**Docker Container LAB**

**Docker Tutorial**

Go through this tutorial before starting the lab as it would clear all your basic concepts.

<https://www.youtube.com/watch?v=zJ6WbK9zFpI>

**Docker concepts**

Docker is a platform for developers and sysadmins to develop, deploy, and run applications with containers. The use of Linux containers to deploy applications is called containerization. Containers are not new, but their use for easily deploying applications is.

Containerization is increasingly popular because containers are:

• Flexible: Even the most complex applications can be containerized.

• Lightweight: Containers leverage and share the host kernel.

• Interchangeable: You can deploy updates and upgrades on-the-fly.

• Portable: You can build locally, deploy to the cloud, and run anywhere.

• Scalable: You can increase and automatically distribute container replicas.

• Stackable: You can stack services vertically and on-the-fly.

**Images and containers**

A container is launched by running an image. An image is an executable package that includes everything needed to run an application--the code, a runtime, libraries, environment variables, and configuration files.

Images of docker container are indexed and saved in online repository which is managed by docker

<https://hub.docker.com/>

A container is a runtime instance of an image--what the image becomes in memory when executed (that is, an image with state, or a user process). You can see a list of your running containers with the command, docker ps, just as you would in Linux.

**Containers and virtual machines**

A container runs natively on Linux and shares the kernel of the host machine with other containers. It runs a discrete process, taking no more memory than any other executable, making it lightweight.

By contrast, a virtual machine (VM) runs a full-blown “guest” operating system with virtual access to host resources through a hypervisor. In general, VMs provide an environment with more resources than most applications need.

We will be using UBUNTU 16.04 machine [Run it in VirtualBox]

docker installation <https://docs.docker.com/install/linux/docker-ce/ubuntu/>

1. Download Docker CE for your linux kernel.
2. Do you know which are two flavors of Docker?
3. Docker can be installed on Windows, Linux and MAC but can only run applications designed to work on?
4. Check whether Docker is installed?
5. Update the repositories sources in ubuntu and the prerequisites for docker.
6. If you get error then the system is searching for new packages, look for dpkg and if it found kill the process and start installing the prerequisites again.
7. Using curl command to find and download gpg key found by docker to verify as a security measure.
8. Use command to check the correct version of docker.
9. Verify if docker has been installed (by running test hello-world image)
10. List basic docker commands.
11. What do you mean docker0?
12. What are three models of networking in docker?
13. Built a container on ‘busybox’
14. Perform the closed networking container and after running it which interface you will get?
15. Perform the bridged networking model and which interface you will get?
16. What will you see when you run Process status command in Docker?
17. What do you mean by port 32768 under port?
18. Statically assign port forwarding instead docker assigning port forwarding.
19. What will you do to stop and remove webserver.
20. Write the command in which docker should forward traffic which is coming on port 8080 to 80 on the container.
21. How will you run on local loop back address?
22. Use the Open Networking model using Containerization with Docker.
23. Which file system does docker use?
24. What do you mean by docker volume?
25. Which are three methods of storage in docker?
26. How will you set Bind mount and find the current directory?
27. Create a file called exam.html and check and check if you can see it.
28. Why volumes are better than bind-mounts
29. Use volume with a container, create it and inspect it.
30. What is tmfs mount?
31. Create a busybox container that will write a one-time hash directory on the host machine, how will you access the container.
32. Login to container and write text to file and observe its start and restart nature.
33. Docker v/s Vagrant.