**Docker record**

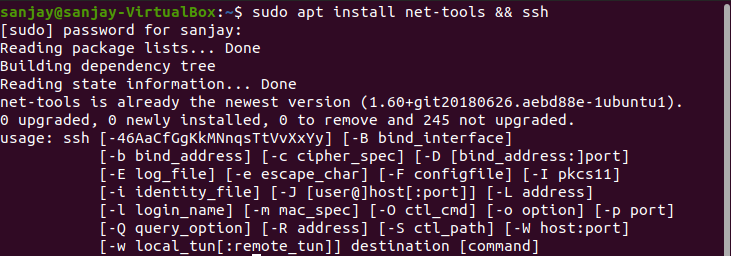
**1934012**

**1934036**

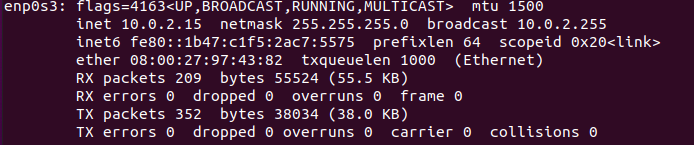
**1934044**

**Creating a Docker Container**

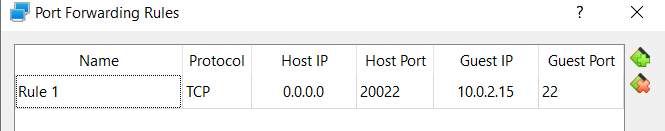
1. Install a Linux OS in VirtualBox.
2. Install net-tools and ssh.



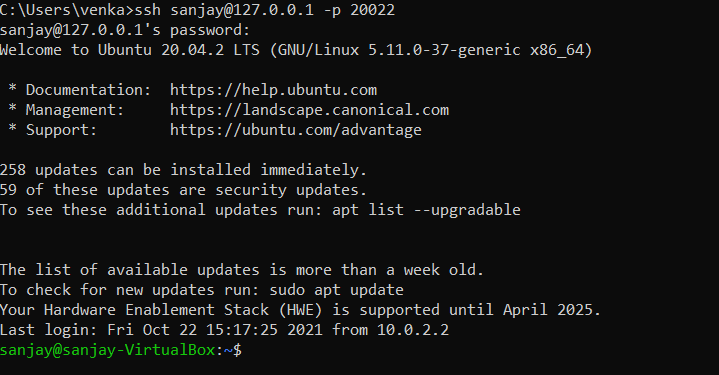
1. Find IP of the Linux guest machine.



1. In the VirtualBox Setting for this OS, Go to Network -> Advanced -> Port Forwarding, Give Host port as 20022, Guest IP found from ifconfig, and Guest port as 22. Save the changes.



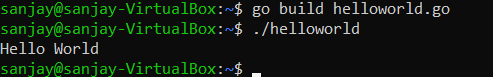
1. In the host machine, Open Command prompt. Type ssh Linux-username@127.0.0.1 -p 20022. Grant access if asked and type the guest Linux OS password.



1. Create a simple HelloWorld program in the language of your choice. (GoLang here)



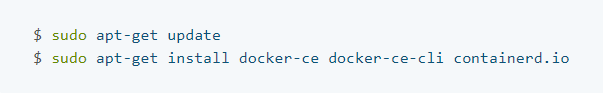
1. Build the program and verify output



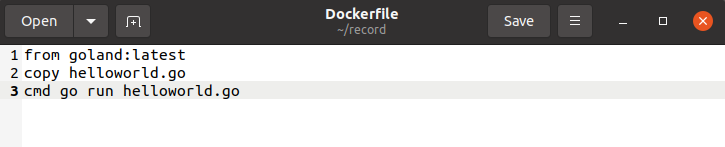
1. Go to https://docs.docker.com/engine/install/ubuntu/ and add the docker repository to the Ubuntu OS.



1. Install Docker



1. In Linux create a “Dockerfile” with the following contents



1. To build the container type “sudo docker build -t test:1 .”
2. To view the image do “sudo docker images”
3. To run the image do “sudo docker run test:1”

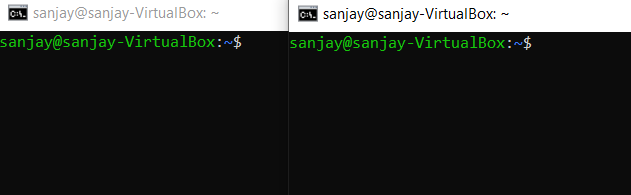


**Entering into Container making Logs of the activities**

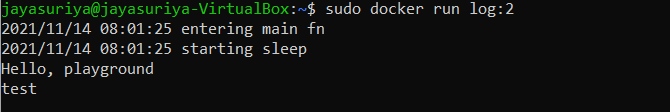
1. Modify the helloworld program to run for 5 minutes



1. Open two SSH terminals from host

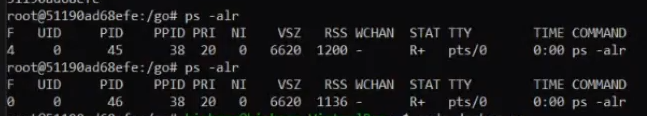


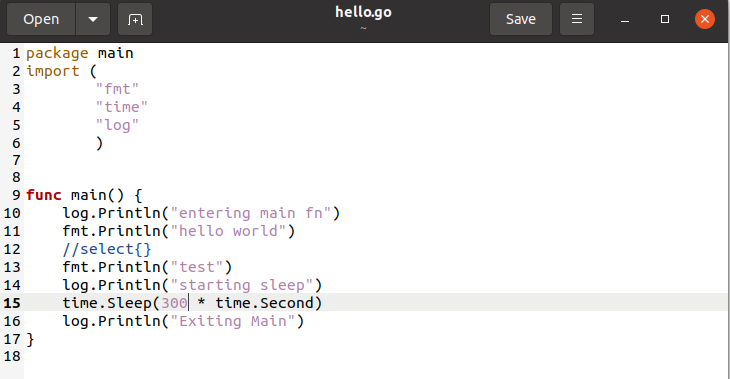
1. Build the modified hellworld program as a docker container and run it in SSH one terminal.



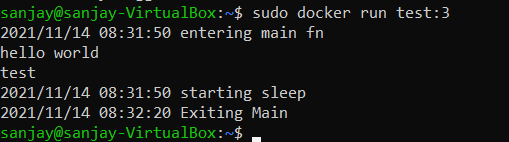
1. Enter into the running container from the other terminal using the following commands

sudo docker exec -it <Docker Image ID> bash

1. To view the process running inside the container do 
2. Add log comments to the file to check for any runtime errors



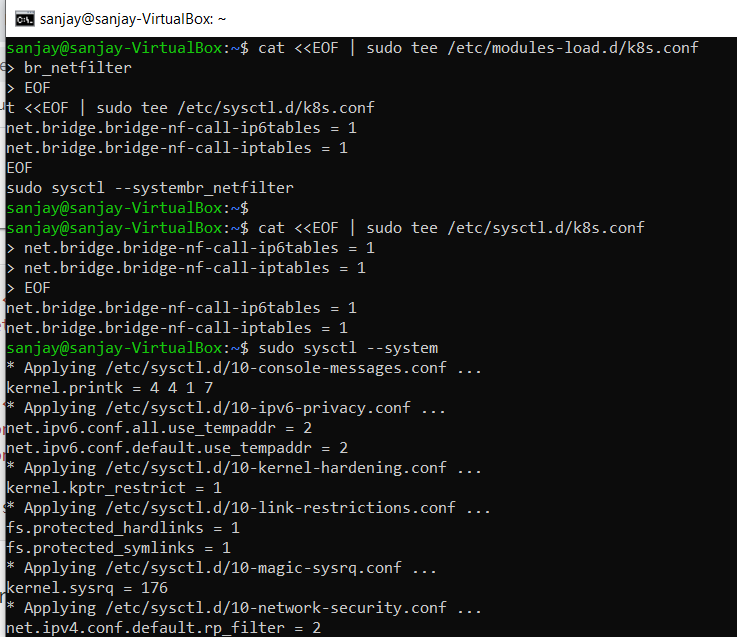
1. To run the file



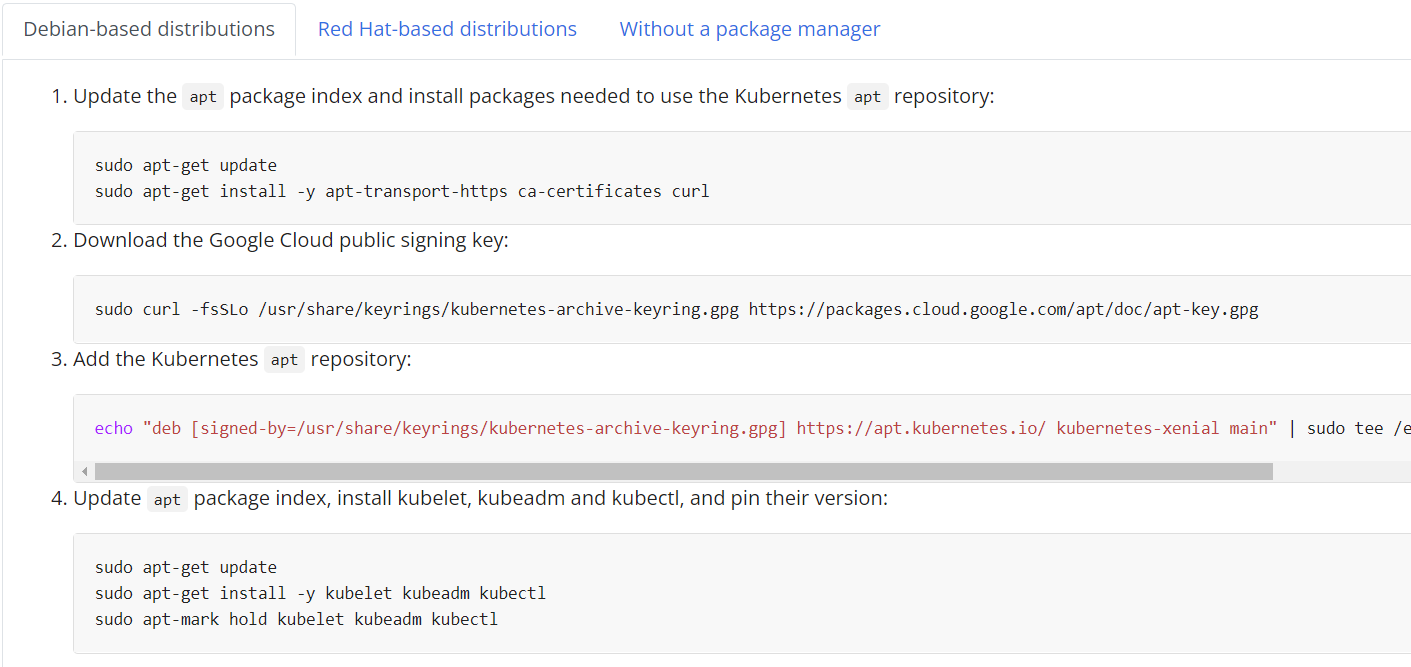
1. Instead of using the entire golang, we can use only the required packages using alpine

**Installing Kubernetes**

1. Go to <https://kubernetes.io/docs/setup/production-environment/tools/kubeadm/install-kubeadm/>
2. Install the prerequisites as mentioned in the website



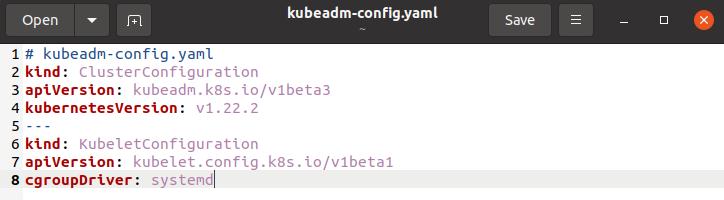
1. To install kubeadm, kubectl, kubelet type the commands as mentioned in the website



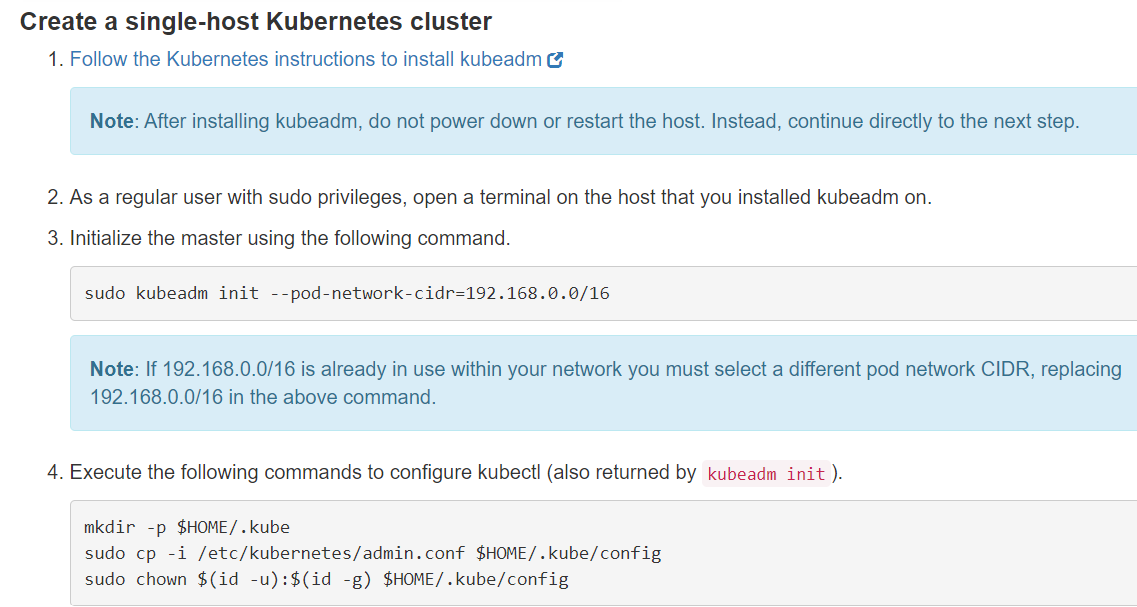
1. To initialize kubeadm, use sudo kubeadm init.



1. Turn off swap using swap -a
2. Create a kube configuration yaml file



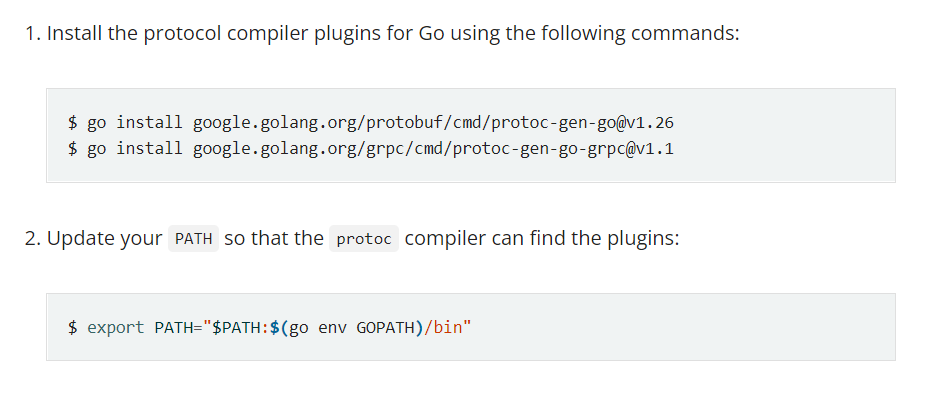
1. Use “kubectl get pods -n kube-system” to see the pods.
2. Add Calico for Kuberentes from <https://docs.projectcalico.org/getting-started/kubernetes/quickstart>
3. Create a helloworld kubernetes file by following the steps in the GitHub page <https://github.com/paulbouwer/hello-kubernetes>



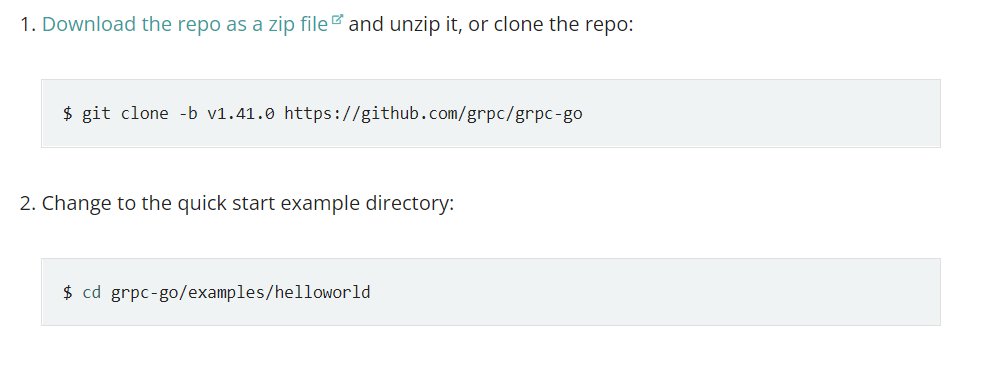
1. Create a helloworld kubernetes file by following the steps in the GitHub page <https://github.com/paulbouwer/hello-kubernetes>

**Creating a GRPC server client program.**

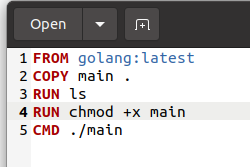
1. Install grpc plugins for golang



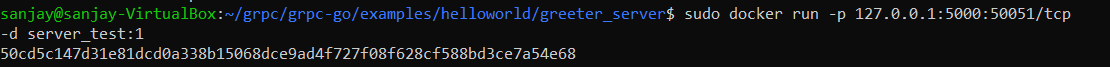
1. Get the example code



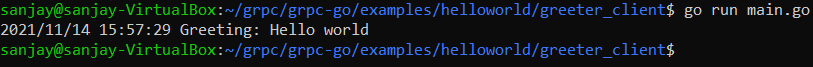
1. Using the command go build main.go to build the main.go file
2. Create a docker file for the grpc server and client program



1. Build both the docker files
2. Change port as 50051 in the greeter go file and enable port forwarding in server using

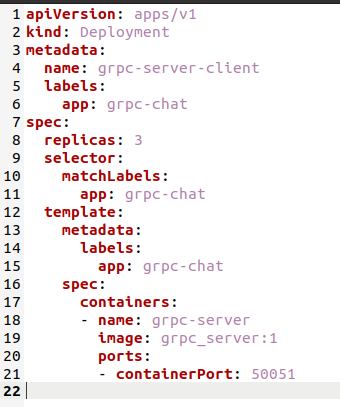


1. Build both the docker files



**Containerizing the GRPC server client program using Kubernetes**

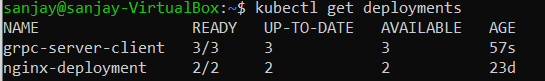
1. Create a Kubernetes deployment YAML file from <https://kubernetes.io/docs/concepts/workloads/controllers/deployment/>
2. Make the necessary changes in the file as below



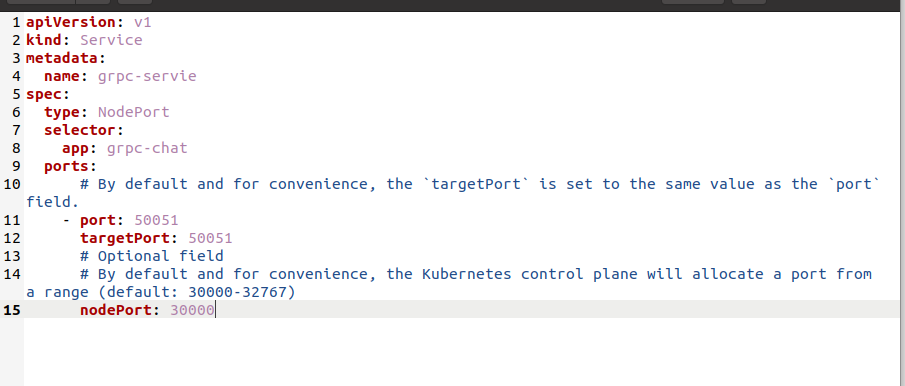
1. Create the deployment using



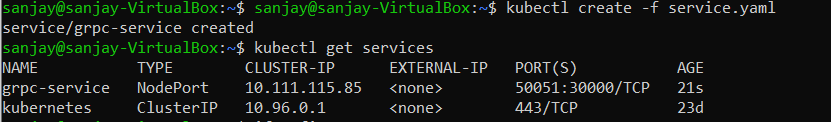
1. See the deployments using kubectl get deployments



1. Create service.yaml file from <https://kubernetes.io/docs/concepts/services-networking/service/>



1. Deploy the services



1. Change the IP in the greeter\_client file and run the program to see the results.



