Set up Docker

Kubernetes requires an existing Docker installation. Install and enable Docker on each server node by following the steps below :

1. Update the package list:

```
sudo apt update -y
```

2. Next, install Docker with the command:

```
sudo apt install docker.io -y
```

```
marko@pnap:~$ sudo apt install docker.io -y
[sudo] password for marko:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base libidn11 pigz runc ubuntu-fan
Suggested packages:
  ifupdown aufs-tools cgroupfs-mount | cgroup-lite debootstrap docker-doc rinse
  zfs-fuse | zfsutils
The following NEW packages will be installed:
  bridge-utils containerd dns-root-data dnsmasq-base docker.io libidn11 pigz runc
  ubuntu-fan
0 upgraded, 9 newly installed, 0 to remove and 67 not upgraded.
```

3. Set Docker to launch on boot by entering the following:

```
sudo systemctl enable docker
```

4. Verify Docker is running:

```
sudo systemctl status docker
```

```
marko@pnap:~$ sudo systemctl status docker
● docker.service - Docker Application Container Engine
     Loaded: loaded (/lib/systemd/system/docker.service; enabled; vendor preset: enabl>
     Active: active (running) since Thu 2022-11-24 11:26:27 UTC; 3min 26s ago
TriggeredBy: ● docker.socket
       Docs: https://docs.docker.com
   Main PID: 2887 (dockerd)
      Tasks: 8
     Memory: 29.2M
     CGroup: /system.slice/docker.service
             └─2887 /usr/bin/dockerd -H fd:// --containerd=/run/containerd/containerd.>
```

5. Start Docker if it is not running:

```
sudo systemctl start docker
```

Install Kubernetes

Step 1: Add Kubernetes Signing Key

Since you are downloading Kubernetes from a non-standard repository, it is essential to ensure that the software is authentic. This is done by adding a signing key.

On each node, use the curl command to download the key, then store it in a safe place (default is `/usr/share/keyrings`):

```
sudo mkdir -p /etc/apt/keyrings/

curl -fsSL https://packages.cloud.google.com/apt/doc/apt-key.gpg
| sudo gpg --dearmor -o
/etc/apt/keyrings/kubernetes-archive-keyring.gpg
```

Step 2: Add Software Repositories

Kubernetes is not included in the default repositories. To add the Kubernetes repository to your list, enter the following on each node:

```
sudo echo "deb
[signed-by=/etc/apt/keyrings/kubernetes-archive-keyring.gpg]
https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee
/etc/apt/sources.list.d/kubernetes.list

sudo apt-get update -y
```

- #Turn Off Swap Space

```
swapoff -a
sudo sed -i 's/\(.*swap.*\)/# \1/g' /etc/fstab

sudo usermod -aG docker $(whoami)
```

Step 3: Kubernetes Installation Tools

Kubeadm (Kubernetes Admin) is a tool that helps initialize a cluster. It fast-tracks setup by using community-sourced best practices. Kubelet is the work package, which runs on every node and starts containers. The tool gives you command-line access to clusters.

Execute the following commands on each server node.

1. Install Kubernetes tools with the command:

```
sudo apt install kubeadm kubelet kubectl kubernetes-cni  -y
```

```
marko@pnap:~$ sudo apt install kubeadm kubelet kubectl -y
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following additional packages will be installed:
  conntrack cri-tools ebtables kubernetes-cni socat
Suggested packages:
  nftables
The following NEW packages will be installed:
  conntrack cri-tools ebtables kubeadm kubectl kubelet kubernetes-cni socat
0 upgraded, 8 newly installed, 0 to remove and 67 not upgraded.
Need to get 81.6 MB of archives.
After this operation, 327 MB of additional disk space will be used.
```

```
sudo apt-mark hold kubeadm kubelet kubectl
```

```
marko@pnap:~$ sudo apt-mark hold kubeadm kubelet kubectl
kubeadm set on hold.
kubelet set on hold.
kubectl set on hold.
marko@pnap:~$
```

Allow the process to complete.

Enable and start kubelet service

systemctl daemon-reload

systemctl start kubelet

systemctl enable kubelet.service

2. Verify the installation with:

```
kubeadm version
```

```
marko@pnap:~$ kubeadm version
kubeadm version: &version.Info{Major:"1", Minor:"25", GitVersion:"v1.25.4", GitCommit:"
872a965c6c6526caa949f0c6ac028ef7aff3fb78", GitTreeState:"clean", BuildDate:"2022-11-09T
13:35:06Z", GoVersion:"go1.19.3", Compiler:"gc", Platform:"linux/amd64"}
marko@pnap:~$
```

—---------- following steps only execute on master —----------

Switch to the root user.

sudo su -

Initialize Kubernates master by executing below command.

kubeadm init

```
vishal@vishal-VirtualBox:~/Desktop$ sudo su -
[sudo] password for vishal:
root@vishal-VirtualBox:~# kubeadm init
[init] Using Kubernetes version: v1.27.3
[preflight] Running pre-flight checks
[preflight] Pulling images required for setting up a Kubernetes cluster
[preflight] This might take a minute or two, depending on the speed of your internet connection
[preflight] You can also perform this action in beforehand using 'kubeadm config images pull'
W0719 14:51:51.248695    2550 checks.go:835] detected that the sandbox image "registry.k8s.io/pause:3.6" of the container runtime i
s inconsistent with that used by kubeadm. It is recommended that using "registry.k8s.io/pause:3.9" as the CRI sandbox image.



[certs] Using certificateDir folder "/etc/kubernetes/pki"
[certs] Generating "ca" certificate and key
[certs] Generating "apiserver" certificate and key
[certs] apiserver serving cert is signed for DNS names [kubernetes kubernetes.default kubernetes.default.svc kubernetes.default.svc
.cluster.local vishal-virtualbox] and IPs [10.96.0.1 192.168.1.12]
[certs] Generating "apiserver-kubelet-client" certificate and key
[certs] Generating "front-proxy-ca" certificate and key
[certs] Generating "front-proxy-client" certificate and key
[certs] Generating "etcd/ca" certificate and key
[certs] Generating "etcd/server" certificate and key
[certs] etcd/server serving cert is signed for DNS names [localhost vishal-virtualbox] and IPs [192.168.1.12 127.0.0.1 ::1]
```

```
[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/

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

kubeadm join 192.168.1.12:6443 --token zm7ef4.bivyddd03jyxizh4 \
        --discovery-token-ca-cert-hash sha256:5c76cc9883593bf00964d3f15da8f14bd90e27bf09e3e541da8b36c0935aee93
root@vishal-VirtualBox:~# exit
logout
vishal@vishal-VirtualBox:~$
```

#exit root user & exeucte as normal user

exit

- run next commands as appeared in the terminal:

```
root@vishal-VirtualBox:~#
root@vishal-VirtualBox:~# exit
logout
vishal@vishal-VirtualBox:~/Desktop$   mkdir -p $HOME/.kube
vishal@vishal-VirtualBox:~/Desktop$   sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config
vishal@vishal-VirtualBox:~/Desktop$   sudo chown $(id -u):$(id -g) $HOME/.kube/config
vishal@vishal-VirtualBox:~/Desktop$
```

To verify, if kubectl is working or not, run the following command.

kubectl get pods -o wide –all-namespaces

```
vishal@vishal-VirtualBox:~$
vishal@vishal-VirtualBox:~$ kubectl get pods -o wide
No resources found in default namespace.
vishal@vishal-VirtualBox:~$ kubectl get nodes -o wide
NAME              STATUS     ROLES           AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
vishal-virtualbox   NotReady   control-plane   2m40s   v1.27.3   192.168.1.12   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal@vishal-VirtualBox:~$
vishal@vishal-VirtualBox:~$ kubectl get pods -o wide --all-namespaces
NAMESPACE     NAME                                        READY   STATUS    RESTARTS   AGE   IP             NODE                NOMINATED NODE   READINESS GATES
kube-system   coredns-5d78c9869d-c7n5n                    0/1     Pending   0          6m3s  <none>         <none>              <none>           <none>
kube-system   coredns-5d78c9869d-tdzld                    0/1     Pending   0          6m3s  <none>         <none>              <none>           <none>
kube-system   etcd-vishal-virtualbox                      1/1     Running   1          6m6s  192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-apiserver-vishal-virtualbox            1/1     Running   1          6m6s  192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-controller-manager-vishal-virtualbox   1/1     Running   1          6m7s  192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-proxy-cfsmz                            1/1     Running   0          6m3s  192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-scheduler-vishal-virtualbox            1/1     Running   1          6m6s  192.168.1.12   vishal-virtualbox   <none>           <none>
vishal@vishal-VirtualBox:~$
```

#You will notice from the previous command, that all the pods are running except one: ‘kube-dns’. For resolving this we will install a # pod network. To install the weave pod network, run the following command:

releases : https://github.com/weaveworks/weave/releases/

installing latest version:

kubectl apply -f
https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s-1.11.yaml

Now, status should have changed

kubectl get pods --all-namespaces

```
vishal@vishal-VirtualBox:~/Desktop$
vishal@vishal-VirtualBox:~/Desktop$ kubectl apply -f https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s-1.11.yaml
serviceaccount/weave-net created
clusterrole.rbac.authorization.k8s.io/weave-net created
clusterrolebinding.rbac.authorization.k8s.io/weave-net created
role.rbac.authorization.k8s.io/weave-net created
rolebinding.rbac.authorization.k8s.io/weave-net created
daemonset.apps/weave-net created
vishal@vishal-VirtualBox:~/Desktop$
vishal@vishal-VirtualBox:~/Desktop$ kubectl get pods -o wide --all-namespaces
NAMESPACE     NAME                                        READY   STATUS            RESTARTS      AGE   IP             NODE                NOMINATED NODE   READINESS GATES
kube-system   coredns-5d78c9869d-c7n5n                    0/1     Pending           0             34m   <none>         <none>              <none>           <none>
kube-system   coredns-5d78c9869d-tdzld                    0/1     Pending           0             34m   <none>         <none>              <none>           <none>
kube-system   etcd-vishal-virtualbox                      1/1     Running           3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-apiserver-vishal-virtualbox            1/1     Running           3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-controller-manager-vishal-virtualbox   1/1     Running           3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-proxy-cfsmz                            1/1     Running           2 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-scheduler-vishal-virtualbox            1/1     Running           3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   weave-net-ldxfb                             0/2     PodInitializing   0             24s   192.168.1.12   vishal-virtualbox   <none>           <none>
vishal@vishal-VirtualBox:~/Desktop$ kubectl get pods -o wide --all-namespaces
NAMESPACE     NAME                                        READY   STATUS    RESTARTS      AGE   IP             NODE                NOMINATED NODE   READINESS GATES
kube-system   coredns-5d78c9869d-c7n5n                    1/1     Running   0             34m   10.32.0.3      vishal-virtualbox   <none>           <none>
kube-system   coredns-5d78c9869d-tdzld                    1/1     Running   0             34m   10.32.0.2      vishal-virtualbox   <none>           <none>
kube-system   etcd-vishal-virtualbox                      1/1     Running   3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-apiserver-vishal-virtualbox            1/1     Running   3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-controller-manager-vishal-virtualbox   1/1     Running   3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-proxy-cfsmz                            1/1     Running   2 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   kube-scheduler-vishal-virtualbox            1/1     Running   3 (10m ago)   34m   192.168.1.12   vishal-virtualbox   <none>           <none>
kube-system   weave-net-ldxfb                             2/2     Running   1 (25s ago)   45s   192.168.1.12   vishal-virtualbox   <none>           <none>
vishal@vishal-VirtualBox:~/Desktop$
```

Get token

- kubeadm token create --print-join-command

```
kube-system   kube-scheduler-vishal-virtualbox            1/1     Running   5 (10m ago)   54m   192.168.
kube-system   weave-net-ldxfb                             2/2     Running   1 (25s ago)   45s   192.168.
vishal@vishal-VirtualBox:~/Desktop$ kubeadm token create --print-join-command
kubeadm join 192.168.1.12:6443 --token e5dfat.87hhatngu1awssdz --discovery-token-ca-cert-hash sha256:5c
vishal@vishal-VirtualBox:~/Desktop$
```

—---------- following steps only execute on master end —----------

- copy paste url in other worker node to join the cluster

```
vishal@amol:~/Desktop$
vishal@amol:~/Desktop$ kubeadm join 192.168.1.12:6443 --token e5dfat.87hhatngu1awssdz --discovery-token-ca
883593bf00964d3f15da8f14bd90e27bf09e3e541da8b36c0935aee93
[preflight] Running pre-flight checks
error execution phase preflight: [preflight] Some fatal errors occurred:
        [ERROR IsPrivilegedUser]: user is not running as root
[preflight] If you know what you are doing, you can make a check non-fatal with `--ignore-preflight-errors
To see the stack trace of this error execute with --v=5 or higher
vishal@amol:~/Desktop$
vishal@amol:~/Desktop$ sudo kubeadm join 192.168.1.12:6443 --token e5dfat.87hhatngu1awssdz --discovery-tok
76cc9883593bf00964d3f15da8f14bd90e27bf09e3e541da8b36c0935aee93
[sudo] password for vishal:
[preflight] Running pre-flight checks
[preflight] Reading configuration from the cluster...
[preflight] FYI: You can look at this config file with 'kubectl -n kube-system get cm kubeadm-config -o ya
[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.

vishal@amol:~/Desktop$
```

check master for their presence

- kubectl get nodes -o wide

```
vishal@vishal-VirtualBox:~/Desktop$ kubectl get nodes -o wide
NAME              STATUS     ROLES           AGE     VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
amol              NotReady   <none>          6m14s   v1.27.3   192.168.1.11   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
manoj             Ready      <none>          5m39s   v1.27.3   192.168.1.10   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal-virtualbox Ready      control-plane   47m     v1.27.3   192.168.1.12   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal@vishal-VirtualBox:~/Desktop$
```

Lets test it,

Create sample deployment.yml

- vi deployment.yml

```
kind: Pod
apiVersion: v1
metadata:
  name: testpod
spec:
  containers:
    - name: c00
      image: ubuntu
      command: ["/bin/bash", "-c", "while true; do echo Hello-vishalk17; sleep
5 ; done"]
  restartPolicy: Never          # Defaults to Always
```

- Kubectl apply -f deployment.yml

```
vishal@vishal-VirtualBox:~$
vishal@vishal-VirtualBox:~$ vi deployment.yml
vishal@vishal-VirtualBox:~$ kubectl apply -f deployment.yml
pod/testpod created
vishal@vishal-VirtualBox:~$ kubectl get nodes
NAME               STATUS   ROLES           AGE   VERSION
amol               Ready    <none>          45m   v1.27.3
manoj              Ready    <none>          44m   v1.27.3
vishal-virtualbox   Ready    control-plane   86m   v1.27.3
vishal@vishal-VirtualBox:~$ kubectl get nodes -o wide
NAME               STATUS   ROLES           AGE   VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
amol               Ready    <none>          45m   v1.27.3   192.168.1.11   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
manoj              Ready    <none>          44m   v1.27.3   192.168.1.10   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal-virtualbox   Ready    control-plane   86m   v1.27.3   192.168.1.12   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal@vishal-VirtualBox:~$    kubectl get pods -o wide
NAME      READY   STATUS              RESTARTS   AGE   IP       NODE    NOMINATED NODE   READINESS GATES
testpod   0/1     ContainerCreating   0          66s   <none>   manoj   <none>           <none>
vishal@vishal-VirtualBox:~$ kubectl get nodes -o wide
NAME               STATUS   ROLES           AGE   VERSION   INTERNAL-IP    EXTERNAL-IP   OS-IMAGE             KERNEL-VERSION      CONTAINER-RUNTIME
amol               Ready    <none>          46m   v1.27.3   192.168.1.11   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
manoj              Ready    <none>          45m   v1.27.3   192.168.1.10   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal-virtualbox   Ready    control-plane   87m   v1.27.3   192.168.1.12   <none>        Ubuntu 20.04.6 LTS   5.15.0-76-generic   containerd://1.6.12
vishal@vishal-VirtualBox:~$ kubectl get pods -o wide
NAME      READY   STATUS    RESTARTS   AGE   IP          NODE    NOMINATED NODE   READINESS GATES
testpod   1/1     Running   0          90s   10.36.0.1   manoj   <none>           <none>
vishal@vishal-VirtualBox:~$
```

OBJ

Sourcecode :

- https://github.com/vishalk17/devops/tree/main/kubernetes

My devops repo :

- https://github.com/vishalk17/devops

My telegram channel:

- https://t.me/vishalk17_devops

Contact:

Telegram : t.me/vishalk17

vishalk17 My youtube Channel :

- YouTube https://www.youtube.com/@vishalk17