**Running Kubernetes single node cluster in Azure**

1. We need to upgrade the basic packages in the linux server by running the following commands

sudo apt-get update  
 sudo apt upgrade

1. Firewalls like UFW can block these communications if not configured correctly. We need to disable the firewall

sudo ufw disable

1. Disables Swap Usage: Stops the operating system from using swap space for virtual memory.

swapoff -a

1. Temporary Effect: The swap will be disabled only until the next reboot, unless you also remove the swap entry from the /etc/fstab file.

sudo sed -i '/swap/d' /etc/fstab

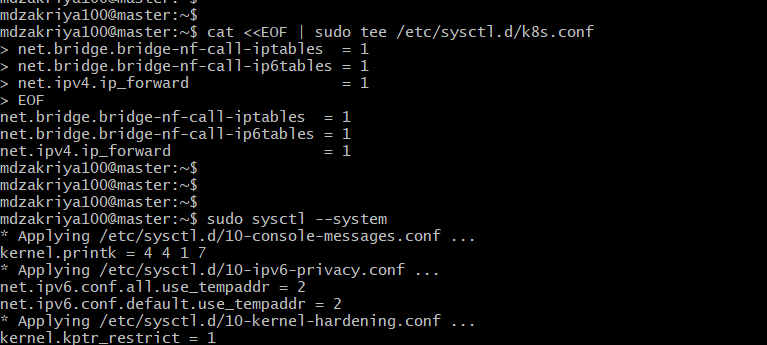
1. Why These Settings Are Needed for Kubernetes

Bridge Networking Support:  
Kubernetes uses network plugins (like Flannel, Calico, etc.) that rely on Linux bridges to forward packets. The bridge-nf-call-\* settings ensure the Linux kernel properly routes bridged traffic through iptables rules.

IP Forwarding:

Kubernetes needs IP forwarding to enable communication between pods, services, and external networks. Without this, traffic between nodes or pods might fail.

cat <<EOF | sudo tee /etc/sysctl.d/k8s.conf  
 net.bridge.bridge-nf-call-iptables = 1  
 net.bridge.bridge-nf-call-ip6tables = 1  
 net.ipv4.ip\_forward = 1  
 EOF  
 sudo sysctl –system

  
  
 6. Installing the packages required for the Kubernetes  
  
 sudo apt install -y apt-transport-https ca-certificates curl gnupg-agent software-properties-common

7. Lets install docker , which is required for the containerization, we can use contianerd, cri-o, podman  
  
sudo su  
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 docker.io

8. Lets install these as well , for autocompletion in kubectl commands:   
  
sudo mkdir -p -m 755 /etc/apt/keyrings

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.32/deb/Release.key | sudo gpg --dearmor -o /etc/apt/keyrings/kubernetes-apt-keyring.gpg

echo 'deb [signed-by=/etc/apt/keyrings/kubernetes-apt-keyring.gpg] https://pkgs.k8s.io/core:/stable:/v1.32/deb/ /' | sudo tee /etc/apt/sources.list.d/kubernetes.list

apt-get update

apt-get install -y kubelet kubeadm kubectl

sudo systemctl enable --now kubelet  
  
 9. Lets install the below packages which will help for the autocompletion for kubectl commands

apt-get install bash-completion

echo 'source <(kubectl completion bash)' >>~/.bashrc

#source /usr/share/bash-completion/bash\_completion

kubectl completion bash >/etc/bash\_completion.d/kubectl  
  
10. Lets initate the Kubernetes cluster   
  
kubeadm init --apiserver-advertise-address=<paste-the-apiserver-advertise-address-ip> --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=all

A computer screen shot of a black screen

Description automatically generated

We need to use the eth0 ip address : 10.0.0.4 , lets run the below command to initiate the kubeneretes

kubeadm init --apiserver-advertise-address=<paste-the-apiserver-advertise-address-ip> --pod-network-cidr=192.168.0.0/16 --ignore-preflight-errors=all

export KUBECONFIG=/etc/kubernetes/admin.conf

A screenshot of a computer program

Description automatically generated

As we can see now, the Kubernetes cluster is deployed successfully , we can run the join command to add the worker node to the cluster

11. Go out of the root and run the below commands

mkdir -p $HOME/.kube  
 sudo cp -i /etc/kubernetes/admin.conf $HOME/.kube/config  
 sudo chown $(id -u):$(id -g) $HOME/.kube/config

12. Now we need to install the cni plugin for the clusters ,  
  
 kubectl apply -f <https://github.com/weaveworks/weave/releases/download/v2.8.1/weave-daemonset-k8s-1.11.yaml>  
  
now we can check the kubenetes cluster status   
  
kubectl get all -A  
kubectl get nodes

A black screen with white text

Description automatically generated