Créer un cluster Kubernetes avec KIND (1 seule machine)

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

Étape 1

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

```
#!/bin/bash

# Configuration des variables
RESOURCE_GROUP="k8s-cluster"
LOCATION="westus"
VNET_NAME="k8sVNet"
SUBNET_MASTER="k8sSubnetMaster"
SUBNET_WORKER="k8sSubnetWorker"
NSG_NAME="k8sMasterNSG"
NIC_NAME="k8sMasterNIC"

# Création du groupe de ressources
az group create --name $RESOURCE_GROUP --location $LOCATION

# Création du réseau virtuel et des sous-réseaux
az network vnet create --resource-group $RESOURCE_GROUP --name $VNET_NAME --address-
prefix 10.0.0.0/16 \
    --subnet-name $SUBNET_MASTER --subnet-prefix 10.0.1.0/24

az network vnet subnet create --resource-group $RESOURCE_GROUP --vnet-name $VNET_NAME
\
    --name $SUBNET_WORKER --address-prefix 10.0.2.0/24

# Création du groupe de sécurité réseau (NSG) pour le master
az network nsg create --resource-group $RESOURCE_GROUP --name $NSG_NAME --location
$LOCATION

# Création de la règle NSG pour ouvrir le port 6443 (Kubernetes API Server)
az network nsg rule create --resource-group $RESOURCE_GROUP --nsg-name $NSG_NAME \
    --name AllowKubernetesAPI --priority 1000 --direction Inbound --access Allow \
    --protocol Tcp --destination-port-range 6443 --destination-address-prefix "*" \
    --source-address-prefix "*" --source-port-range "*"

# Association du NSG à la NIC de la VM master
# Note: Cette commande suppose que la NIC existe déjà. Si la NIC n'existe pas encore,
# elle doit être créée avec la VM ou explicitement avant d'exécuter cette commande.
# Cette étape est normalement effectuée lors de la création de la VM Master dans
create_master.sh
# az network nic update --resource-group $RESOURCE_GROUP --name $NIC_NAME --network-
security-group $NSG_NAME

echo "Environnement Azure pour Kubernetes est prêt."
```

Étape 2

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

```
#!/bin/bash

# Configuration des variables pour le Master
RESOURCE_GROUP="k8s-cluster"
VM_NAME="k8sMaster"
VNET_NAME="k8sVNet"
SUBNET_NAME="k8sSubnetMaster"
VM_SIZE="Standard_D2s_v3"
IMAGE="Ubuntu2204"
USERNAME="azureuser"

# Création de la VM Master avec une taille ajustée
az vm create \
    --resource-group $RESOURCE_GROUP \
    --name $VM_NAME \
    --image $IMAGE \
    --size $VM_SIZE \
    --vnet-name $VNET_NAME \
    --subnet $SUBNET_NAME \
    --admin-username $USERNAME \
    --generate-ssh-keys

echo "VM Master créée."
```

Étape 3

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

```
#!/bin/bash

# Configuration des variables
RESOURCE_GROUP="k8s-cluster"

echo "Toutes les VM ont été créées. Installation des composants Kubernetes sur le
master..."

# Récupération de l'IP publique du Master
MASTER_IP=$(az vm show --resource-group $RESOURCE_GROUP --name k8sMaster --show-
details --query publicIps -o tsv)

# Connexion SSH à la VM Master
ssh -o StrictHostKeyChecking=no azureuser@$MASTER_IP << 'EOF'
sudo bash << 'ENDSSH'

# Mise à jour des packages
apt-get update
apt-get install -y ca-certificates curl software-properties-common

# Installation de Docker
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | gpg --dearmor -o
/usr/share/keyrings/docker-archive-keyring.gpg
echo "deb [arch=amd64 signed-by=/usr/share/keyrings/docker-archive-keyring.gpg]
https://download.docker.com/linux/ubuntu $(lsb_release -cs) stable" | tee
/etc/apt/sources.list.d/docker.list > /dev/null
apt-get update
apt-get install -y docker-ce docker-ce-cli containerd.io

# Installation de Kind
curl -Lo ./kind https://kind.sigs.k8s.io/dl/v0.11.1/kind-linux-amd64
chmod +x ./kind
mv ./kind /usr/local/bin/

# Installation de kubectl
curl -LO "https://dl.k8s.io/release/$(curl -L -s
https://dl.k8s.io/release/stable.txt)/bin/linux/amd64/kubectl"
chmod +x kubectl
mv kubectl /usr/local/bin/

# Suppression de l'ancien cluster Kind s'il existe
echo "Suppression de l'ancien cluster Kind s'il existe..."
kind delete cluster --name kind

# Création du fichier de configuration Kind pour un cluster avec 1 nœud de contrôle et
2 workers
cat <<EOT > kind-config.yaml
kind: Cluster
apiVersion: kind.x-k8s.io/v1alpha4
nodes:
- role: control-plane
- role: worker
- role: worker
EOT

# Création du cluster Kind
echo "Création du nouveau cluster Kind..."
kind create cluster --config kind-config.yaml

# Configuration de kubectl pour utiliser le cluster Kind
kubectl cluster-info --context kind-kind

echo "Le cluster Kind est configuré avec succès."

ENDSSH
EOF

echo "Configuration du master Kubernetes avec Kind et kubectl sur le master terminée."
```

Étape 4

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

****
**********POUR TESTER KIND **********
****

MACHINE MASTER

```
ssh -o StrictHostKeyChecking=no azureuser@20.237.198.42
sudo -s
kubectl config get-contexts
kubectl get nodes -o wide
kubectl get po -o wide
kubectl run nginx --image=nginx --dry-run=client -o yaml
kubectl get po
kubectl get po -n kube-system
kubectl get po -n kube-system -o wide
```

```
alias kpo='kubectl get po -o wide'
kpo
```

```
alias k='kubectl'
k get po
k get no
```

```
k run mynginx --image=nginx --dry-run=client -o yaml > pod.yaml
k get po
k apply -f pod.yaml
```

Remarque : Ce qui est surligné est pareil
k run mynginx --image=nginx --dry-run=client -o yaml > pod.yaml
k apply -f pod.yaml

Étape 5

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

Azure_CLI_Shell

```
ssh -o StrictHostKeyChecking=no azureuser@20.237.198.42
sudo -s
./kubernetes_reset.sh
```

script5_kubernetes_reset.sh

```
#!/bin/bash

echo "Début de la réinitialisation de l'état de Kubernetes sur ce nœud..."

# Réinitialisation de l'état de kubeadm
echo "Exécution de kubeadm reset pour nettoyer l'état de Kubernetes..."
sudo kubeadm reset -f

# Nettoyage des répertoires de données de Kubernetes
echo "Nettoyage des répertoires de données de Kubernetes..."
sudo rm -rf /etc/kubernetes/manifests/*
sudo rm -rf /var/lib/etcd/*
sudo rm -rf /var/lib/kubelet/*
sudo rm -rf $HOME/.kube/config

# Redémarrage des services de conteneurs
echo "Redémarrage des services Docker et Containerd..."
sudo systemctl restart docker
sudo systemctl restart containerd

# Vérification et libération des ports utilisés par Kubernetes (OPTIONNEL)
echo "Vérification des ports utilisés par Kubernetes (10250, 2379, 2380)..."
sudo netstat -tulnpe | grep -E '10250|2379|2380'

echo "La réinitialisation est terminée. Vous pouvez maintenant réinitialiser votre
cluster Kubernetes avec 'kubeadm init'."
```

Étape 6

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

```
exit
exit
./script3_configure_master.sh
```

```
exit (pour quitter le mode admin)
exit (pour quitter la VM)
./script3_configure_master.sh (au niveau du cloud shell azure CLI)
```

Utilisez le même script que dans l’étape 3

Étape 7

Étape 1 - script1_setup_azure.sh
Étape 2 - script2_create_master.sh
Étape 3 - script3_configure_master.sh
Étape 4 - manipulations
Étape 5 - script5_kubernetes_reset.sh (OPTIONNEL 1)
Étape 6 -script3_configure_master2.sh (OPTIONNEL 2)
Étape 7 – script7_cleanup.sh

```
#!/bin/bash

# Configuration des variables
RESOURCE_GROUP="k8s-cluster"

# Suppression du groupe de ressources et de toutes les ressources associées
az group delete --name $RESOURCE_GROUP --yes --no-wait

echo "Azure resources are being deleted..."
```

ou

az group delete --name k8s-cluster --yes --no-wait