**Network Raptors**

1. **Objective:**

To integrate openvSwitch into docker for communication between multiple containers in different hosts.

1. **Technologies:**

Dockers

OpenDaylight

OpenvSwitch

WireShark

**Why Docker?**

To make it easier to create, deploy, and run applications by using containers that allow a developer to package up an application with all the parts it needs, such as libraries and other dependencies, and ship it all out as one package.

For fun!

[<https://www.powtoon.com/embed/cjDwT4Fa3hX/>](https://www.powtoon.com/embed/cjDwT4Fa3hX/)

**OpenDayLight**

OpenDaylight (ODL) is a modular open platform for customizing and automating networks of any size and scale. In this project, we are going to launch OpenDaylight in Docker.

**OpenvSwitch**

# Open vSwitch is an open-source implementation of a distributed virtual multilayer switch. In this project we are going to implement “Multi-Host Overlay Networking with Open vSwitch using dockers”.

**WireShark**

Wireshark is a free and open-source packet analyzer. It is used for network troubleshooting, analysis, software and communications protocol development, and education. In this project we are implementing WireShark via docker for analyzing the status of docker.

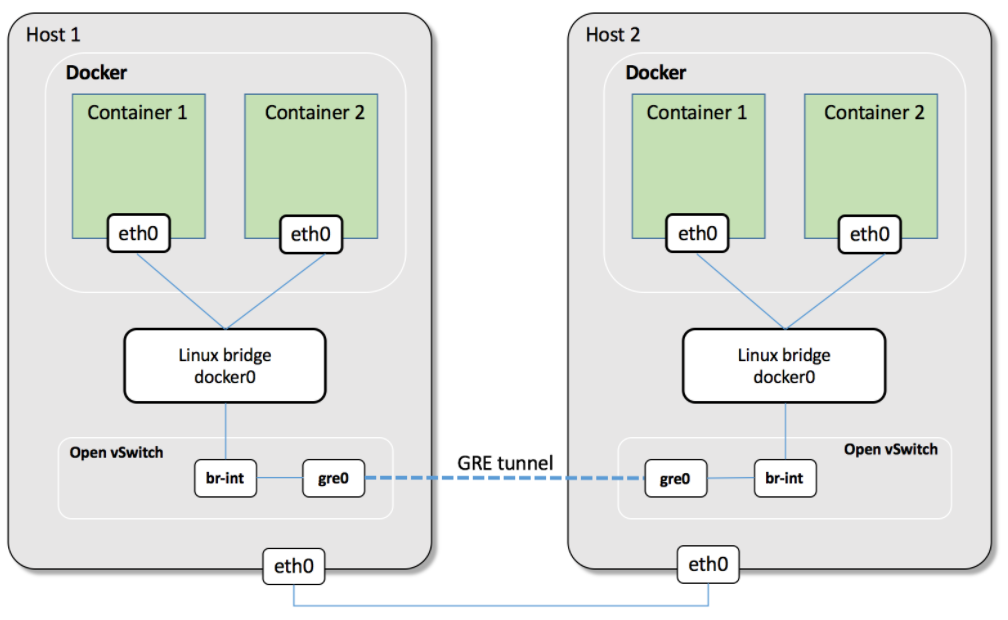
1. Implementation:
2. **Docker networking with openvSwitch**

Containers in different hosts (Two Ubuntu Nodes) can communicate with each other by exchanging packets of information. There are two ways to connect containers with open vSwitch.

1. connect default docker0 with ovs bridge
2. connect container with ovs bridge directly through veth pair.

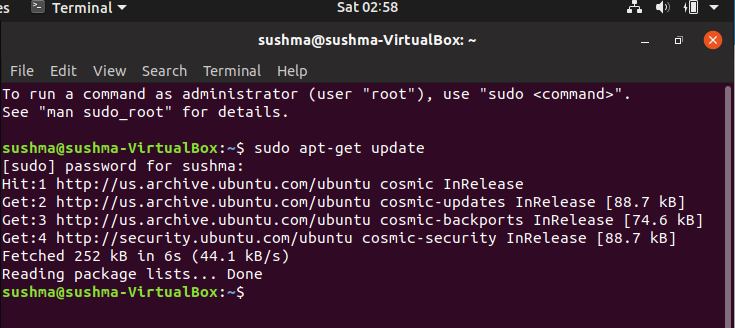
We chose the second way, because it’s bit complicated as we will need to do more work to connect containers with ovs and it helps us to get a better understanding of the concepts.

**Topology:**

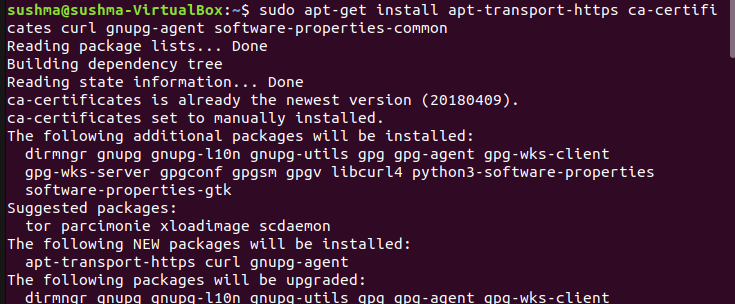


**Installation of Docker:**

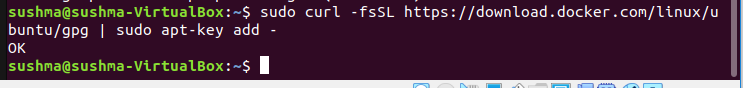
* Update all the packages

****

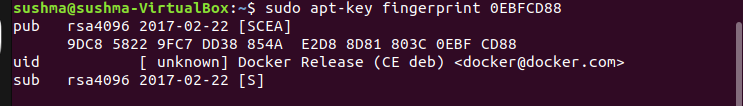
* Next, install a few prerequisite packages which let apt use packages over HTTPS

****

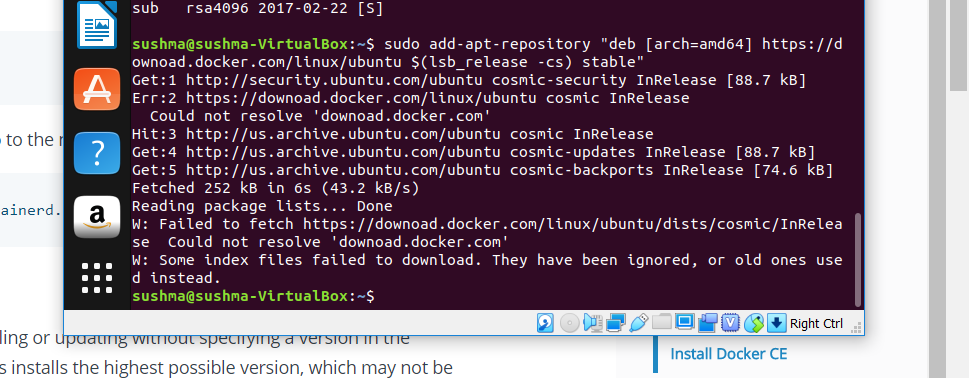
* Then add the GPG key for the official Docker repository to the system

****

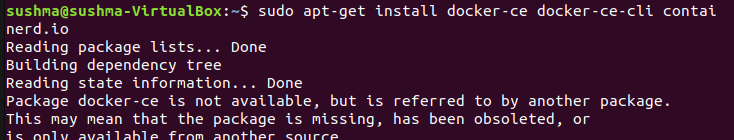
* Verifying that we now have the key with the fingerprint 9DC8 5822 9FC7 DD38 854A E2D8 8D81 803C 0EBF CD88, by searching for the last 8 characters of the fingerprint.

****

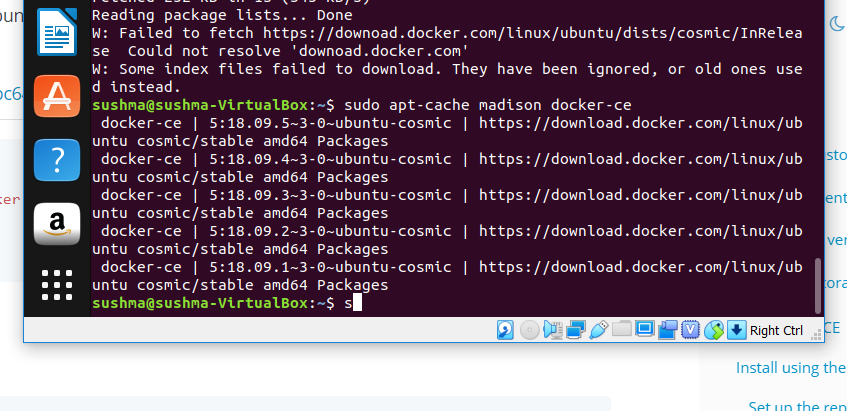
* Setting up the **stable** repository.



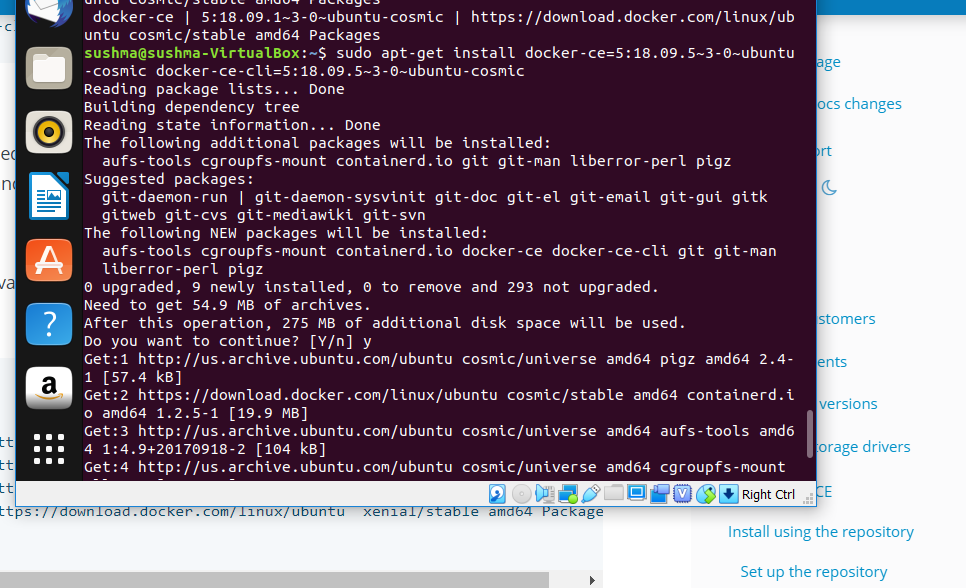
* Installing the latest version of Docker CE and container, or go to the next step to install a specific version

****

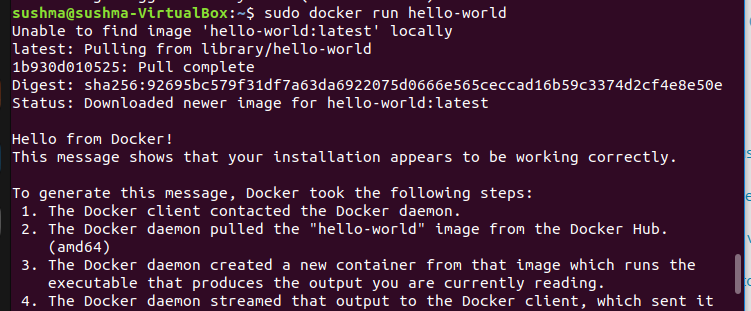
* To install a specific version of Docker CE, list the available versions in the repo, then select and install, list of the versions available in our repo:



* Install a specific version using the version string from the second column



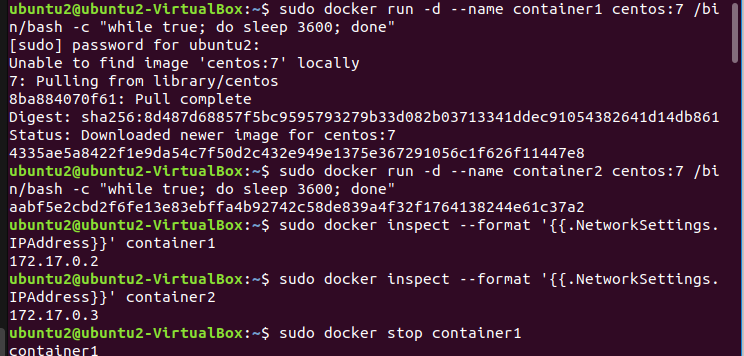
* Verifying that Docker CE is installed correctly by running the hello-world image.

****

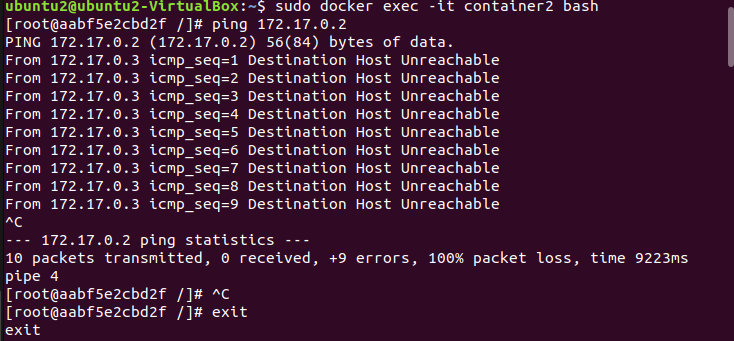
**OpenvSwitch and multihost communication set up**

**On Host 1**

* Start two containers on host 1 and Stop container 1 on host 1, because it has them same IP address as container 1 on host 2

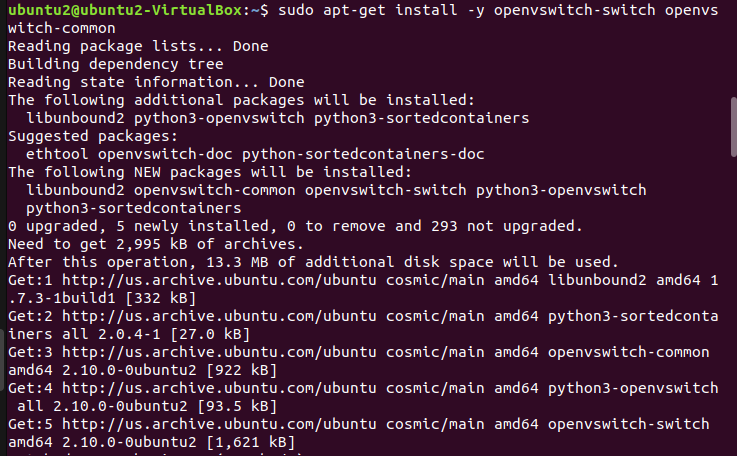
****

* Container 2 on host 1 cannot access container 1 on host 2 on ping

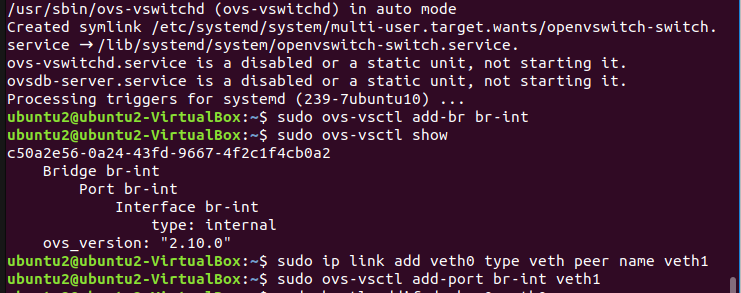
****

**Setting up the tunnel and Configure OVS:**

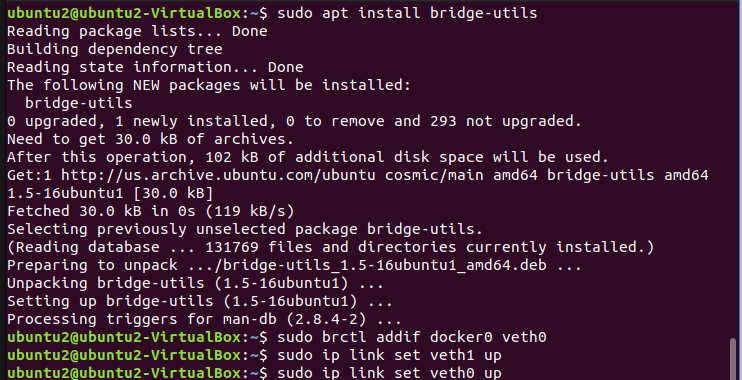
* Install OVS:

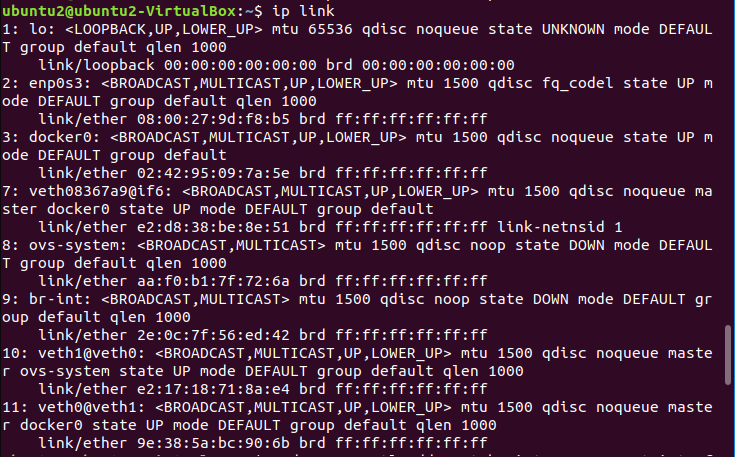
****

* Creating a ovs bridge and a veth pair

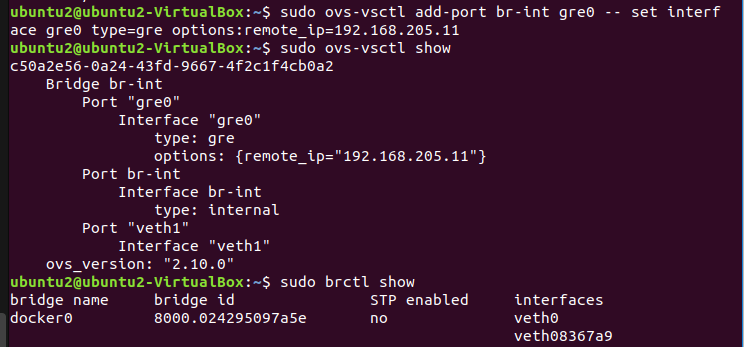
****

* Connecting veth pair with dockre0 and ovs bridge br-int, set them up.

****

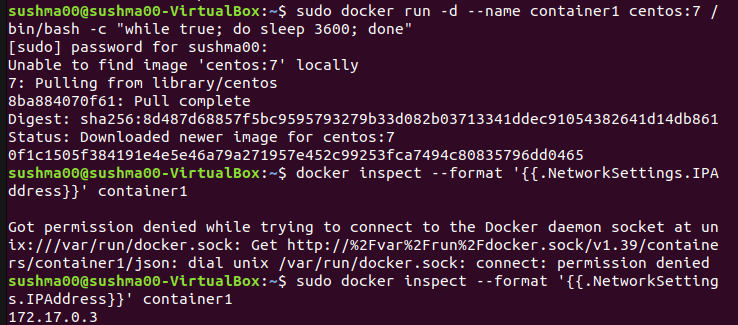
****

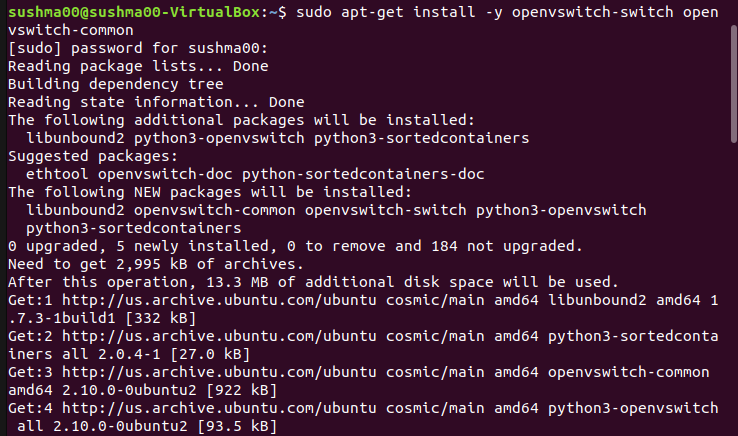
**Set up the GRE tunnel**

****

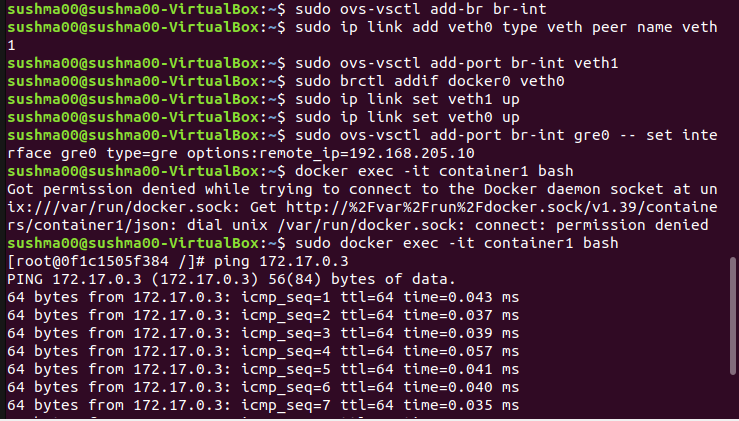
**On Host-2**

* Running all the commands to configure OVS and set up tunnel

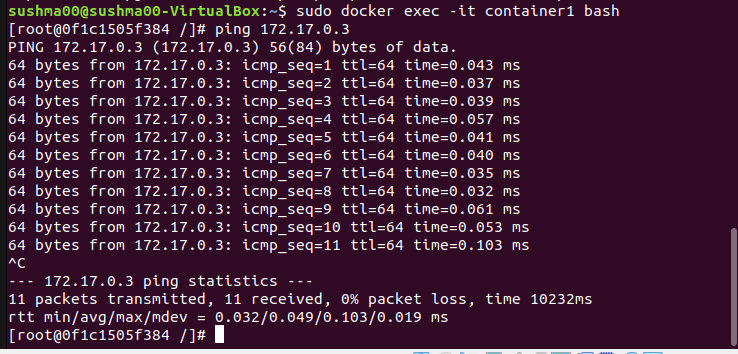
****

****

* Doing a ping to check if packets get exchanged.

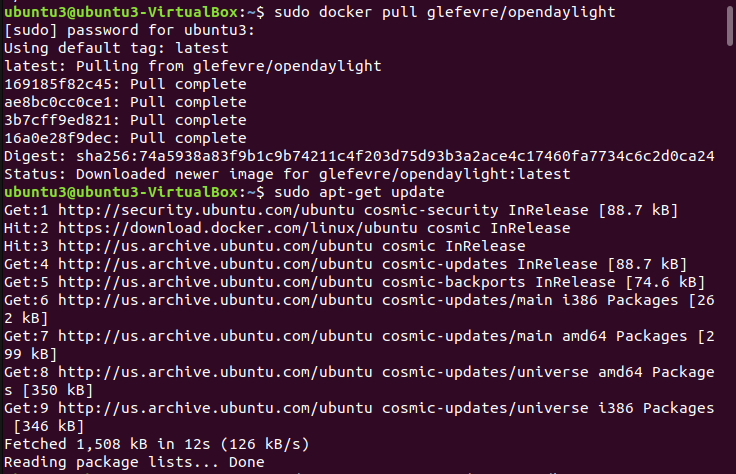
****

* Packet transmission takes place andin container1 on host 2 ping container 2 on host 1

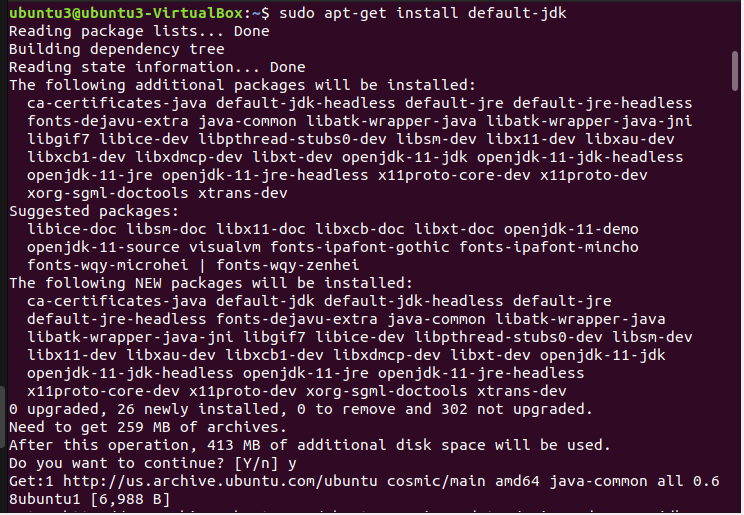
****

**Launching Opendaylight via Docker**

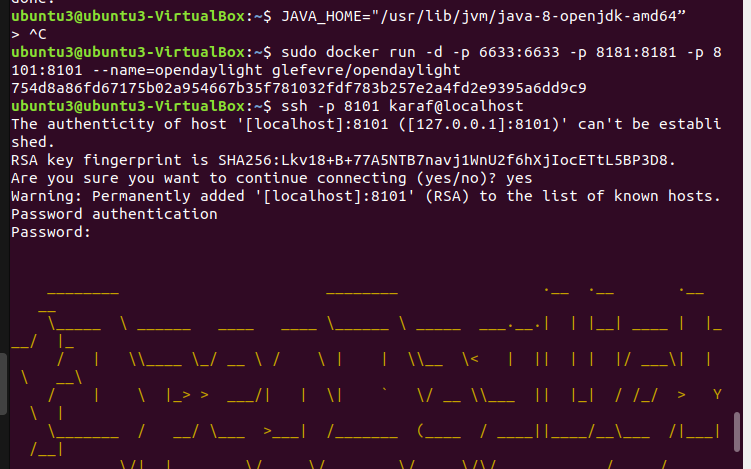
* Pull docker Image from Docker Hub

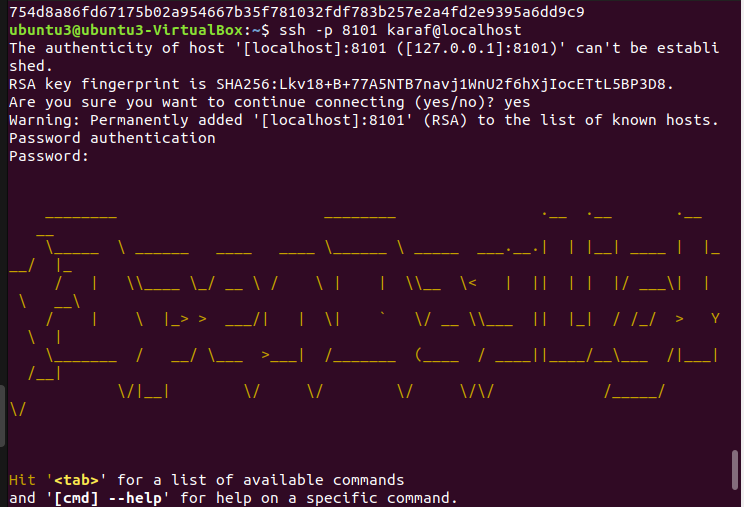
****

* Install JAVA Package

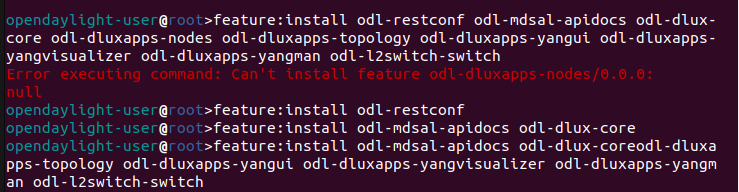
****

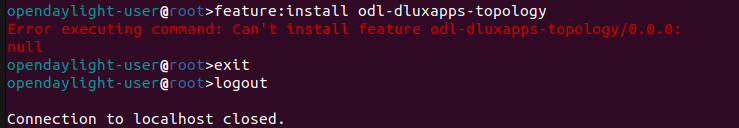
* Set JAVA HOME\_PATH and launch on the karaf CLI

****

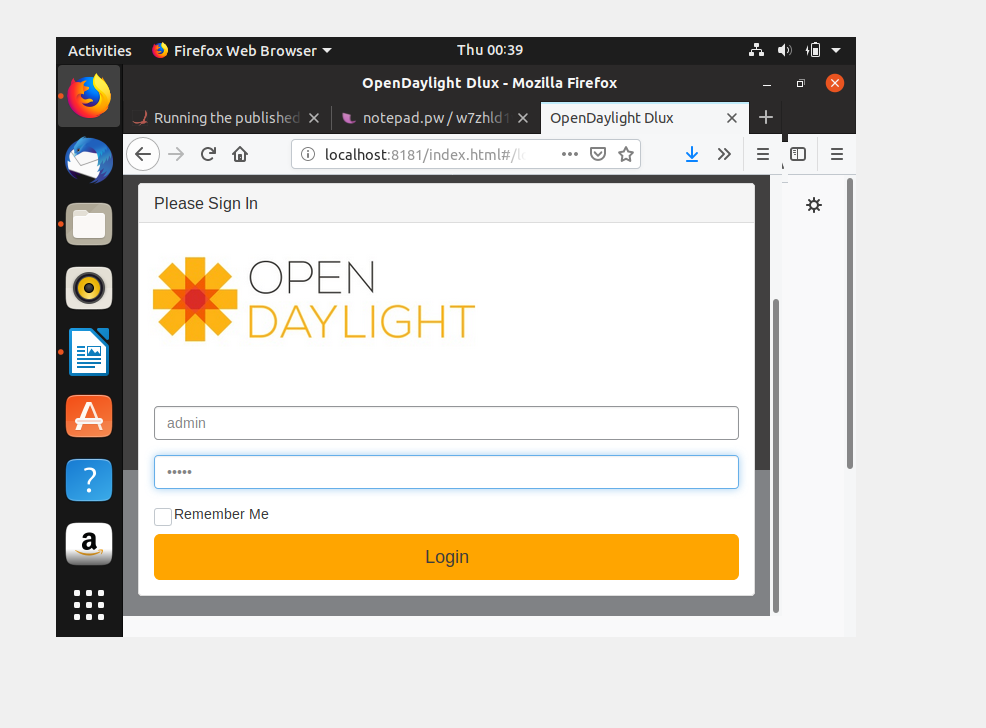
****

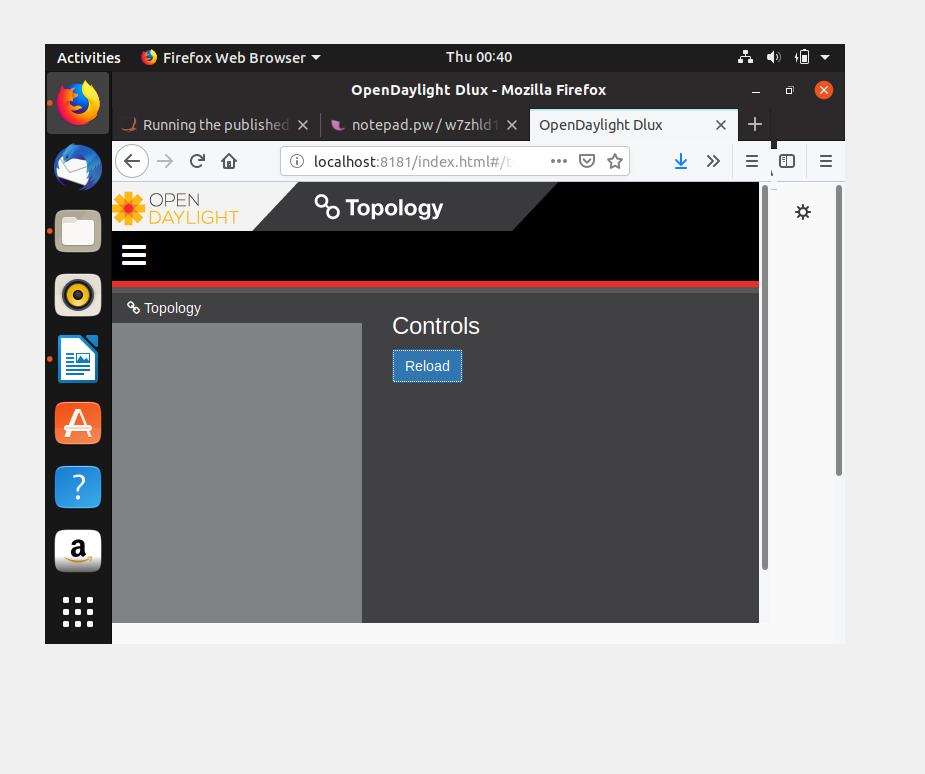
* Install features for opendaylight

****

****

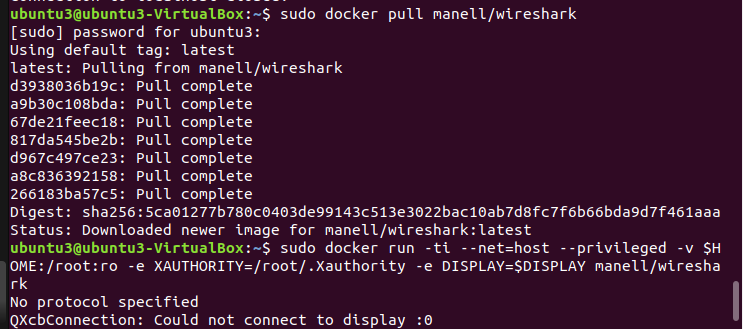
* Launching opendaylight using REST call in browser

****

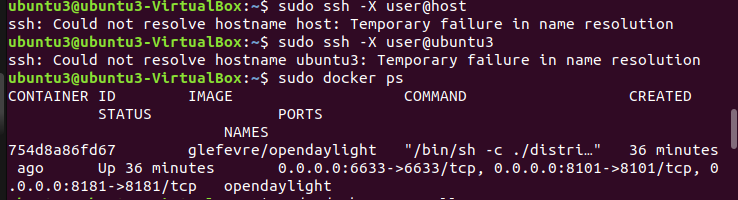
****

**Wireshark via Docker for analyzing status of Docker**

* Pull Wireshark Image

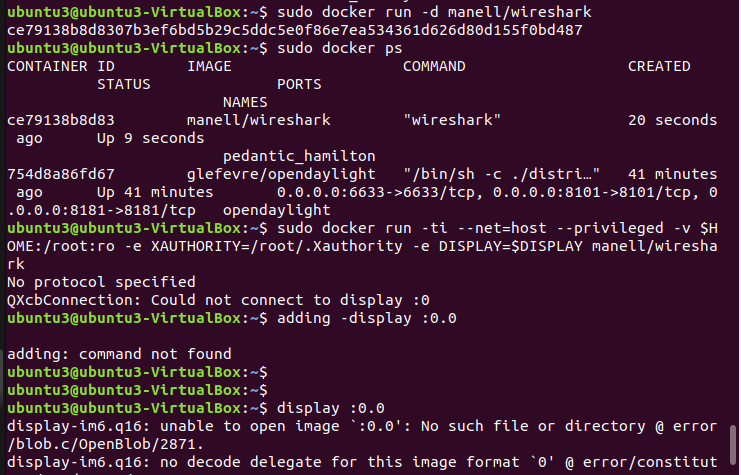
****

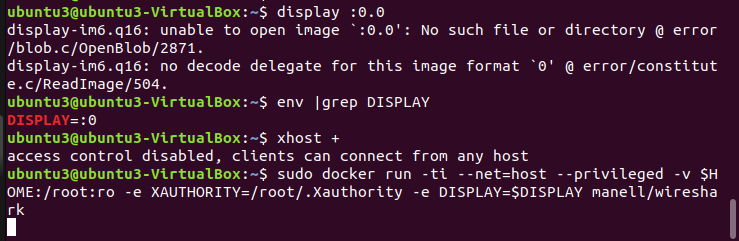
* Running the Wireshark container

****

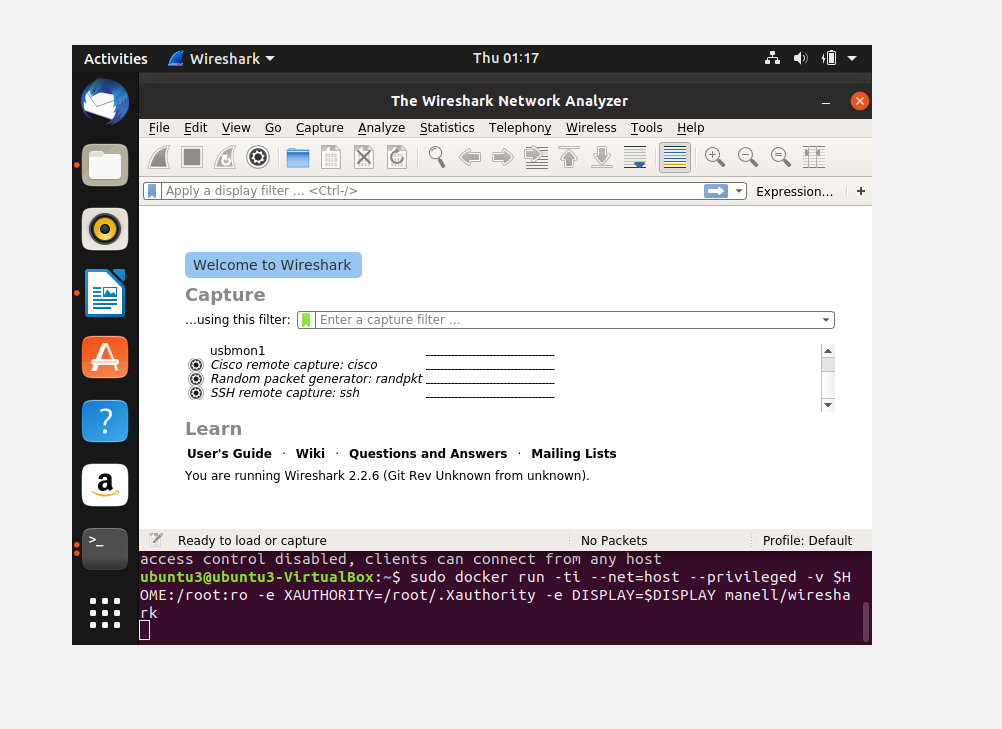
****

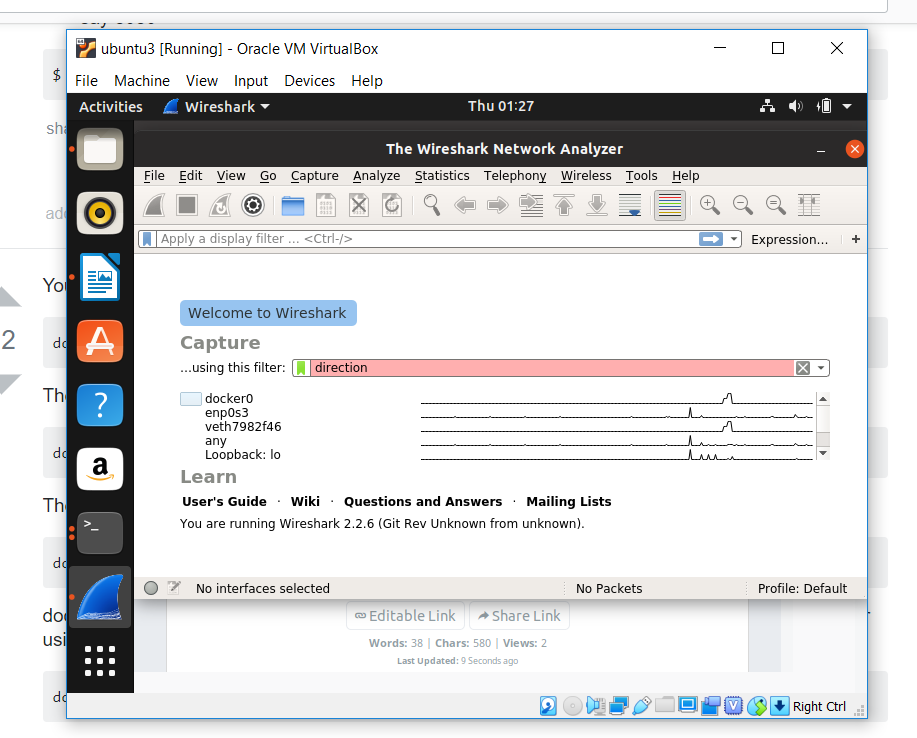
* Run the Wirshark image on the same host and open the display package required to launch Wireshark

****

****

* Launched the wireshark and analyzed the status of docker

****

****

1. **Opendaylight on Docker:**
2. Installation and configuration of docker.
3. Pull default image from public docker registry.
4. Run the docker image.
5. Start karaf console.
6. Install some features of opendaylight to test it.
7. **Challenges faced:**
8. Understanding networking in docker.
9. Container communication.
10. Creating machines as two different host with different IP’s.
11. Installation of opendaylight on docker because of dependency and packages issues.
12. Configuring tunnel connection between two hosts.
13. If container/tunnel creation fails, must delete the existing container/tunnel and create new tunnel.
14. Tunnel Creation.
15. Suo and SSH issues.
16. JAVA dependency issues.

**5. Conclusion**

1. With the increase in the requirements for a project, dockers will be of paramount important.
2. Based on the the requirements of the organization, we may use different techniques.
3. In this project we tried to implement and launch SDN based controller, switches and analyzers using dockers.
4. Containerization using Dockers will be more useful in a practical team-based environment.
5. Configure cluster of opendaylight instances where each hosted on docker containers.
6. Create a new docker network instead of using default docker0 bridge.

**Team:**

Priyanka Gadde – pxg170630

Venkatesh Gotimukul – vxg173330

Hasanth Sista – hxs171730

Sushma Nagaraj Sunkollu – sxs173433

Karthik Reddy Vudagandla – kxv170930