

**Aim:** To study and Implement Containerization using Docker

## 2. Theory:

- Docker is a tool designed to make it easier to create, deploy and run applications by using containers, which are formed with the help of Docker Engine.
- Docker containers are light-weight alternatives to virtual machines and use host OS.
- With dockers, there is no need to pre-allocate the RAM or disk space, it takes RAM and disk space according to the requirement.

## 3. Procedure:

### Step1: How to install Docker on EC2 instance.

To install Docker on the EC2 Instance. Go to the location path where your Key Pair is saved, open terminal and type the following command:

**ssh -i keypair.pem ec2-user@<EC2-INSTANCE-PUBLIC-IP-ADDRESS>**

```
chandupriyam@MTLP-616 MINGW64 /d/Chandu Priya/DOCKER
$ ssh -i AWSDOCKERS.pem ec2-user@54.218.250.170
The authenticity of host '54.218.250.170 (54.218.250.170)' can't be established.
ECDSA key fingerprint is SHA256:qEV70ogYXdQ8ktBdJFwaaIKDOz2NwImdztww2Rx1VjU.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '54.218.250.170' (ECDSA) to the list of known hosts.

  _ | _ | _ )
  _ | ( _ | /   Amazon Linux AMI
  _ | \ _ | _ |

https://aws.amazon.com/amazon-linux-ami/2018.03-release-notes/
3 package(s) needed for security, out of 7 available
Run "sudo yum update" to apply all updates.
[ec2-user@ip-172-31-19-85 ~]$ |
```

You have connected to your EC2 instance, Now you can install docker by using the below command:

**sudo yum install -y docker**

Package	Arch	Version
Installing: docker	x86_64	18.06.1ce-10.32.amzn1
Installing for dependencies:		
libseccomp	x86_64	2.3.1-2.4.amzn1
libtool-ltdl	x86_64	2.4.2-20.4.8.5.32.amzn1
pigz	x86_64	2.3.3-1.6.amzn1
xfsprogs	x86_64	4.5.0-18.23.amzn1
Transaction Summary		
Install 1 Package (+4 Dependent packages)		
Total download size: 47 M		
Installed size: 154 M		
Downloading packages:		
(1/5): libseccomp-2.3.1-2.4.amzn1.x86_64.rpm		
(2/5): libtool-ltdl-2.4.2-20.4.8.5.32.amzn1.x86_64.rpm		
(3/5): pigz-2.3.3-1.6.amzn1.x86_64.rpm		
(4/5): xfsprogs-4.5.0-18.23.amzn1.x86_64.rpm		
(5/5): docker-18.06.1ce-10.32.amzn1.x86_64.rpm		
Total		
Running transaction check		
Running transaction test		
Transaction test succeeded		
Running transaction		
Installing : pigz-2.3.3-1.6.amzn1.x86_64		
Installing : libtool-ltdl-2.4.2-20.4.8.5.32.amzn1.x86_64		
Installing : libseccomp-2.3.1-2.4.amzn1.x86_64		
Installing : xfsprogs-4.5.0-18.23.amzn1.x86_64		
Installing : docker-18.06.1ce-10.32.amzn1.x86_64		
Verifying : xfsprogs-4.5.0-18.23.amzn1.x86_64		
Verifying : docker-18.06.1ce-10.32.amzn1.x86_64		
Verifying : libseccomp-2.3.1-2.4.amzn1.x86_64		
Verifying : libtool-ltdl-2.4.2-20.4.8.5.32.amzn1.x86_64		
Verifying : pigz-2.3.3-1.6.amzn1.x86_64		
Installed:		
docker.x86_64 0:18.06.1ce-10.32.amzn1		
Dependency Installed:		
libseccomp.x86_64 0:2.3.1-2.4.amzn1		
libtool-ltdl.x86_64 0:2.4.2-20.4.8.5.32.amzn1		
pigz.x86_64 0:2.3.3-1.6.amzn1		

Use this command to start the docker:

sudo service docker start

```
[ec2-user@ip-172-31-19-85 ~]$ sudo service docker start
Starting cgconfig service: [ OK ]
Starting docker: [ OK ]
[ec2-user@ip-172-31-19-85 ~]$ |
```

docker info

If you done everything right, the above command will shows info about Docker installation without showing any errors.

```
[root@ip-172-31-19-85 ec2-user]# docker info
Containers: 0
  Running: 0
  Paused: 0
  Stopped: 0
Images: 0
Server Version: 18.06.1-ce
Storage Driver: overlay2
  Backing Filesystem: extfs
  Supports d_type: true
  Native Overlay Diff: true
Logging Driver: json-file
Cgroup Driver: cgroupfs
Plugins:
  Volume: local
  Network: bridge host macvlan null overlay
  Log: awslogs fluentd gcplogs gelf journald json-file logentries splunk
Swarm: inactive
Runtimes: runc
Default Runtime: runc
Init Binary: docker-init
containerd version: 468a545b9edcd5932818eb9de8e72413e616e86e
runc version: 69663f0bd4b60df09991c08812a60108003fa340
init version: fec3683
Security Options:
  seccomp
   Profile: default
Kernel Version: 4.14.138-89.102.amzn1.x86_64
Operating System: Amazon Linux AMI 2018.03
OSType: linux
Architecture: x86_64
CPUs: 1
Total Memory: 985.8MiB
Name: ip-172-31-19-85
ID: TMMZ:IMAX:RDXQ:53FB:2AGU:M22U:JZU4:N75O:GZ7A:MHXM:GJPH:IT5X
Docker Root Dir: /var/lib/docker
Debug Mode (client): false
Debug Mode (server): false
Registry: https://index.docker.io/v1/
Labels:
Experimental: false
Insecure Registries:
  127.0.0.0/8
Live Restore Enabled: false

[root@ip-172-31-19-85 ec2-user]# |
```

Now you can run your image. In this case, I will show you the sample node app to get something like this:

**docker run -d -p 80:8080 chandupriya/nodedemo:0.1**

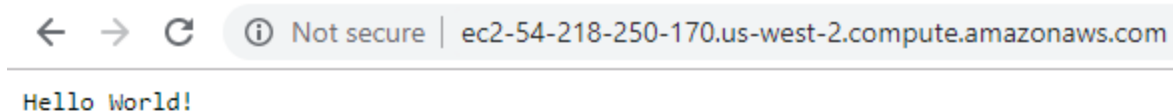
```
[root@ip-172-31-19-85 ec2-user]#  
[root@ip-172-31-19-85 ec2-user]# docker run -d -p 80:8080 chandupriya/nodedemo:0.1  
Unable to find image 'chandupriya/nodedemo:0.1' locally  
0.1: Pulling from chandupriya/nodedemo  
3d77ce4481b1: Pull complete  
7d2f32934963: Pull complete  
0c5cf711b890: Pull complete  
9593dc852d6b: Pull complete  
4e3b8a1eb914: Pull complete  
ddcf13cc1951: Pull complete  
2e460d114172: Pull complete  
d94b1226fbf2: Pull complete  
4e89ff8419a9: Pull complete  
Digest: sha256:c2d53dea92f14e73455ac4f88a14d296cb3c2cde65ae189417d552832488f3c5  
Status: Downloaded newer image for chandupriya/nodedemo:0.1  
be3f9e88bc9754602e2bfb0280c3f21e8909cfd2cdffbfdd67aa2806e69bed3b  
54-218-250-170.us-west-2.compute.amazonaws.com
```

You will note the “-p 80:8080” text which commands Docker on your Container to link port 8080 to port 80 located on the EC2 Instance. Test whether the process has gone through using the following command:

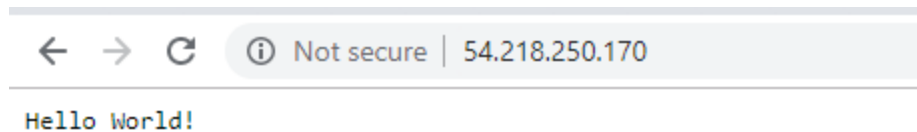
```
[ec2-user@ip-172-31-19-85 ~]$  
[ec2-user@ip-172-31-19-85 ~]$ curl http://localhost  
Hello World! [ec2-user@ip-172-31-19-85 ~]$
```

If you get the message “**Hello world!**” then your Docker container is running in the AWS cloud.

The last step is to visit the public DNS name in your browser, you get the message like this:



The screenshot shows a web browser window with a light gray header bar. On the left, there are navigation icons: a back arrow, a forward arrow, and a circular refresh icon. To the right of these icons is a status bar that reads "Not secure" followed by the URL "ec2-54-218-250-170.us-west-2.compute.amazonaws.com". Below the header bar, the main content area of the browser displays the text "Hello World!" in a monospaced font.



Now that you have successfully deployed Docker Container on AWS EC2 everything looks perfect.

To stop all Docker containers, simply run the following command in your terminal:

```
docker kill $(docker ps -q)
```

### **How to Remove All Docker Containers**

If you don't just want to stop containers and you'd like to go a step further and remove them, simply run the following command:

```
docker rm $(docker ps -a -q)
```

### **How To Remove All Docker Images**

To remove all Docker images, run this command:

```
docker rmi $(docker images -q)
```

### **Conclusion:**

Implementation of Docker containers with AWS EC2 Instance has been done successfully.