Docker and Kubernetes

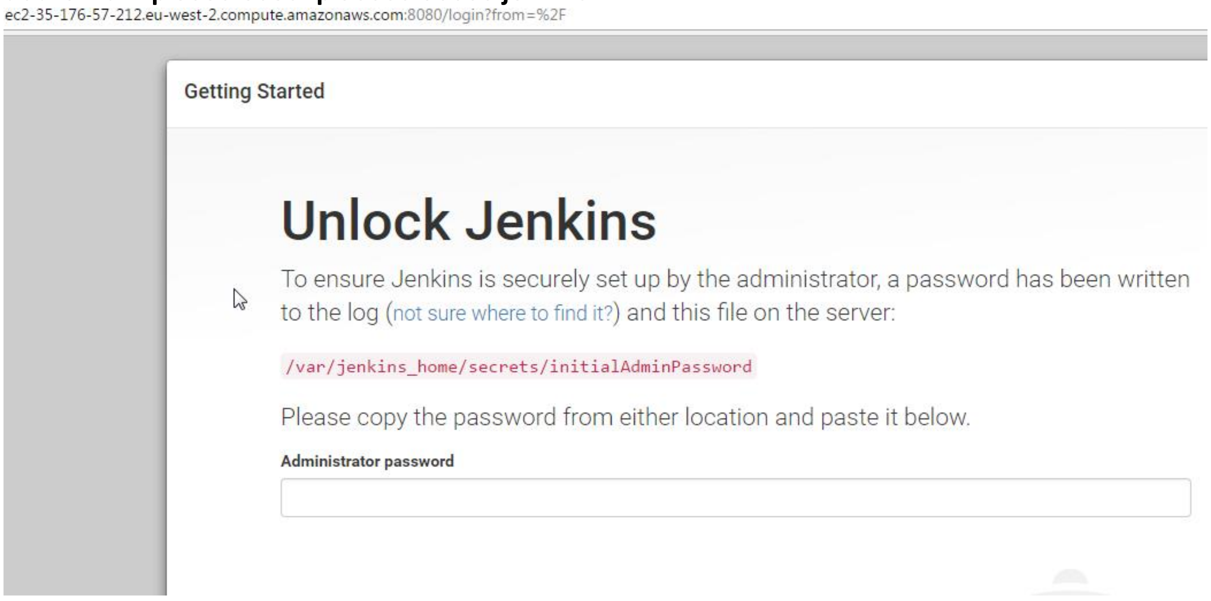
Task 1 – Install docker

Get the script file to install docker and its dependencies from docker website and pipe it to the shell.

wget -qO- https://get.docker.com/ | sh

Task 2 – Deploy a Jenkins Container

Pull a Jenkins image
docker pull Jenkins
Run Jenkins in a container
docker run -p 8080:8080 -p 50000:50000 jenkins


ec2-35-176-57-212.eu-west-2.compute.amazonaws.com:8080/login?from=%2F

Getting Started

Unlock Jenkins

To ensure Jenkins is securely set up by the administrator, a password has been written to the log (not sure where to find it?) and this file on the server:

```
/var/jenkins_home/secrets/initialAdminPassword
```

Please copy the password from either location and paste it below.

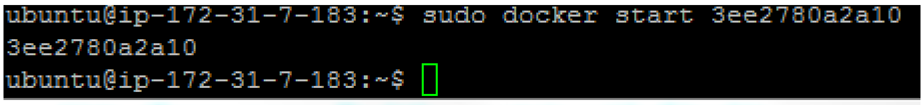
Administrator password

A container of Jenkins has been created, use ctrl+z to get out of the terminal.

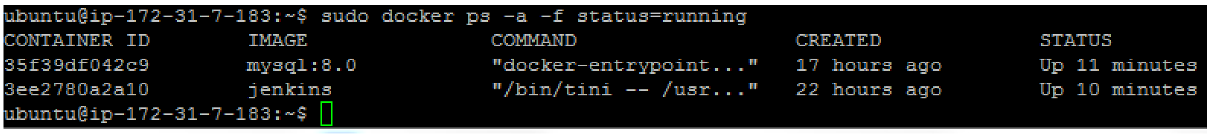
sudo docker ps –a

Lists all processes, find the container id of Jenkins

sudo docker start containerID



```
ubuntu@ip-172-31-7-183:~$ sudo docker start 3ee2780a2a10
3ee2780a2a10
ubuntu@ip-172-31-7-183:~$
```



```
ubuntu@ip-172-31-7-183:~$ sudo docker ps -a -f status=running
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS
35f39df042c9        mysql:8.0           "docker-entrypoint..."   17 hours ago        Up 11 minutes
3ee2780a2a10        jenkins             "/bin/tini -- /usr..."   22 hours ago        Up 10 minutes
ubuntu@ip-172-31-7-183:~$
```

Task 3 – Create a dockerfile

Create dockerfile using the following contents:

```
FROM ubuntu:16.04

#Always update your running system
RUN sudo apt-get update -y

#You may or may not need to run these commands
RUN sudo apt-get install -y wget
RUN sudo apt-get install -y tar

#installs the libraries needed to run the GUI
RUN sudo apt-get install -y libgtk2.0
RUN sudo apt-get install -y mesa-utils
RUN sudo apt-get install -y libXtst6

#RUN sudo apt-get install -y openjdk-7-jre
#RUN java -version

#Now install the Java Compiler
#RUN sudo apt-get install -y openjdk-7-jdk
#RUN javac -version

#Add java from file and install
WORKDIR /opt

ADD files /opt

RUN sudo tar zxvf /opt/java.tar.gz

RUN sudo update-alternatives --install /usr/bin/java java /opt/jdk1.8.0_74/bin/java 100
RUN sudo update-alternatives --install /usr/bin/javac javac /opt/jdk1.8.0_74/bin/javac 100
```

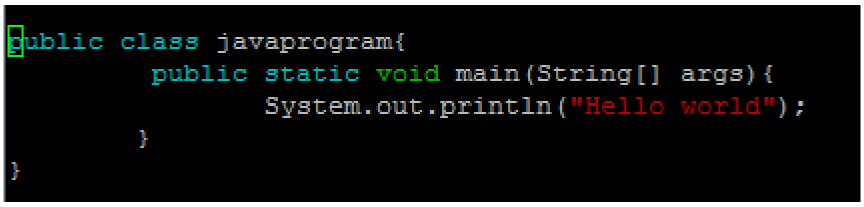
```
# Install OpenJDK-8
RUN apt-get update && \
    apt-get install -y openjdk-8-jdk && \
    apt-get install -y ant && \
    apt-get clean;

# Fix certificate issues
RUN apt-get update && \
    apt-get install ca-certificates-java && \
    apt-get clean && \
    update-ca-certificates -f;

# Setup JAVA_HOME -- useful for docker commandline
ENV JAVA_HOME /usr/lib/jvm/java-8-openjdk-amd64/
RUN export JAVA_HOME
```

Run the file using the following command.

docker build -t [imagename] .

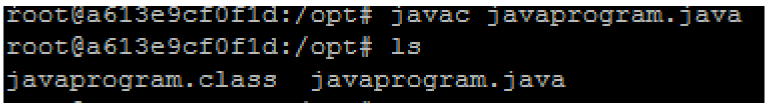
Note: the dot at the end of command must be present; imagename is the name of the image to be run e.g. Ubuntu:16.04

Create java program

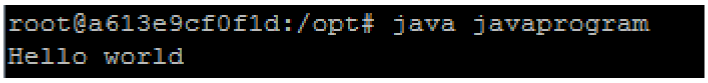


```
public class javaprogram{
         public static void main(String[] args){
                System.out.println("Hello world");
        }
}
```

Compile and run it.



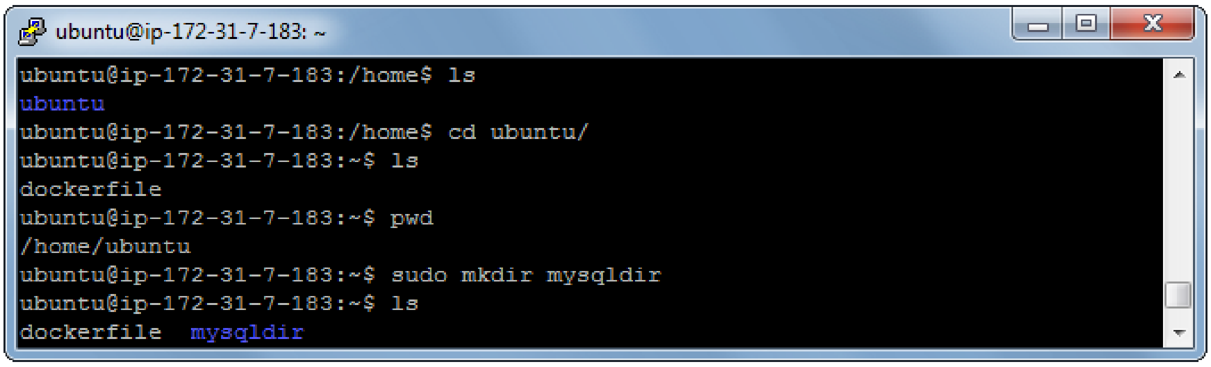
```
root@a613e9cf0f1d:/opt# javac javaprogram.java
root@a613e9cf0f1d:/opt# ls
javaprogram.class  javaprogram.java
```



```
root@a613e9cf0f1d:/opt# java javaprogram
Hello world
```

Task 4 – Create your own linked container

To create mysql container that stores data in the host volume, create a directory



```
ubuntu@ip-172-31-7-183:/home$ ls
ubuntu
ubuntu@ip-172-31-7-183:/home$ cd ubuntu/
ubuntu@ip-172-31-7-183:~$ ls
dockerfile
ubuntu@ip-172-31-7-183:~$ pwd
/home/ubuntu
ubuntu@ip-172-31-7-183:~$ sudo mkdir mysqldir
ubuntu@ip-172-31-7-183:~$ ls
dockerfile  mysqldir
```

docker run --name some-mysql -v /home/ubuntu/mysqldir:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=my-secret-pw -d mysql:tag

where **my-secret-pw** is password, **tag** is version number e.g. 8.0

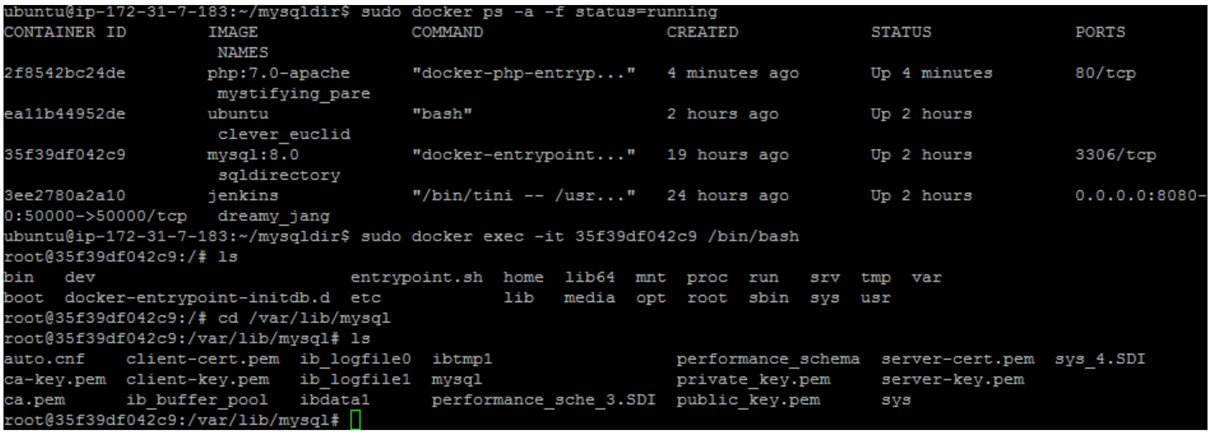
e.g. **docker run --name some-mysql -v /home/ubuntu/mysqldir:/var/lib/mysql -e MYSQL_ROOT_PASSWORD=abcde -d mysql:8.0**

Find running process with

sudo docker ps -a -f status=running

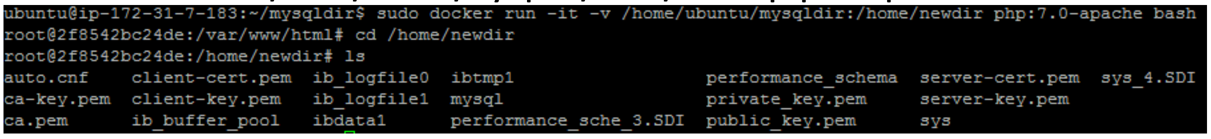
Execute interactive terminal –it inside a container using exec and containerID

sudo docker exec -it containerID /bin/bash



```
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker ps -a -f status=running
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
                     NAMES
2f8542bc24de        php:7.0-apache      "docker-php-entryp..."   4 minutes ago       Up 4 minutes        80/tcp
                     mystifying_pare
ea11b44952de        ubuntu              "bash"                   2 hours ago         Up 2 hours
                     clever_euclid
35f39df042c9        mysql:8.0           "docker-entrypoint..."   19 hours ago        Up 2 hours          3306/tcp
                     sqldirectory
3ee2780a2a10        jenkins             "/bin/tini -- /usr..."   24 hours ago        Up 2 hours          0.0.0.0:8080-
0:50000->50000/tcp   dreamy_jang
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker exec -it 35f39df042c9 /bin/bash
root@35f39df042c9:/# ls
bin   dev                         entrypoint.sh  home  lib64  mnt  proc  run   srv  tmp  var
boot  docker-entrypoint-initdb.d  etc            lib   media  opt  root  sbin  sys  usr
root@35f39df042c9:/# cd /var/lib/mysql
root@35f39df042c9:/var/lib/mysql# ls
auto.cnf    client-cert.pem  ib_logfile0  ibtmp1                  performance_schema  server-cert.pem  sys_4.SDI
ca-key.pem  client-key.pem   ib_logfile1  mysql                   private_key.pem     server-key.pem
ca.pem      ib_buffer_pool   ibdata1      performance_sche_3.SDI  public_key.pem      sys
root@35f39df042c9:/var/lib/mysql#
```

For PHP project, pull php image:
sudo docker pull php
When running the container mount the directory to a newdirectory in php file path
sudo docker run -it -v /home/ubuntu/mysqldir:/home/newdir php:7.0-apache bash

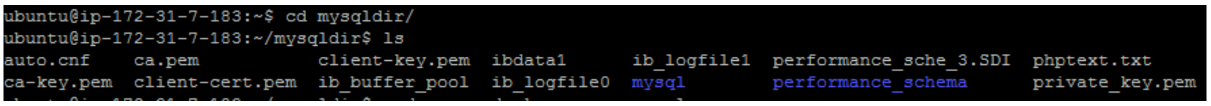


```
ubuntu@ip-172-31-7-183:~/mysqldir$ sudo docker run -it -v /home/ubuntu/mysqldir:/home/newdir php:7.0-apache bash
root@2f8542bc24de:/var/www/html# cd /home/newdir
root@2f8542bc24de:/home/newdir# ls
auto.cnf    client-cert.pem  ib_logfile0  ibtmp1                  performance_schema  server-cert.pem  sys_4.SDI
ca-key.pem  client-key.pem   ib_logfile1  mysql                   private_key.pem     server-key.pem
ca.pem      ib_buffer_pool   ibdata1      performance_sche_3.SDI  public_key.pem      sys
```

Task 5 - Create your own docker-compose file

Create a directory to be used as shared folder.

/home/Ubuntu/mysqldir contains many files from task4



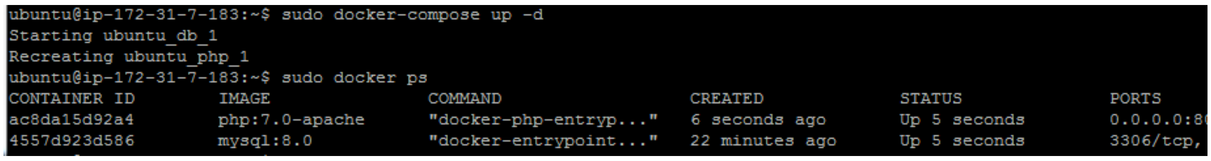
```
ubuntu@ip-172-31-7-183:~$ cd mysqldir/
ubuntu@ip-172-31-7-183:~/mysqldir$ ls
auto.cnf    ca.pem           client-key.pem  ibdata1      ib_logfile1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile0  mysql        performance_schema      private_key.pem
```

Create **docker-compose.yml** file as below:

```
version : '2'

services:
  db:
    image: mysql:8.0
    ports:
      - "3333:3333"
    volumes:
      - /home/ubuntu/mysqldir:/var/www/html
    environment:
      MYSQL_ROOT_PASSWORD: passwerd
  php:
    image: php:7.0-apache
```

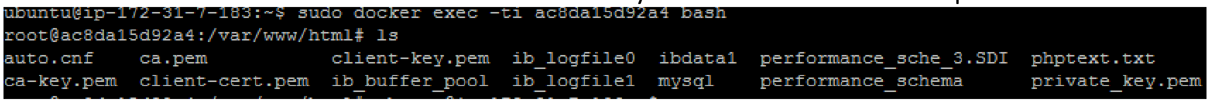
```
links:
  - db:db
ports:
  - "80:80"
volumes:
  - /home/ubuntu/mysqldir:/var/www/html
```

PHP and mysql are running



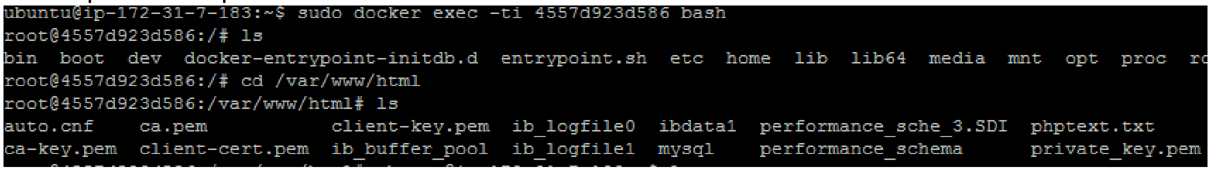
```
ubuntu@ip-172-31-7-183:~$ sudo docker-compose up -d
Starting ubuntu_db_1
Recreating ubuntu_php_1
ubuntu@ip-172-31-7-183:~$ sudo docker ps
CONTAINER ID        IMAGE               COMMAND                  CREATED             STATUS              PORTS
ac8da15d92a4        php:7.0-apache      "docker-php-entryp..."   6 seconds ago       Up 5 seconds        0.0.0.0:8
4557d923d586        mysql:8.0           "docker-entrypoint..."   22 minutes ago      Up 5 seconds        3306/tcp,
```

Execute the first container and check shared directory which contains files as expected.



```
ubuntu@ip-172-31-7-183:~$ sudo docker exec -ti ac8da15d92a4 bash
root@ac8da15d92a4:/var/www/html# ls
auto.cnf    ca.pem           client-key.pem  ib_logfile0  ibdata1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile1  mysql    performance_schema      private_key.pem
```

Execute the second container and it can be seen that the directory is successfully mounted as the files expected are present.



```
ubuntu@ip-172-31-7-183:~$ sudo docker exec -ti 4557d923d586 bash
root@4557d923d586:/# ls
bin  boot  dev  docker-entrypoint-initdb.d  entrypoint.sh  etc  home  lib  lib64  media  mnt  opt  proc  ro
root@4557d923d586:/# cd /var/www/html
root@4557d923d586:/var/www/html# ls
auto.cnf    ca.pem           client-key.pem  ib_logfile0  ibdata1  performance_sche_3.SDI  phptext.txt
ca-key.pem  client-cert.pem  ib_buffer_pool  ib_logfile1  mysql    performance_schema      private_key.pem
```

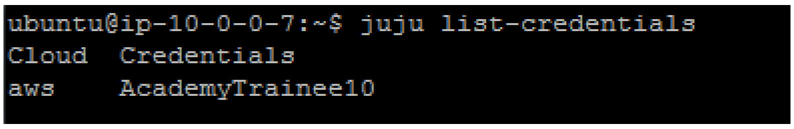
Task 6 – Install Kubernetes

Install conjure up which also installs juju

sudo snap install conjure-up --classic

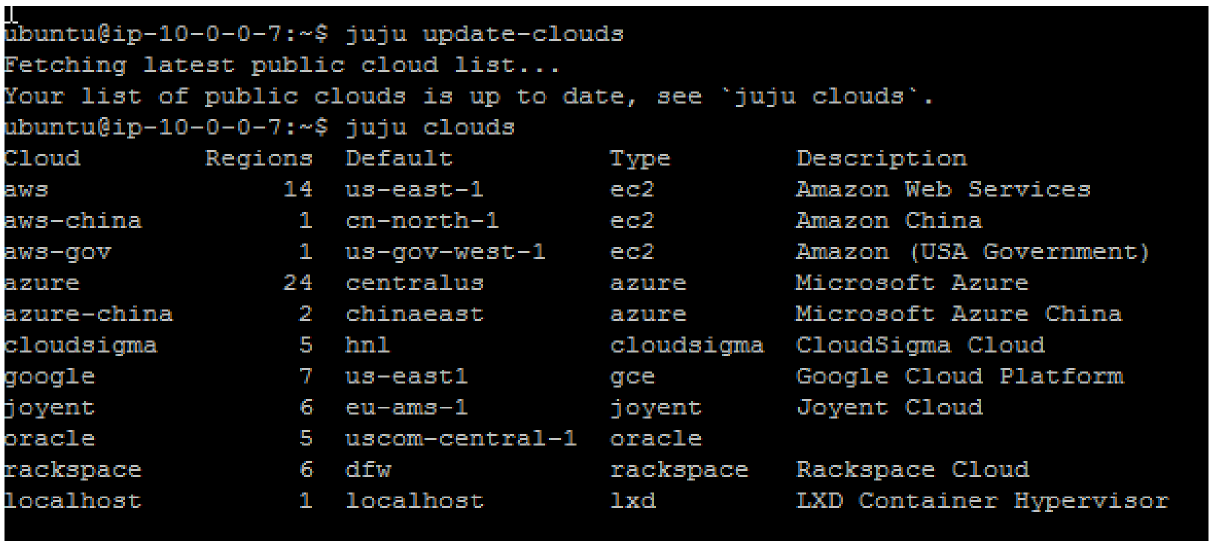
Add credentials of aws using juju

juju add-credential aws



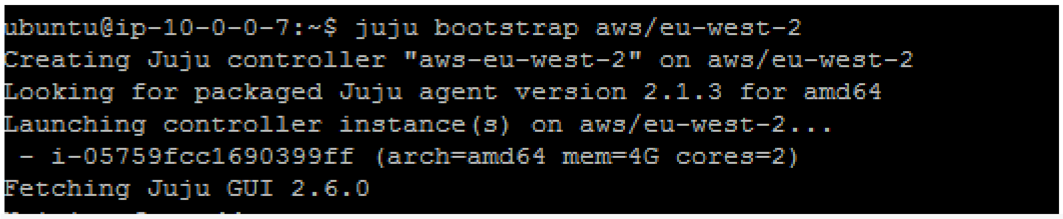
```
ubuntu@ip-10-0-0-7:~$ juju list-credentials
Cloud  Credentials
aws    AcademyTrainee10
```

Update cloud and check it



```
ubuntu@ip-10-0-0-7:~$ juju update-clouds
Fetching latest public cloud list...
Your list of public clouds is up to date, see `juju clouds`.
ubuntu@ip-10-0-0-7:~$ juju clouds
Cloud          Regions  Default          Type        Description
aws                 14  us-east-1        ec2         Amazon Web Services
aws-china            1  cn-north-1       ec2         Amazon China
aws-gov              1  us-gov-west-1    ec2         Amazon (USA Government)
azure               24  centralus        azure       Microsoft Azure
azure-china          2  chinaeast        azure       Microsoft Azure China
cloudsigma           5  hnl              cloudsigma  CloudSigma Cloud
google               7  us-east1         gce         Google Cloud Platform
joyent               6  eu-ams-1         joyent      Joyent Cloud
oracle               5  uscom-central-1  oracle
rackspace            6  dfw              rackspace   Rackspace Cloud
localhost            1  localhost        lxd         LXD Container Hypervisor
```

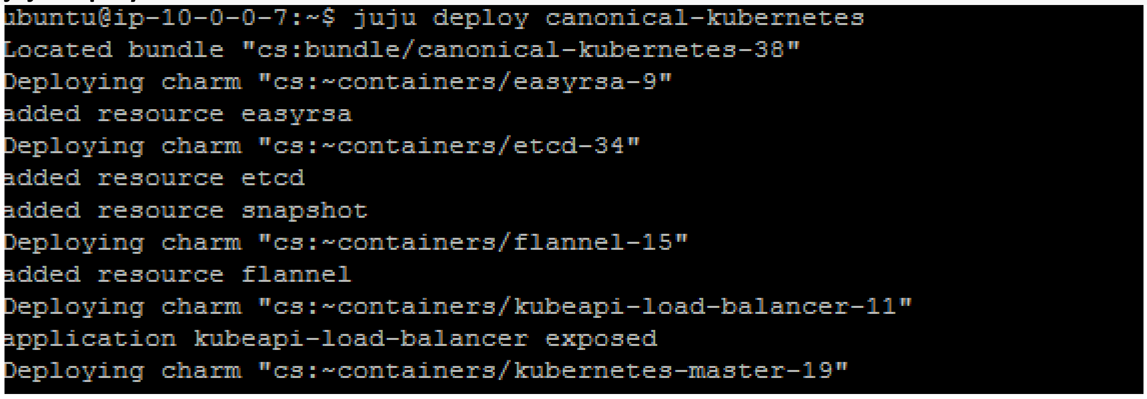
Bootstrap a controller to manage our cluster



```
ubuntu@ip-10-0-0-7:~$ juju bootstrap aws/eu-west-2
Creating Juju controller "aws-eu-west-2" on aws/eu-west-2
Looking for packaged Juju agent version 2.1.3 for amd64
Launching controller instance(s) on aws/eu-west-2...
 - i-05759fcc1690399ff (arch=amd64 mem=4G cores=2)
Fetching Juju GUI 2.6.0
```

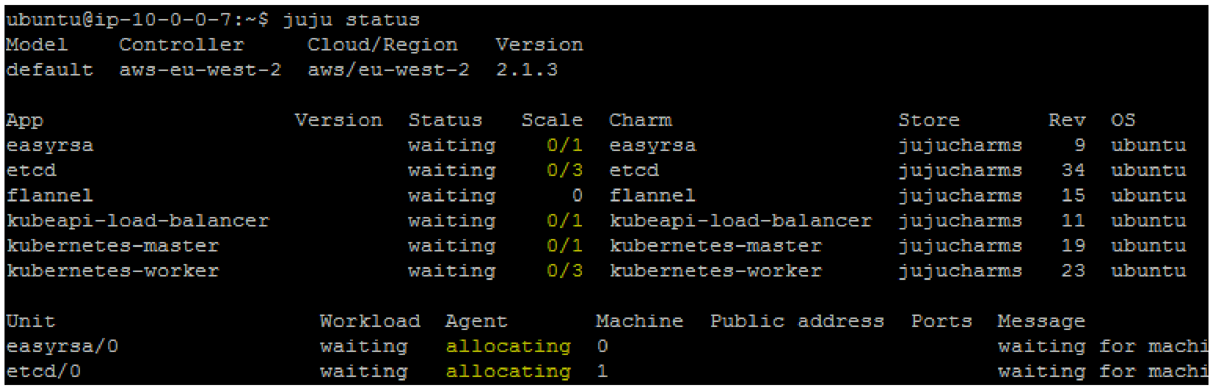
Deploy a cluster of 9 nodes

juju deploy canonical-kubernetes



```
ubuntu@ip-10-0-0-7:~$ juju deploy canonical-kubernetes
Located bundle "cs:bundle/canonical-kubernetes-38"
Deploying charm "cs:~containers/easyrsa-9"
added resource easyrsa
Deploying charm "cs:~containers/etcd-34"
added resource etcd
added resource snapshot
Deploying charm "cs:~containers/flannel-15"
added resource flannel
Deploying charm "cs:~containers/kubeapi-load-balancer-11"
application kubeapi-load-balancer exposed
Deploying charm "cs:~containers/kubernetes-master-19"
```

Check the status



```
ubuntu@ip-10-0-0-7:~$ juju status
Model    Controller     Cloud/Region   Version
default  aws-eu-west-2  aws/eu-west-2  2.1.3

App                    Version  Status   Scale  Charm                  Store       Rev  OS
easyrsa                         waiting    0/1  easyrsa                jujucharms    9  ubuntu
etcd                            waiting    0/3  etcd                   jujucharms   34  ubuntu
flannel                         waiting      0  flannel                jujucharms   15  ubuntu
kubeapi-load-balancer           waiting    0/1  kubeapi-load-balancer  jujucharms   11  ubuntu
kubernetes-master               waiting    0/1  kubernetes-master      jujucharms   19  ubuntu
kubernetes-worker               waiting    0/3  kubernetes-worker      jujucharms   23  ubuntu

Unit                      Workload  Agent       Machine  Public address  Ports  Message
easyrsa/0                 waiting   allocating  0                               waiting for machi
etcd/0                    waiting   allocating  1                               waiting for machi
```

Task 7 – Creating your first Single Container Pod

Install kubectl using curl

```
curl -LO https://storage.googleapis.com/kubernetes-release/release/$(curl -s
https://storage.googleapis.com/kubernetes-release/release/stable.txt)/bin/linux/amd64/kubectl
```

Make it executable

```
chmod +x ./kubectl
```

Move it to environmental PATH

```
sudo mv ./kubectl /usr/local/bin/kubectl
```