**What is Docker?**

Docker is an open platform for developing, shipping, and running applications. Docker enables you to separate your applications from your infrastructure so you can deliver software quickly. With Docker, you can manage your infrastructure in the same ways you manage your applications.

**What is container?**

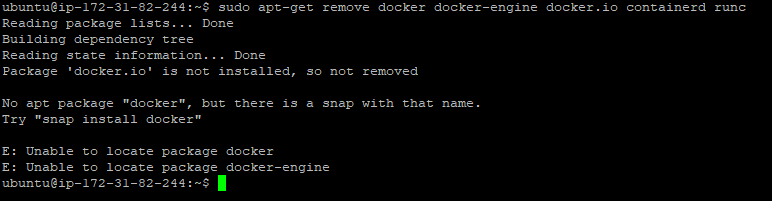
A container is a standard unit of software that packages up code and all its dependencies so the application runs quickly and reliably from one computing environment to another. A Docker container image is a lightweight, standalone, executable package of software that includes everything needed to run an application: code, runtime, system tools, system libraries and settings. Container images become containers at runtime and in the case of Docker containers – images become containers when they run on docker engine.

**Step to install docker in ubuntu**

#### **Set up the repository**

**Step 1: Frist we need to delete the old version of docker installed**

sudo apt-get remove docker docker-engine docker.io containerd runc



NOTE: if no docker is present in the server this output will come.

**Step 2: sudo apt-get update**

Update the apt package index and install packages to allow apt to use a repository over HTTPS:

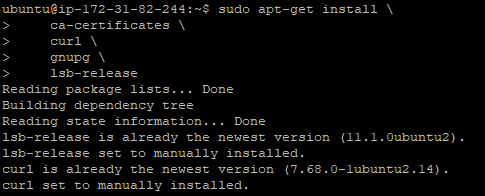
sudo apt-get install \

ca-certificates \

curl \

gnupg \

lsb-release

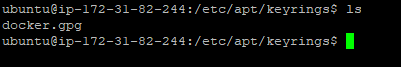


**Step 3: Add Docker’s official GPG key:**

Create a Directory - sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /etc/apt/keyrings/docker.gpg





**Step 4: Use the following command to set up the repository:**

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/etc/apt/keyrings/docker.gpg] https://download.docker.com/linux/ubuntu \

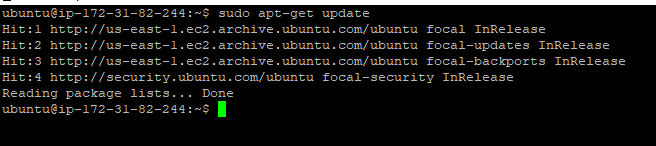
$(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null



#### **Install Docker Engine**

**Step 1: Update the apt package index:**

sudo apt-get update



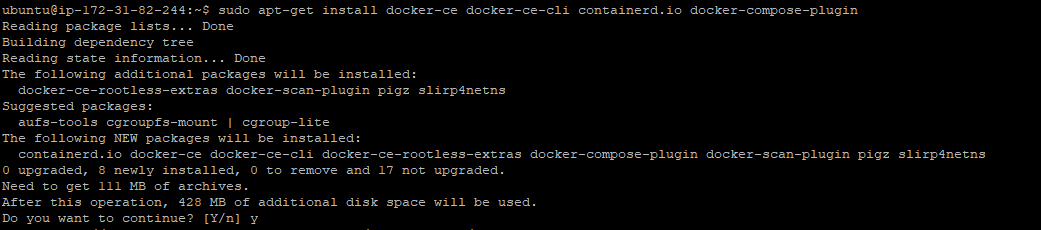
NOTE: If you get GPG error then use the below command

sudo chmod a+r /etc/apt/keyrings/docker.gpg

sudo apt-get update

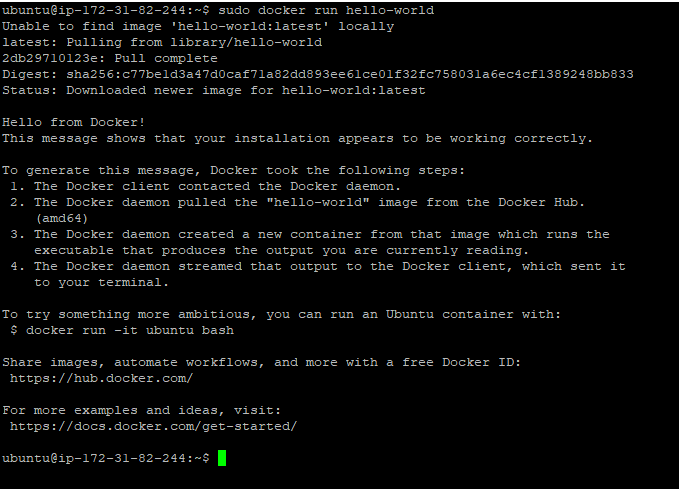
**Step 2: Install Docker Engine, containerd, and Docker Compose.**

sudo apt-get install docker-ce docker-ce-cli containerd.io docker-compose-plugin



To Verify that the Docker Engine installation is successful by running the hello-world image:

sudo docker run hello-world



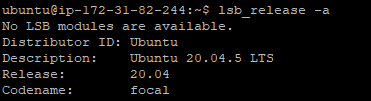
#### **Upgrade Docker Engine**

### Install from a package

**Step 1: Go to** [**https://download.docker.com/linux/ubuntu/dists/**](https://download.docker.com/linux/ubuntu/dists/)

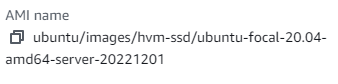
**Step 2: Select your Ubuntu version in the list.**

To know your ubuntu version use : lsb\_release -a



**Step 3: Go to pool/stable/ and select the applicable architecture**

NOTE: we can find the architecture in our aws instance detail ami name



**Step 4: Download the following deb files for the Docker Engine, CLI, containerd, and Docker Compose packages:**

* 1. containerd.io\_<version>\_<arch>.deb
  2. docker-ce\_<version>\_<arch>.deb
  3. docker-ce-cli\_<version>\_<arch>.deb
  4. docker-compose-plugin\_<version>\_<arch>.deb

**Step 5: Install the .deb packages. Update the paths in the following example to where you downloaded the Docker packages.**

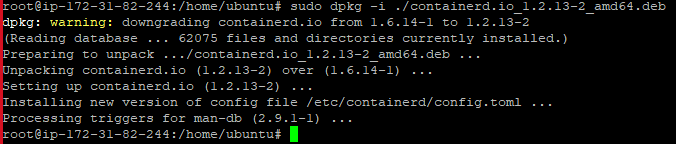
sudo dpkg -i ./containerd.io\_<version>\_<arch>.deb \

./docker-ce\_<version>\_<arch>.deb \

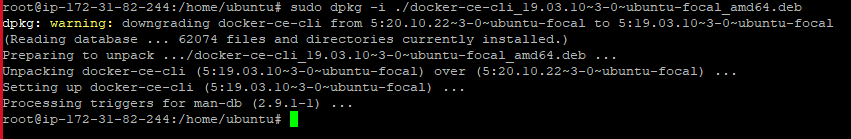
./docker-ce-cli\_<version>\_<arch>.deb \

./docker-compose-plugin\_<version>\_<arch>.deb

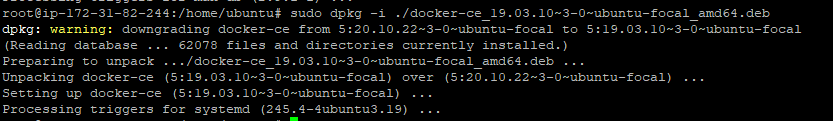
sudo dpkg -i ./containerd.io\_1.2.13-2\_amd64.deb



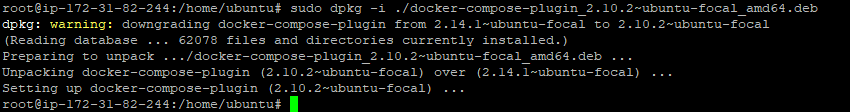
sudo dpkg -i ./docker-ce-cli\_19.03.10~3-0~ubuntu-focal\_amd64.deb



sudo dpkg -i ./docker-ce\_19.03.10~3-0~ubuntu-focal\_amd64.deb



sudo dpkg -i ./docker-compose-plugin\_2.10.2~ubuntu-focal\_amd64.deb



**How to pull image from docker hub**

**Create a account on docker hub**

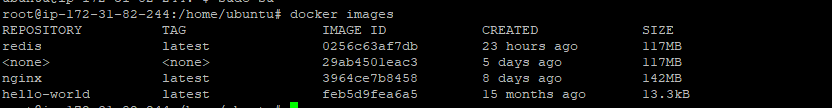
<https://hub.docker.com/> Login with your user id and pass . You wil find various ref images there .Choose any one image which you want to work on .

Here we are doing for Nginx and Redis . SO we will go to nginx image and use the pull command present there .

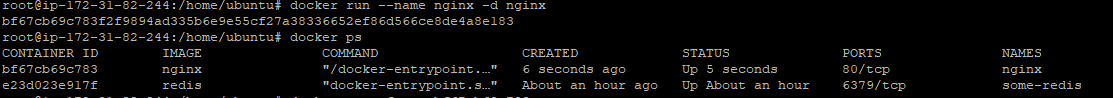
**NGINX setup in Docker**

**Step 1: Command** - docker pull nginx (Using this command we can pull the image)

To check the images present use command docker images (It will list all the images present in the server )

**STEP 2: To create the container using the pulled image we need to use**

docker run --name nginx -d -p 8080:80 nginx

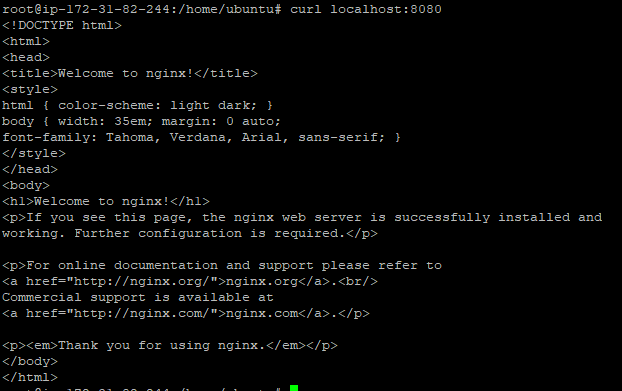
You can see in the above image the running container I.e nginx and redis

To Get inside the container you can use the command

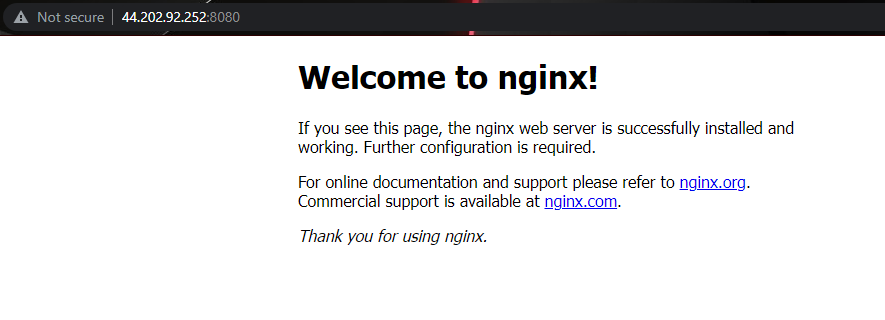
**STEP 3: docker exec -it <container id> bash**

Also to check nginx is running or not we can use two process that is

1. curl localhost:8080



2.) You can go the browser and give your server ip and port on which nginx is running .

You willl see the home page of Nginx

**Basic commands for Docker -**

|  |  |
| --- | --- |
| ***Command*** | ***Description*** |
|  |  |
| **Docker version** | We usually start by finding the installed version of docker that we are working on |
| **Docker search** | The “docker search” command searches for specific images through the Docker hub. This command returns the specific information, including image name, description, automated, official stars, etc. |
| **Docker pull** | As the name suggests, this command pulls a specific image from the Docker Hub. All you have to do is use the command ‘docker pull’ along with the name of the image. |
| **Docker run** | This command is used to create a container from an image. |
| **Docker ps** | This command is used to list all the running containers in the background. |
| **Docker stop** | The ‘docker stop’ command stops a container using the container name or its id. |
| **Docker restart** | his command is used to restart the stopped container. It is recommended to use this after rebooting the system. |
| **Docker kill** | This command is used to stop the container immediately by killing its execution. While the ‘docker stop’ command helps shut down the container in its own time, the ‘docker kill’ command stops it at once |
| **Docker exec** | This command is used to access the container that is running |
| **Docker login** | This command helps you to log into your docker hub. As you try to log in, you will be asked to give your docker hub credentials. |
| **Docker commit** | This command is used to create or save an image of the edited container on the local system. |
| **Docker push** | This command helps push or upload a docker image on the repository or the docker hub |
| **Docker network** | The ‘docker network’ command is used to know the details of the list of networks in the cluster. |
| **Docker rmi** | This command is used to free up some disk space. The image id is used to remove the image while using this command. |
| **Docker ps -a** | This command is used to know the details of all the running, stopped, or exited containers. |
| **Docker copy** | This command copies a file from docker to the local system |
| **Docker logs** | This command is used to check the logs of all the docker containers with the corresponding contained id mentioned in the command |
| **Docker volume** | This command creates a volume so that the docker container can use it to store data. |
| **Docker logout** | This command will log you out of the docker hub |