonaws.com (54.78.205.229)' can't be established.
ED25519 key fingerprint is SHA256:pAdZkByFwFEXSzJYOWJnGSwXb8vFkVJYSG8T54EUzyA.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```

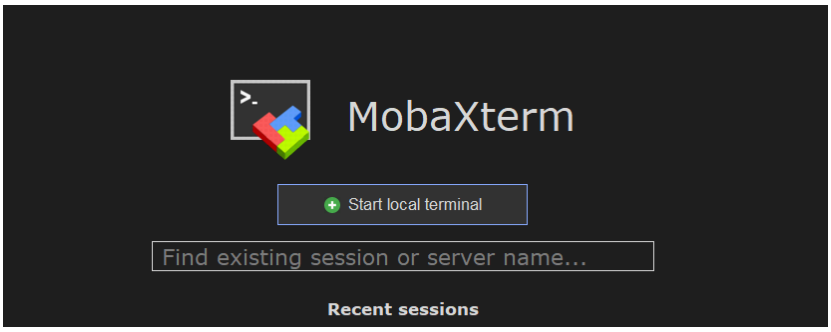
Tip: One usually keeps the .pem files in the .ssh folder in one’s home directory.

For Windows users...

In Windows, things are more complicated. We recommend using MobaXterm (AWS has instructions on how to get this working with putty).

Option 1

- Click ‘**Start local terminal**’ after you open MobaXterm.

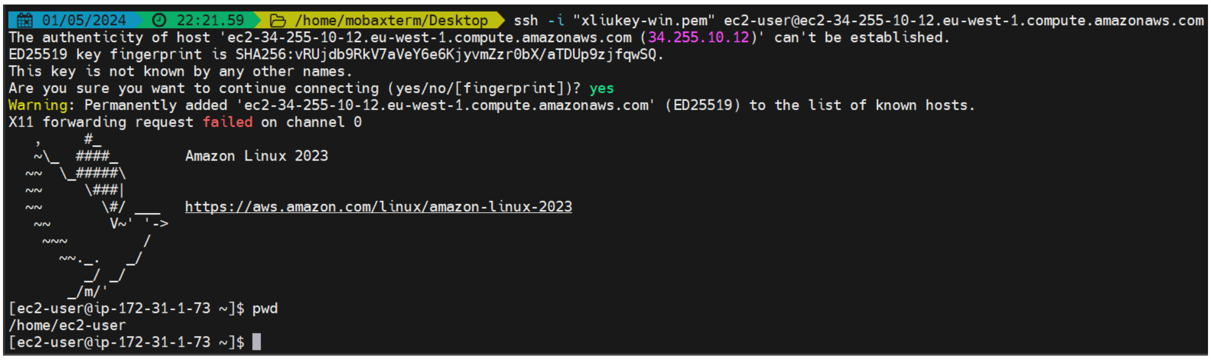


- Follow the guidance of ‘For Linux users’ step by step. The commands are
  1) changing permissions by

  chmod 400 filename.ppk

  2) then sshing:

  ssh -i "filename.ppk" ubuntu@ec2-54-78-205-229.eu-west-1.compute.amazonaws.com



```
 01/05/2024  22:21.59  /home/mobaxterm/Desktop  ssh -i "xliukey-win.pem" ec2-user@ec2-34-255-10-12.eu-west-1.compute.amazonaws.com
The authenticity of host 'ec2-34-255-10-12.eu-west-1.compute.amazonaws.com (34.255.10.12)' can't be established.
ED25519 key fingerprint is SHA256:vRUjdb9RkV7aVeY6e6KjyvmZzr0bX/aTDUp9zjfqwSQ.
This key is not known by any other names.
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
Warning: Permanently added 'ec2-34-255-10-12.eu-west-1.compute.amazonaws.com' (ED25519) to the list of known hosts.
X11 forwarding request failed on channel 0
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
[ec2-user@ip-172-31-1-73 ~]$ pwd
/home/ec2-user
[ec2-user@ip-172-31-1-73 ~]$ 
```

Option 2

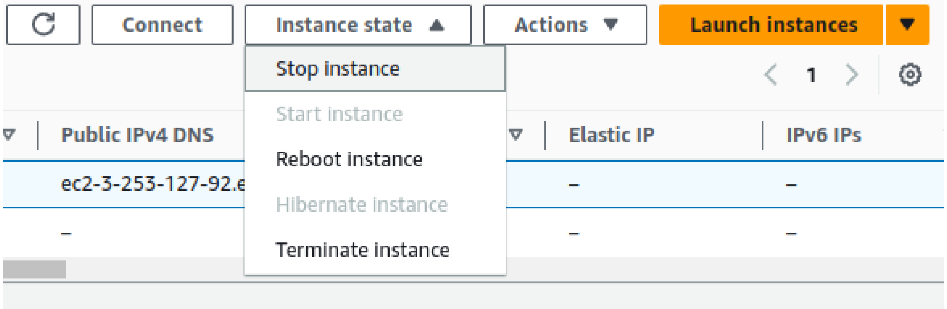
Follow these instructions to set up a new session for reuse.

If you can't get that working, please try alternative SSH clients. *(We have not enabled IAM access to “EC2 instance connect” otherwise anyone could access any other EC2 instance).*

Also, this slightly awkward process is a step that hopefully shows how the key approach to security works etc.

3. Disconnect and stop

- Enter '**exit**' in the terminal to log out of the instance
- Click '**Instance state**' and '**Stop instance**' to power off the instance (Please double-check the instance name to make sure you are stopping your instance)



Elastic Map Reduce (EMR)

We'll set up a cluster to run a simple Recommender system (a little like the one you set up earlier in the course). Here we're not interested in the algorithm, more in some of the aspects of the cloud.

We're using Elastic Map Reduce.

> "The node types in Amazon EMR are as follows:
>
> - Primary node: A node that manages the cluster by running software components to coordinate the distribution of data and tasks among other nodes for processing. The primary node tracks the status of tasks and monitors the health of the cluster. Every cluster has a primary node, and it's possible to create a single-node cluster with only the primary node.
> - Core node: A node with software components that run tasks and store data in the Hadoop Distributed File System (HDFS) on your cluster. Multi-node clusters have at least one core node.
> - Task node: A node with software components that only runs tasks and does not store data in HDFS. **Task nodes are optional.**"
>
> – AWS documentation.

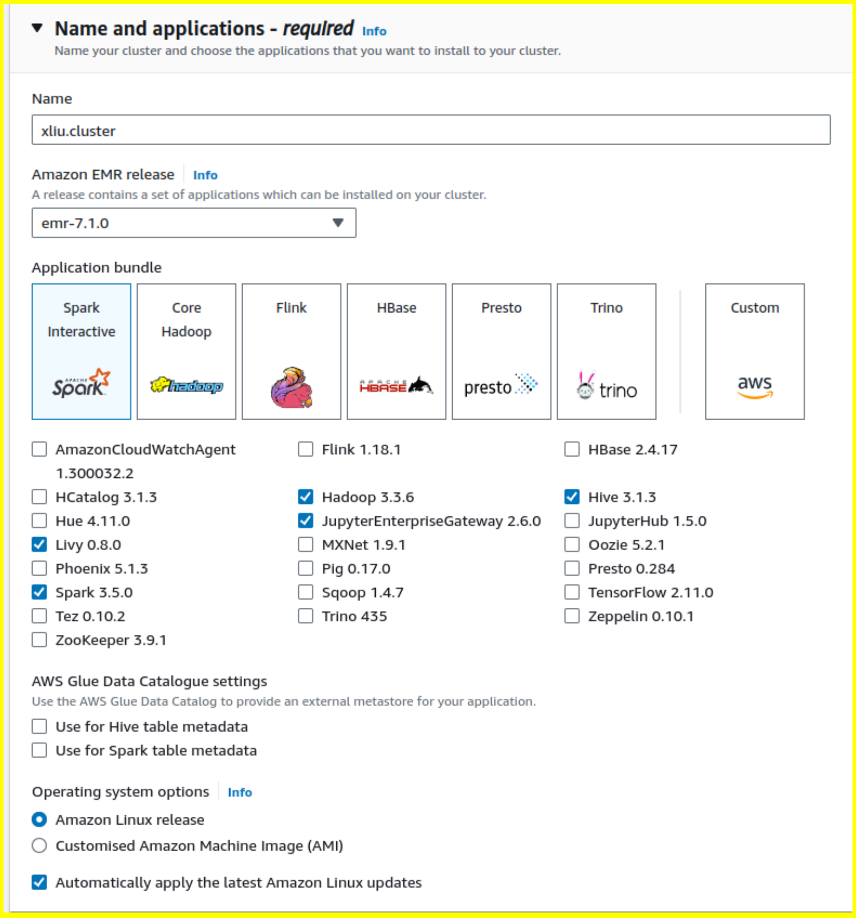
1. Creating cluster

We will make a cluster with one primary node and about three core nodes.

- Click on **Services**, and find **EMR** (under analytics), then click **Create cluster**.

Step 1: Name and applications

Enter a unique name (that won't be confused with someone else's cluster), e.g. the first part of your email address, e.g. 'xliu.cluster'. **Select the 'Spark' application bundle.**



▼ **Name and applications - *required*** Info

Name your cluster and choose the applications that you want to install to your cluster.

Name

xliu.cluster

Amazon EMR release Info

A release contains a set of applications which can be installed on your cluster.

emr-7.1.0 ▼

Application bundle

□ AmazonCloudWatchAgent 1.300032.2
□ HCatalog 3.1.3
□ Hue 4.11.0
☑ Livy 0.8.0
□ Phoenix 5.1.3
☑ Spark 3.5.0
□ Tez 0.10.2
□ ZooKeeper 3.9.1
□ Flink 1.18.1
☑ Hadoop 3.3.6
☑ JupyterEnterpriseGateway 2.6.0
□ MXNet 1.9.1
□ Pig 0.17.0
□ Sqoop 1.4.7
□ Trino 435
□ HBase 2.4.17
☑ Hive 3.1.3
□ JupyterHub 1.5.0
□ Oozie 5.2.1
□ Presto 0.284
□ TensorFlow 2.11.0
□ Zeppelin 0.10.1

AWS Glue Data Catalogue settings

Use the AWS Glue Data Catalog to provide an external metastore for your application.

□ Use for Hive table metadata
□ Use for Spark table metadata

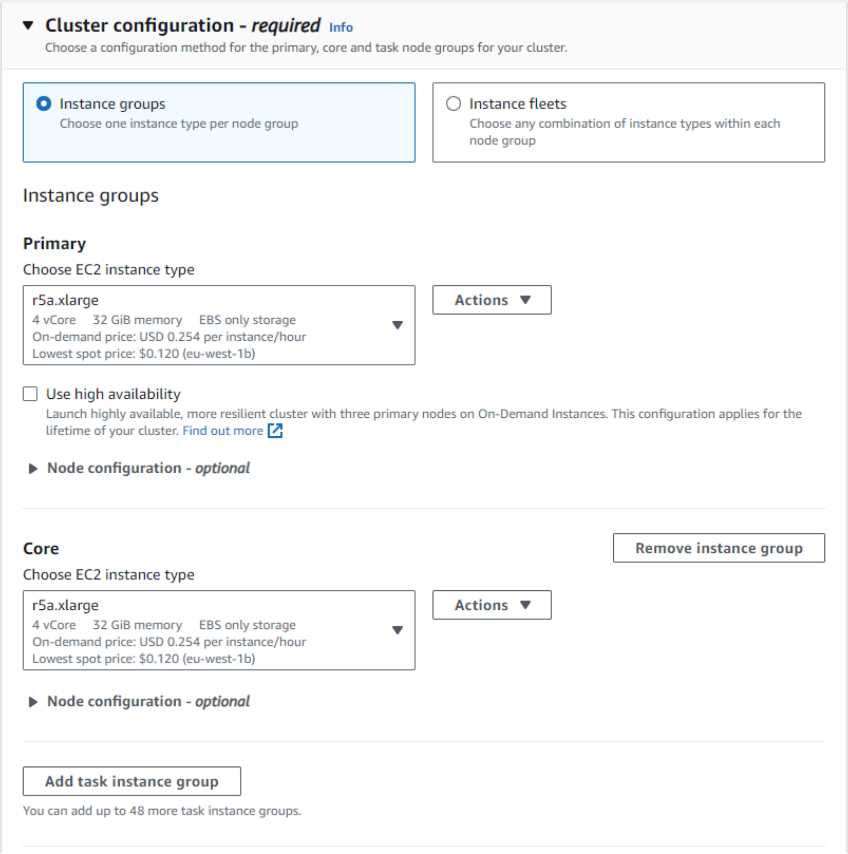
Operating system options Info

● Amazon Linux release
○ Customised Amazon Machine Image (AMI)

☑ Automatically apply the latest Amazon Linux updates

Step 2: Cluster configuration

- Edit **BOTH** the **Primary** and **Core** to use the “**r5a.xlarge**” instance type (we've increased the AWS limits on this type. You'll find that using other instance types will probably fail).
- **Remove** the “**Task 1 of 1**” instance group (we don’t need Task nodes).



▼ **Cluster configuration - *required***

Choose a configuration method for the primary, core and task node groups for your cluster.

● Instance groups
Choose one instance type per node group

○ Instance fleets
Choose any combination of instance types within each node group

Instance groups

Primary

Choose EC2 instance type

r5a.xlarge
4 vCore 32 GiB memory EBS only storage
On-demand price: USD 0.254 per instance/hour
Lowest spot price: $0.120 (eu-west-1b)

□ Use high availability
Launch highly available, more resilient cluster with three primary nodes on On-Demand Instances. This configuration applies for the lifetime of your cluster. Find out more

▶ **Node configuration - *optional***

Core

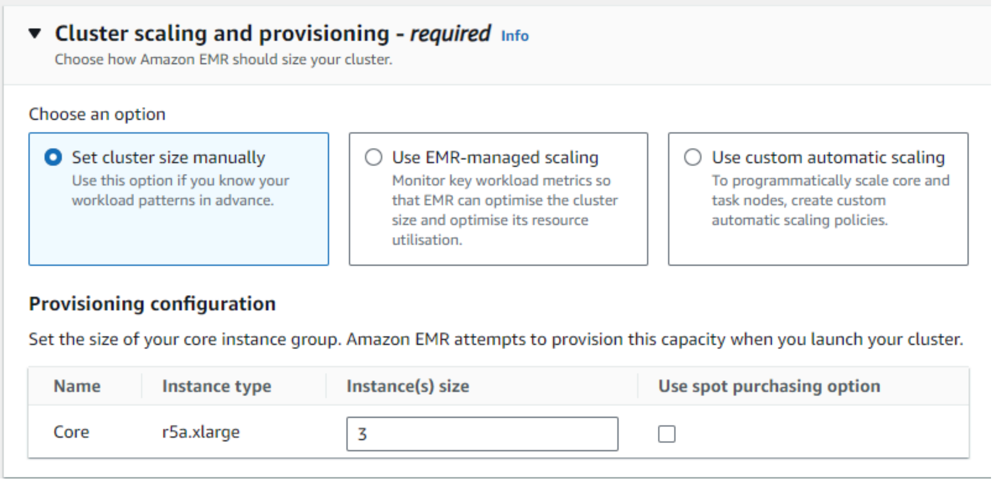
Choose EC2 instance type

r5a.xlarge
4 vCore 32 GiB memory EBS only storage
On-demand price: USD 0.254 per instance/hour
Lowest spot price: $0.120 (eu-west-1b)

▶ **Node configuration - *optional***

You can add up to 48 more task instance groups.

- Set the Core group cluster size to 2 (or 3).



▼ **Cluster scaling and provisioning - *required***

Choose how Amazon EMR should size your cluster.

Choose an option

● Set cluster size manually
Use this option if you know your workload patterns in advance.

○ Use EMR-managed scaling
Monitor key workload metrics so that EMR can optimise the cluster size and optimise its resource utilisation.

○ Use custom automatic scaling
To programmatically scale core and task nodes, create custom automatic scaling policies.

Provisioning configuration

Set the size of your core instance group. Amazon EMR attempts to provision this capacity when you launch your cluster.

Name	Instance type	Instance(s) size	Use spot purchasing option
Core	r5a.xlarge	3	□

You can save us money if you select ‘Use spot purchasing option’, but there is a really small risk your instances will get terminated by AWS during a spike in demand.

- Leave the rest of the page as its default: Don’t enable cluster scaling. Auto-termination can be left on. EBS Root volume can be left at 15Gb.

- Tags: add the same name you gave before, i.e. xliu.cluster.

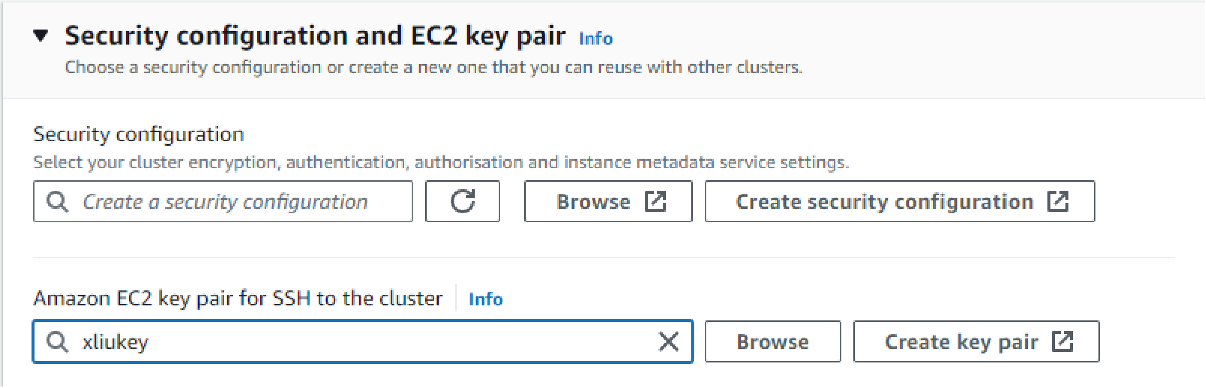
Step 2: Bootstrap actions

Bootstrap actions can help us to install software or customise your instance configuration. We have created and added the install-numpy.sh before the lab, so you can set the right directory to load it:

- Choose **Bootstrap actions**
- Choose **Add**
- Enter
  InstallNumpy for Name and
  s3://install-numpy/install-numpy.sh for Script location
- Choose **Add bootstrap action**

Step 4: Security configuration and EC2 key pair

IMPORTANT: In this section you need to select the EC2 keypair you created previously. (If you don't do this you won't be able to connect to the cluster!). Please make sure you choose the correct one!



▼ Security configuration and EC2 key pair Info

Choose a security configuration or create a new one that you can reuse with other clusters.

Security configuration

Select your cluster encryption, authentication, authorisation and instance metadata service settings.

Create a security configuration
Browse
Create security configuration

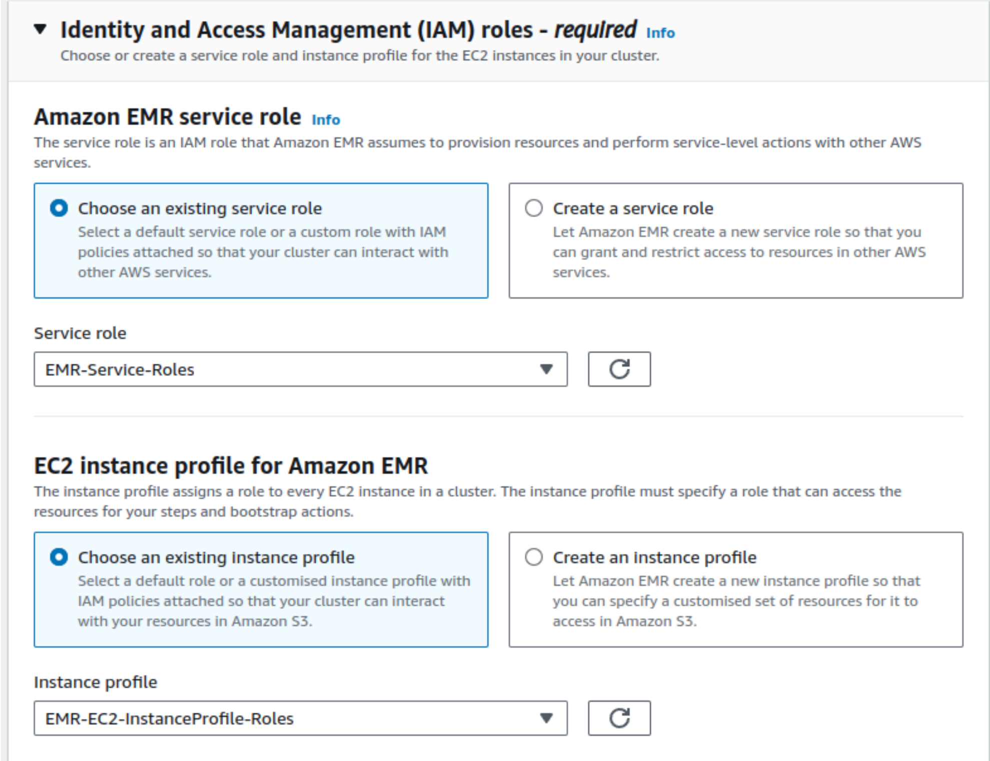
Amazon EC2 key pair for SSH to the cluster Info

xliukey

Step 5: Identity and Access Management (IAM) roles

The cluster and its constituent instances need to be able to do things in your AWS environment. We use roles to control what they can or can't do. For our use case,

- select the **EMR-Sevice-Roles** for the Amazon EMR service role, and
- **EMR-EC2-InstanceProfile-Roles** for the EC2 instance profile for Amazon EMR:



▼ **Identity and Access Management (IAM) roles - *required*** Info

Choose or create a service role and instance profile for the EC2 instances in your cluster.

Amazon EMR service role Info

The service role is an IAM role that Amazon EMR assumes to provision resources and perform service-level actions with other AWS services.

● Choose an existing service role
Select a default service role or a custom role with IAM policies attached so that your cluster can interact with other AWS services.

○ Create a service role
Let Amazon EMR create a new service role so that you can grant and restrict access to resources in other AWS services.

Service role: EMR-Service-Roles

EC2 instance profile for Amazon EMR

The instance profile assigns a role to every EC2 instance in a cluster. The instance profile must specify a role that can access the resources for your steps and bootstrap actions.

● Choose an existing instance profile
Select a default role or a customised instance profile with IAM policies attached so that your cluster can interact with your resources in Amazon S3.

○ Create an instance profile
Let Amazon EMR create a new instance profile so that you can specify a customised set of resources for it to access in Amazon S3.

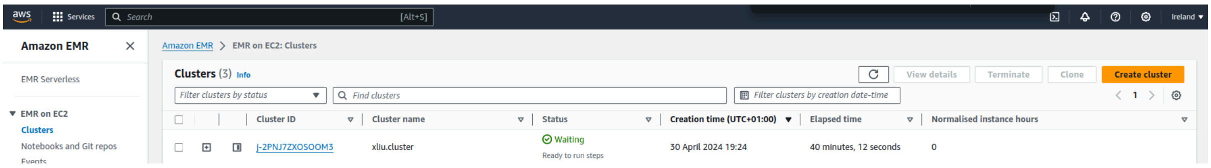
Instance profile: EMR-EC2-InstanceProfile-Roles

Step 6: Launch

- Finally, click “**Create cluster**” and wait.

2. Connecting

You will be able to select the cluster from the EMR Clusters page:



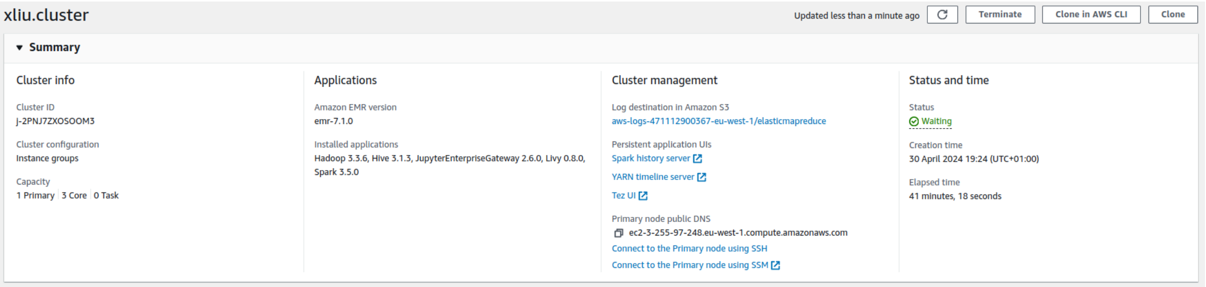
Amazon EMR > EMR on EC2: Clusters

Clusters (3) Info

Filter clusters by status | Find clusters | Filter clusters by creation date-time

Cluster ID	Cluster name	Status	Creation time (UTC+01:00)	Elapsed time	Normalised instance hours
j-2PNJ7ZXOSOOM3	xliu.cluster	Waiting Ready to run steps	30 April 2024 19:24	40 minutes, 12 seconds	0

- **Click** your cluster’s ID (or select it and click “view details”) to check the details. Please be patient, it takes a while to get the cluster spun up. Initially, SSH won't be available, and even when it is PySpark etc might not be installed when you first arrive - wait a minute and click the refresh button again.



xliu.cluster

▼ Summary

Cluster info
Cluster ID: j-2PNJ7ZXOSOOM3
Cluster configuration: Instance groups
Capacity: 1 Primary | 3 Core | 0 Task

Applications
Amazon EMR version: emr-7.1.0
Installed applications: Hadoop 3.3.6, Hive 3.1.3, JupyterEnterpriseGateway 2.6.0, Livy 0.8.0, Spark 3.5.0

Cluster management
Log destination in Amazon S3: aws-logs-471112900367-eu-west-1/elasticmapreduce
Persistent application UIs: Spark history server, YARN timeline server, Tez UI
Primary node public DNS: ec2-3-255-97-248.eu-west-1.compute.amazonaws.com
Connect to the Primary node using SSH
Connect to the Primary node using SSM

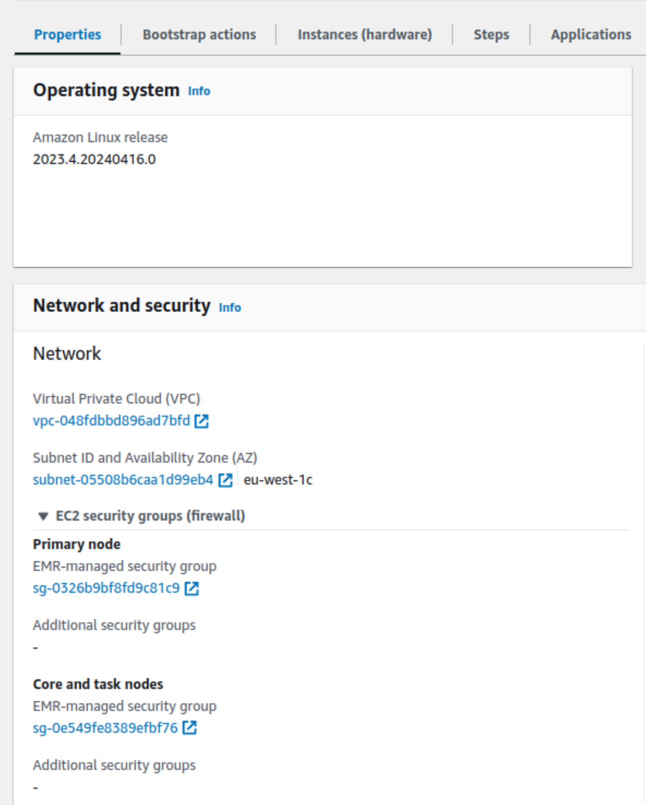
Status and time
Status: Waiting
Creation time: 30 April 2024 19:24 (UTC+01:00)
Elapsed time: 41 minutes, 18 seconds

- Click ‘**Connect to the Primary node using SSH**’. You can find guidance on a connection using SSH on different systems, which you can follow accordingly.

3. Authorise inbound traffic

Authorising for primary node and core and task nodes. The guidance refers to https://docs.aws.amazon.com/emr/latest/ManagementGuide/emr-connect-ssh-prereqs.html

- The **Properties** tab on this page will be pre-selected.
- Under ‘**Network and security**’ in the Properties tab, select the **arrow next to EC2 security groups (firewall)** to expand this section



Operating system

Amazon Linux release
2023.4.20240416.0

Network and security

Network

Virtual Private Cloud (VPC)
vpc-048fdbbd896ad7bfd

Subnet ID and Availability Zone (AZ)
subnet-05508b6caa1d99eb4 eu-west-1c

▼ EC2 security groups (firewall)

Primary node
EMR-managed security group
sg-0326b9bf8fd9c81c9

Additional security groups
-

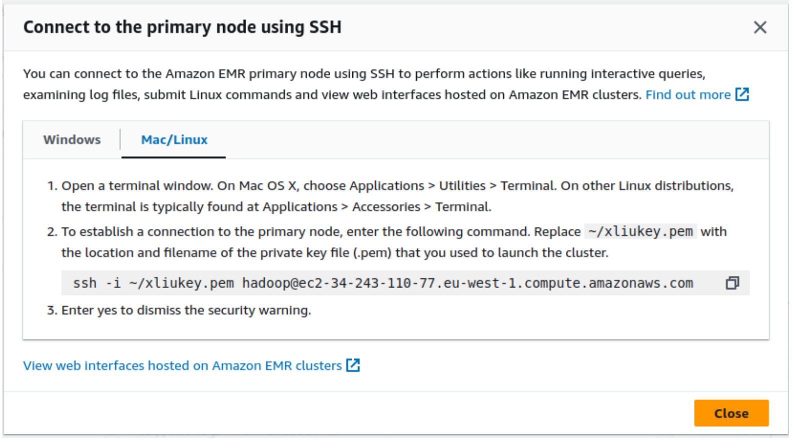
Core and task nodes
EMR-managed security group
sg-0e549fe8389efbf76

Additional security groups
-

If you would like to update the **Primary** node, please select the **security group link** below primary node. You can update **core and task** nodes with the same way as follows. Please update **both** in this lab.

- Choose the **Inbound rules tab** and then choose **Edit inbound rules**.
- Check for an inbound rule that allows public access with the following settings. If it exists, choose **Delete** to remove it, which is (Type: SSH, Port: 22, Source: Custom 0.0.0.0/0)
- **Scroll** to the **bottom** of the list of rules and choose **Add Rule**.
- For Type, select **SSH**. This selection automatically enters TCP for Protocol and 22 for Port Range.
- For source, select **My IP** to automatically add your IP address as the source address.
- Choose **Save**.

For Linux users…



Connect to the primary node using SSH

You can connect to the Amazon EMR primary node using SSH to perform actions like running interactive queries, examining log files, submit Linux commands and view web interfaces hosted on Amazon EMR clusters. Find out more

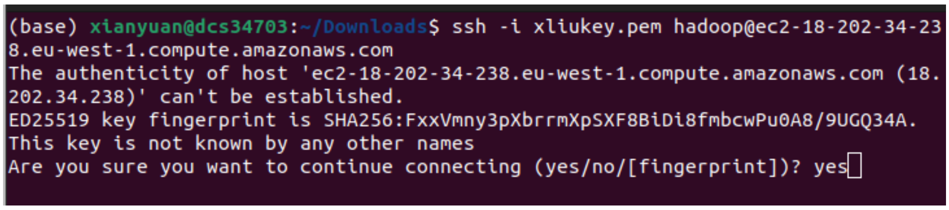
1. Open a terminal window. On Mac OS X, choose Applications > Utilities > Terminal. On other Linux distributions, the terminal is typically found at Applications > Accessories > Terminal.
2. To establish a connection to the primary node, enter the following command. Replace `~/xliukey.pem` with the location and filename of the private key file (.pem) that you used to launch the cluster.

```
ssh -i ~/xliukey.pem hadoop@ec2-34-243-110-77.eu-west-1.compute.amazonaws.com
```

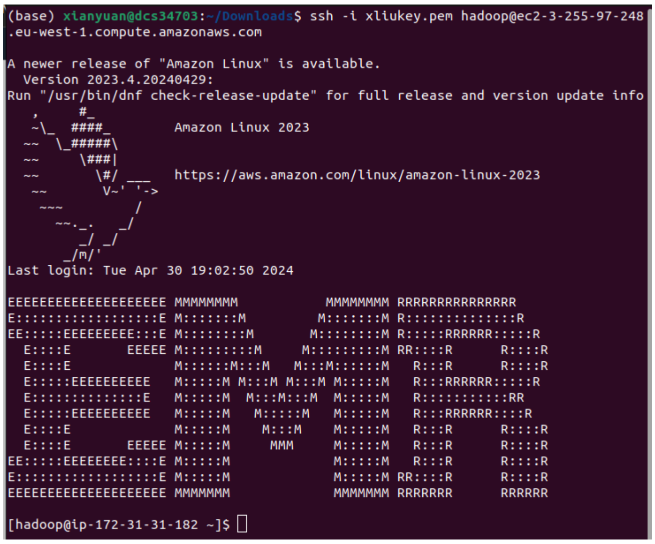
3. Enter yes to dismiss the security warning.

View web interfaces hosted on Amazon EMR clusters

When you connect you will see…



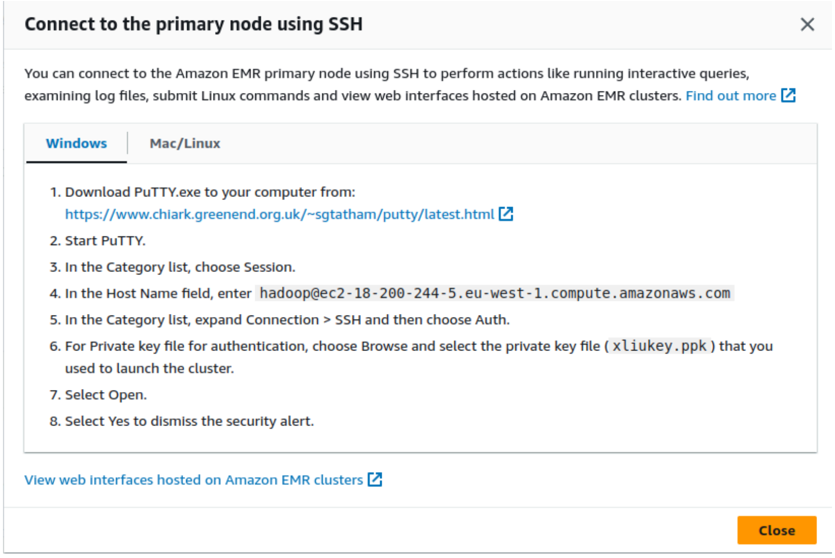
```
(base) xianyuan@dcs34703:~/Downloads$ ssh -i xliukey.pem hadoop@ec2-18-202-34-23
8.eu-west-1.compute.amazonaws.com
The authenticity of host 'ec2-18-202-34-238.eu-west-1.compute.amazonaws.com (18.
202.34.238)' can't be established.
ED25519 key fingerprint is SHA256:FxxVmny3pXbrrmXpSXF8BiDi8fmbcwPu0A8/9UGQ34A.
This key is not known by any other names
Are you sure you want to continue connecting (yes/no/[fingerprint])? yes
```



```
(base) xianyuan@dcs34703:~/Downloads$ ssh -i xliukey.pem hadoop@ec2-3-255-97-248
.eu-west-1.compute.amazonaws.com

A newer release of "Amazon Linux" is available.
  Version 2023.4.20240429:
Run "/usr/bin/dnf check-release-update" for full release and version update info
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'
Last login: Tue Apr 30 19:02:50 2024

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-31-182 ~]$
```

For Windows users…

Please follow the guidance provided by AWS EMR



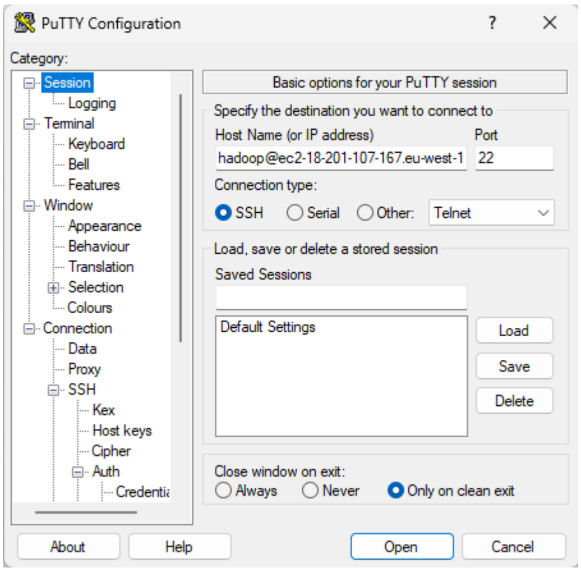
Connect to the primary node using SSH

You can connect to the Amazon EMR primary node using SSH to perform actions like running interactive queries, examining log files, submit Linux commands and view web interfaces hosted on Amazon EMR clusters. Find out more

1. Download PuTTY.exe to your computer from: https://www.chiark.greenend.org.uk/~sgtatham/putty/latest.html
2. Start PuTTY.
3. In the Category list, choose Session.
4. In the Host Name field, enter `hadoop@ec2-18-200-244-5.eu-west-1.compute.amazonaws.com`
5. In the Category list, expand Connection > SSH and then choose Auth.
6. For Private key file for authentication, choose Browse and select the private key file (`xliukey.ppk`) that you used to launch the cluster.
7. Select Open.
8. Select Yes to dismiss the security alert.

View web interfaces hosted on Amazon EMR clusters

- Complete the **Host Name** field, like



PuTTY Configuration

Basic options for your PuTTY session

Specify the destination you want to connect to

Host Name (or IP address): hadoop@ec2-18-201-107-167.eu-west-1

Port: 22

Connection type: ● SSH ○ Serial ○ Other: Telnet

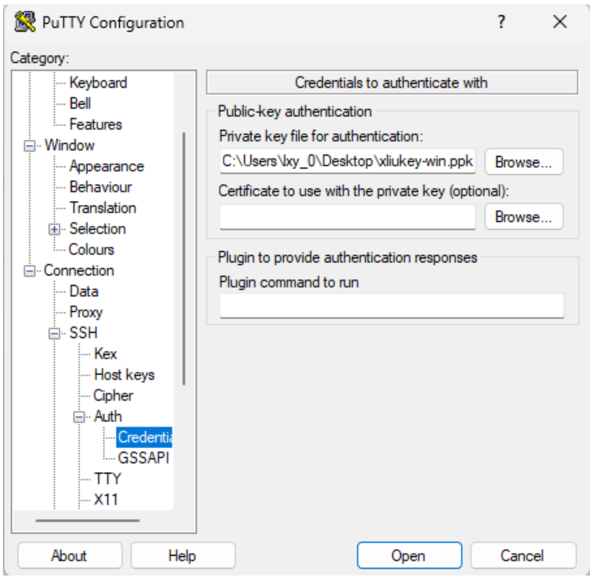
Load, save or delete a stored session

Saved Sessions

Default Settings

Close window on exit: ○ Always ○ Never ● Only on clean exit

- Select the **private key file**, like



PuTTY Configuration

Credentials to authenticate with

Public-key authentication

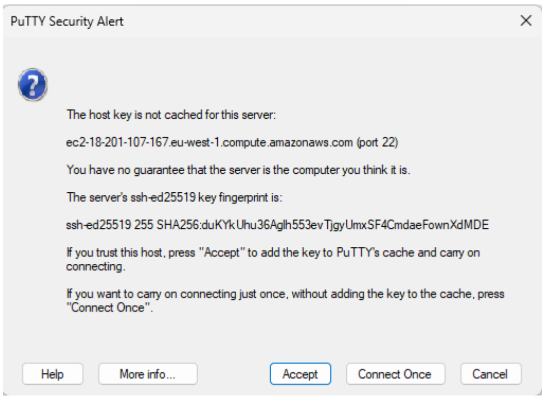
Private key file for authentication: C:\Users\lxy_0\Desktop\xliukey-win.ppk

Certificate to use with the private key (optional):

Plugin to provide authentication responses

Plugin command to run

- Select **Accept** to dismiss the security alert



PuTTY Security Alert

The host key is not cached for this server:

ec2-18-201-107-167.eu-west-1.compute.amazonaws.com (port 22)

You have no guarantee that the server is the computer you think it is.

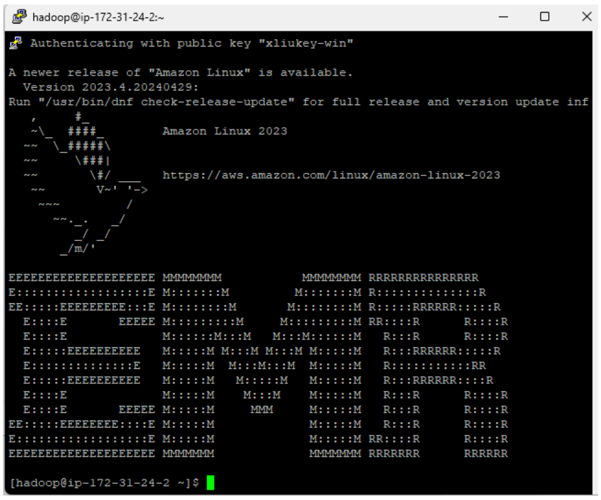
The server's ssh-ed25519 key fingerprint is:

ssh-ed25519 255 SHA256:duKYkUhu36Aglh553evTjgyUmxSF4CmdaeFownXdMDE

If you trust this host, press "Accept" to add the key to PuTTY's cache and carry on connecting.

If you want to carry on connecting just once, without adding the key to the cache, press "Connect Once".

The result is



hadoop@ip-172-31-24-2:~

```
Authenticating with public key "xliukey-win"

A newer release of "Amazon Linux" is available.
  Version 2023.4.20240429:
Run "/usr/bin/dnf check-release-update" for full release and version update inf
   ,     #_
   ~\_  ####_        Amazon Linux 2023
  ~~  \_#####\
  ~~     \###|
  ~~       \#/ ___   https://aws.amazon.com/linux/amazon-linux-2023
   ~~       V~' '->
    ~~~         /
      ~~._.   _/
         _/ _/
       _/m/'

EEEEEEEEEEEEEEEEEEEE MMMMMMMM           MMMMMMMM RRRRRRRRRRRRRRR
E::::::::::::::::::E M:::::::M         M:::::::M R::::::::::::::R
EE:::::EEEEEEEEE:::E M::::::::M       M::::::::M R:::::RRRRRR:::::R
  E::::E       EEEEE M:::::::::M     M:::::::::M RR::::R      R::::R
  E::::E             M::::::M:::M   M:::M::::::M   R:::R      R::::R
  E:::::EEEEEEEEEE   M:::::M M:::M M:::M M:::::M   R:::RRRRRR:::::R
  E::::::::::::::E   M:::::M  M:::M:::M  M:::::M   R:::::::::::RR
  E:::::EEEEEEEEEE   M:::::M   M:::::M   M:::::M   R:::RRRRRR::::R
  E::::E             M:::::M    M:::M    M:::::M   R:::R      R::::R
  E::::E       EEEEE M:::::M     MMM     M:::::M   R:::R      R::::R
EE:::::EEEEEEEE::::E M:::::M             M:::::M   R:::R      R::::R
E::::::::::::::::::E M:::::M             M:::::M RR::::R      R::::R
EEEEEEEEEEEEEEEEEEEE MMMMMMM             MMMMMMM RRRRRRR      RRRRRR

[hadoop@ip-172-31-24-2 ~]$
```

The AWS command line interface

We now have a cluster running and ready for our commands.

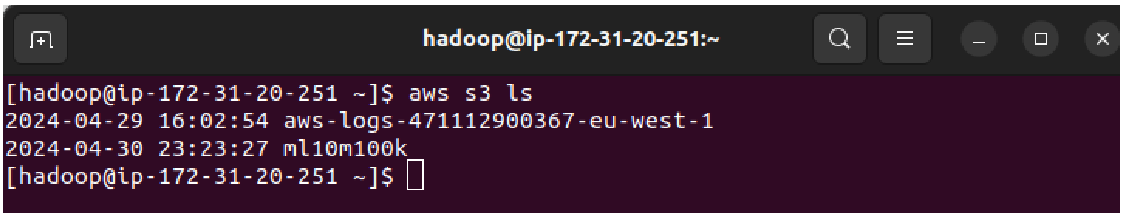
Let's try out the **AWS command line interface**. It is often the case that one will want to perform a console operation repeatedly or describe a complex action programmatically to make it easy to see what has been done. To this end, Amazon provides both a command line tool and an API. All the actions you can do with the console can be done via these alternative interfaces. As we're now logged into a machine that's got the AWS CLI installed, we could quickly try it out.

1. s3 access

First, we're going to look at Amazon's simple storage service (s3). This has a web interface, but we'll be accessing it using the command line. On your new instance type:

aws s3 ls

This will list the bucket(s):

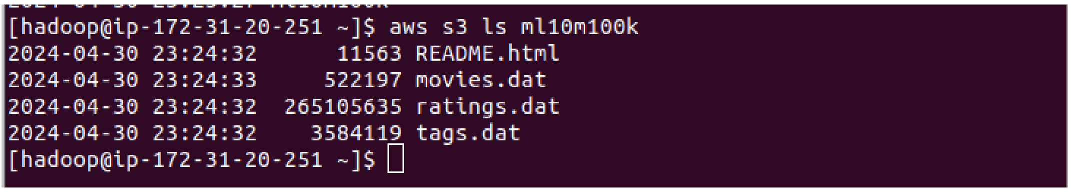


```
[hadoop@ip-172-31-20-251 ~]$ aws s3 ls
2024-04-29 16:02:54 aws-logs-471112900367-eu-west-1
2024-04-30 23:23:27 ml10m100k
[hadoop@ip-172-31-20-251 ~]$
```

To list the contents of the bucket:

aws s3 ls ml10m100k

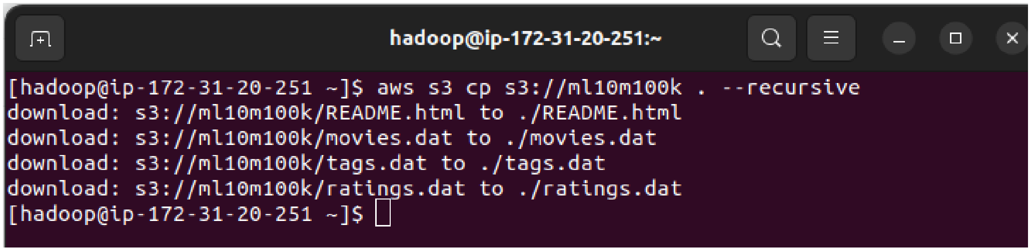
The result is



```
[hadoop@ip-172-31-20-251 ~]$ aws s3 ls ml10m100k
2024-04-30 23:24:32      11563 README.html
2024-04-30 23:24:33     522197 movies.dat
2024-04-30 23:24:32  265105635 ratings.dat
2024-04-30 23:24:32    3584119 tags.dat
[hadoop@ip-172-31-20-251 ~]$
```

Then to copy the data to our instance,

aws s3 cp s3://ml10m100k . --recursive



```
[hadoop@ip-172-31-20-251 ~]$ aws s3 cp s3://ml10m100k . --recursive
download: s3://ml10m100k/README.html to ./README.html
download: s3://ml10m100k/movies.dat to ./movies.dat
download: s3://ml10m100k/tags.dat to ./tags.dat
download: s3://ml10m100k/ratings.dat to ./ratings.dat
[hadoop@ip-172-31-20-251 ~]$
```

2. ec2 via the command line

Also, try `` `aws ec2 describe-instances` `` to list all the instances currently in this account. Enter **q** to exit.

The command line interface allows you to create and destroy instances, configure users, launch clusters, and do any other operation that you can also do with the web interface. You may find you don't have access to some of these commands, however.

Be careful, you can terminate your fellow students' instances as you are all in the same account and have sufficient permissions. For more help on a particular aspect, you can type something like,

aws ec2 help

3. Control via the API

We can also access AWS using the API.

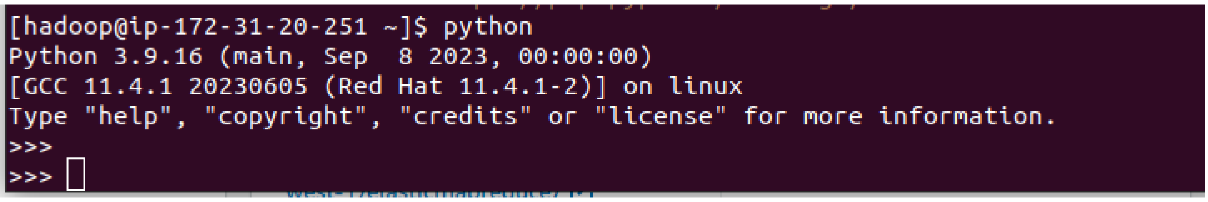
On the command line, we need to install boto3, a library for accessing the AWS API from Python, by entering

sudo pip3 install boto3

While still logged into the cluster, start a Python terminal via

python

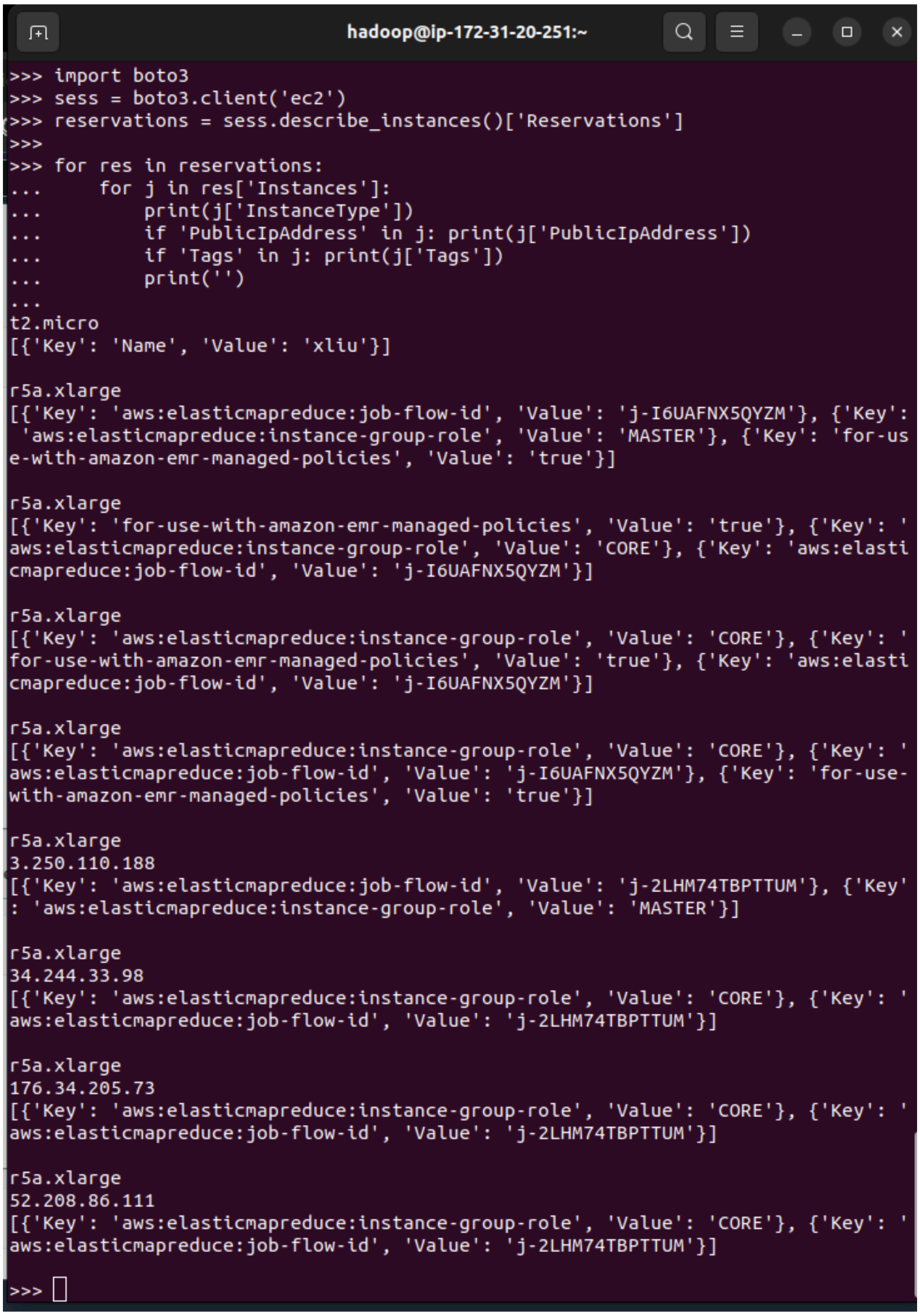
The result will be



```
[hadoop@ip-172-31-20-251 ~]$ python
Python 3.9.16 (main, Sep  8 2023, 00:00:00)
[GCC 11.4.1 20230605 (Red Hat 11.4.1-2)] on linux
Type "help", "copyright", "credits" or "license" for more information.
>>>
>>> 
```

We can then initiate a session to control EC2 by connecting to the AWS API. This is done by entering the following commands:

```
import boto3
sess = boto3.client('ec2')
reservations = sess.describe_instances()['Reservations']

for res in reservations:
   for j in res['Instances']:
      print(j['InstanceType'])
      if 'PublicIpAddress' in j: print(j['PublicIpAddress'])
      if 'Tags' in j: print(j['Tags'])
      print('')
```



```
>>> import boto3
>>> sess = boto3.client('ec2')
>>> reservations = sess.describe_instances()['Reservations']
>>>
>>> for res in reservations:
...     for j in res['Instances']:
...         print(j['InstanceType'])
...         if 'PublicIpAddress' in j: print(j['PublicIpAddress'])
...         if 'Tags' in j: print(j['Tags'])
...         print('')
...
t2.micro
[{'Key': 'Name', 'Value': 'xliu'}]

r5a.xlarge
[{'Key': 'aws:elasticmapreduce:job-flow-id', 'Value': 'j-I6UAFNX5QYZM'}, {'Key':
 'aws:elasticmapreduce:instance-group-role', 'Value': 'MASTER'}, {'Key': 'for-us
e-with-amazon-emr-managed-policies', 'Value': 'true'}]

r5a.xlarge
[{'Key': 'for-use-with-amazon-emr-managed-policies', 'Value': 'true'}, {'Key': '
aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': 'aws:elasti
cmapreduce:job-flow-id', 'Value': 'j-I6UAFNX5QYZM'}]

r5a.xlarge
[{'Key': 'aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': '
for-use-with-amazon-emr-managed-policies', 'Value': 'true'}, {'Key': 'aws:elasti
cmapreduce:job-flow-id', 'Value': 'j-I6UAFNX5QYZM'}]

r5a.xlarge
[{'Key': 'aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': '
aws:elasticmapreduce:job-flow-id', 'Value': 'j-I6UAFNX5QYZM'}, {'Key': 'for-use-
with-amazon-emr-managed-policies', 'Value': 'true'}]

r5a.xlarge
3.250.110.188
[{'Key': 'aws:elasticmapreduce:job-flow-id', 'Value': 'j-2LHM74TBPTTUM'}, {'Key'
: 'aws:elasticmapreduce:instance-group-role', 'Value': 'MASTER'}]

r5a.xlarge
34.244.33.98
[{'Key': 'aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': '
aws:elasticmapreduce:job-flow-id', 'Value': 'j-2LHM74TBPTTUM'}]

r5a.xlarge
176.34.205.73
[{'Key': 'aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': '
aws:elasticmapreduce:job-flow-id', 'Value': 'j-2LHM74TBPTTUM'}]

r5a.xlarge
52.208.86.111
[{'Key': 'aws:elasticmapreduce:instance-group-role', 'Value': 'CORE'}, {'Key': '
aws:elasticmapreduce:job-flow-id', 'Value': 'j-2LHM74TBPTTUM'}]

>>> 
```

This will print out a list of all the instance types, and their tags.
Leave the Python interface [**ctrl-D** or **exit()**] so we can use PySpark instead...

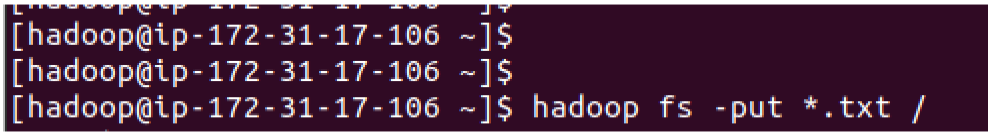
4. Word counting using PySpark

We need to put the datafiles we're going to use within reach of Hadoop….We simply run (on the command line). Referring to https://spark.apache.org/examples.html.
First, copy the data to our instance

aws s3 cp s3://some-texts . --recursive

Then, put the data within reach of Hadoop (Please try again, if error ‘8020 failed on connection exception: java.net.ConnectException: Connection refused’ occurs.)

```
hadoop fs -put *.txt /
```



```
[hadoop@ip-172-31-17-106 ~]$
[hadoop@ip-172-31-17-106 ~]$
[hadoop@ip-172-31-17-106 ~]$ hadoop fs -put *.txt /
```

This copies the data to the Hadoop Distributed File System.

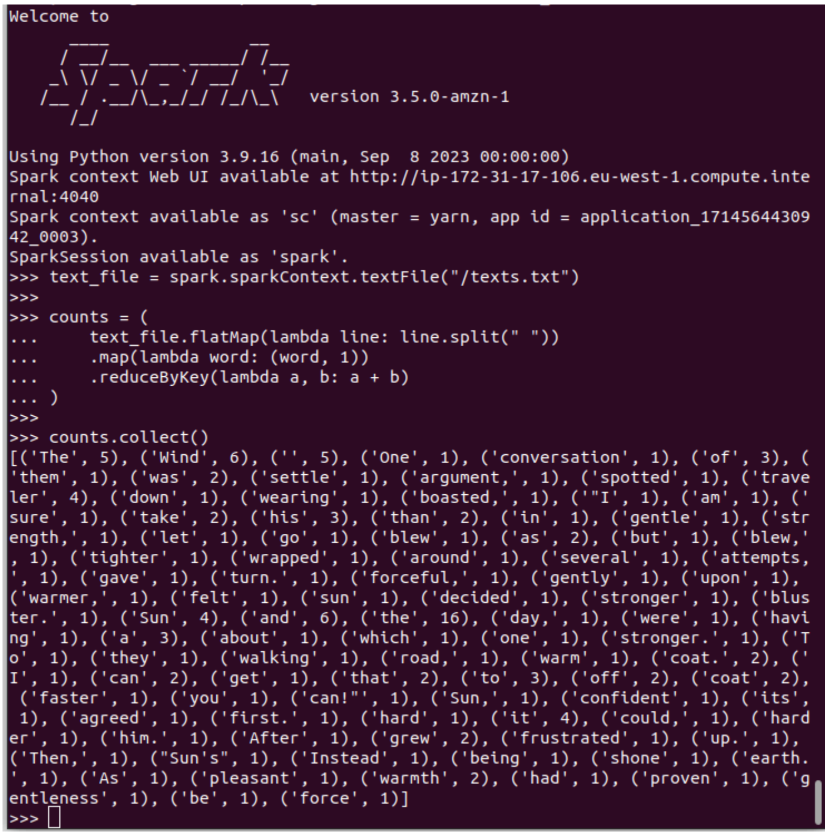
Spark RDD Example

You would like to compute the count of each word in the text file. Spark allows for efficient execution of the query because it parallelizes this computation. Many other query engines aren’t capable of parallelizing computations.

First, **Enter** ‘`pyspark`’ to open the interactive python/spark command line (not just python!).

Then, performing the computation with Spark RDDs:

```
text_file = spark.sparkContext.textFile("/texts.txt")

counts = (
   text_file.flatMap(lambda line: line.split(" "))
   .map(lambda word: (word, 1))
   .reduceByKey(lambda a, b: a + b)
)

counts.collect()
```



```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.0-amzn-1
      /_/

Using Python version 3.9.16 (main, Sep  8 2023 00:00:00)
Spark context Web UI available at http://ip-172-31-17-106.eu-west-1.compute.inte
rnal:4040
Spark context available as 'sc' (master = yarn, app id = application_17145644309
42_0003).
SparkSession available as 'spark'.
>>> text_file = spark.sparkContext.textFile("/texts.txt")
>>>
>>> counts = (
...     text_file.flatMap(lambda line: line.split(" "))
...     .map(lambda word: (word, 1))
...     .reduceByKey(lambda a, b: a + b)
... )
>>>
>>> counts.collect()
[('The', 5), ('Wind', 6), ('', 5), ('One', 1), ('conversation', 1), ('of', 3), (
'them', 1), ('was', 2), ('settle', 1), ('argument,', 1), ('spotted', 1), ('trave
ler', 4), ('down', 1), ('wearing', 1), ('boasted,', 1), ('"I', 1), ('am', 1), ('
sure', 1), ('take', 2), ('his', 3), ('than', 2), ('in', 1), ('gentle', 1), ('str
ength,', 1), ('let', 1), ('go', 1), ('blew', 1), ('as', 2), ('but', 1), ('blew,'
, 1), ('tighter', 1), ('wrapped', 1), ('around', 1), ('several', 1), ('attempts,
', 1), ('gave', 1), ('turn.', 1), ('forceful,', 1), ('gently', 1), ('upon', 1),
('warmer,', 1), ('felt', 1), ('sun', 1), ('decided', 1), ('stronger', 1), ('blus
ter.', 1), ('Sun', 4), ('and', 6), ('the', 16), ('day,', 1), ('were', 1), ('havi
ng', 1), ('a', 3), ('about', 1), ('which', 1), ('one', 1), ('stronger.', 1), ('T
o', 1), ('they', 1), ('walking', 1), ('road,', 1), ('warm', 1), ('coat.', 2), ('
I', 1), ('can', 2), ('get', 1), ('that', 2), ('to', 3), ('off', 2), ('coat', 2),
 ('faster', 1), ('you', 1), ('can!"', 1), ('Sun,', 1), ('confident', 1), ('its',
 1), ('agreed', 1), ('first.', 1), ('hard', 1), ('it', 4), ('could,', 1), ('hard
er', 1), ('him.', 1), ('After', 1), ('grew', 2), ('frustrated', 1), ('up.', 1),
('Then,', 1), ("Sun's", 1), ('Instead', 1), ('being', 1), ('shone', 1), ('earth.
', 1), ('As', 1), ('pleasant', 1), ('warmth', 2), ('had', 1), ('proven', 1), ('g
entleness', 1), ('be', 1), ('force', 1)]
>>> 
```

- Exit by using **Ctrl-D**

5. Make a recommendation using PySpark

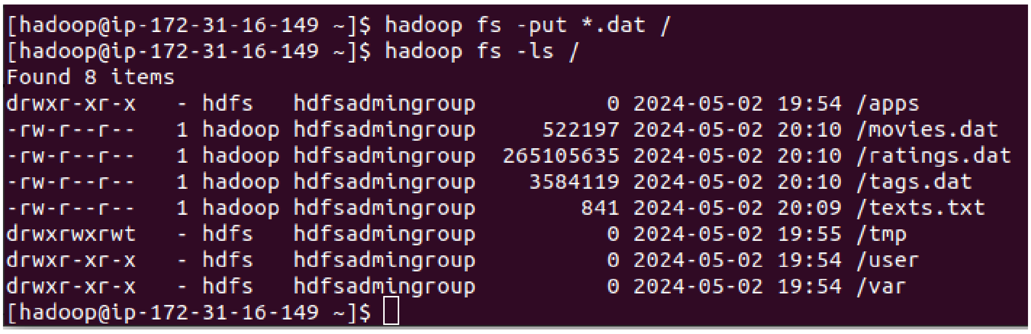
We need to put the datafiles we're going to use within reach of Hadoop….We simply run (on the command line).

hadoop fs -put *.dat /

This copies the data to the Hadoop Distributed File System.
You can check if the files there, via

hadoop fs -ls /

The result is



```
[hadoop@ip-172-31-16-149 ~]$ hadoop fs -put *.dat /
[hadoop@ip-172-31-16-149 ~]$ hadoop fs -ls /
Found 8 items
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2024-05-02 19:54 /apps
-rw-r--r--   1 hadoop hdfsadmingroup     522197 2024-05-02 20:10 /movies.dat
-rw-r--r--   1 hadoop hdfsadmingroup  265105635 2024-05-02 20:10 /ratings.dat
-rw-r--r--   1 hadoop hdfsadmingroup    3584119 2024-05-02 20:10 /tags.dat
-rw-r--r--   1 hadoop hdfsadmingroup        841 2024-05-02 20:09 /texts.txt
drwxrwxrwt   - hdfs   hdfsadmingroup          0 2024-05-02 19:55 /tmp
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2024-05-02 19:54 /user
drwxr-xr-x   - hdfs   hdfsadmingroup          0 2024-05-02 19:54 /var
[hadoop@ip-172-31-16-149 ~]$ 
```

The PySpark command line

Enter '`pyspark`' to open the interactive python/spark command line (not just python!).

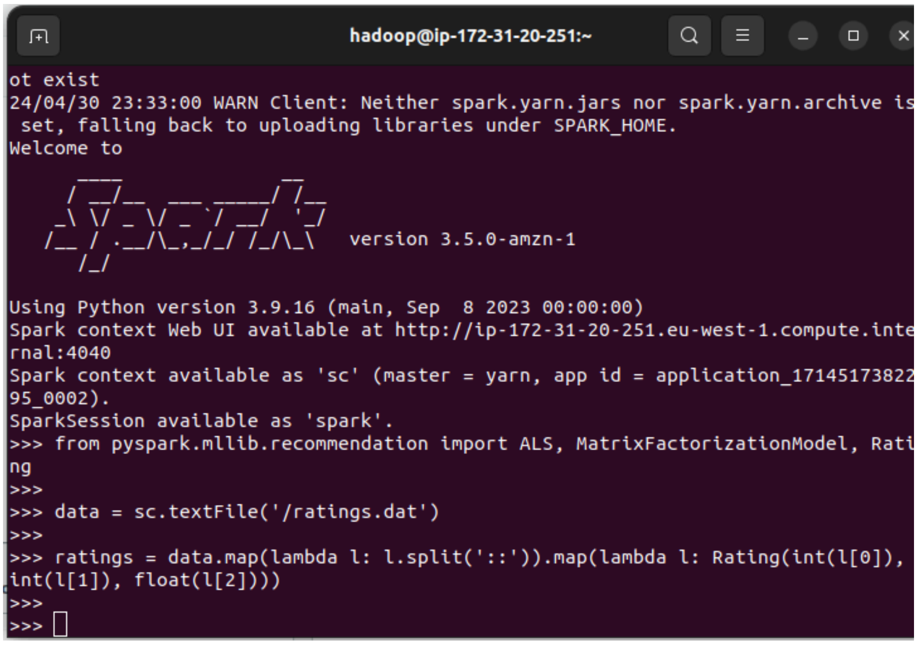
We load the rating data from the Hadoop file system. This is a '**::**' separated table of ratings. The lambda/map functions are to split by these '**::**' and assign the types to the three columns.

from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rating

data = sc.textFile('/ratings.dat')

ratings = data.map(lambda l: l.split('::')).map(lambda l: Rating(int(l[0]), int(l[1]), float(l[2])))

This will be very quick as **Spark only runs things when it needs to** (hence none of the above is run until the following lines are run...). The result is



```
ot exist
24/04/30 23:33:00 WARN Client: Neither spark.yarn.jars nor spark.yarn.archive is
 set, falling back to uploading libraries under SPARK_HOME.
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.0-amzn-1
      /_/

Using Python version 3.9.16 (main, Sep  8 2023 00:00:00)
Spark context Web UI available at http://ip-172-31-20-251.eu-west-1.compute.inte
rnal:4040
Spark context available as 'sc' (master = yarn, app id = application_17145173822
95_0002).
SparkSession available as 'spark'.
>>> from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rati
ng
>>>
>>> data = sc.textFile('/ratings.dat')
>>>
>>> ratings = data.map(lambda l: l.split('::')).map(lambda l: Rating(int(l[0]),
int(l[1]), float(l[2])))
>>>
>>> 
```

Let's ask for the first item, to check it's working:

ratings.first()

A quick recap of recommender-systems/collaborative filtering

Let's set up the model...we're trying to guess the ratings people give to particular movies. We assume for this model that the full matrix of all reviews by everyone of every movie could be approximated by multiplying two low-rank matrices… for example, if there are N people and M movies, we believe there is an NxM matrix we want to approximate potentially. We could approximate this with the product of an Nx2 and a 2xM matrix (to help intuition imagine if the first column of the Nx2 is how much each person likes funny movies, and the second column is how much they like scary movies, then the first row of the 2xM matrix is how funny each movie is and the second row is how scary, realistically we don’t anticipate looking for such patterns here).

Back to the application

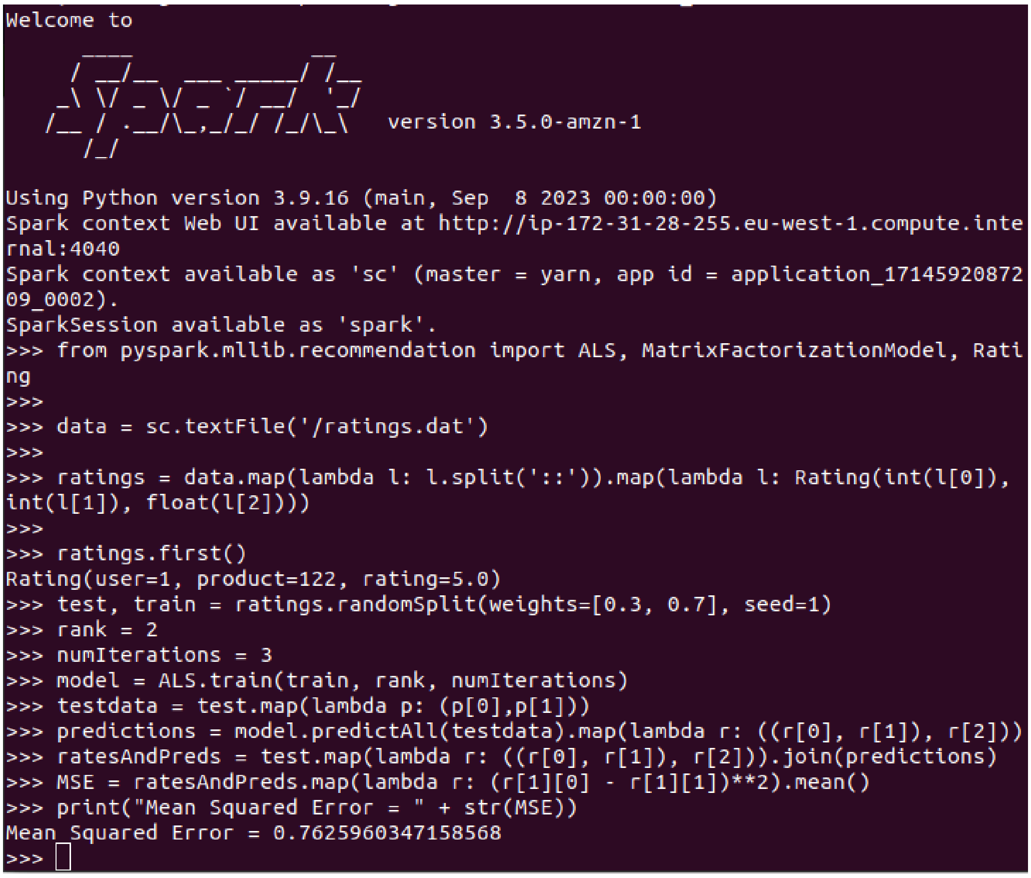
We split our dataset randomly into a training and test set, then build a model using the Alternating Least Squares approach[1] to factorising a matrix...

```
test, train = ratings.randomSplit(weights=[0.3, 0.7], seed=1)
rank = 2
numIterations = 2
model = ALS.train(train, rank, numIterations)

testdata = test.map(lambda p: (p[0],p[1]))
predictions = model.predictAll(testdata).map(lambda r: ((r[0], r[1]), r[2]))
ratesAndPreds = test.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1])**2).mean()
print("Mean Squared Error = " + str(MSE))
```

The result shows the value of the Mean Squared Error.

[1] *Alternating Least Squares (ALS) matrix factorization.*

> *ALS attempts to estimate the ratings matrix R as the product of two lower-rank matrices, X and Y, i.e. X * Yt = R. Typically these approximations are called ‘factor’ matrices. The general approach is iterative. During each iteration, one of the factor matrices is held constant, while the other is solved for using least squares. The newly-solved factor matrix is then held constant while solving for the other factor matrix.*
>
> *From the spark API reference.*

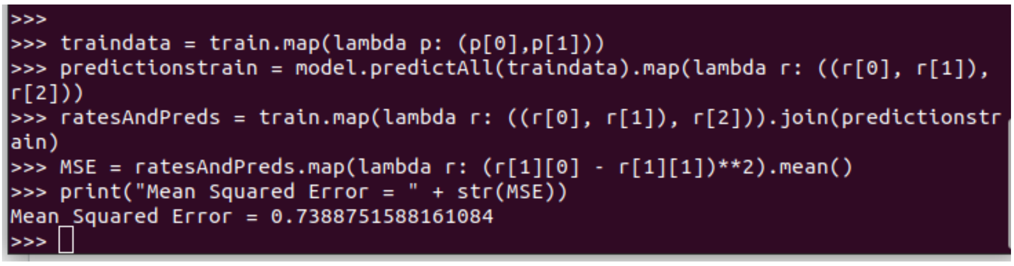


```
Welcome to
      ____              __
     / __/__  ___ _____/ /__
    _\ \/ _ \/ _ `/ __/  '_/
   /__ / .__/\_,_/_/ /_/\_\   version 3.5.0-amzn-1
      /_/

Using Python version 3.9.16 (main, Sep  8 2023 00:00:00)
Spark context Web UI available at http://ip-172-31-28-255.eu-west-1.compute.inte
rnal:4040
Spark context available as 'sc' (master = yarn, app id = application_17145920872
09_0002).
SparkSession available as 'spark'.
>>> from pyspark.mllib.recommendation import ALS, MatrixFactorizationModel, Rati
ng
>>>
>>> data = sc.textFile('/ratings.dat')
>>>
>>> ratings = data.map(lambda l: l.split('::')).map(lambda l: Rating(int(l[0]),
int(l[1]), float(l[2])))
>>>
>>> ratings.first()
Rating(user=1, product=122, rating=5.0)
>>> test, train = ratings.randomSplit(weights=[0.3, 0.7], seed=1)
>>> rank = 2
>>> numIterations = 3
>>> model = ALS.train(train, rank, numIterations)
>>> testdata = test.map(lambda p: (p[0],p[1]))
>>> predictions = model.predictAll(testdata).map(lambda r: ((r[0], r[1]), r[2]))
>>> ratesAndPreds = test.map(lambda r: ((r[0], r[1]), r[2])).join(predictions)
>>> MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1])**2).mean()
>>> print("Mean Squared Error = " + str(MSE))
Mean Squared Error = 0.7625960347158568
>>> 
```

Finally, a quick look at the MSE on the training data;

```
traindata = train.map(lambda p: (p[0],p[1]))
predictionstrain = model.predictAll(traindata).map(lambda r: ((r[0], r[1]), r[2]))
ratesAndPreds = train.map(lambda r: ((r[0], r[1]), r[2])).join(predictionstrain)
MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1])**2).mean()
print("Mean Squared Error = " + str(MSE))
```

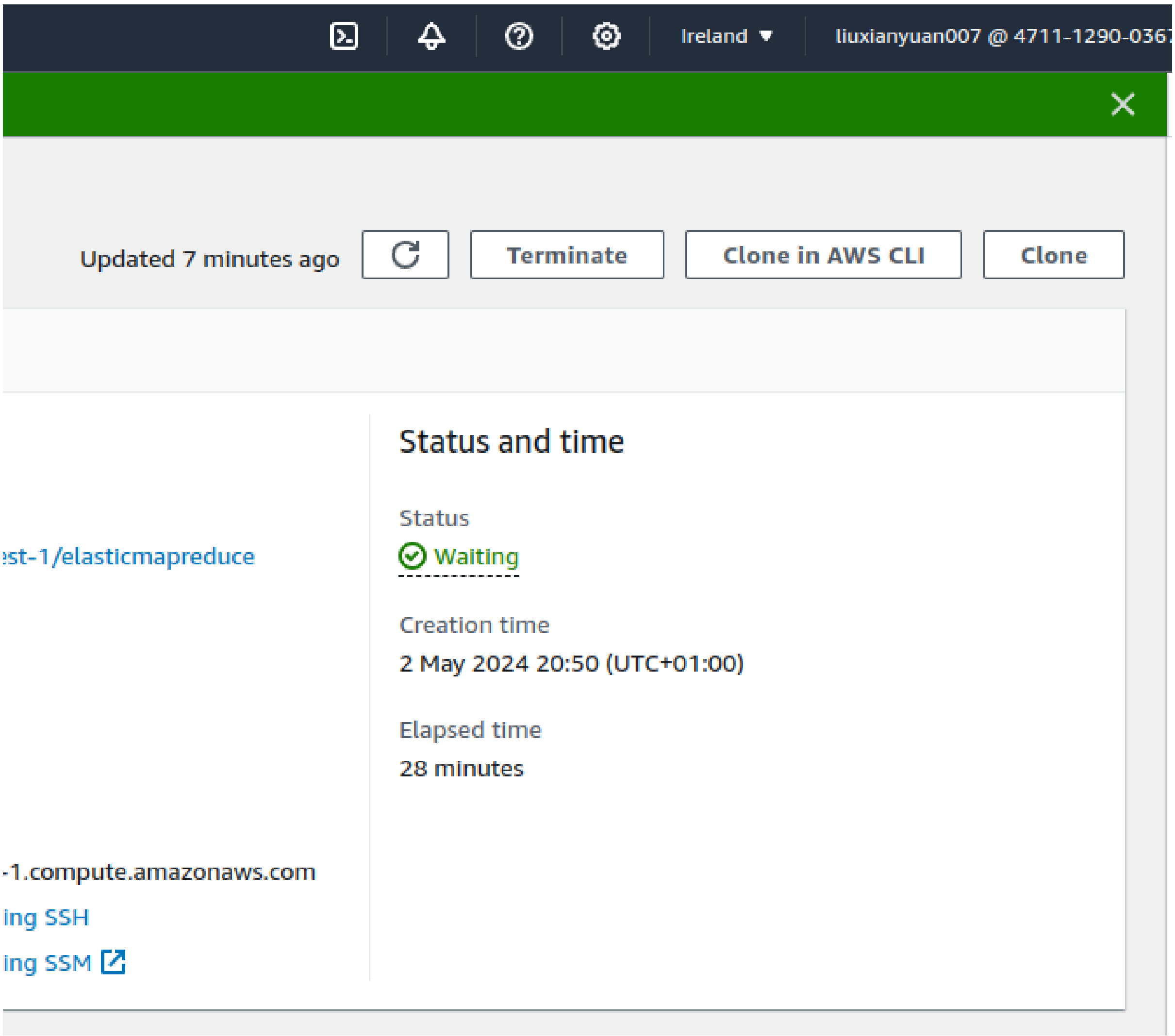
The result is



```
>>>
>>> traindata = train.map(lambda p: (p[0],p[1]))
>>> predictionstrain = model.predictAll(traindata).map(lambda r: ((r[0], r[1]),
r[2]))
>>> ratesAndPreds = train.map(lambda r: ((r[0], r[1]), r[2])).join(predictionstr
ain)
>>> MSE = ratesAndPreds.map(lambda r: (r[1][0] - r[1][1])**2).mean()
>>> print("Mean Squared Error = " + str(MSE))
Mean Squared Error = 0.7388751588161084
>>> 
```

6. Terminate

- Enter '**exit**' in the terminal or [Ctrl + d] to exit pyspark
- Enter **'exit'** to log out of the instance
- Click **'Terminate'** to power off the cluster



est-1/elasticmapreduce

-1.compute.amazonaws.com

ing SSH

ing SSM

Status and time

Status

✓ Waiting

Creation time

2 May 2024 20:50 (UTC+01:00)

Elapsed time

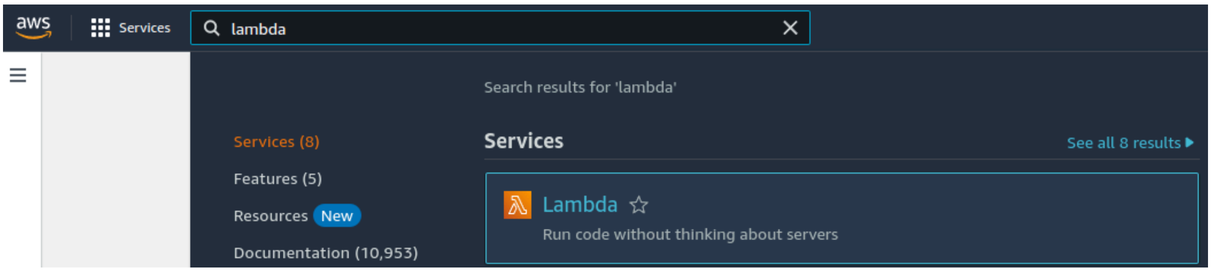
28 minutes

Using Lambda

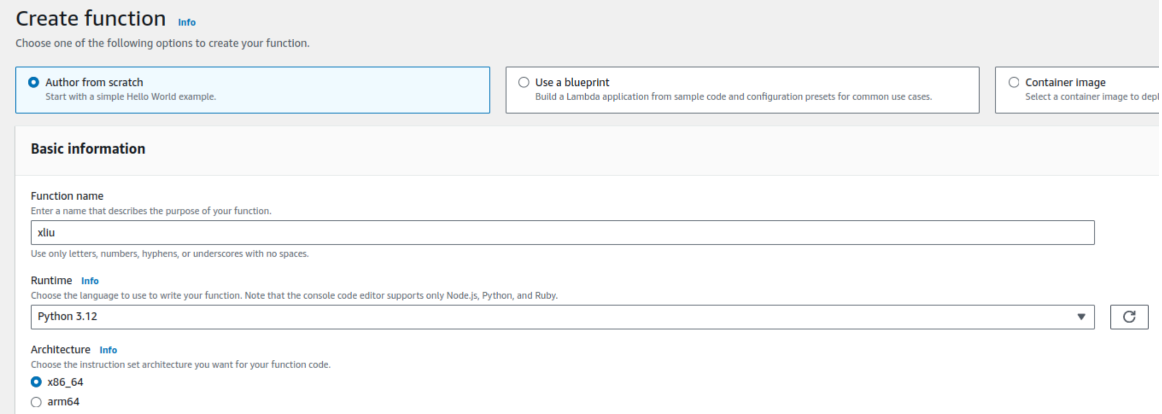
Another exercise: You can create a lambda function!

1. Create function

Find the lambda service…



- Click '**Create function**'
- Author from scratch
- Function name: pick something **unique**, such as adding your ID
- Runtime: Python 3.12



Create function Info

Choose one of the following options to create your function.

● Author from scratch
Start with a simple Hello World example.

○ Use a blueprint
Build a Lambda application from sample code and configuration presets for common use cases.

○ Container image
Select a container image to depl

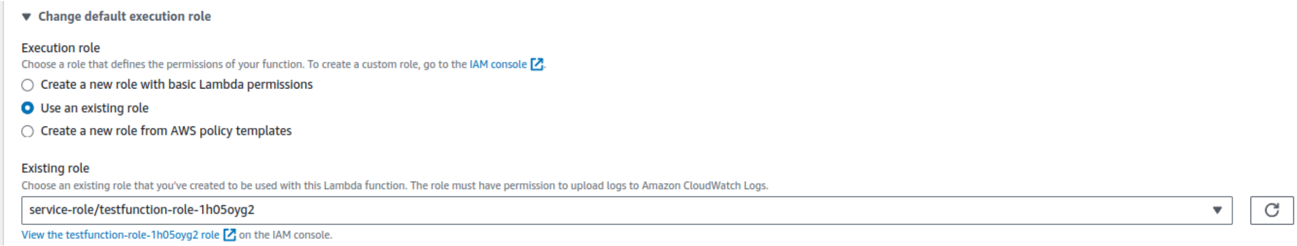
Basic information

Function name
Enter a name that describes the purpose of your function.
xliu
Use only letters, numbers, hyphens, or underscores with no spaces.

Runtime Info
Choose the language to use to write your function. Note that the console code editor supports only Node.js, Python, and Ruby.
Python 3.12

Architecture Info
Choose the instruction set architecture you want for your function code.
● x86_64
○ arm64

- Click '**Change default execution role**', choose '**use an existing role**' and then choose 'service-role/testfunction-role-X'.

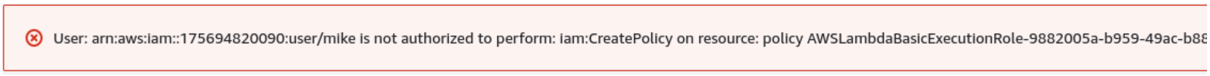


▼ Change default execution role

Execution role
Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.
○ Create a new role with basic Lambda permissions
● Use an existing role
○ Create a new role from AWS policy templates

Existing role
Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.
service-role/testfunction-role-1h05oyg2
View the testfunction-role-1h05oyg2 role on the IAM console.

Note: you need to change the execution role, as it, by default, tries to create a new role with lambda permissions (**the IAM profile you're using doesn't have permissions for that**). Example error message:



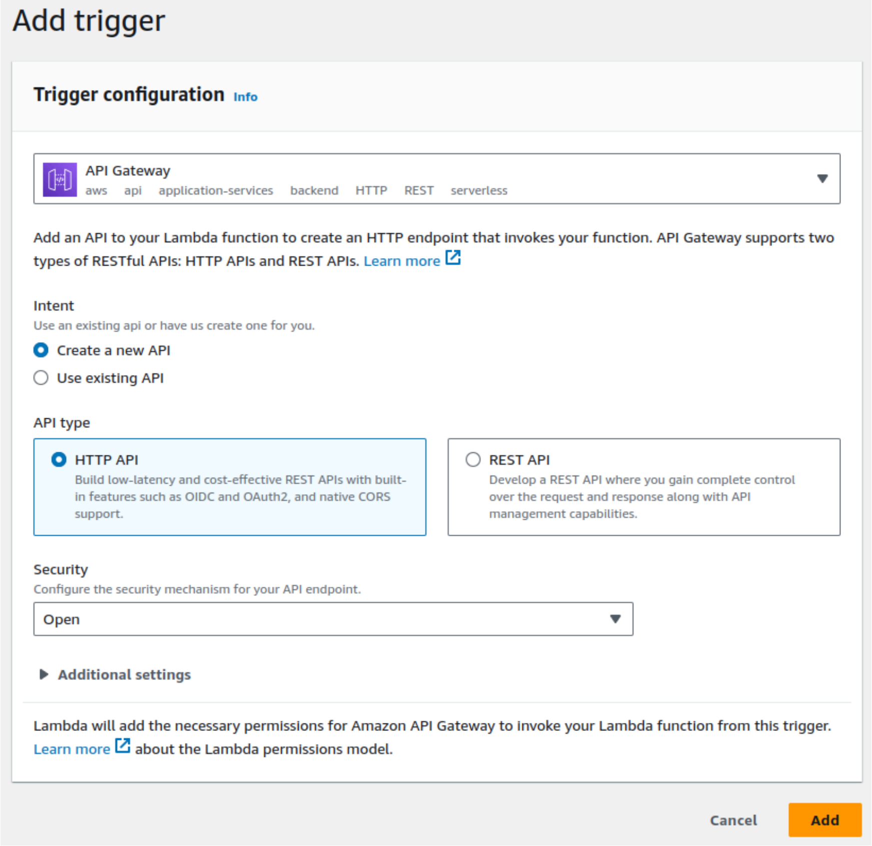
User: arn:aws:iam::175694820090:user/mike is not authorized to perform: iam:CreatePolicy on resource: policy AWSLambdaBasicExecutionRole-9882005a-b959-49ac-b88

So please use the ‘testfunction’ role.

- Click

2. Add Trigger

- Click **Add Trigger** -> select **‘API Gateway’** from the drop-down list
- Select **‘Create a new API’**
- For Security select **‘open’**
- Click **‘Add’**. Here’s the API Gateway configuration



Add trigger

Trigger configuration Info

API Gateway
aws api application-services backend HTTP REST serverless ▼

Add an API to your Lambda function to create an HTTP endpoint that invokes your function. API Gateway supports two types of RESTful APIs: HTTP APIs and REST APIs. Learn more

Intent
Use an existing api or have us create one for you.
● Create a new API
○ Use existing API

API type
● HTTP API
Build low-latency and cost-effective REST APIs with built-in features such as OIDC and OAuth2, and native CORS support.
○ REST API
Develop a REST API where you gain complete control over the request and response along with API management capabilities.

Security
Configure the security mechanism for your API endpoint.
Open ▼

▶ Additional settings

Lambda will add the necessary permissions for Amazon API Gateway to invoke your Lambda function from this trigger. Learn more about the Lambda permissions model.

Cancel Add

- Click ‘Add’

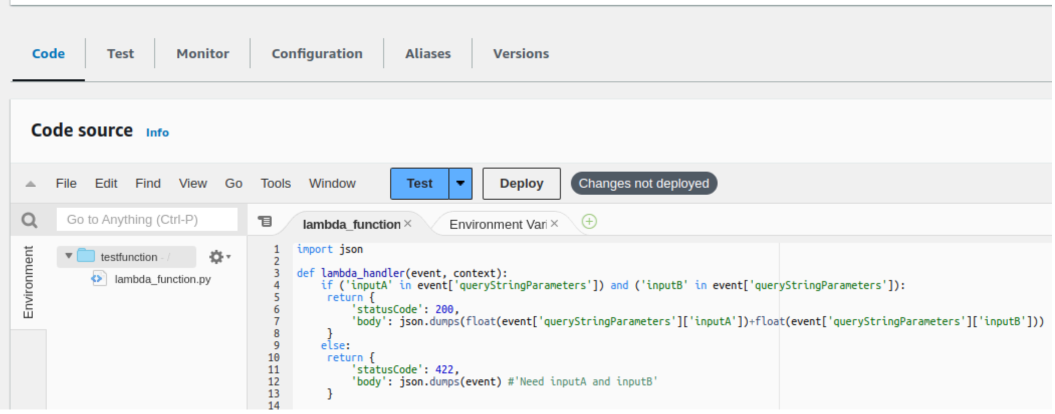
3. Update code

- Click **‘code’**

Here’s some example code for the lambda that adds together two numbers.

```
import json

def lambda_handler(event, context):
    if ('inputA' in event['queryStringParameters']) and ('inputB' in
event['queryStringParameters']):
        return {
                'statusCode': 200,
                'body':
json.dumps(float(event['queryStringParameters']['inputA'])+float(event['queryStringParameters']['inputB']))
        }
    else:
        return {
                'statusCode': 422,
                'body': json.dumps(event) #'Need inputA and inputB'
        }
```

- Click **'Deploy'**

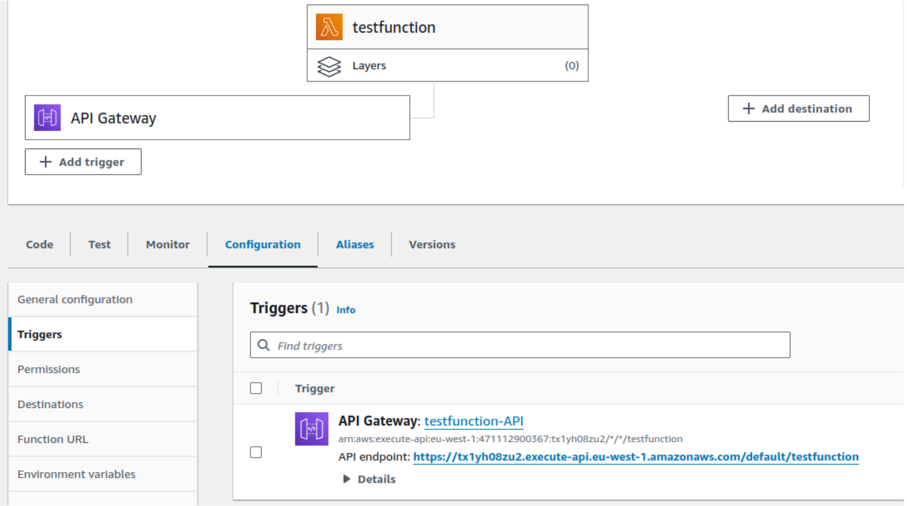


```
import json

def lambda_handler(event, context):
    if ('inputA' in event['queryStringParameters']) and ('inputB' in event['queryStringParameters']):
     return {
         'statusCode': 200,
         'body': json.dumps(float(event['queryStringParameters']['inputA'])+float(event['queryStringParameters']['inputB']))
     }
    else:
     return {
         'statusCode': 422,
         'body': json.dumps(event) #'Need inputA and inputB'
     }
```

4. Test

Option 1

- Visit the API interface address (click on the API Gateway again to find the URL):



API Gateway: testfunction-API

arn:aws:execute-api:eu-west-1:471112900367:tx1yh08zu2/*/*/testfunction

API endpoint: https://tx1yh08zu2.execute-api.eu-west-1.amazonaws.com/default/testfunction

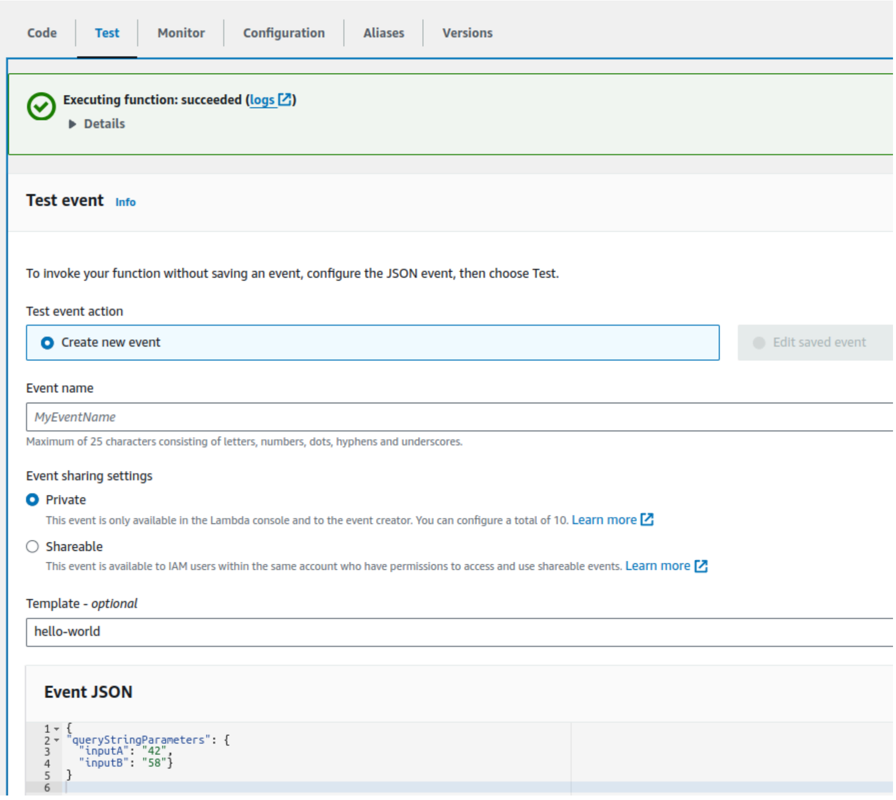
- Click the link to the '**API endpoint**'.

- You’ll probably get an “{"message":"Internal Server Error"}”, as the code we wrote expects two arguments.
- Edit the URL by adding arguments after the URL, e.g.
  If the endpoint was at
  https://tx1yh08zu2.execute-api.eu-west-1.amazonaws.com/default/xliu,
  I will add
  ?inputA=42&inputB=58
  to the end of the URL:
  https://tx1yh08zu2.execute-api.eu-west-1.amazonaws.com/default/xliu?inputA=3&inputB=4
  That seems to work! The result is 100.

Tip: Copying and pasting from the Google Doc sometimes jumbles the indentation in the code - so that might need sorting.

Option 2

You can also configure a test event (see Test tab).



Executing function: succeeded (logs)

▸ Details

Test event Info

To invoke your function without saving an event, configure the JSON event, then choose Test.

Test event action

● Create new event ○ Edit saved event

Event name

MyEventName

Maximum of 25 characters consisting of letters, numbers, dots, hyphens and underscores.

Event sharing settings

● Private
This event is only available in the Lambda console and to the event creator. You can configure a total of 10. Learn more

○ Shareable
This event is available to IAM users within the same account who have permissions to access and use shareable events. Learn more

Template - *optional*

hello-world

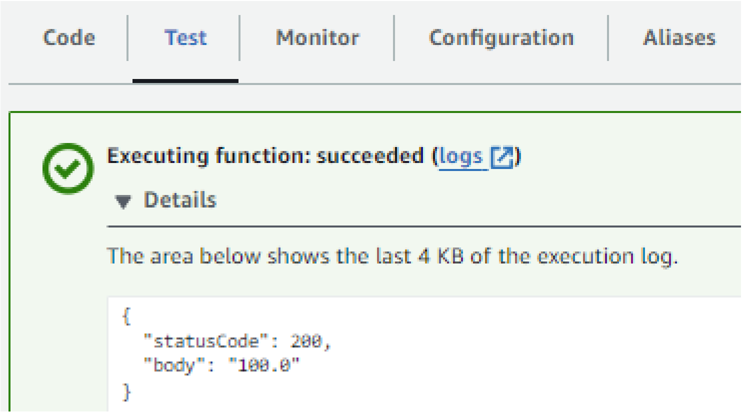
Event JSON

```
{
"queryStringParameters": {
  "inputA": "42",
  "inputB": "58"}
}

```

Test event:

{
"queryStringParameters": {
  "inputA": "42",
  "inputB": "58"}
}

- Scroll down Details, and check the output: 100



Executing function: succeeded (logs)

The area below shows the last 4 KB of the execution log.

```
{
  "statusCode": 200,
  "body": "100.0"
}
```

Additional exercises (SKIP)

Trigger on a file being added to an s3 bucket…
https://docs.aws.amazon.com/lambda/latest/dg/with-s3-example.html

Back to the command line…

A global network of air pollution sensors is aggregated by openAQ. They archive this data on s3 (see here: https://openaq-data.s3.amazonaws.com/index.html)

(tip: you can leave PySpark using Ctrl-D).

We can copy one of these csv files to our instance from the command line using:

```
aws s3 cp s3://openaq-data/2019-01-01.csv .
```

Tip: This might be faster if the instance were in the same region as the data!

Or, from inside PySpark, can read the data directly:

```
df = spark.read.csv("s3a://openaq-data/2019-01-01.csv")
```

This could be improved by treating the headers properly for a start. Add options to do this (hint set `header=True`)

Feel free to try analysing this data. For example: count the number of measurements in Uganda...

Hint the 'country' column would equal 'UG'...

```
df[df['country']=='UG'].count()
```

Note that without the `.count()` the result is immediate - spark doesn't do any computation until it has to.

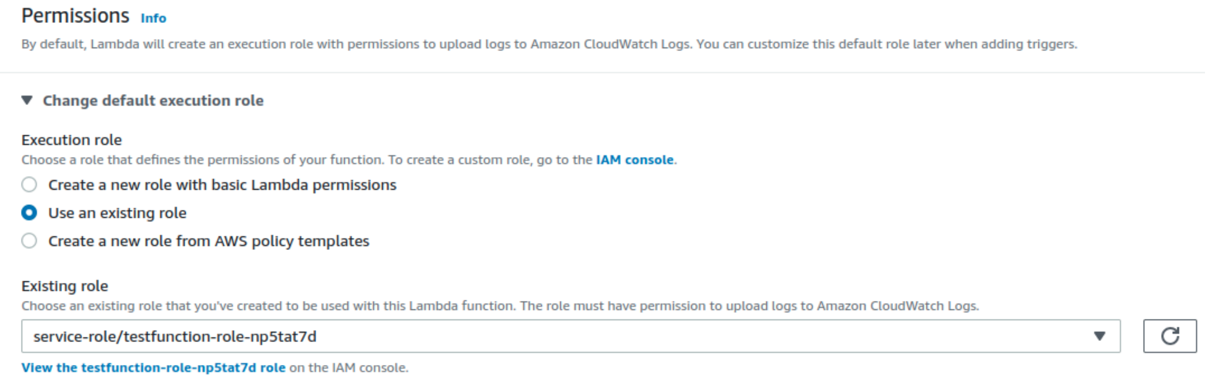
You'll find the operation is far quicker the second time thanks to caching. Also look at `.persist()` here.

Starting and stopping EC2 instances from the command line

Example code, modify to start an instance yourself.

```
aws ec2 run-instances --image-id ami-xxxxxxxx --count 1
--instance-type t2.micro --key-name MyKeyPair --security-group-ids
sg-903004f8 --subnet-id subnet-6e7f829e
```

Think about where you can get the **security-group-id**. (hint: either from the web console or for bonus credit from the command line interface!
hint: `aws ec2 describe-security-groups`).



Permissions Info

By default, Lambda will create an execution role with permissions to upload logs to Amazon CloudWatch Logs. You can customize this default role later when adding triggers.

▼ Change default execution role

Execution role

Choose a role that defines the permissions of your function. To create a custom role, go to the IAM console.

○ Create a new role with basic Lambda permissions

● Use an existing role

○ Create a new role from AWS policy templates

Existing role

Choose an existing role that you've created to be used with this Lambda function. The role must have permission to upload logs to Amazon CloudWatch Logs.

service-role/testfunction-role-np5tat7d

View the testfunction-role-np5tat7d role on the IAM console.

Preparation (SKIP, set by Xianyuan before the lab)

Creating policies

(Please Skip this process, as this is for the lecturer)

First, we will create suitable IAM policies and roles for our lab.

- *Click on **Services**, and find **IAM***
- *Choose **Policies***

List Buckets

- *Choose **Create policy***
- *Choose **s3** in the service*
- *In Filter Actions, choose **List***
- *Tick **ListBucket**, **ListAllMyBuckets***
- *Tick **any** in **bucket***
- *Choose **Next***
- *Enter Policy name: **s3ListBuckets***
- *Choose **Create policy***

Describe Instances

- *Choose **Create policy***
- *Choose **ec2** in the service*
- *In Filter Actions, enter **DescribeInstances***
- *Tick **DescribeInstances***
- *Choose **Next***
- *Enter Policy name: **ec2DescribeInstances***
- *Choose **Create policy***

Create Role for EMR Sevices

- *Choose **Create role***
- *Choose **AWS service**, and in the Use case, choose **EMR**, and click **Next***
- *Keep the default settings, and click **Next***
- *In the Role name, enter **EMR-Sevice-Roles**, and click **Create role***

Create Role for EMR EC2 Instance Profile

- *Choose **Create role***
- *Choose **AWS service**, and in the Use case, choose **EC2**, and click **Next***
- *Add **ec2DescribeInstances** and **s3ListBuckets**, and click **Next***
- *In the Role name, enter **EMR-EC2-InstanceProfile-Roles**, and click **Create role***

Install_Numpy bootstrap actions

- *Go to s3, Create bucket, Name: install-numpy, Create bucket*
- *Click install-numpy, upload install-numpy.sh*

- In install-numpy.sh, the code is

  ```
  #!/bin/bash
  sudo pip3 install numpy
  ```

Upload text data for word counting

- Go to s3, Create bucket, Name: some-texts, Create bucket
- Click some-texts, upload texts.txt
- The texts.txt includes some paragraph found or generated by ChatGPT.

Data downloading

1. Download MovieLens 10M Dataset (ml10m100k) from https://grouplens.org/datasets/movielens/10m/.
2. Upload to s3