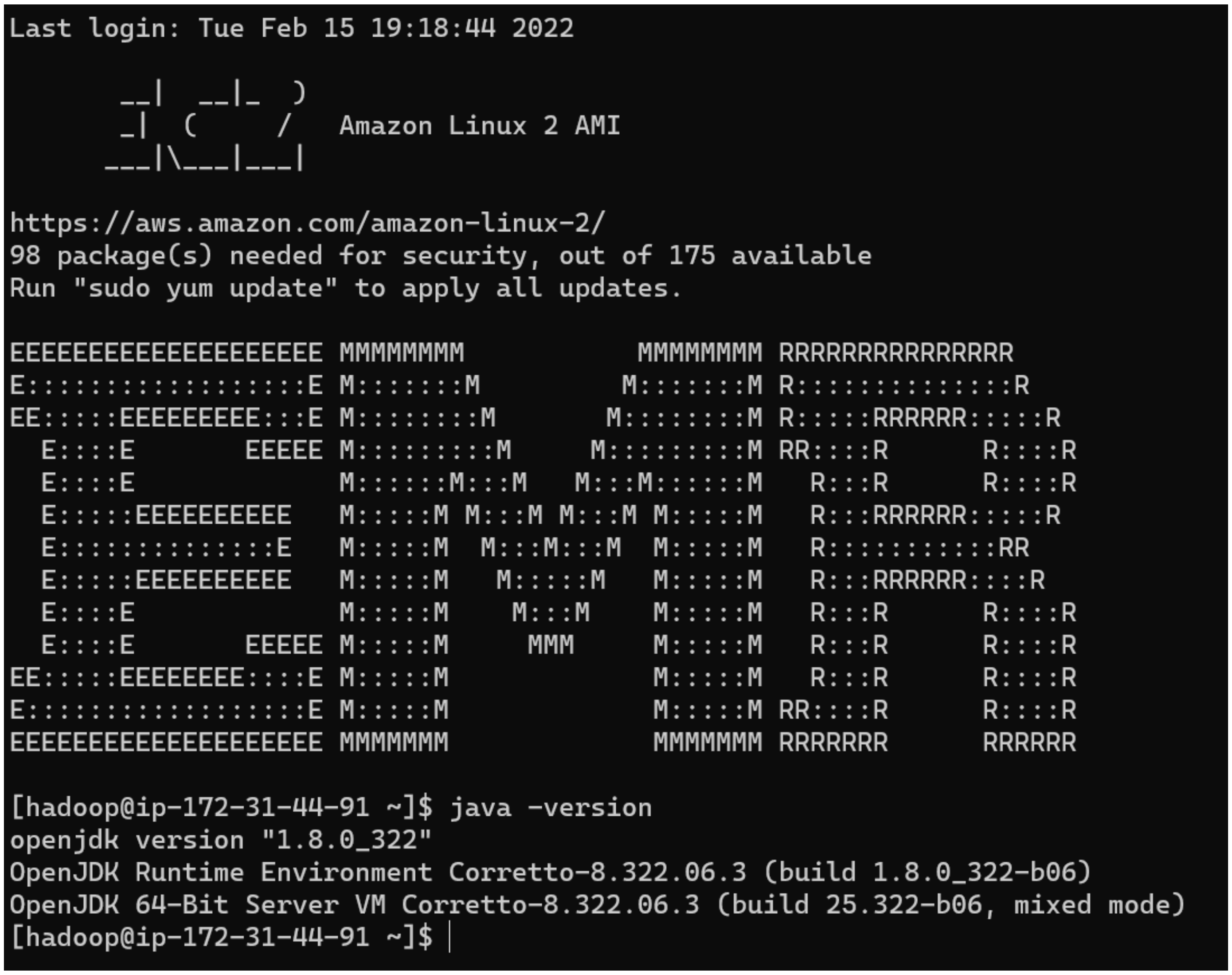
Installing Airflow on EMR

This document contains the steps to install and configure Airflow on your EMR cluster. Make sure that you are using the “hadoop” user while following these instructions.

Setting up Airflow installation scripts

- Make sure that you are using the “hadoop” user while following these instructions.



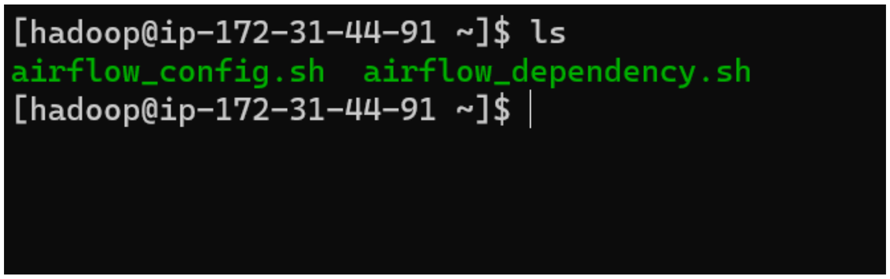
```
Last login: Tue Feb 15 19:18:44 2022

       __|  __|_  )
       _|  (     /   Amazon Linux 2 AMI
      ___|\___|___|

https://aws.amazon.com/amazon-linux-2/
98 package(s) needed for security, out of 175 available
Run "sudo yum update" to apply all updates.

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-172-31-44-91 ~]$ java -version
openjdk version "1.8.0_322"
OpenJDK Runtime Environment Corretto-8.322.06.3 (build 1.8.0_322-b06)
OpenJDK 64-Bit Server VM Corretto-8.322.06.3 (build 25.322-b06, mixed mode)
[hadoop@ip-172-31-44-91 ~]$ 
```

- First, you need to download the Airflow bash scripts by running the following commands.

```
wget
https://airflow-installation.s3.amazonaws.com/airflow-emr/airflow_config.sh
```

```
wget
https://airflow-installation.s3.amazonaws.com/airflow-emr/airflow_dependenc
y.sh
```

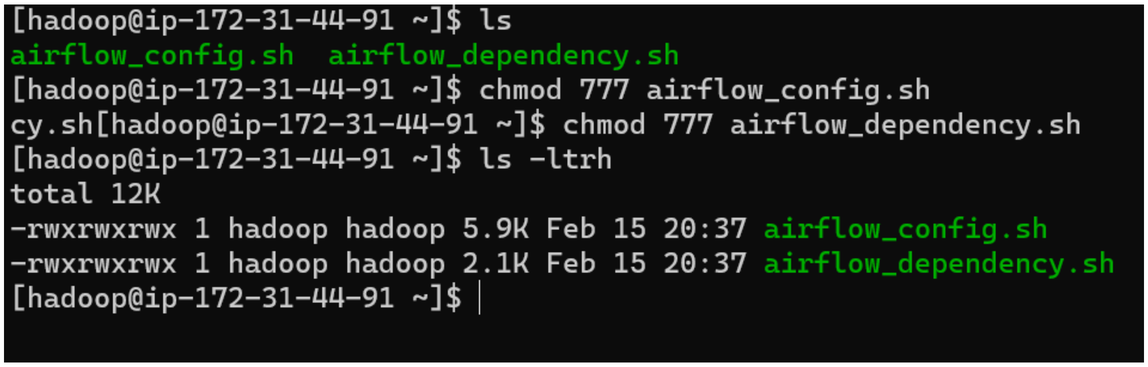
- Use the **ls** command to make sure the bash scripts have been installed correctly.



```
[hadoop@ip-172-31-44-91 ~]$ ls
airflow_config.sh  airflow_dependency.sh
[hadoop@ip-172-31-44-91 ~]$
```

- Now we need to change the permissions of the downloaded file so that we can execute it. Run the following command to do the same:

```
chmod 777 airflow_config.sh
chmod 777 airflow_dependency.sh
ls -ltrh
```



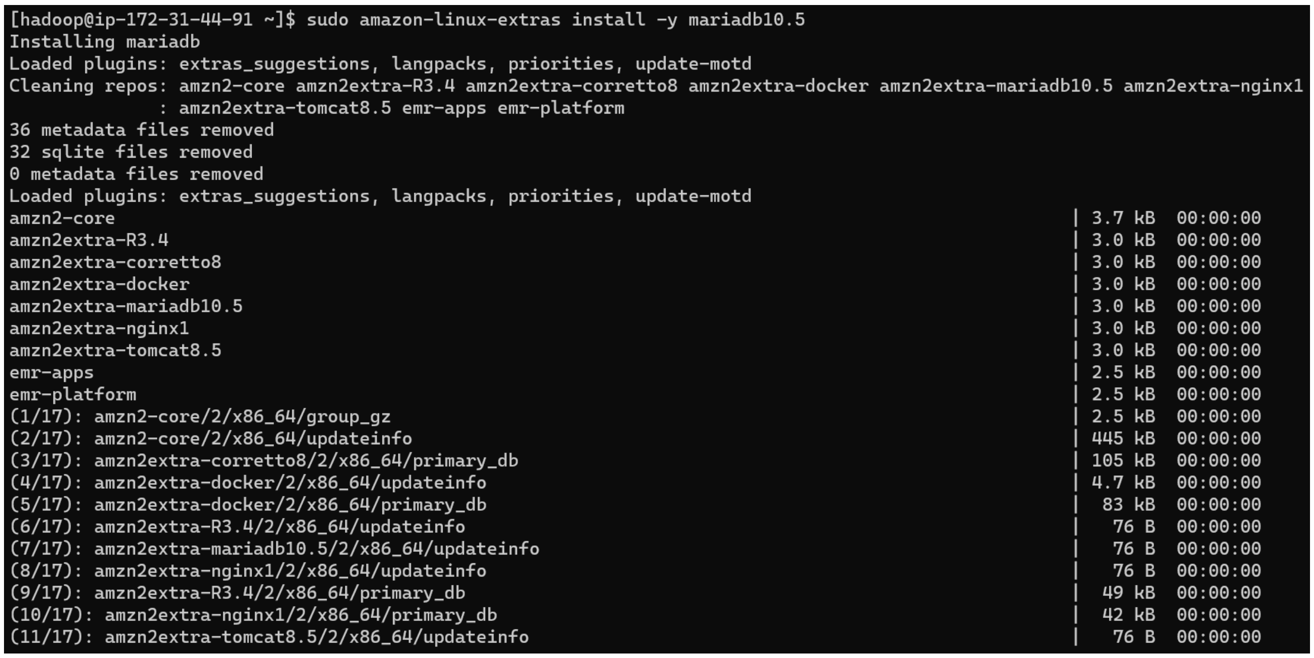
```
[hadoop@ip-172-31-44-91 ~]$ ls
airflow_config.sh  airflow_dependency.sh
[hadoop@ip-172-31-44-91 ~]$ chmod 777 airflow_config.sh
cy.sh[hadoop@ip-172-31-44-91 ~]$ chmod 777 airflow_dependency.sh
[hadoop@ip-172-31-44-91 ~]$ ls -ltrh
total 12K
-rwxrwxrwx 1 hadoop hadoop 5.9K Feb 15 20:37 airflow_config.sh
-rwxrwxrwx 1 hadoop hadoop 2.1K Feb 15 20:37 airflow_dependency.sh
[hadoop@ip-172-31-44-91 ~]$
```

Now before you start executing the Airflow installation scripts, you need to set up MySQL for Airflow. Follow the steps on the next page.

Setting up MySQL for Airflow

- Before you start installing Airflow, you need to update MySQL and set it up. For this, run the following command to update MySQL (For EMR, we use MariaDB and so we will be updating it)

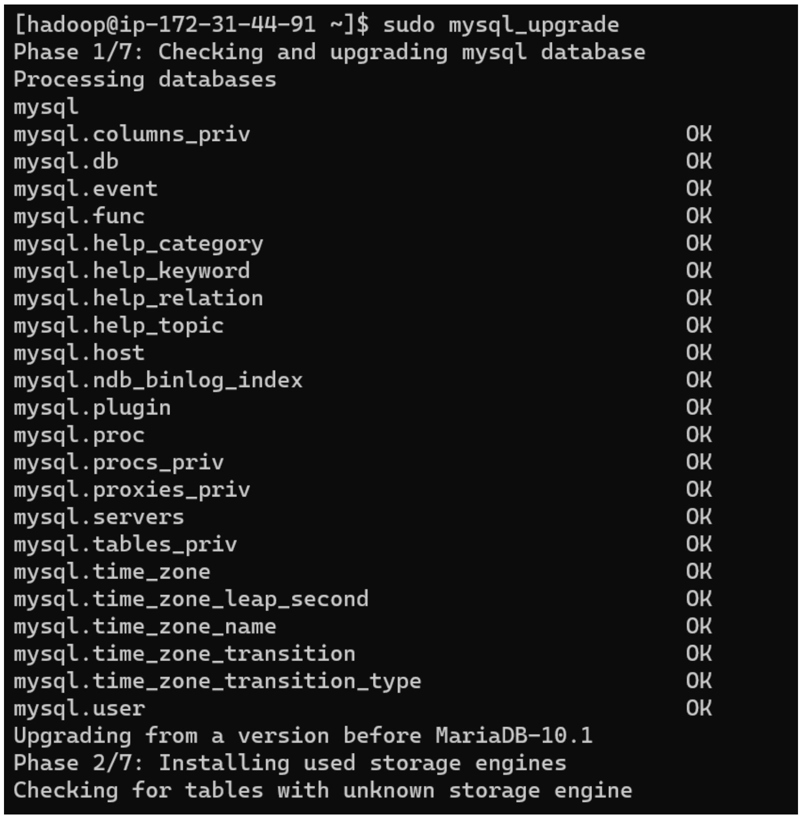
```
sudo amazon-linux-extras install -y mariadb10.5
```



```
[hadoop@ip-172-31-44-91 ~]$ sudo amazon-linux-extras install -y mariadb10.5
Installing mariadb
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
Cleaning repos: amzn2-core amzn2extra-R3.4 amzn2extra-corretto8 amzn2extra-docker amzn2extra-mariadb10.5 amzn2extra-nginx1
              : amzn2extra-tomcat8.5 emr-apps emr-platform
36 metadata files removed
32 sqlite files removed
0 metadata files removed
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
amzn2-core                                                                                     | 3.7 kB  00:00:00
amzn2extra-R3.4                                                                                | 3.0 kB  00:00:00
amzn2extra-corretto8                                                                           | 3.0 kB  00:00:00
amzn2extra-docker                                                                              | 3.0 kB  00:00:00
amzn2extra-mariadb10.5                                                                         | 3.0 kB  00:00:00
amzn2extra-nginx1                                                                              | 3.0 kB  00:00:00
amzn2extra-tomcat8.5                                                                           | 3.0 kB  00:00:00
emr-apps                                                                                       | 2.5 kB  00:00:00
emr-platform                                                                                   | 2.5 kB  00:00:00
(1/17): amzn2-core/2/x86_64/group_gz                                                           | 2.5 kB  00:00:00
(2/17): amzn2-core/2/x86_64/updateinfo                                                         | 445 kB  00:00:00
(3/17): amzn2extra-corretto8/2/x86_64/primary_db                                               | 105 kB  00:00:00
(4/17): amzn2extra-docker/2/x86_64/updateinfo                                                  | 4.7 kB  00:00:00
(5/17): amzn2extra-docker/2/x86_64/primary_db                                                  |  83 kB  00:00:00
(6/17): amzn2extra-R3.4/2/x86_64/updateinfo                                                    |   76 B  00:00:00
(7/17): amzn2extra-mariadb10.5/2/x86_64/updateinfo                                             |   76 B  00:00:00
(8/17): amzn2extra-nginx1/2/x86_64/updateinfo                                                  |   76 B  00:00:00
(9/17): amzn2extra-R3.4/2/x86_64/primary_db                                                    |  49 kB  00:00:00
(10/17): amzn2extra-nginx1/2/x86_64/primary_db                                                 |  42 kB  00:00:00
(11/17): amzn2extra-tomcat8.5/2/x86_64/updateinfo                                              |   76 B  00:00:00
```

- After this, you need to run the following command to configure MySQL tables.

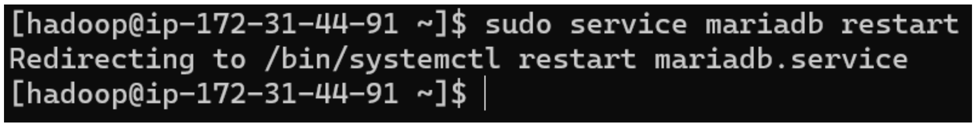
```
sudo mysql_upgrade
```



```
[hadoop@ip-172-31-44-91 ~]$ sudo mysql_upgrade
Phase 1/7: Checking and upgrading mysql database
Processing databases
mysql
mysql.columns_priv                                 OK
mysql.db                                           OK
mysql.event                                        OK
mysql.func                                         OK
mysql.help_category                                OK
mysql.help_keyword                                 OK
mysql.help_relation                                OK
mysql.help_topic                                   OK
mysql.host                                         OK
mysql.ndb_binlog_index                             OK
mysql.plugin                                       OK
mysql.proc                                         OK
mysql.procs_priv                                   OK
mysql.proxies_priv                                 OK
mysql.servers                                      OK
mysql.tables_priv                                  OK
mysql.time_zone                                    OK
mysql.time_zone_leap_second                        OK
mysql.time_zone_name                               OK
mysql.time_zone_transition                         OK
mysql.time_zone_transition_type                    OK
mysql.user                                         OK
Upgrading from a version before MariaDB-10.1
Phase 2/7: Installing used storage engines
Checking for tables with unknown storage engine
```

- Run the following command to restart the **mariadb** service.

```
sudo service mariadb restart
```



```
[hadoop@ip-172-31-44-91 ~]$ sudo service mariadb restart
Redirecting to /bin/systemctl restart mariadb.service
[hadoop@ip-172-31-44-91 ~]$ 
```

- Now for the final step for MySQL setup, run the following command to set up MySQL.

```
mysql_secure_installation
```

After running this command, you will have to go through several prompts where you have to answer in **Y** or **n.** There will also be a prompt in between where you will need to set up your root password. The prompts and the answer to these questions(in Italics) will be in this order:

```
Enter current password for root (enter for none):
```

Press Enter without entering anything

```
Enable unix_socket authentication? [Y/n]
```
*Type **n** and press Enter*

```
Set root password? [Y/n]
```
*Type **Y** and press Enter*

```
New password:
```
*Type **123** and press Enter*

```
Re-enter new password:
```
*Type **123** and press Enter*

```
Remove anonymous users? [Y/n]
```
*Type **Y** and press Enter*

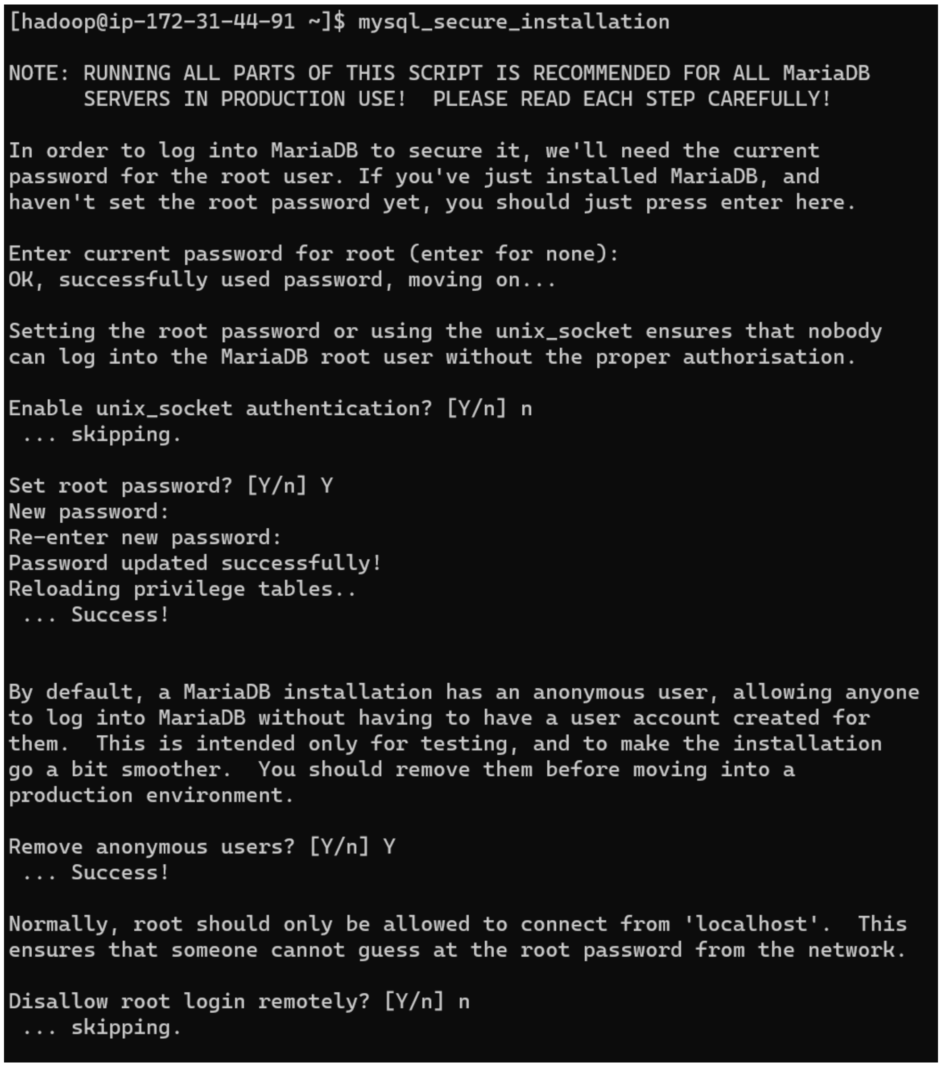
```
Disallow root login remotely? [Y/n]
```
*Type **n** and press Enter*

```
Remove test database and access to it? [Y/n]
```
*Type **Y** and press Enter*

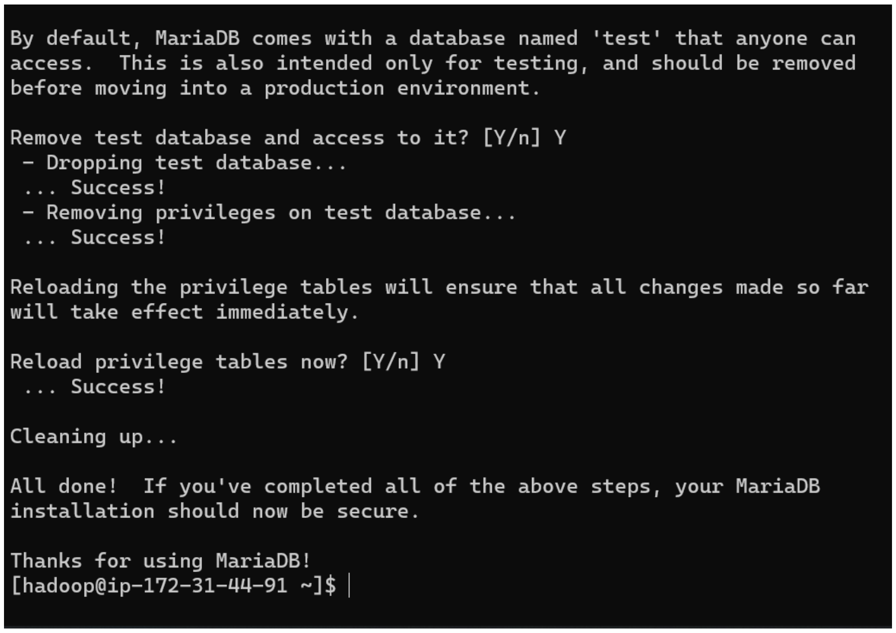
```
Reload privilege tables now? [Y/n]
```
*Type **Y** and press Enter*

With this you have setup MySQL for your EMR cluster. Now you can continue to setup Airflow.

The screenshots of the **mysql_secure_installation** command are present below for your reference.



```
[hadoop@ip-172-31-44-91 ~]$ mysql_secure_installation

NOTE: RUNNING ALL PARTS OF THIS SCRIPT IS RECOMMENDED FOR ALL MariaDB
      SERVERS IN PRODUCTION USE!  PLEASE READ EACH STEP CAREFULLY!

In order to log into MariaDB to secure it, we'll need the current
password for the root user. If you've just installed MariaDB, and
haven't set the root password yet, you should just press enter here.

Enter current password for root (enter for none):
OK, successfully used password, moving on...

Setting the root password or using the unix_socket ensures that nobody
can log into the MariaDB root user without the proper authorisation.

Enable unix_socket authentication? [Y/n] n
 ... skipping.

Set root password? [Y/n] Y
New password:
Re-enter new password:
Password updated successfully!
Reloading privilege tables..
 ... Success!


By default, a MariaDB installation has an anonymous user, allowing anyone
to log into MariaDB without having to have a user account created for
them.  This is intended only for testing, and to make the installation
go a bit smoother.  You should remove them before moving into a
production environment.

Remove anonymous users? [Y/n] Y
 ... Success!

Normally, root should only be allowed to connect from 'localhost'.  This
ensures that someone cannot guess at the root password from the network.

Disallow root login remotely? [Y/n] n
 ... skipping.
```



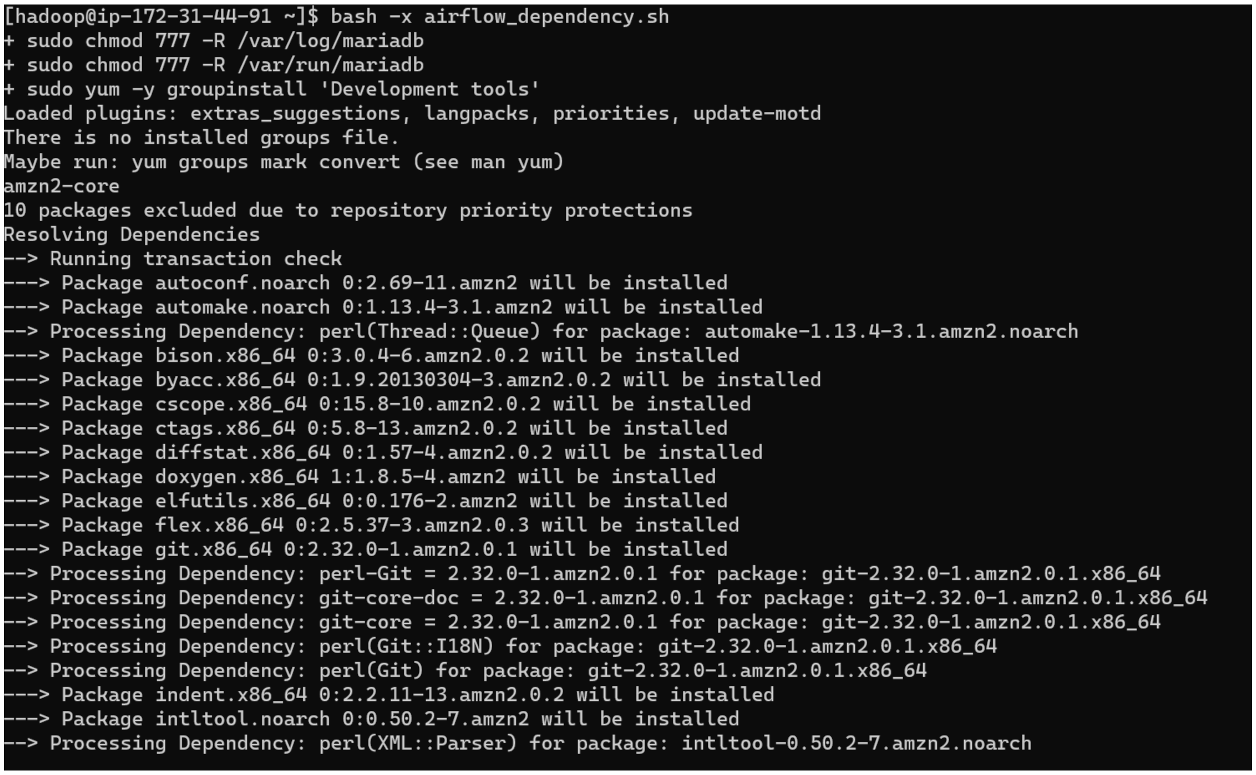
```
By default, MariaDB comes with a database named 'test' that anyone can
access.  This is also intended only for testing, and should be removed
before moving into a production environment.

Remove test database and access to it? [Y/n] Y
 - Dropping test database...
 ... Success!
 - Removing privileges on test database...
 ... Success!

Reloading the privilege tables will ensure that all changes made so far
will take effect immediately.

Reload privilege tables now? [Y/n] Y
 ... Success!

Cleaning up...

All done!  If you've completed all of the above steps, your MariaDB
installation should now be secure.

Thanks for using MariaDB!
[hadoop@ip-172-31-44-91 ~]$
```

Installing Airflow

- Now you can start executing the Airflow scripts. You need to first run the airflow_dependecy.sh script with the following command.

```
bash -x airflow_dependency.sh
```

(This may take around 10-15 minutes so please wait patiently)

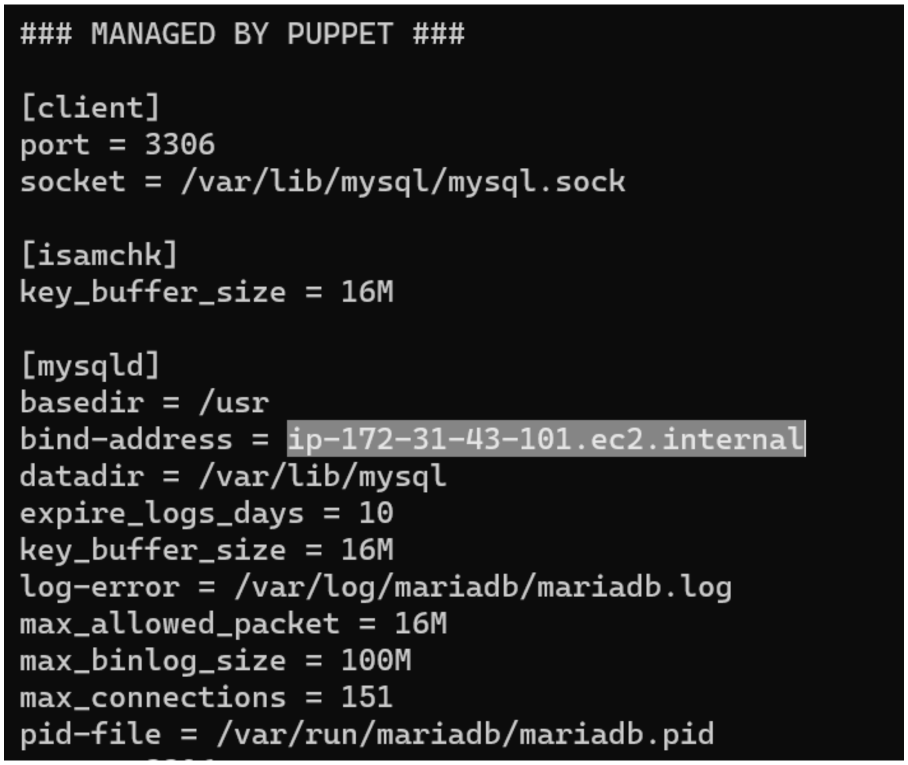


```
[hadoop@ip-172-31-44-91 ~]$ bash -x airflow_dependency.sh
+ sudo chmod 777 -R /var/log/mariadb
+ sudo chmod 777 -R /var/run/mariadb
+ sudo yum -y groupinstall 'Development tools'
Loaded plugins: extras_suggestions, langpacks, priorities, update-motd
There is no installed groups file.
Maybe run: yum groups mark convert (see man yum)
amzn2-core
10 packages excluded due to repository priority protections
Resolving Dependencies
--> Running transaction check
---> Package autoconf.noarch 0:2.69-11.amzn2 will be installed
---> Package automake.noarch 0:1.13.4-3.1.amzn2 will be installed
--> Processing Dependency: perl(Thread::Queue) for package: automake-1.13.4-3.1.amzn2.noarch
---> Package bison.x86_64 0:3.0.4-6.amzn2.0.2 will be installed
---> Package byacc.x86_64 0:1.9.20130304-3.amzn2.0.2 will be installed
---> Package cscope.x86_64 0:15.8-10.amzn2.0.2 will be installed
---> Package ctags.x86_64 0:5.8-13.amzn2.0.2 will be installed
---> Package diffstat.x86_64 0:1.57-4.amzn2.0.2 will be installed
---> Package doxygen.x86_64 1:1.8.5-4.amzn2 will be installed
---> Package elfutils.x86_64 0:0.176-2.amzn2 will be installed
---> Package flex.x86_64 0:2.5.37-3.amzn2.0.3 will be installed
---> Package git.x86_64 0:2.32.0-1.amzn2.0.1 will be installed
--> Processing Dependency: perl-Git = 2.32.0-1.amzn2.0.1 for package: git-2.32.0-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core-doc = 2.32.0-1.amzn2.0.1 for package: git-2.32.0-1.amzn2.0.1.x86_64
--> Processing Dependency: git-core = 2.32.0-1.amzn2.0.1 for package: git-2.32.0-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git::I18N) for package: git-2.32.0-1.amzn2.0.1.x86_64
--> Processing Dependency: perl(Git) for package: git-2.32.0-1.amzn2.0.1.x86_64
---> Package indent.x86_64 0:2.2.11-13.amzn2.0.2 will be installed
---> Package intltool.noarch 0:0.50.2-7.amzn2 will be installed
--> Processing Dependency: perl(XML::Parser) for package: intltool-0.50.2-7.amzn2.noarch
```

- Now you need to update the **my.cnf** file. For this run the following command:

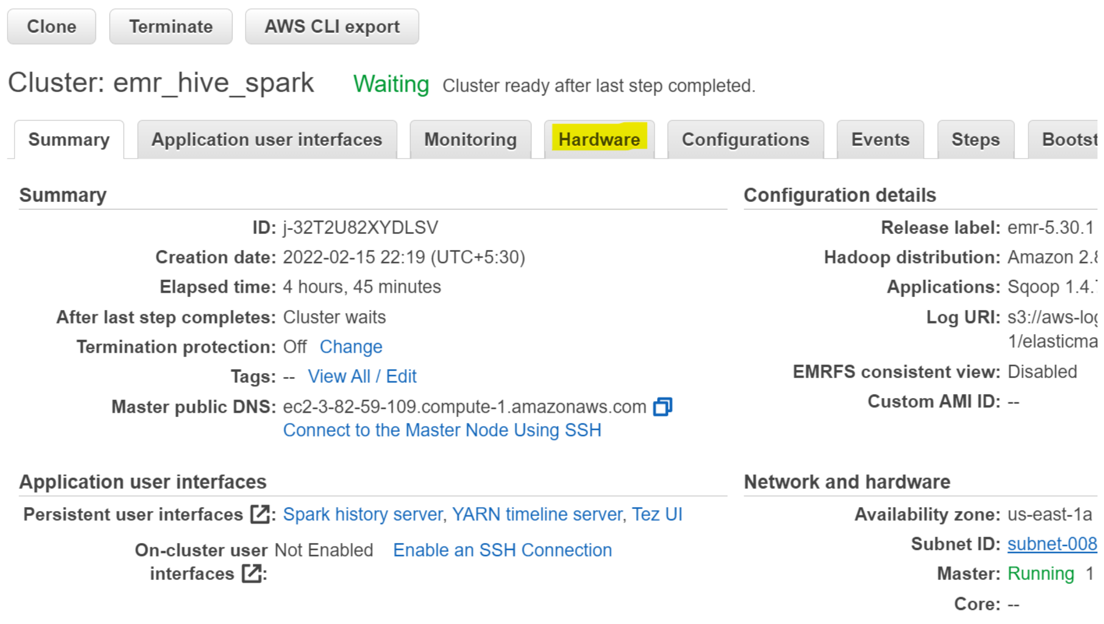
```
sudo vi /etc/my.cnf
```

You need to replace the **bind-address** in the screenshot below with the **private DNS** for your own EMR cluster. Enter **i** to go into INPUT mode for **vi** and then erase the bind-address.



```
### MANAGED BY PUPPET ###

[client]
port = 3306
socket = /var/lib/mysql/mysql.sock

[isamchk]
key_buffer_size = 16M

[mysqld]
basedir = /usr
bind-address = ip-172-31-43-101.ec2.internal
datadir = /var/lib/mysql
expire_logs_days = 10
key_buffer_size = 16M
log-error = /var/log/mariadb/mariadb.log
max_allowed_packet = 16M
max_binlog_size = 100M
max_connections = 151
pid-file = /var/run/mariadb/mariadb.pid
```

To find the private DNS for your EMR cluster, open your EMR cluster console. Click on the **Hardware** tab.



Cluster: emr_hive_spark Waiting Cluster ready after last step completed.

Summary

ID:	j-32T2U82XYDLSV
Creation date:	2022-02-15 22:19 (UTC+5:30)
Elapsed time:	4 hours, 45 minutes
After last step completes:	Cluster waits
Termination protection:	Off Change
Tags:	-- View All / Edit
Master public DNS:	ec2-3-82-59-109.compute-1.amazonaws.com Connect to the Master Node Using SSH

Configuration details

Release label:	emr-5.30.1
Hadoop distribution:	Amazon 2.8
Applications:	Sqoop 1.4.7
Log URI:	s3://aws-log 1/elasticma
EMRFS consistent view:	Disabled
Custom AMI ID:	--

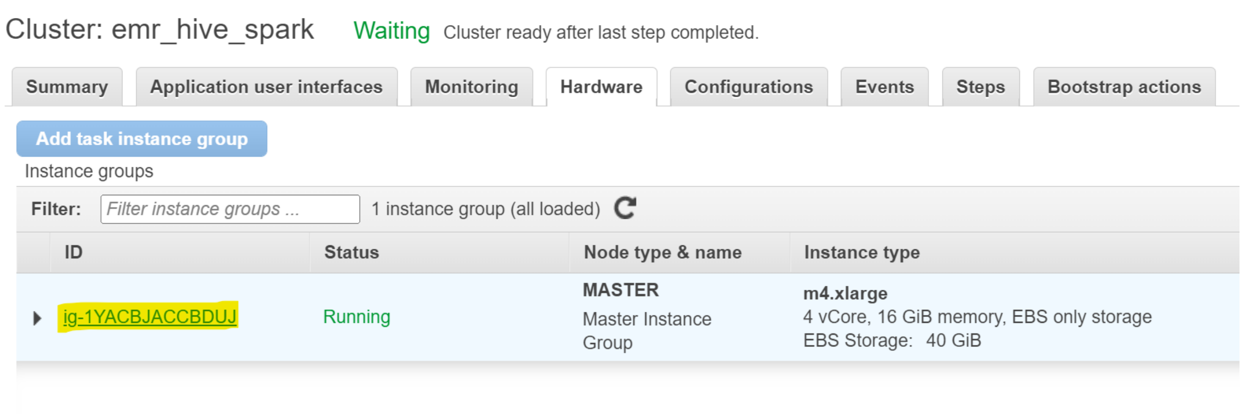
Application user interfaces

Persistent user interfaces:	Spark history server, YARN timeline server, Tez UI
On-cluster user interfaces:	Not Enabled Enable an SSH Connection

Network and hardware

Availability zone:	us-east-1a
Subnet ID:	subnet-008
Master:	Running 1
Core:	--

Click on the instance ID highlighted below.



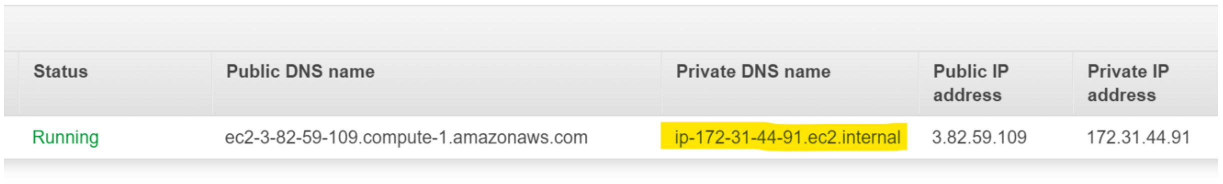
Cluster: emr_hive_spark Waiting Cluster ready after last step completed.

Instance groups

Filter: Filter instance groups ... 1 instance group (all loaded)

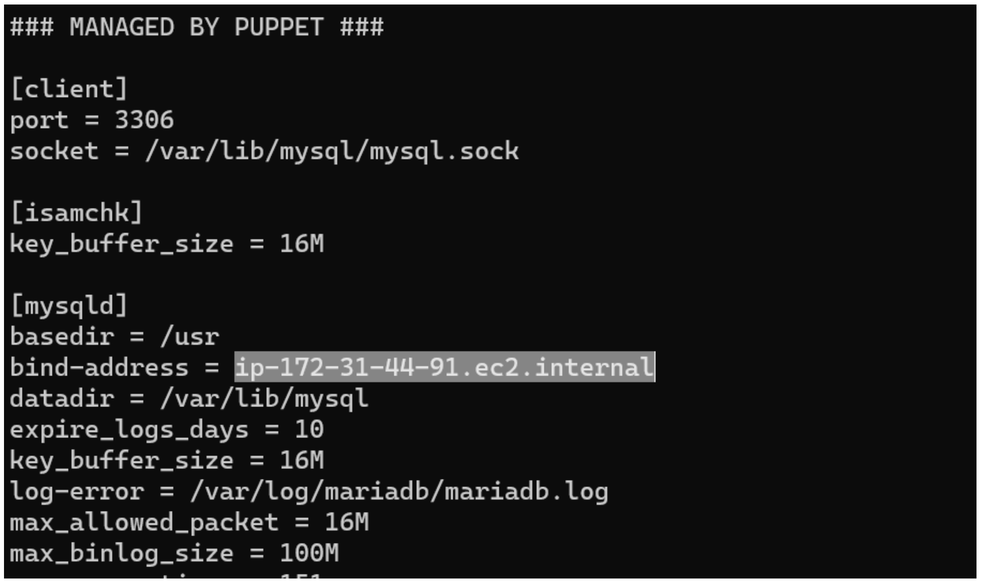
ID	Status	Node type & name	Instance type
ig-1YACBJACCBDUJ	Running	MASTER Master Instance Group	m4.xlarge 4 vCore, 16 GiB memory, EBS only storage EBS Storage: 40 GiB

In the following screen, scroll to the right and there you will see your private IP and private DNS. Copy the private DNS name from here with the **Ctrl + C** keyboard shortcut. You can also manually type it.



Status	Public DNS name	Private DNS name	Public IP address	Private IP address
Running	ec2-3-82-59-109.compute-1.amazonaws.com	ip-172-31-44-91.ec2.internal	3.82.59.109	172.31.44.91

Replace bind-address with your private DNS name by clicking on right click button or manually typing it. After this, click on **Esc** and then type **:wq** to save the file.

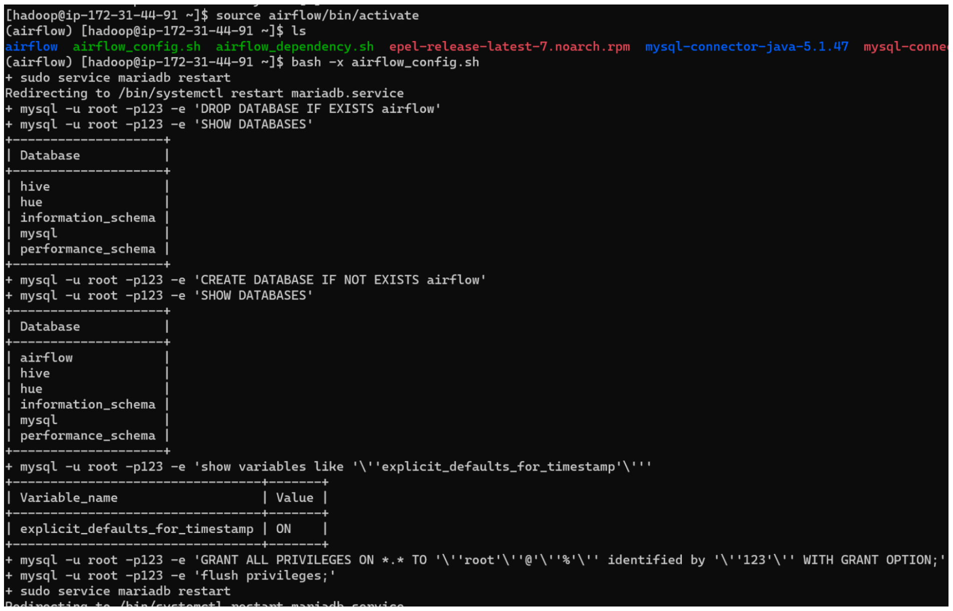


```
### MANAGED BY PUPPET ###

[client]
port = 3306
socket = /var/lib/mysql/mysql.sock

[isamchk]
key_buffer_size = 16M

[mysqld]
basedir = /usr
bind-address = ip-172-31-44-91.ec2.internal
datadir = /var/lib/mysql
expire_logs_days = 10
key_buffer_size = 16M
log-error = /var/log/mariadb/mariadb.log
max_allowed_packet = 16M
max_binlog_size = 100M
```

- Now you can execute the 2nd airflow bash script. You need to first go into the airflow python environment and then execute the airflow_dependecy.sh script with the following commands.

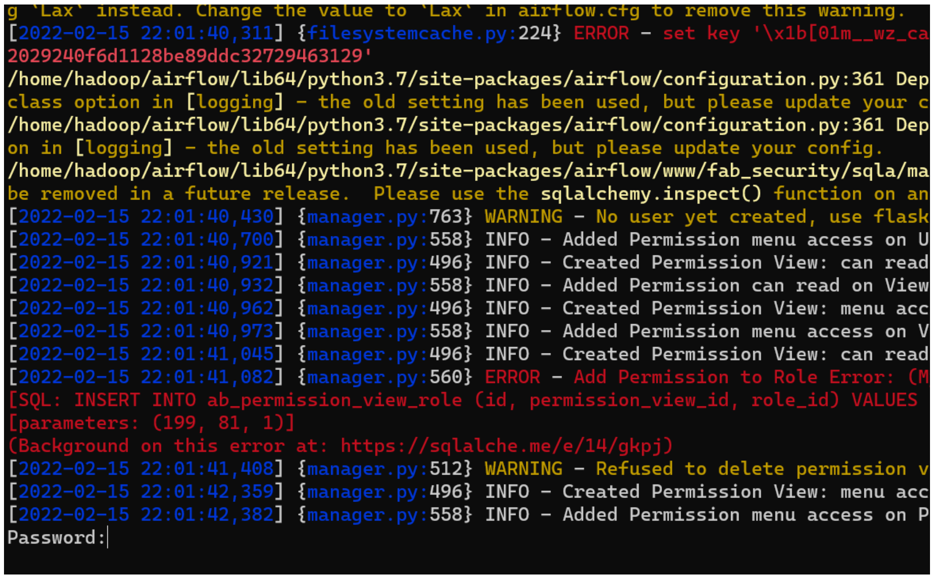
```
source airflow/bin/activate
bash -x airflow_config.sh
```

(This may take around 10-15 minutes so please wait patiently)



```
[hadoop@ip-172-31-44-91 ~]$ source airflow/bin/activate
(airflow) [hadoop@ip-172-31-44-91 ~]$ ls
airflow  airflow_config.sh  airflow_dependency.sh  epel-release-latest-7.noarch.rpm  mysql-connector-java-5.1.47  mysql-conne
(airflow) [hadoop@ip-172-31-44-91 ~]$ bash -x airflow_config.sh
+ sudo service mariadb restart
Redirecting to /bin/systemctl restart mariadb.service
+ mysql -u root -p123 -e 'DROP DATABASE IF EXISTS airflow'
+ mysql -u root -p123 -e 'SHOW DATABASES'
+--------------------+
| Database           |
+--------------------+
| hive               |
| hue                |
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
+ mysql -u root -p123 -e 'CREATE DATABASE IF NOT EXISTS airflow'
+ mysql -u root -p123 -e 'SHOW DATABASES'
+--------------------+
| Database           |
+--------------------+
| airflow            |
| hive               |
| hue                |
| information_schema |
| mysql              |
| performance_schema |
+--------------------+
+ mysql -u root -p123 -e 'show variables like '\''explicit_defaults_for_timestamp'\'''
+---------------------------------+-------+
| Variable_name                   | Value |
+---------------------------------+-------+
| explicit_defaults_for_timestamp | ON    |
+---------------------------------+-------+
+ mysql -u root -p123 -e 'GRANT ALL PRIVILEGES ON *.* TO '\''root'\''@'\''%'\'' identified by '\''123'\'' WITH GRANT OPTION;'
+ mysql -u root -p123 -e 'flush privileges;'
+ sudo service mariadb restart
Redirecting to /bin/systemctl restart mariadb.service
```

You will be prompted to enter the password for the Airflow admin user. Type **admin** here and press Enter and then type **admin** again and then press Enter to confirm your password.



```
g `Lax` instead. Change the value to `Lax` in airflow.cfg to remove this warning.
[2022-02-15 22:01:40,311] {filesystemcache.py:224} ERROR - set key '\x1b[01m__wz_ca
2029240f6d1128be89ddc32729463129'
/home/hadoop/airflow/lib64/python3.7/site-packages/airflow/configuration.py:361 Dep
class option in [logging] - the old setting has been used, but please update your c
/home/hadoop/airflow/lib64/python3.7/site-packages/airflow/configuration.py:361 Dep
on in [logging] - the old setting has been used, but please update your config.
/home/hadoop/airflow/lib64/python3.7/site-packages/airflow/www/fab_security/sqla/ma
be removed in a future release.  Please use the sqlalchemy.inspect() function on an
[2022-02-15 22:01:40,430] {manager.py:763} WARNING - No user yet created, use flask
[2022-02-15 22:01:40,700] {manager.py:558} INFO - Added Permission menu access on U
[2022-02-15 22:01:40,921] {manager.py:496} INFO - Created Permission View: can read
[2022-02-15 22:01:40,932] {manager.py:558} INFO - Added Permission can read on View
[2022-02-15 22:01:40,962] {manager.py:496} INFO - Created Permission View: menu acc
[2022-02-15 22:01:40,973] {manager.py:558} INFO - Added Permission menu access on V
[2022-02-15 22:01:41,045] {manager.py:496} INFO - Created Permission View: can read
[2022-02-15 22:01:41,082] {manager.py:560} ERROR - Add Permission to Role Error: (M
[SQL: INSERT INTO ab_permission_view_role (id, permission_view_id, role_id) VALUES
[parameters: (199, 81, 1)]
(Background on this error at: https://sqlalche.me/e/14/gkpj)
[2022-02-15 22:01:41,408] {manager.py:512} WARNING - Refused to delete permission v
[2022-02-15 22:01:42,359] {manager.py:496} INFO - Created Permission View: menu acc
[2022-02-15 22:01:42,382] {manager.py:558} INFO - Added Permission menu access on P
Password:
```

With this, you have successfully installed Airflow on your EMR cluster.

To check if you have installed Airflow, you can now try and open the Airflow Webserver.

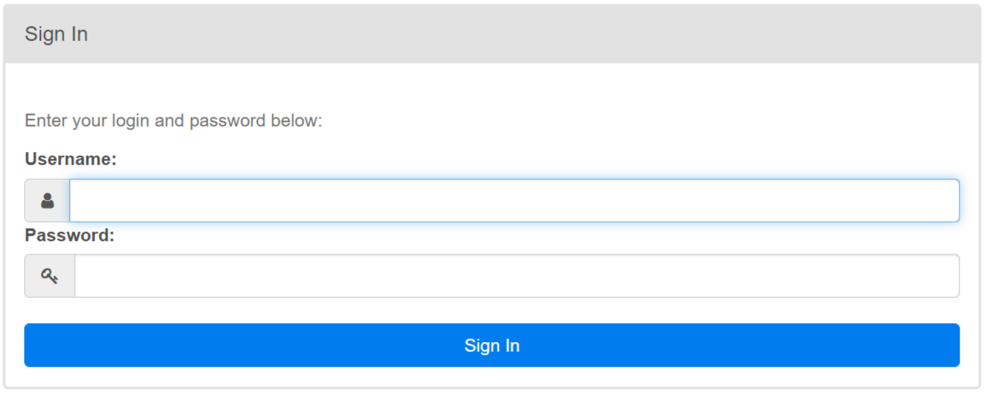
Note: Make sure that you have opened the 8082 port for your EMR cluster security group. You would have followed this step while setting up your EMR cluster.

For this, copy the public DNS for your EMR cluster along with the port 8082 such as the following:

<public DNS>:8082

For example: **ec2-3-82-59-109.compute-1.amazonaws.com:8082**

You can now paste this on your web browser and open the Airflow Webserver UI.

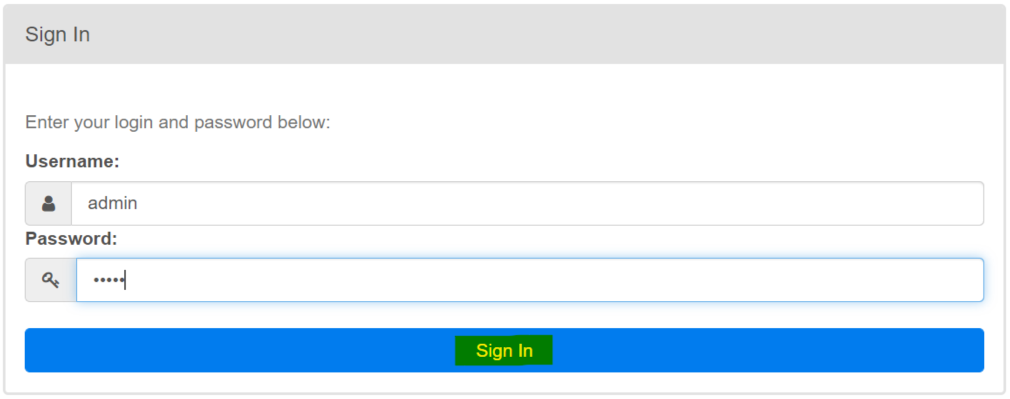

Sign In

Enter your login and password below:

Username:

Password:

You can sign in to this page with username and password as **admin** and **admin**. Click on Sign in after entering the username and password.

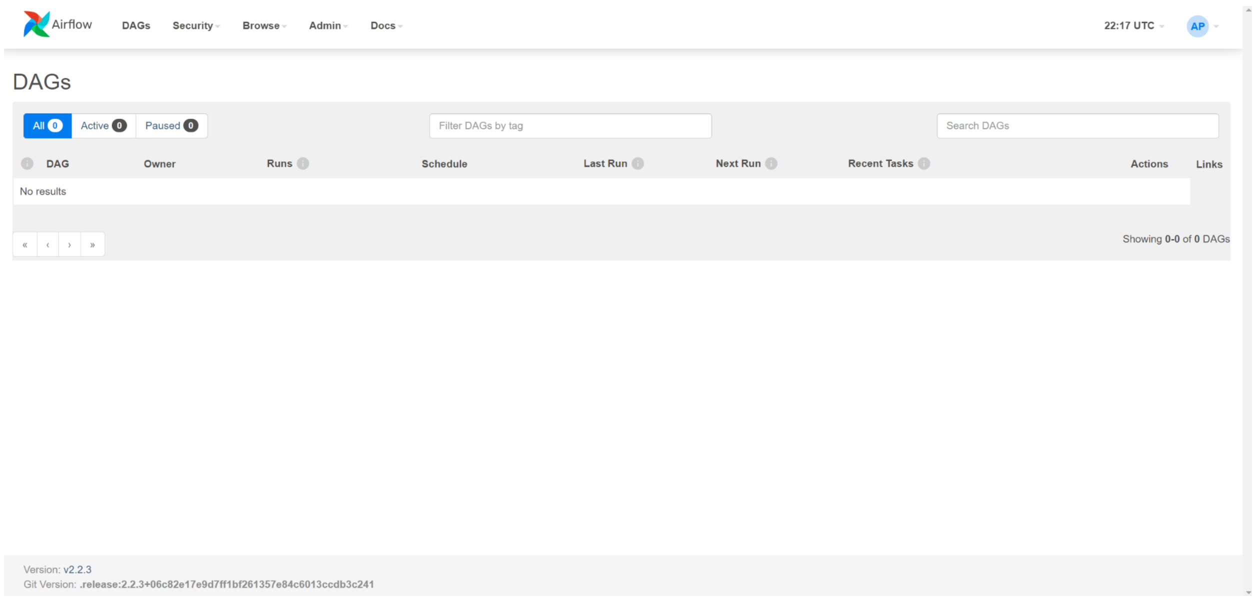

Sign In

Enter your login and password below:

Username: admin

Password: •••••

You should see the following screen.



No results

Showing 0-0 of 0 DAGs

Version: v2.2.3
Git Version: .release:2.2.3+06c82e17e9d7ff1bf261357e84c6013ccdb3c241

Now, you can go ahead and start working with Airflow DAGs.

Note: By default, after this installation, Java 11 will be activated on your EMR cluster. If you want to run Hive and Spark operators then you will have to switch back to Java 8 by running the following command:

```
sudo alternatives --config java <<< 1
```