Create Kubernetes cluster with Kubeadm

Including a master node and a worker node

Create virtual machines

- - It is needed to create two virtual machines (one for each node)
- - Ubuntu 20.04 TLS
- - Need to create two virtual machines with the normal process

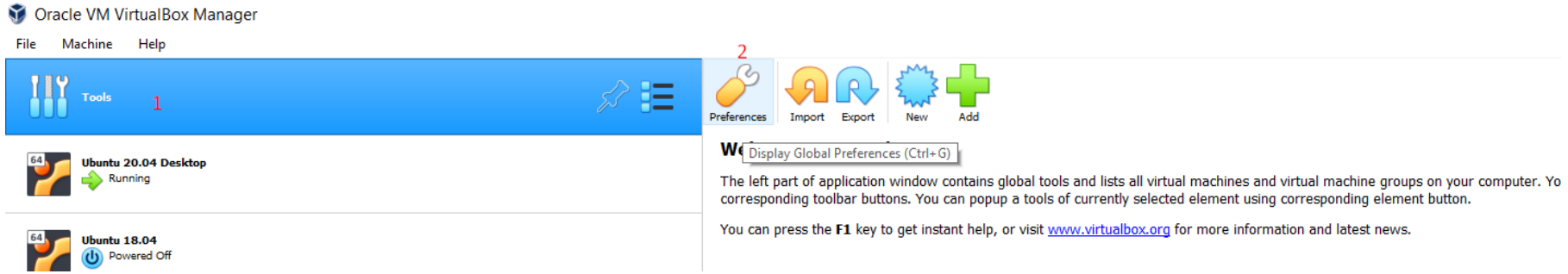
Role	IP	OS	RAM	CPU
Master	172.16.0.4*	Ubuntu 20.04 TLS	2Gb	2
Worker	172.16.0.5*	Ubuntu 20.04 TLS	1Gb	1

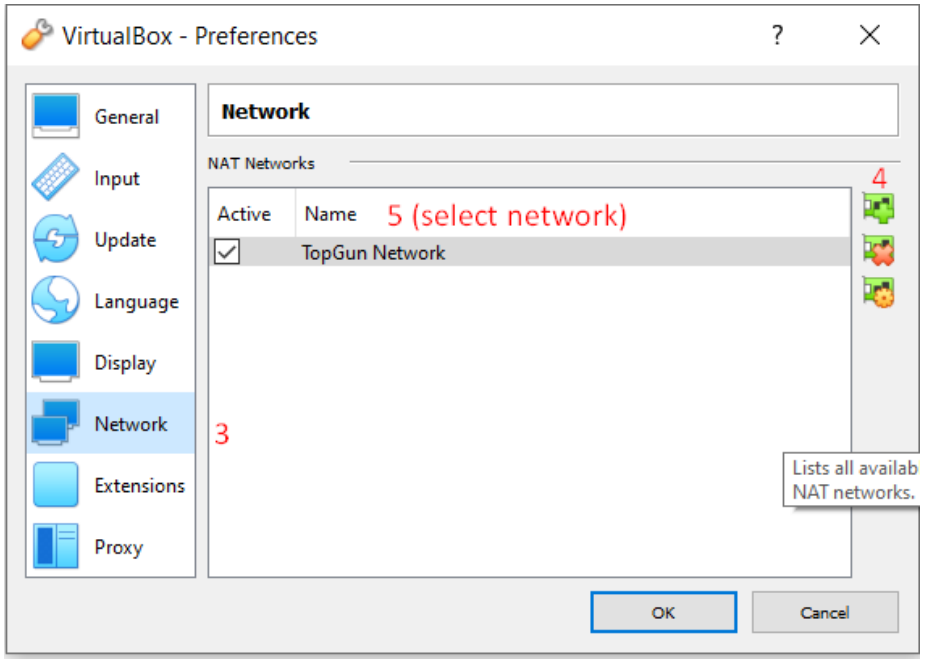
- After creating VM, they must be started. Then, you use the Ubuntu 20.04 TLS ISO file to put the OS.
- Install Ubuntu with minimum requirements, nothing else needed. Give the VM names like master, worker, or easy identifiable names, as they will be much used in future.

* The Ips are assigned automatically by the DHCP after we create a virtual box network (in next slide)

Create network for virtual machines

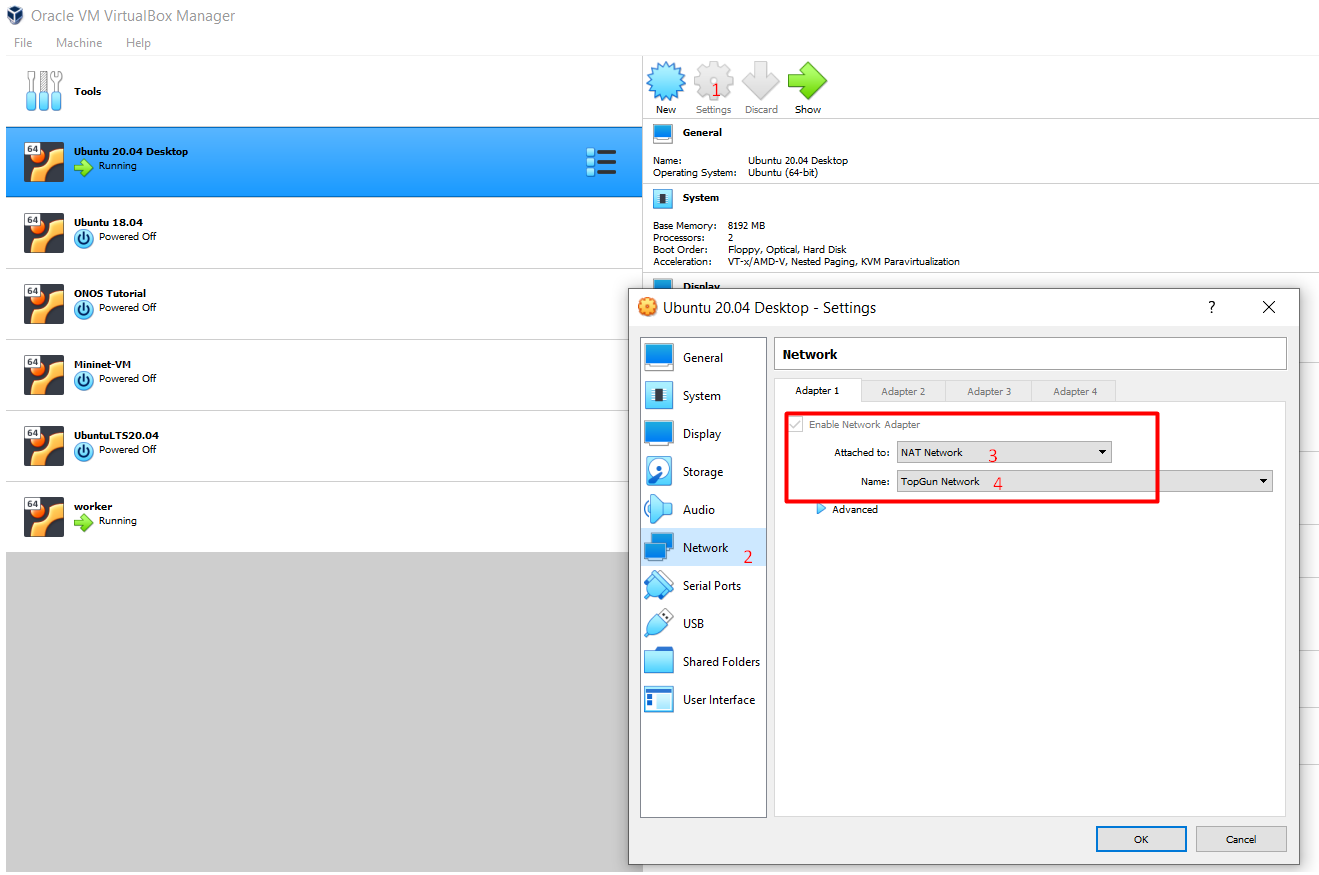
- In virtual Box, a network must be created for the VMs to talk among them





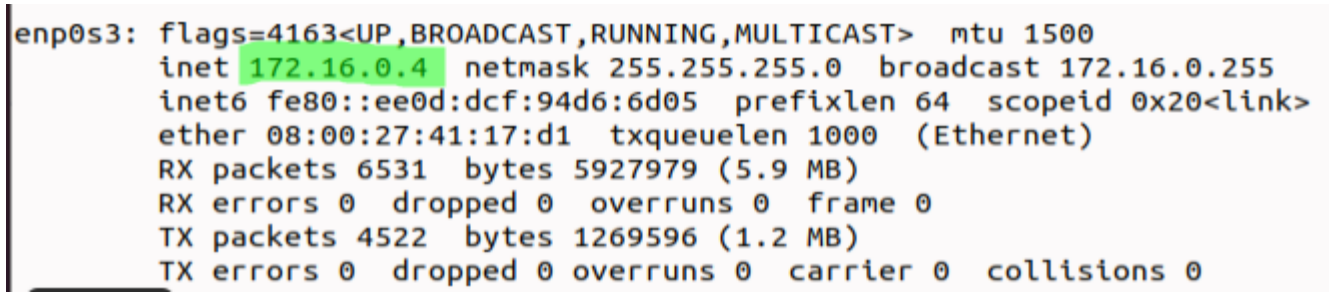
Create network for virtual machines (2)

- Need to access the settings of each VM (two in our case, which will be master node and worker node), and select in the network settings NAT network (in our case, TopGun)



Steps for both VM to work before installing Kubernetes

- Execute commands:
  - sudo apt install net-tools -> Install ifconfig commands and etc
  - sudo apt-get install openssh-server -> Install to allow to do ssh connections among VMs if needed (probably needed in future for testing purposes).
- Check IP of VM
  - Execute command ifconfig -> As a result, it gives several interfaces. The one with name epn0s3 is the IP of the VM.



Test Ips, SSH

- - Try a SSH connection from one VM (master) to the other (worker), as a last step to check that the network is working fine
  - Execute command ssh *user@ip**
  - **Ej. ssh mastervm@172.16.0.4*
- In this moment, everything is ready to create Kubernetes cluster with Kubeadm.

Steps for both VM (Master and worker)

- Log in as root user (every command from now on needs to be executed as root): *sudo su –* *(or putting sudo before every command, whichever you prefer)*
- Disable Firewall: *ufw disable*
- Disable swap: *swapoff -a; sed -i '/swap/d' /etc/fstab*
- Update sysctl settings for Kubernetes networking: cat *>>/etc/sysctl.d/kubernetes.conf<<EOF*

net.bridge.bridge-nf-call-ip6tables = 1

net.bridge.bridge-nf-call-iptables = 1

EOF

sysctl --system

https://www.youtube.com/watch?v=mMmxMoprxiY

https://github.com/justmeandopensource/kubernetes/blob/master/docs/install-cluster-ubuntu-20.md

Install Docker Engine

- *apt install -y apt-transport-https ca-certificates curl gnupg-agent software-properties-common*
- *curl -fsSL https://download.docker.com/linux/ubuntu/gpg | apt-key add -*
- *add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable"*
- *apt update*
- *apt install -y docker-ce=5:19.03.10~3-0~ubuntu-focal containerd.io*

Install Kubernetes

- curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | apt-key add -
- echo "deb https://apt.kubernetes.io/ kubernetes-xenial main" > /etc/apt/sources.list.d/kubernetes.list
- *apt update && apt install -y kubeadm=1.18.5-00 kubelet=1.18.5-00 kubectl=1.18.5-00*

- ** Already installs kubectl, kubeadm and kubelet*

https://www.youtube.com/watch?v=mMmxMoprxiY

https://github.com/justmeandopensource/kubernetes/blob/master/docs/install-cluster-ubuntu-20.md

Steps for only Master node (1)

Always with root permission, with command sudo su -) or putting sudo in front of every command

- Initialize Kubernetes : sudo kubeadm init --pod-network-cidr=192.168.0.0/16
- Ejecutar los comandos que vienen a continuacion:

mkdir -p $HOME/.kube

sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

sudo chown $(id -u):$(id -g) $HOME/.kube/config

- **Dowload Calico,yaml** file from website ***https://docs.projectcalico.org/manifests/calico.yaml***
- **Edit the Calico.yaml file**. Since the BGP has a method that the pods connect to the first IP that is available and working, it can lead to errors when you have many interfaces in a machine. To solve this, we need to edit the Calico.yaml file and change the following values corresponding to section *auto-detect the BGP IP address*. Instead, the type (or name) should be "**IP_AUTODETECTION_METHOD**" and the value selected is "**can-reach=172.16.0.4**" which is the IP of my master node. That means, BGP will choose only an IP for the node that can connect to my master node.

```
3848  # Auto-detect the BGP IP address.
3849  - name: IP
3850    value: "autodetect"
```

Value by default: BAD

```
# Auto-detect the BGP IP address.
- name: IP_AUTODETECTION_METHOD
  value: "can-reach=172.16.0.4"
```

Personalized value: GOOD

https://www.youtube.com/watch?v=mMmxMoprxiY

https://docs.projectcalico.org/reference/node/configuration#ip-autodetection-methods

https://github.com/justmeandopensource/kubernetes/blob/master/docs/install-cluster-ubuntu-20.md

Steps for only Master node (2)

Always with root permission, with command sudo su -) or putting sudo in front of every command

- *Once Calico.yaml file is edited, execute command: kubectl apply –f calico.yaml (file is supposed to be in same directory where command is executed)*
- After a couple minutes, execute command: *watch kubectl get pods -n kube-system (if OK, everypod is running with status 1/1)*

```
root@crubio-VirtualBox:~# kubectl get pods -n kube-system -o wide
NAME                                          READY   STATUS    RESTARTS   AGE   IP               NODE              NOMINATED NODE   READINESS GATES
calico-kube-controllers-746f9d75cb-9j8zq      1/1     Running   1          81m   192.168.63.70    crubio-virtualbox <none>           <none>
calico-node-6hklg                             1/1     Running   1          81m   172.16.0.4       crubio-virtualbox <none>           <none>
calico-node-7rskg                             1/1     Running   1          77m   172.16.0.5       worker            <none>           <none>
calico-node-mhcps                             1/1     Running   1          41m   172.16.0.6       worker2           <none>           <none>
coredns-66bff467f8-q55kj                      1/1     Running   1          83m   192.168.63.68    crubio-virtualbox <none>           <none>
coredns-66bff467f8-xh58w                      1/1     Running   1          83m   192.168.63.69    crubio-virtualbox <none>           <none>
etcd-crubio-virtualbox                        1/1     Running   1          83m   172.16.0.4       crubio-virtualbox <none>           <none>
kube-apiserver-crubio-virtualbox              1/1     Running   1          83m   172.16.0.4       crubio-virtualbox <none>           <none>
kube-controller-manager-crubio-virtualbox     1/1     Running   1          83m   172.16.0.4       crubio-virtualbox <none>           <none>
kube-proxy-5qtcx                              1/1     Running   1          41m   172.16.0.6       worker2           <none>           <none>
kube-proxy-w6g9h                              1/1     Running   1          83m   172.16.0.4       crubio-virtualbox <none>           <none>
kube-proxy-x5brk                              1/1     Running   1          77m   172.16.0.5       worker            <none>           <none>
kube-scheduler-crubio-virtualbox              1/1     Running   1          83m   172.16.0.4       crubio-virtualbox <none>           <none>
root@crubio-VirtualBox:~#
```

Master
Worker 1
Worker 2

This picture is from the end, but it is only to show how columns “READY” and “STATUS ” should look like

- *Create Cluster join command (to create worker nodes for the master node): kubeadm token create --print-join-command*
- *It will generate something like this (it is the command needed to be executed in worker node to join to a master)*:

```
root@crubio-VirtualBox:~# kubeadm token create --print-join-command
W0902 14:06:49.787789   36815 configset.go:202] WARNING: kubeadm cannot validate component configs for API groups [kubelet.config.k8s.io kubeproxy.config.k8s.io]
kubeadm join 172.16.0.4:6443 --token u6w38m.zxq0h0b1rub5sh14     --discovery-token-ca-cert-hash sha256:7960f7074722ef522c00813db8f4e352ce0c0c43e9390f98e83712ca689f2f1c
root@crubio-VirtualBox:~#
```

https://www.youtube.com/watch?v=mMmxMoprxiY

https://github.com/justmeandopensource/kubernetes/blob/master/docs/install-cluster-ubuntu-20.md

Steps for each worker node

- Copy that command in yellow (result of previous slide) and execute in each node (Virtual Machine) that you want to join the cluster.

```
root@crubio-VirtualBox:~# kubeadm token create --print-join-command
W0902 14:06:49.787789   36815 configset.go:202] WARNING: kubeadm cannot validate component configs for API groups [kubelet.config.k8s.io kubeproxy.config.k8s.io]
kubeadm join 172.16.0.4:6443 --token u6w38m.zxq0h0b1rub5sh14     --discovery-token-ca-cert-hash sha256:7960f7074722ef522c00813db8f4e352ce0c0c43e9390f98e83712ca689f2f1c
root@crubio-VirtualBox:~#
```

- Expected result is a success message saying that “Execute kubectl get nodes in your control-plane node (master node)” to view nodes in the cluster.

```
root@worker:~# kubeadm join 172.16.0.4:6443 --token swt0zy.v7y5ux302295p3z1 --discovery-token-ca-cert-hash sha256:7960f7074722ef522c00813db8f4e352ce0c0c43e9390f98e83712ca689f2f1c
W0902 14:09:16.124100    7775 join.go:346] [preflight] WARNING: JoinControlPane.controlPlane settings will be ignored when control-plane flag is not set.
[preflight] Running pre-flight checks
        [WARNING IsDockerSystemdCheck]: detected "cgroupfs" as the Docker cgroup driver. The recommended driver is "systemd". Please follow the guide at https://kubernetes.io/docs/setup/cri/
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -oyaml'
[kubelet-start] Downloading configuration for the kubelet from the "kubelet-config-1.18" ConfigMap in the kube-system namespace
[kubelet-start] Writing kubelet configuration to file "/var/lib/kubelet/config.yaml"
[kubelet-start] Writing kubelet environment file with flags to file "/var/lib/kubelet/kubeadm-flags.env"
[kubelet-start] Starting the kubelet
[kubelet-start] Waiting for the kubelet to perform the TLS Bootstrap...

This node has joined the cluster:
* Certificate signing request was sent to apiserver and a response was received.
* The Kubelet was informed of the new secure connection details.

Run 'kubectl get nodes' on the control-plane to see this node join the cluster.

root@worker:~#
```

https://www.youtube.com/watch?v=mMmxMoprxiY

https://github.com/justmeandopensource/kubernetes/blob/master/docs/install-cluster-ubuntu-20.md

Checking that cluster is up

- Only from the master node, executing the command *kubectl get nodes*, the output should be:

```
root@crubio-VirtualBox:~# kubectl get nodes
NAME                STATUS   ROLES    AGE   VERSION
crubio-virtualbox   Ready    master   87m   v1.18.5
worker              Ready    <none>   81m   v1.18.5
worker2             Ready    <none>   44m   v1.18.5
root@crubio-VirtualBox:~#
```

- If everything is good, after joining two nodes to the cluster, the command kubectl get pods –n kube-system –o wide should look like this:

```
root@crubio-VirtualBox:~# kubectl get pods -n kube-system -o wide
NAME                                        READY   STATUS    RESTARTS   AGE   IP              NODE                NOMINATED NODE   READINESS GATES
calico-kube-controllers-746f9d75cb-9j8zq    1/1     Running   1          81m   192.168.63.70   crubio-virtualbox   <none>           <none>
calico-node-6hklg                           1/1     Running   1          81m   172.16.0.4      crubio-virtualbox   <none>           <none>
calico-node-7rskg                           1/1     Running   1          77m   172.16.0.5      worker              <none>           <none>
calico-node-mhcps                           1/1     Running   1          41m   172.16.0.6      worker2             <none>           <none>
coredns-66bff467f8-q55kj                    1/1     Running   1          83m   192.168.63.68   crubio-virtualbox   <none>           <none>
coredns-66bff467f8-xh58w                    1/1     Running   1          83m   192.168.63.69   crubio-virtualbox   <none>           <none>
etcd-crubio-virtualbox                      1/1     Running   1          83m   172.16.0.4      crubio-virtualbox   <none>           <none>
kube-apiserver-crubio-virtualbox            1/1     Running   1          83m   172.16.0.4      crubio-virtualbox   <none>           <none>
kube-controller-manager-crubio-virtualbox   1/1     Running   1          83m   172.16.0.4      crubio-virtualbox   <none>           <none>
kube-proxy-5qtcx                            1/1     Running   1          41m   172.16.0.6      worker2             <none>           <none>
kube-proxy-w6g9h                            1/1     Running   1          83m   172.16.0.4      crubio-virtualbox   <none>           <none>
kube-proxy-x5brk                            1/1     Running   1          77m   172.16.0.5      worker              <none>           <none>
kube-scheduler-crubio-virtualbox            1/1     Running   1          83m   172.16.0.4      crubio-virtualbox   <none>           <none>
root@crubio-VirtualBox:~#
```

Master
Worker 1
Worker 2