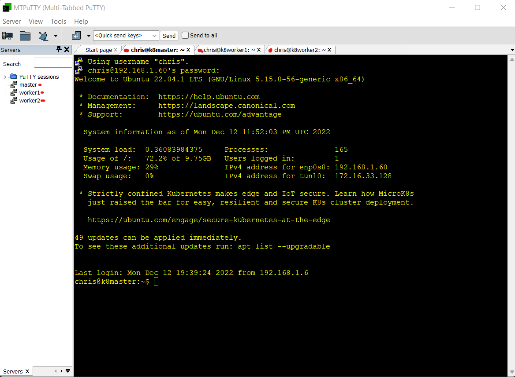
Creating a Kubernetes Cluster on your very own Windows 10/11 client for self-testing.

Can be used to prep for the CKA and CKAD certifications.

1. Test Bed Requirements:
2. Your laptop – This guide will be for Windows 10/11 users. (for MAC users – to be continued…)
3. Oracle VM Box – download oracle vm box: <https://www.virtualbox.org/wiki/Downloads>
4. Ubuntu22.04.1 Server “.iso” image – image name: ubuntu-22.04.1-live-server-amd64.iso
   1. download site: <https://ubuntu.com/download/server>
   2. Note: of course, you can and should grab the latest Ubuntu version. As of this publication 22.04.1 was the latest and greatest.
5. Your test bed will look like this:
   1. Master vm **##Note:** In the creation of master vm, you will see “master1” further into this part of the demo, just ignore the 1 when you see it. Think of it as just “master”
   2. Worker1 vm
   3. Worker2 vm
6. HIGHLY SUGGESTED: Download MPutty, very good tool so you can copy and paste between nodes. It will be a struggle to copy and paste between vm’s, so you tons of freedom when Mputty into all 3 vm’s(master, worker1, worker2). Here is a screen shot of my final setup using Mputty emulator



To create a mputty connection to your vm’s, I will show you how to create a connect to master.

Graphical user interface, application

Description automatically generated

1. Setting up your test bed
2. Download and install Oracle VM Box – this is a common exe install, follow defaults
3. Download Ununtu22.04.1 Server .iso
   1. Download Ubuntu22.04.1 .iso into your Downloads Folder
   2. Create a directory called: ubuntu24
      1. Example: /Users/ubuntu24
   3. Bring up Oracle VM Virtual Box Manager click on “New”

Graphical user interface, text, application

Description automatically generated

* 1. Create a VM called “master”

Graphical user interface, text, application

Description automatically generated

Name: master

Machine Folder: C:\Users\ubuntu24 ##same directory you created earlier

Type: Linux

Version: Ubuntu (64-bit)

* 1. Up the memory from default of 1024 to 3024

Graphical user interface, text, application, email

Description automatically generated

* 1. For Hard Disk choose “Create a virtual disk now”
     1. Choose for Hard disk file type: **VDI**
     2. For Storage on physical hard disk, choose: Dynamically allocated
     3. For File location and size up from default value to 20 GB for the “master”

**NOTE**: use 15GB for worker1 and worker2 node.

* + 1. Click on Create
  1. You will see your “master” vm in Powered Off mode
  2. We need to make some modifications before we start the “master” vm
  3. With master node highlighted in your Oracle VM Virtual Box Manager. Click on Settings

Graphical user interface, application

Description automatically generated

* 1. In System -> Processor tab up the Processor from 1 to 3 and click ok

**NOTE:** for worker1 and worker2, choose 2 cpu’s.

Graphical user interface, text, application

Description automatically generated

* 1. Graphical user interface, application

     Description automatically generatedClick on Network and add a NAT adapter in Adapter 1 tab
  2. In the Adapter 2 tab, choose in Attached to: BRIDGED Adapter and leave defaults and click ok.

Graphical user interface, application

Description automatically generated

* 1. Start your VM from the Oracle VM Virtual Machine Manager.
  2. As your master vm is starting, you will be prompted, and this is where you will choose your Ubuntu24.
  3. Select Startup Disk page
     1. Click on the folder icon and then the + icon on the next page and go to where you download the ubuntu24 .iso file.

Graphical user interface, text, application, Word

Description automatically generated

* + 1. A dialogue display will appear. Follow the prompts and enter the values listed below or in the images.

Choose “Try or Install Ubuntu Server”, it will run for a bit and it will produce a ton of output, you will see green OK’s throughout.

**NOTE:** toggle through on the open screen using your up and down arrows. Use Enter when you land on the desired prompt/line/area.

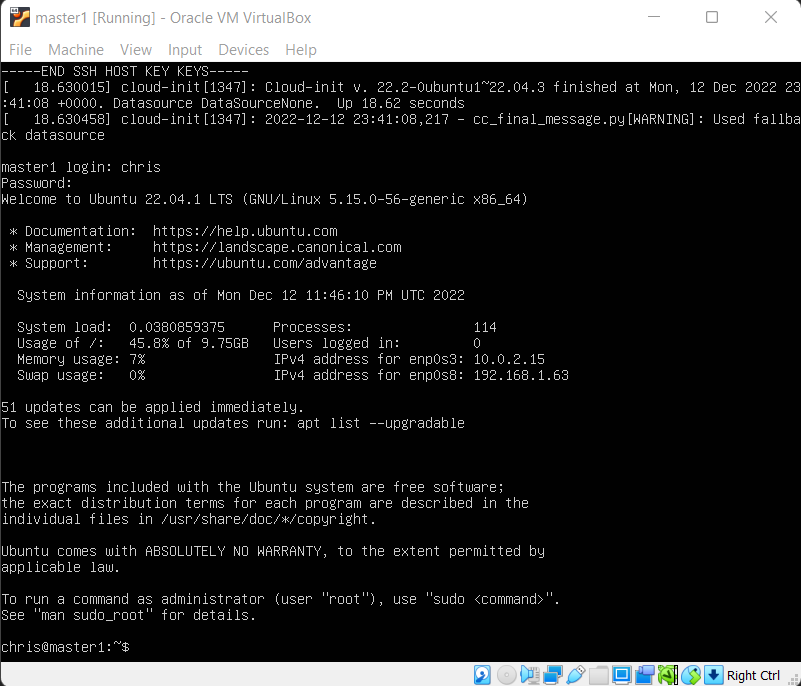
Text

Description automatically generated

* + 1. Choose the following while in the Ubuntu install screen. You will see an orange border.
       1. Language: choose your desired language. It should be defaulted to English if so, then just it Enter. Arrow up or down to your desired language if default is not desired
       2. Optional, choose using up and down arrows then hit Enter:
          1. Update to the new installer ## I choose this, and it will run for a bit.
          2. Continue without updating
          3. Back
       3. For Layout and Variant screen, arrow down to Done and hit Enter
       4. Choose Ubuntu Server and it should be defaulted and leave other areas in their default mode and choose Done
       5. Note your Network connections, you should have two
          1. Enp0s3
          2. Enp0s8
          3. Leave as is and Enter on Done
       6. Leave Proxy Address blank which is default Arrow to Done and Enter
       7. Leave Mirror address default Arrow to Done and Enter
       8. On the Guided storage configuration screen, leave defaults. Arrow down to Done and click Enter
       9. Storage Configuration screen, leave all default values as is, click on Done
       10. On the Confirm destructive action screen, arrow to continue and Enter
       11. On the Profile Setup screen:
           1. Your Name: chris
           2. Your server’s name: master
           3. Pick a username: chris
           4. Choose a password: mypassword
           5. Confirm your password: mypassword
           6. Arrow to Done and Enter
       12. On the SSH Setup screen, hit Enter when inside the [ ] symbols i.e. [X] . Arrow down to Done and Enter
       13. On the Featured Server Snaps screen, just skip this and arrow down to Done and Enter
       14. On the Install Complete, Arrow down to Reboot Now and Enter
       15. If ERROR: if you get a “[FAILED] Failed unmounting /cdrom…”, just Enter
       16. You will get a big dump of info after rebooting, host keys and such.. Just Enter when the dump stops. You will end up at the master login screen.

Text

Description automatically generated



* + - 1. **IMPORTAT:** Repeat “ALL STEPS 2 more times” starting from bullet B2c to B2o to create worker1 and worker2 vm’s. Tip: don’t continue on until you’ve completed this bullet 17 step. Trust me…

1. Installing Kubernetes Cluster:

Assumptions: You have your master, worker1 and worker2 vm’s are running.

**NOTE:** Pay close attention to line items where I mention ON ALL NODES, this means run the command on master, worker1 and worker2

======================================================================

======================================================================

######master = IPv4 address for enp0s8: 192.168.1.60 k8master.tj.net

######worker1= IPv4 address for enp0s8: 192.168.1.61 k8worker1.tj.net

######worker2= IPv4 address for enp0s8: 192.168.1.62 k8worker2.tj.net

## Run below commands **on ALL NODES**

chris@k8master:~$ sudo hostnamectl set-hostname "k8master.tj.net"

chris@k8master:~$ exec bash

##add to /etc/hosts file **on each node**.

chris@k8master:~$ sudo vi /etc/hosts

192.168.1.60 k8master.tj.net

192.168.1.61 k8worker1.tj.net

192.168.1.62 k8worker2.tj.net

sudo swapoff -a

chris@k8master:~$ sudo sed -i '/ swap / s/^\(.\*\)$/#\1/g' /etc/fstab

chris@k8master:~$ sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

chris@k8master:~$ sudo modprobe overlay

chris@k8master:~$ sudo modprobe br\_netfilter

chris@k8master:~$ sudo tee /etc/sysctl.d/kubernetes.conf <<EOF

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

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

net.ipv4.ip\_forward = 1

EOF

chris@k8master:~$ sudo sysctl --system

## install containerd run time on all nodes

chris@k8master:~$ sudo apt install -y curl gnupg2 software-properties-common apt-transport-https ca-certificates

chris@k8master:~$ sudo curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmour -o /etc/apt/trusted.gpg.d/docker.gpg

sudo add-apt-repository "deb [arch=amd64] https://download.docker.com/linux/ubuntu $(lsb\_release -cs) stable"

chris@k8master:~$ sudo apt update

chris@k8master:~$ sudo apt install -y containerd.io

chris@k8master:~$ containerd config default | sudo tee /etc/containerd/config.toml >/dev/null 2>&1

chris@k8master:~$ sudo sed -i 's/SystemdCgroup \= false/SystemdCgroup \= true/g' /etc/containerd/config.toml

## Restart, enable and verify containerd service

chris@k8master:~$ sudo systemctl restart containerd

chris@k8master:~$ sudo systemctl enable containerd

chris@k8master:~$ sudo systemctl status containerd

## Execute below commands to add apt repository for Kubernetes **on all nodes**

chris@k8master:~$ curl -s https://packages.cloud.google.com/apt/doc/apt-key.gpg | sudo apt-key add -

chris@k8master:~$ sudo apt-add-repository "deb http://apt.kubernetes.io/ kubernetes-xenial main"

## Install Kubernetes components like kubectl, kubelet and Kubeadm utility on **all the nodes**

chris@k8master:~$ sudo apt update

chris@k8master:~$ sudo apt install -y kubelet kubeadm kubectl

chris@k8master:~$ sudo apt-mark hold kubelet kubeadm kubectl

**## DO THIS STEP VERY LAST:** Set to initialize Kubernetes cluster. **On the master node only**

**## Note:** Remember to change the k8smaster.example.net to your value.

chris@k8master:~$ sudo kubeadm init --control-plane-endpoint=k8smaster.example.net

**NOTE:** You will want to remember the value that is produced in your kubeadm init output, sample below. Note the value in the red box below. Take your version and paste into your worker1 and worker2 nodes.

##TERMINAL RESPONSE Start

[bootstrap-token] Creating the "cluster-info" ConfigMap in the "kube-public" namespace

[kubelet-finalize] Updating "/etc/kubernetes/kubelet.conf" to point to a rotatable kubelet c lient certificate and key

[addons] Applied essential addon: CoreDNS

[addons] Applied essential addon: kube-proxy

Your Kubernetes control-plane has initialized successfully!

To start using your cluster, you need to run the following as a regular user:

mkdir -p $HOME/.kube

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

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

Alternatively, if you are the root user, you can run:

export KUBECONFIG=/etc/kubernetes/admin.conf

You should now deploy a pod network to the cluster.

Run "kubectl apply -f [podnetwork].yaml" with one of the options listed at:

https://kubernetes.io/docs/concepts/cluster-administration/addons/

You can now join any number of control-plane nodes by copying certificate authorities

and service account keys on each node and then running the following as root:

kubeadm join k8master.tj.net:6443 --token 4uqe3s.dsfukloaleka43p7 \

--discovery-token-ca-cert-hash sha256:39d02431c3bbe3a054101c4c98fe1502cd668f1c348a58 105869af40d069c28d \

--control-plane

Then you can join any number of worker nodes by running the following on each as root:

sudo kubeadm join k8master.tj.net:6443 --token 4uqe3s.dsfukloaleka43p7 \

--discovery-token-ca-cert-hash sha256:39d02431c3bbe3a054101c4c98fe1502cd668f1c348a58105869af40d069c28d

chris@k8master:~$

####TERMINAL RESPONSE End

## Run commands from above output **on master only**

chris@k8master:~$ mkdir -p $HOME/.kube

chris@k8master:~$ sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config

chris@k8master:~$ sudo chown $(id -u):$(id -g) $HOME/.kube/config

## View cluster and node status **on master only**

chris@k8master:~$ kubectl cluster-info

chris@k8master:~$ kubectl get nodes

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl cluster-info

Kubernetes control plane is running at https://k8master.tj.net:6443

CoreDNS is running at https://k8master.tj.net:6443/api/v1/namespaces/kube-system/services/kube-dns:dns/proxy

## To further debug and diagnose cluster problems, use 'kubectl cluster-info dump'.

chris@k8master:~$ kubectl get nodes

####TERMINAL OUTPUT Start

NAME STATUS ROLES AGE VERSION

k8master.tj.net NotReady control-plane 11m v1.26.0

chris@k8master:~$

####TERMINAL OUTPUT End

## Join worker1 to cluster – run on worker1 – remember this command is taken from the kubeadm init output. See red box above

chris@k8worker1:~$ sudo kubeadm join k8master.tj.net:6443 --token 4uqe3s.dsfukloaleka43p7 \

--discovery-token-ca-cert-hash sha256:39d02431c3bbe3a054101c4c98fe1502cd668f1c348a58105869af40d069c28d

## Join worker2 to cluster – run onworker2 – remember this command is taken from the kubeadm init output. See red box above

chris@k8worker2:~$ sudo kubeadm join k8master.tj.net:6443 --token 4uqe3s.dsfukloaleka43p7 \

--discovery-token-ca-cert-hash sha256:39d02431c3bbe3a054101c4c98fe1502cd668f1c348a58105869af40d069c28d

##Checking NODE status after you joined worker1 and worker2 – **On Master node**

####TERMINAL OUTPUT Start

chris@**k8master**:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

k8master.tj.net NotReady control-plane 21m v1.26.0

k8worker1.tj.net NotReady <none> 73s v1.26.0

k8worker2.tj.net NotReady <none> 31s v1.26.0

chris@k8master:~$

####TERMINAL OUTPUT End

####Not ready status is expected, we need to install

#### Container Network Interface or network add-on plugins

#### possibly (Flannel, Weave-Net and Calico)

##Installing Calico plugin on Master node

curl https://projectcalico.docs.tigera.io/manifests/calico.yaml -O

kubectl apply -f calico.yaml

##To verify status

kubectl get pods -n kube-system

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl get pods -n kube-system

NAME READY STATUS RESTARTS AGE

calico-kube-controllers-7bdbfc669-g66zd 1/1 Running 0 94s

calico-node-2ccpl 0/1 Init:1/3 1 (25s ago) 94s

calico-node-6k22c 1/1 Running 0 94s

calico-node-r5g96 1/1 Running 0 94s

coredns-787d4945fb-k4chv 1/1 Running 0 41m

coredns-787d4945fb-xdd65 1/1 Running 0 41m

etcd-k8master.tj.net 1/1 Running 1 (7m57s ago) 41m

kube-apiserver-k8master.tj.net 1/1 Running 1 (7m57s ago) 41m

kube-controller-manager-k8master.tj.net 1/1 Running 2 (7m57s ago) 41m

kube-proxy-k46nl 1/1 Running 1 (7m57s ago) 41m

kube-proxy-m68bp 1/1 Running 0 20m

kube-proxy-xp7qb 1/1 Running 0 21m

kube-scheduler-k8master.tj.net 1/1 Running 2 (7m57s ago) 41m

####TERMINAL OUTPUT End

####Keep refreshing the command if status is not in Running state

###Checking the status of your node

kubectl get nodes

####TERMINAL OUTPUT start

chris@k8master:~$ kubectl get nodes

NAME STATUS ROLES AGE VERSION

k8master.tj.net Ready control-plane 44m v1.26.0

k8worker1.tj.net Ready <none> 24m v1.26.0

k8worker2.tj.net Ready <none> 23m v1.26.0

chris@k8master:~$

####TERMINAL OUTPUT end

======================================================================

======================================================================

##### IF ERRORS

A possible error you may get:

chris@k8master:~$ kubectl get nodes

The connection to the server k8master.tj.net:6443 was refused - did you specify the right host or port?

**Solution:**

Run below on the “master” node

sudo -i

swapoff -a

exit

strace -eopenat kubectl version

======================================================================

======================================================================

##### TESTING YOUR DEPLOYMENT

########TESTING KUBERNETES INSTALLATION

##Testing: Deploy nginx based application and access it

kubectl create deployment nginx-app --image=nginx --replicas=2

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl create deployment nginx-app --image=nginx --replicas=2

deployment.apps/nginx-app created

chris@k8master:~$

####TERMINAL OUTPUT End

##Checking Status of your nginx deployment

kubectl get deployment nginx-app

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl get deployment nginx-app

NAME READY UP-TO-DATE AVAILABLE AGE

nginx-app 1/2 2 1 109s

chris@k8master:~$

####TERMINAL OUTPUT End

##Expose the deployment as NodePort

kubectl expose deployment nginx-app --type=NodePort --port=80

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl expose deployment nginx-app --type=NodePort --port=80

service/nginx-app exposed

chris@k8master:~$

####TERMINAL OUTPUT End

##Run the below commands to view the service status

kubectl get svc nginx-app

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl get svc nginx-app

NAME TYPE CLUSTER-IP EXTERNAL-IP PORT(S) AGE

nginx-app NodePort 10.106.118.229 <none> 80:31081/TCP 97s

##Run a describe on the service

kubectl describe svc nginx-app

####TERMINAL OUTPUT Start

chris@k8master:~$ kubectl describe svc nginx-app

Name: nginx-app

Namespace: default

Labels: app=nginx-app

Annotations: <none>

Selector: app=nginx-app

Type: NodePort

IP Family Policy: SingleStack

IP Families: IPv4

IP: 10.106.118.229

IPs: 10.106.118.229

Port: <unset> 80/TCP

TargetPort: 80/TCP

NodePort: <unset> 31081/TCP

Endpoints: 172.16.1.1:80

Session Affinity: None

External Traffic Policy: Cluster

Events: <none>

chris@k8master:~$

####TERMINAL OUTPUT End

##Testing your deployment is successful from the master

##Curl worker1 ip and port

chris@k8master:~$ curl http://192.168.1.61:31081

####TERMINAL OUTPUT Start

<!DOCTYPE html>

<html>

<head>

<title>Welcome to nginx!</title>

<style>

html { color-scheme: light dark; }

body { width: 35em; margin: 0 auto;

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

</style>

</head>

<body>

<h1>Welcome to nginx!</h1>

<p>If you see this page, the nginx web server is successfully installed and

working. Further configuration is required.</p>

<p>For online documentation and support please refer to

<a href="http://nginx.org/">nginx.org</a>.<br/>

Commercial support is available at

<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>

</body>

</html>

chris@k8master:~$

####TERMINAL OUTPUT End

##Curling worker2 node

chris@k8master:~$ curl http://192.168.1.62:31081

###TERMINAL OUTPUT Start

<!DOCTYPE html>

<html>

<head>

<title>Welcome to nginx!</title>

<style>

html { color-scheme: light dark; }

body { width: 35em; margin: 0 auto;

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

</style>

</head>

<body>

<h1>Welcome to nginx!</h1>

<p>If you see this page, the nginx web server is successfully installed and

working. Further configuration is required.</p>

<p>For online documentation and support please refer to

<a href="http://nginx.org/">nginx.org</a>.<br/>

Commercial support is available at

<a href="http://nginx.com/">nginx.com</a>.</p>

<p><em>Thank you for using nginx.</em></p>

</body>

</html>

chris@k8master:~$

####TERMINAL OUTPUT End

======================================================================

**You have successfully implemented k8 cluster – now learn the commands!**

======================================================================

Graphical user interface

Description automatically generatedBirds Eyer View