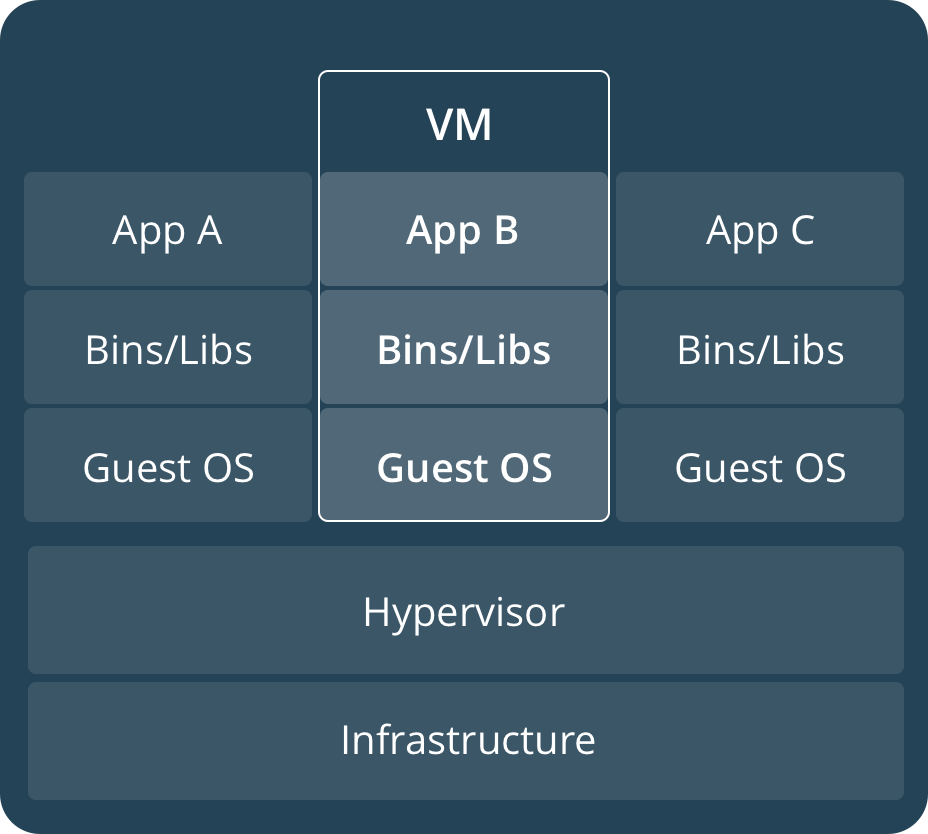
Containers and Dockers.

Southern Methodist University

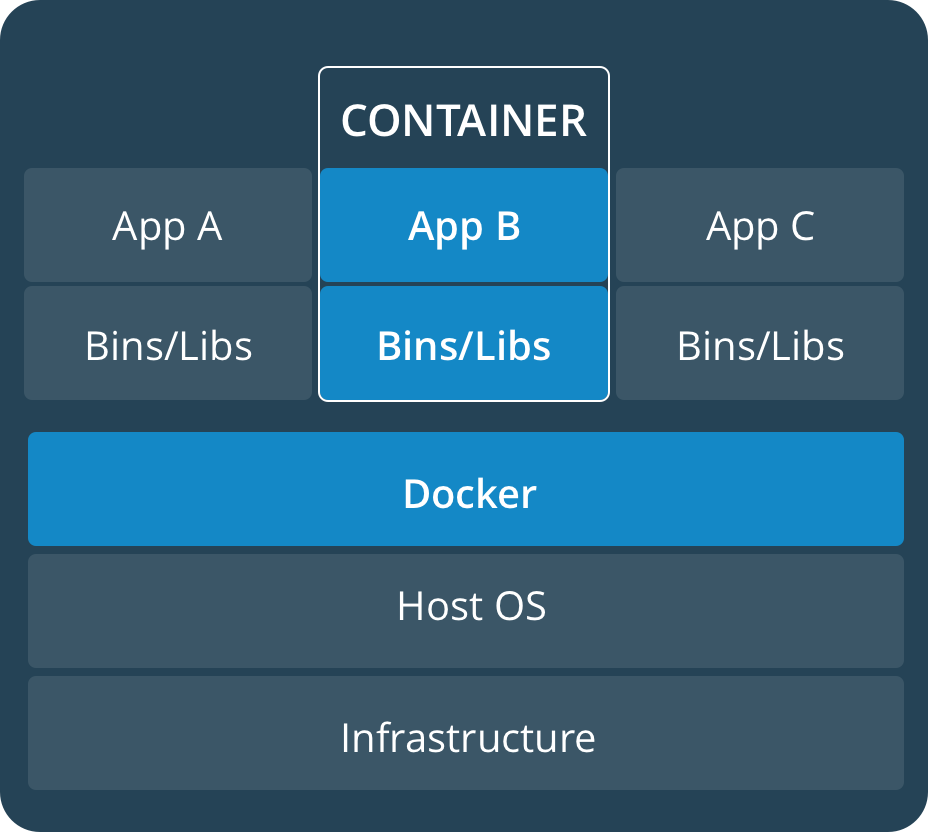
Telecommunication and network engineering

Virtualization is very important, useful and widely used technology in Data Centers today, and we all know that by now. All the labs we have covered were based on technologies which virtualize the Data Centers. We will continue further in exploring few more technologies in the same category. In today’s lab we will discover and learn another widely used technology (which is quite old, but has found a new way of implementation and hence gained popularity and wide acceptance) called containers.

Virtual Machine:



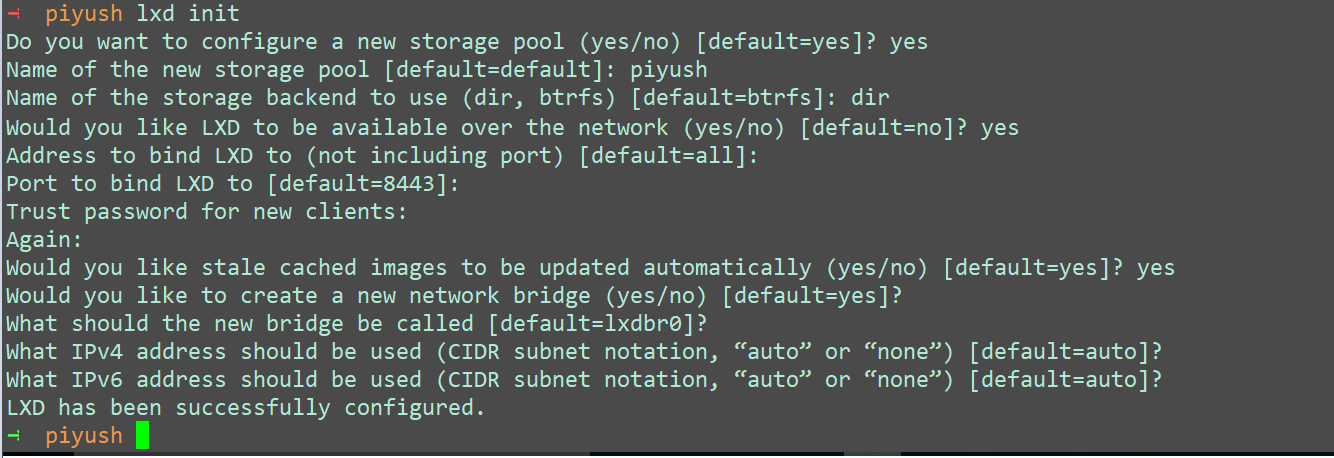
Containers:

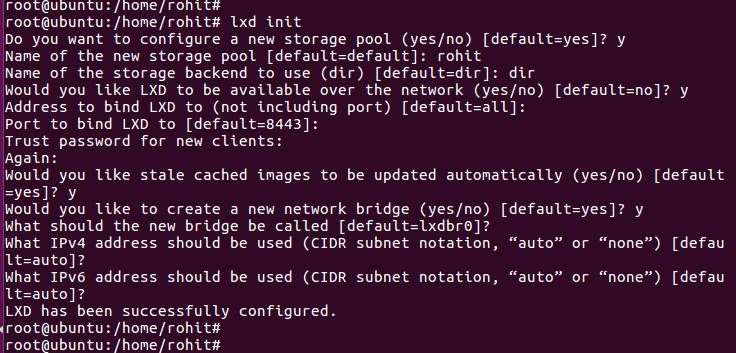


1. Note the IP address of your machine and ssh into it using putty. Use following credentials;
2. Verify connectivity to internet, and upgrade your machines
3. To get started with containers, we need to install the packages related to it first, download the packages using following commands;

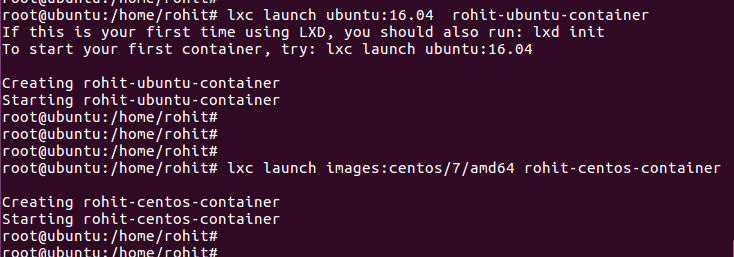
*apt install -t xenial-backports lxd lxd-client*

1. Configure your container by issuing following command and screenshot below;
   1. sudo lxd init

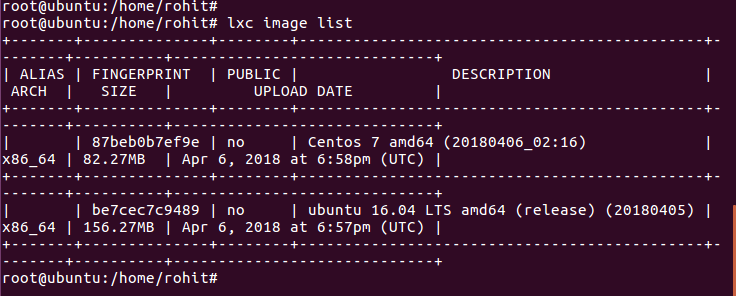


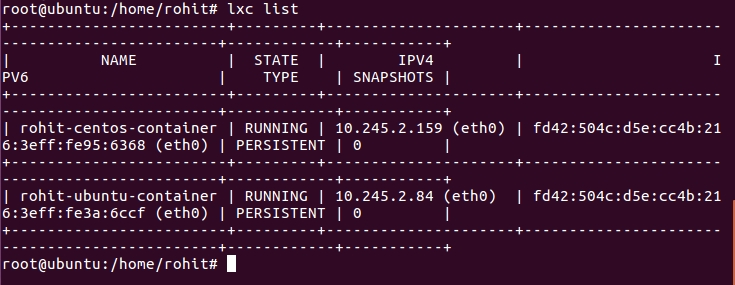


1. Download and launch an Ubuntu and a centos container by issuing following commands
   1. lxc launch ubuntu:16.04 *xyz*-ubuntu-container # Replace xyz with your name
   2. lxc launch images:centos/7/amd64 xyz-centos-container # Replace xyz with your name

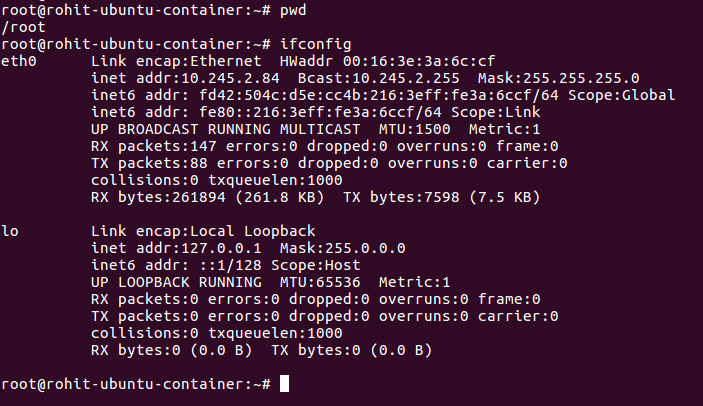


1. List the available images using command ‘lxc image list’ and list the currently running containers using command ‘lxc list’

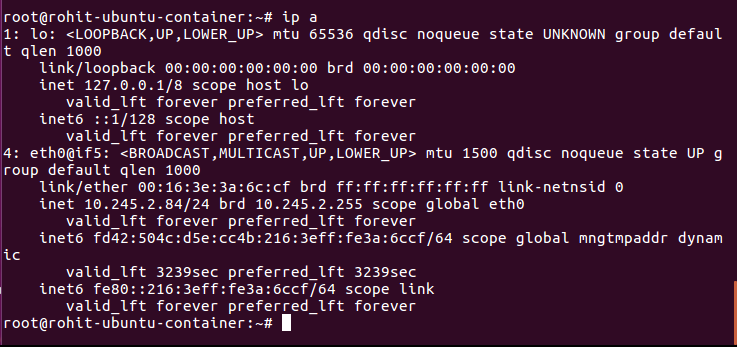


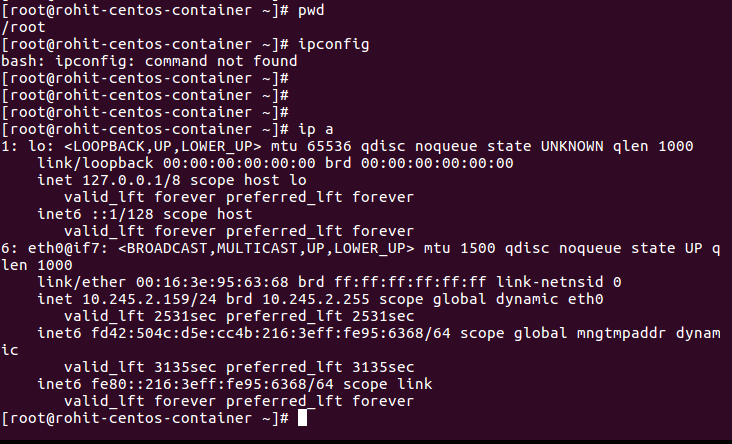


1. To access the shell of a container you can use following commands;
   1. lxc exec ‘*container-name’* -- /bin/bash. You can use ‘ash’ instead of /bin/bash if bash is not installed by default.
   2. Launch shell of ubuntu and centos containers you created and provide the output of following;
      1. Current working directory

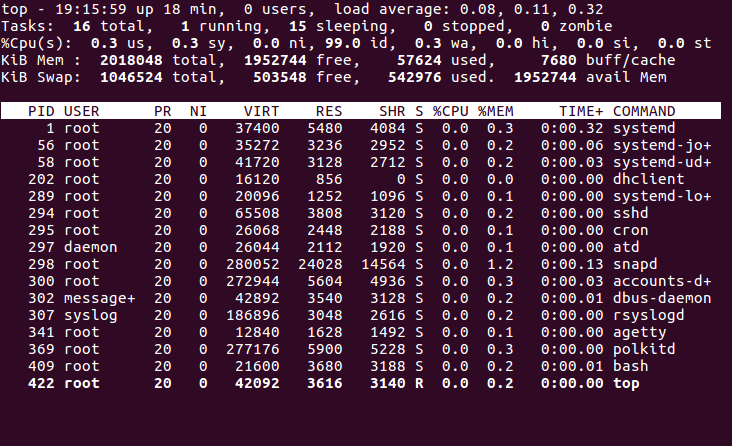


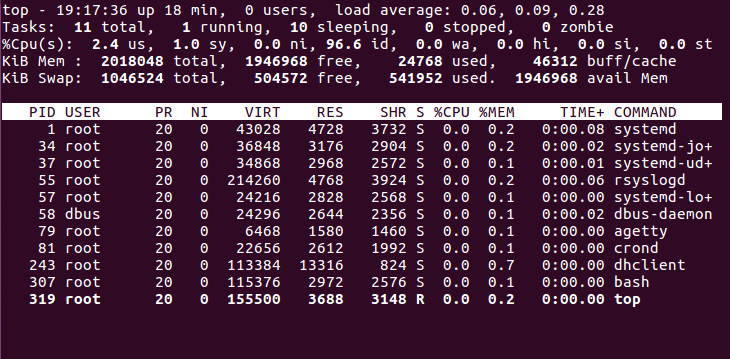
* + 1. IP address information



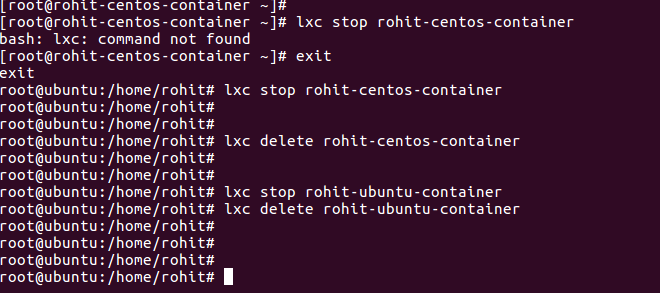


* + 1. Current utilization



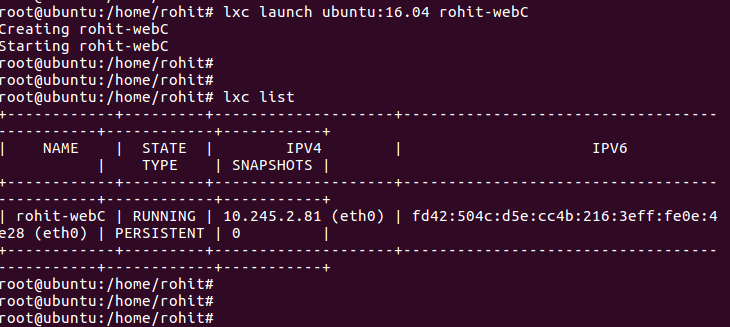


1. To stop a container, issue the command ‘lxc stop ‘*container-name’* and to delete it, use ‘lxc delete ‘*container-name’.* Delete both the containers you created. Take screenshots of before and after deleting the containers.

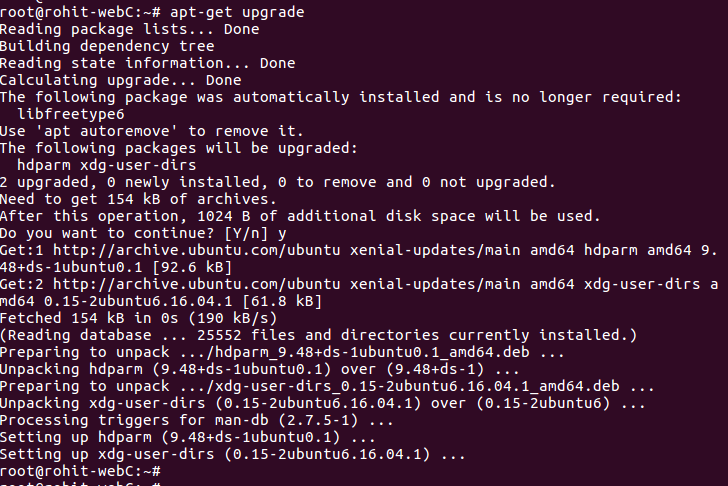


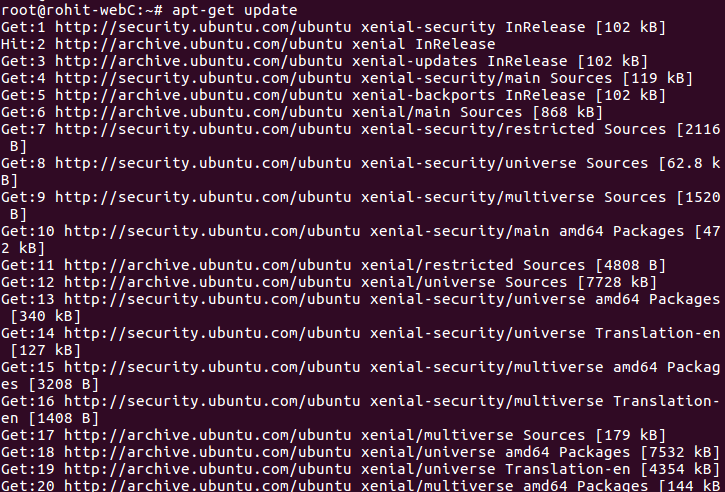
Task – Launch a web server inside a container.

1. Launch a container using ubuntu 16.04 image and name it as ‘*your-name’*-webC



1. Login to the shell and then update and upgrade your container





1. Install nginx and then edit the default index file located at /var/www/html/ /index.nginx-debian.html;

<!DOCTYPE html>

<html>

<head>

<title>\*\*\*\*\*\*\*\*\*\*\*\*Welcome\*\*\*\*\*\*\*\*\*\*\*\*</title>

<style>

body {

width: 35em;

margin: 0 auto;

font-family: Tahoma, Verdana, Arial, sans-serif;

}

</style>

</head>

<body>

<h1>Welcome to your first web container!</h1>

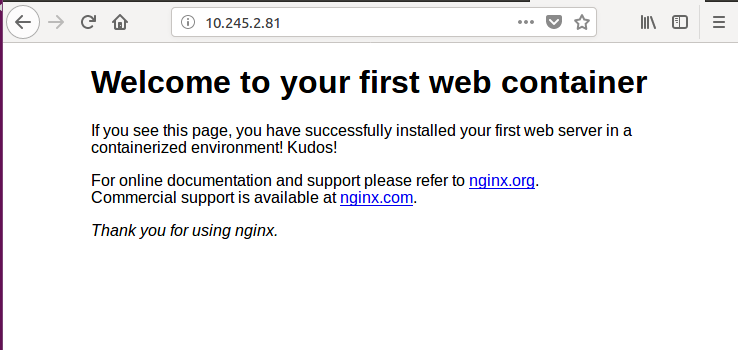
<p>If you see this page, You have successfully installed you first web

server in a containerized environment! Kudos!</p>

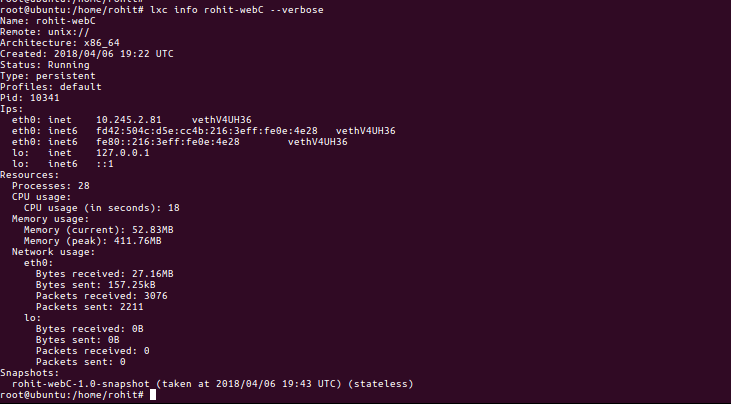
</body>

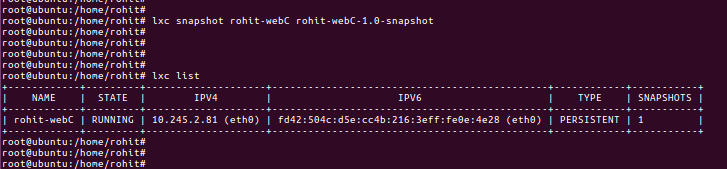
</html>

1. Restart nginx server and verify by accessing the URL on your web browser. (Use your container’s IP)



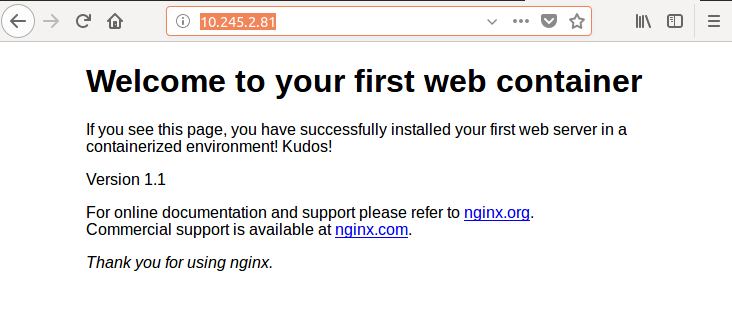
1. Create a snapshot of your container and name it as ‘*your-name’*-webC-1.0



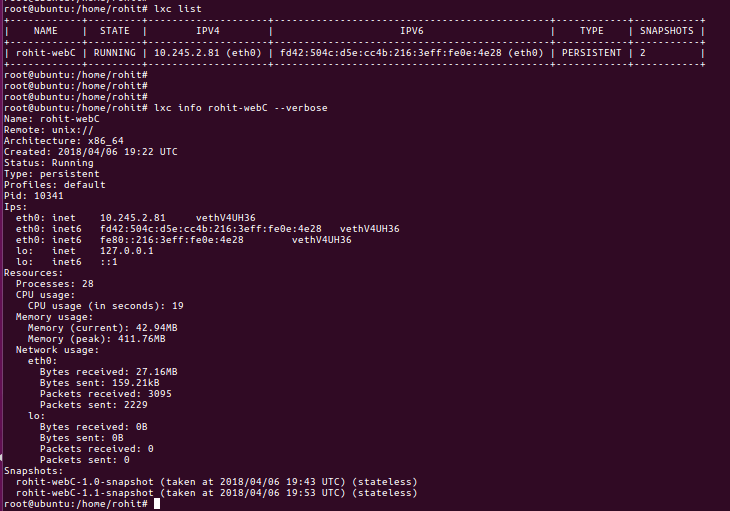
1. Verify whether the snapshot was created or not. 
2. Add the following line in the body of index file of your web container and restart nginx.

<p>Version 1.1</p>

1. Verify the change by reloading the web page.



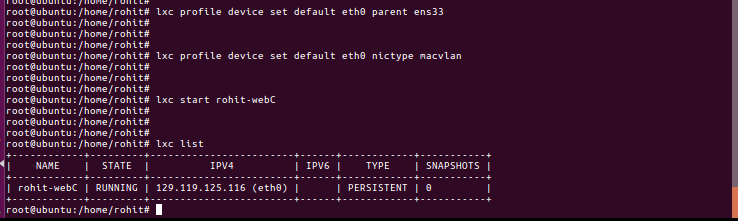
1. Create a new snapshot and name it as ‘*your-name’*-webC-1.1



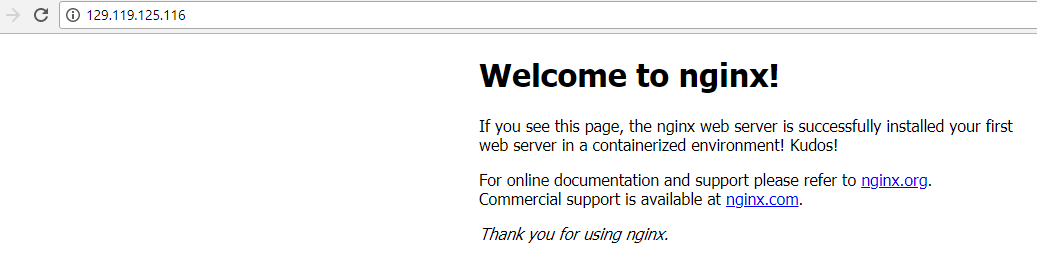
1. Restore your container to ‘*your-name’*-webC-1.0 and reload the page to see the webpage going back to the previous stage.
2. Delete the latest snapshot you created.
3. Currently, the web server can only be accessed from your local host machine which is running the container. To make this web server available on the network, we need to bridge the physical network of the host to container network. To do so, first stop the container and then paste the following commands;

lxc profile device set default eth0 parent ens160 (replace ens160 with the interface name of your ubuntu host connecting to internet)

lxc profile device set default eth0 nictype macvlan



1. Now, start your web container again and notice the new IP address of the container. Verify the web page by accessing the new IP from your browser.



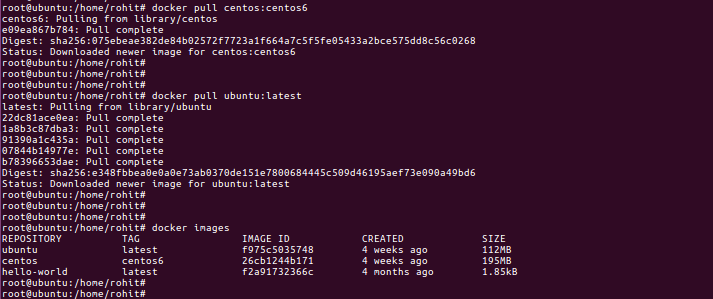
Dockers

1. Use the official installation guide to install Docker CE version on ubuntu

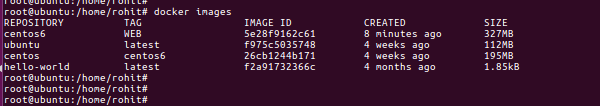
<https://docs.docker.com/engine/installation/linux/docker-ce/ubuntu/>



1. The docker service by default can be run only by root. If a user needs to use docker, it needs to be added in the docker group. This can be done by issuing the command ‘usermod -aG docker user-name’. Add your user in the docker group and verify by issuing command ‘id *user-name’*.
2. Once this is done, reboot the system for the changes to take effect.
3. Login back in to your host and check which version of docker is running. Issue the command ‘docker info’ to get more detailed information such as number of containers running/paused/stopped, root directory, server mode, runtime, logging drivers, etc...
4. Pull the centos6 and latest ubuntu image from repository using the following command
   1. docker pull centos:centos6
   2. docker pull ubuntu:latest



1. To list the currently available images, issue ‘docker images’.

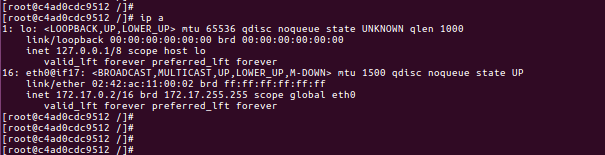


1. To run a container using docker, issue ‘docker run -it docker.io/centos:centos6 /bin/bash’
2. Run two containers based on ubuntu and centos6 images which were pulled before.
3. Once you run a docker container, your shell session will get attached to the container you deploy. Here you will notice that you are inside your container and can explore or start performing your tasks specific to that container such as creating an application or webserver etc. SSH connect to your docker host (Ubuntu desktop machine) again and run the following command to see the list of containers running; ‘docker ps’ . Issue ‘docker ps -a’ to list the containers which are stopped/exited.
4. To delete a container, use ‘docker rm *container-name\_OR\_ID’*

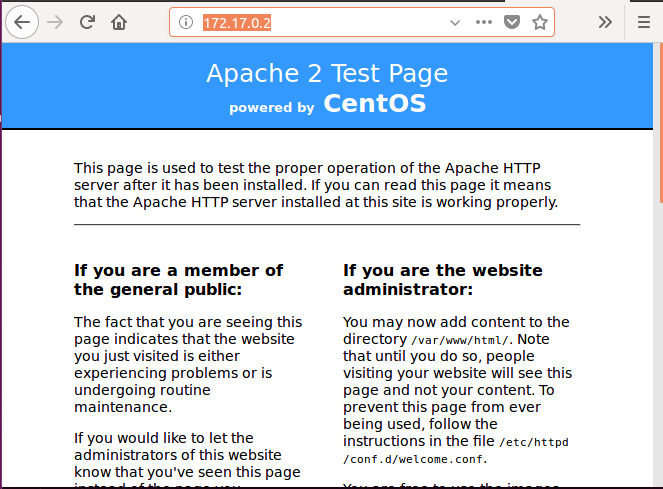
Delete the both the containers which we created in step 8.

Modifying and creating your own image:

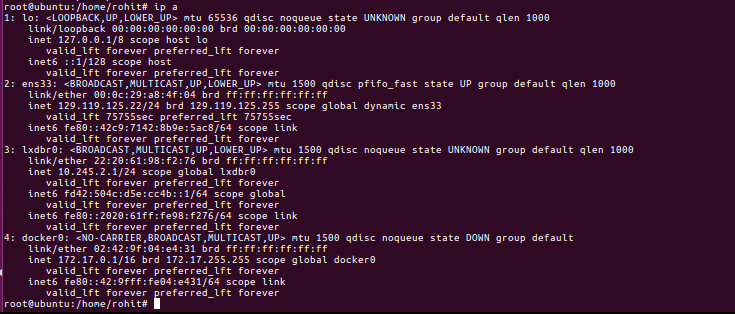
1. Launch a container based on centos6 image in docker

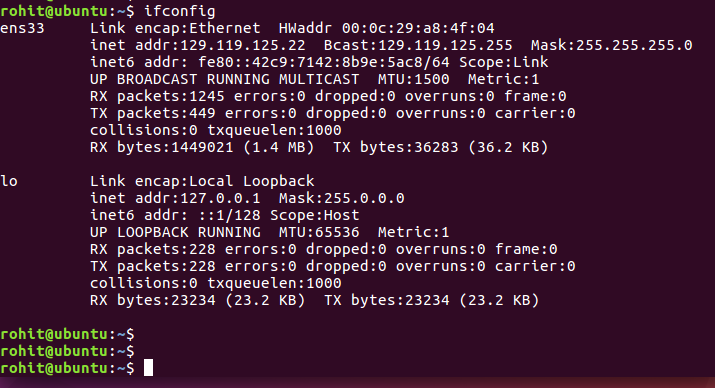


1. Once inside the container’s shell, install and enable the following packages;
   1. initscripts
   2. nano
   3. which
   4. httpd
   5. iproute
   6. sudo
   7. openssh-server
   8. telnet
   9. enable sshd and httpd service on boot by using command ‘chkconfig ssh on’ and ‘chkconfig httpd on’
   10. start sshd and httpd services
2. Once the above packages and services are installed, verify the ssh and http service are running or not.



1. exit the container. The container will be stopped. Now create a new image from this container and name it as centos6:WEB
2. Now, Launch a new container using the newly created image. You will notice the new container will already have the packages installed and you just need to enable the services now. This is the most important feature of docker which makes it an integral part of CI/CD.





1. Now again, the containers we launch on a docker host are running inside the hosted network. The services you enable on them are not available over the network. To expose the services running inside a container, we will have to expose the TCP/UDP port running on a container to local host. Exit the container you launched in previous step and relaunch it using the following command;
   1. docker run -it --name="test\_web" -p 8080:80 docker.io/centos6:WEB /bin/bash

Once the container is launched, start the httpd service and now, verify the web page by accessing your docker host’s IP with port 8080. (example <http://10.44.104.117:8080>, assuming 10.44.104.117 is your docker host).

