**Nom :** DIAWARA NANA  
**Filière :** CyberSecurity  
**Cours :** Distributed Applications

## ****I. Introduction****

Kubernetes est une plateforme open-source développée initialement par Google, aujourd’hui gérée par la Cloud Native Computing Foundation (CNCF). Elle permet d’orchestrer des conteneurs (comme ceux créés avec Docker) à grande échelle. Grâce à Kubernetes, il est possible de gérer automatiquement le déploiement, la montée en charge, la réplication, la tolérance aux pannes, ainsi que la mise à jour continue des applications conteneurisées.

Ce TP a pour objectif d’installer un cluster Kubernetes à partir de machines Ubuntu, en configurant un **nœud maître (master)** et plusieurs **nœuds travailleurs (workers)**.

## ****II. Objectifs****

* Comprendre l’architecture de Kubernetes.
* Installer un cluster Kubernetes fonctionnel (1 master + N workers).
* Déployer un réseau de communication entre les pods.
* Tester le bon fonctionnement du cluster.

## ****III. Architecture de Kubernetes****

Un cluster Kubernetes est constitué de plusieurs composants :

* **Nœud maître (Control Plane)** : gère l’état du cluster (API Server, Controller Manager, Scheduler, etcd).
* **Nœuds travailleurs (Workers)** : hébergent les **pods** (groupes de conteneurs).
* **Kubelet** : agent présent sur chaque nœud pour communiquer avec le maître.
* **Container Runtime (Docker, containerd)** : exécute les conteneurs.
* **Plugin CNI (Calico, Flannel, Weave...)** : gère le réseau des pods.

## ****IV. Prérequis****

* 3 machines virtuelles (ou physiques) sous **Ubuntu 20.04/22.04**
* Accès internet
* Accès root (sudo)
* SWAP désactivé (swapoff -a)
* Communication réseau entre les nœuds
* NTP synchronisé (recommandé)

## ****V. Étapes d’installation détaillées****

### ****1. Installation sur le Nœud Master****

#### a. Mise à jour et installation de dépendances

sudo apt update && sudo apt upgrade -y

sudo apt install curl apt-transport-https ca-certificates software-properties-common

#### b. Installation de Docker/Containerd

curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo gpg --dearmor -o /usr/share/keyrings/docker-archive-keyring.gpg

echo \

"deb [arch=$(dpkg --print-architecture) signed-by=/usr/share/keyrings/docker-archive-keyring.gpg] https://download.docker.com/linux/ubuntu \

$(lsb\_release -cs) stable" | sudo tee /etc/apt/sources.list.d/docker.list > /dev/null

sudo apt update

sudo apt install -y docker-ce docker-ce-cli containerd.io

#### c. Désactivation du SWAP

sudo swapoff -a

sudo sed -i '/ swap / s/^/#/' /etc/fstab

#### d. Chargement des modules nécessaires

sudo tee /etc/modules-load.d/containerd.conf <<EOF

overlay

br\_netfilter

EOF

sudo modprobe overlay

sudo modprobe br\_netfilter

#### e. Configuration réseau système

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

sudo sysctl –system

#### f. Configuration de containerd

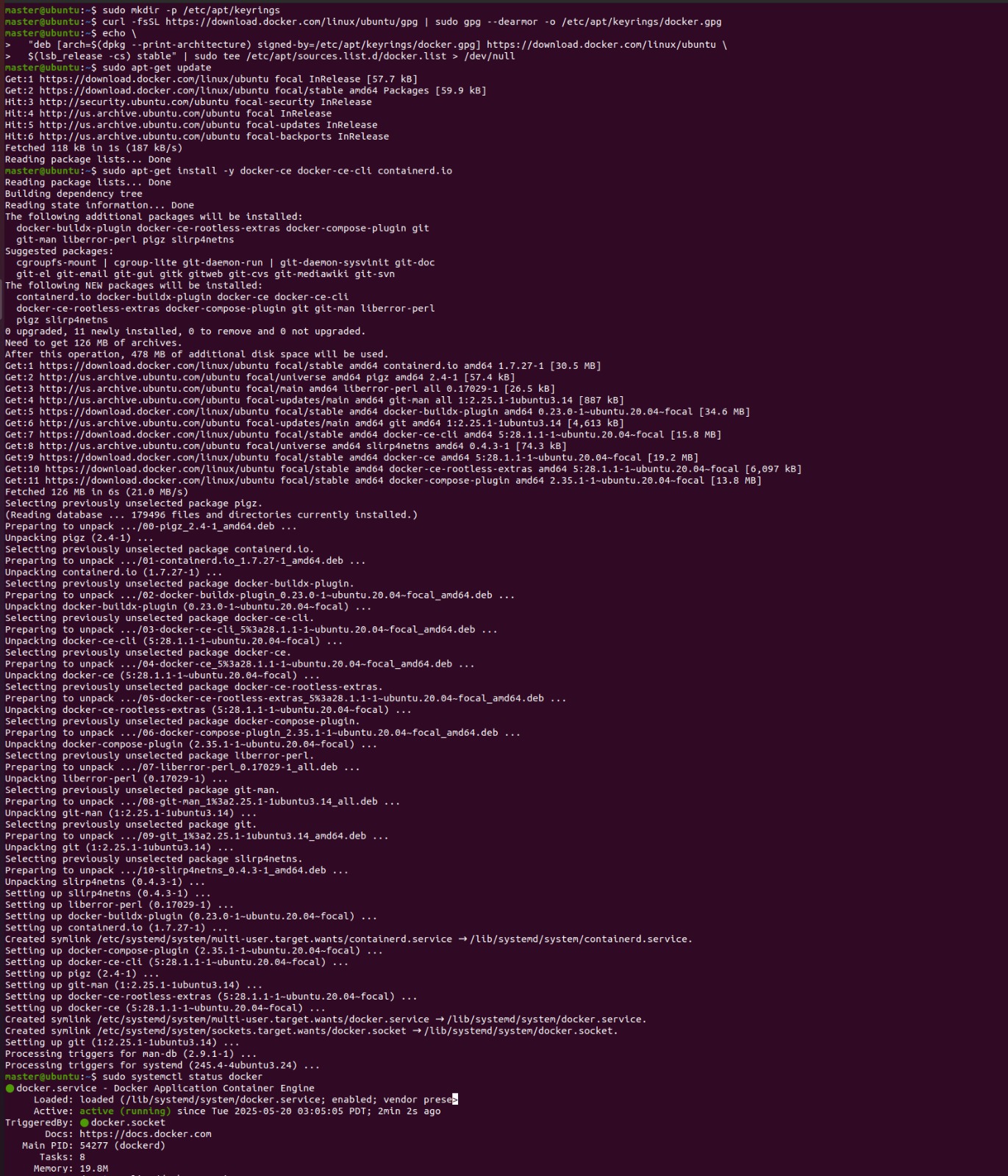
containerd config default | sudo tee /etc/containerd/config.toml >/dev/null

sudo sed -i 's/SystemdCgroup = false/SystemdCgroup = true/g' /etc/containerd/config.toml

sudo systemctl restart containerd

sudo systemctl enable containerd





#### g. Installation de Kubernetes (kubeadm, kubelet, kubectl)

sudo mkdir -p /etc/apt/keyrings

curl -fsSL https://pkgs.k8s.io/core:/stable:/v1.30/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:/v1.30/deb/ /" | sudo tee /etc/apt/sources.list.d/kubernetes.list

sudo apt update

sudo apt install -y kubelet kubeadm kubectl

sudo apt-mark hold kubelet kubeadm kubectl



#### h. Initialisation du cluster

sudo kubeadm init

Après l'initialisation, suivre les instructions affichées pour configurer kubectl :

bash

CopierModifier

mkdir -p $HOME/.kube

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

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

#### i. Installation du plugin réseau (CNI)

Avec **Calico** :

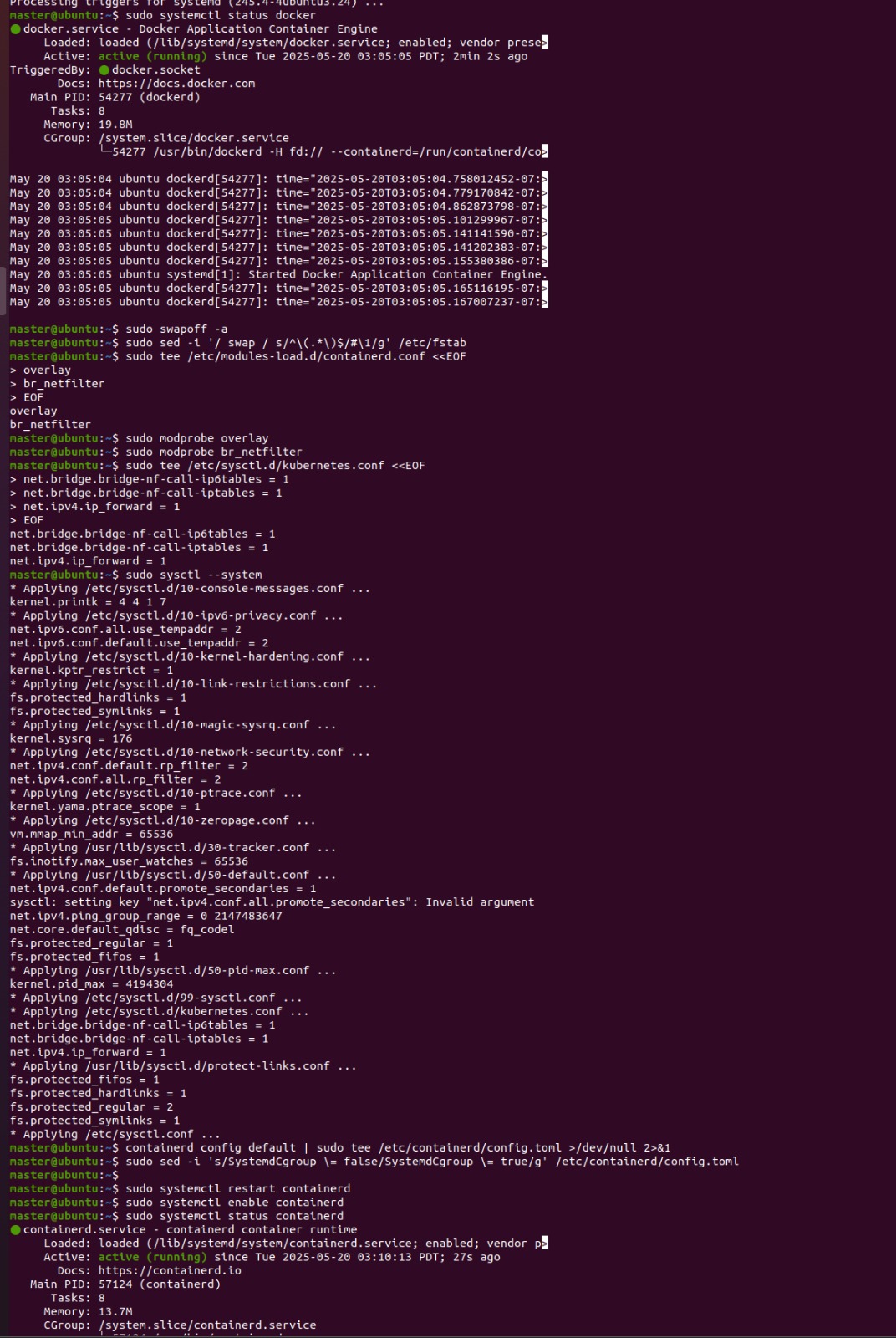
kubectl apply -f https://raw.githubusercontent.com/projectcalico/calico/v3.25.0/manifests/calico.yaml

Ou avec **Weave Net** :

bash

CopierModifier

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



### ****2. Installation sur les Nœuds Workers****

Effectuer les mêmes étapes que pour le master, jusqu’à l’installation de kubelet, kubeadm et kubectl.

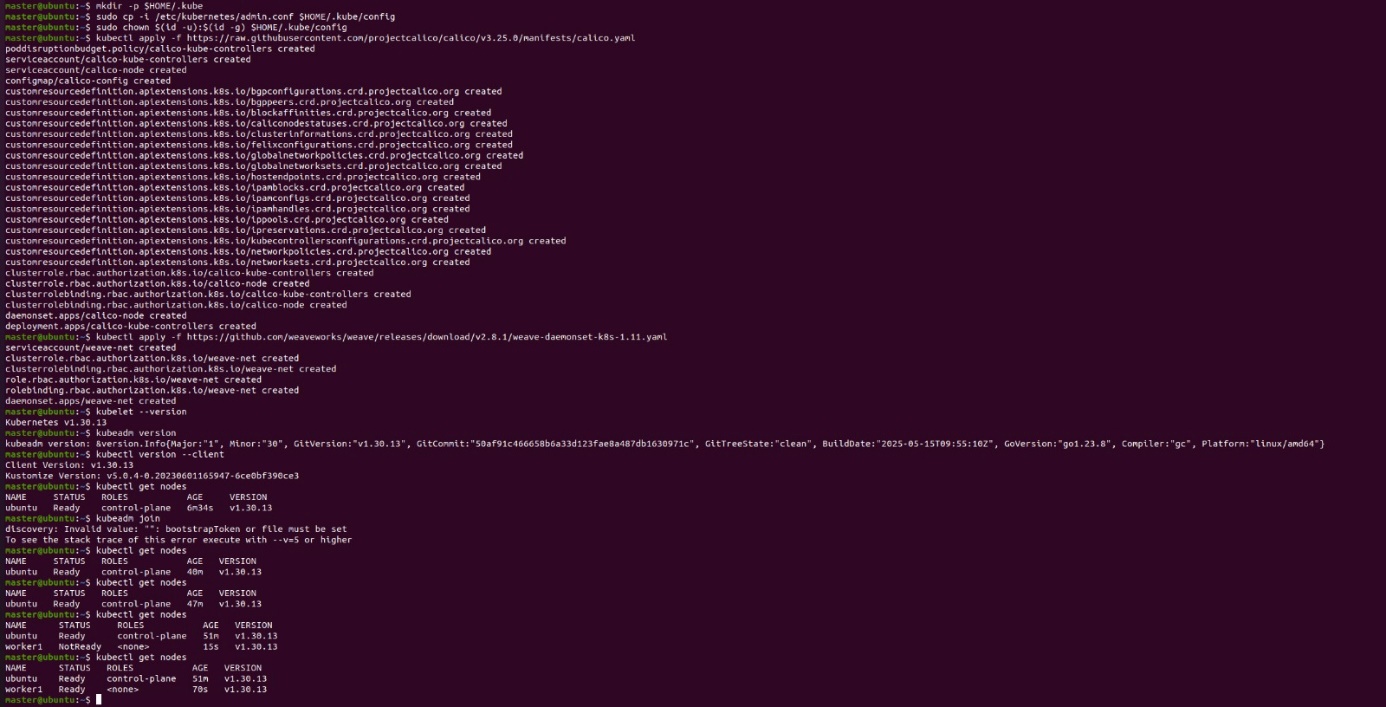
Ensuite, rejoindre le cluster à l’aide de la commande kubeadm join fournie à la fin de l’initialisation du master. Exemple :

sudo kubeadm join <IP\_MASTER>:6443 --token <TOKEN> \

--discovery-token-ca-cert-hash sha256:<HASH>

Après quelques instants, vérifier sur le master :

kubectl get nodes



Tous les nœuds doivent apparaître en statut **Ready**.

## ****VI. Avantages de Kubernetes****

* **Scalabilité** : capacité à gérer automatiquement la montée en charge.
* **Haute disponibilité** : redémarrage automatique des conteneurs en cas de panne.
* **Déploiement simplifié** : mise à jour continue sans interruption.
* **Portabilité** : fonctionne sur le cloud, sur site ou en hybride.
* **Gestion automatisée** : planification, réplication, supervision, mise à l’échelle.
* **Écosystème riche** : intégration avec Helm, Prometheus, Istio, etc.

## ****VII. Conclusion****

Ce TP a permis de mettre en place un cluster Kubernetes fonctionnel, depuis la configuration du nœud maître jusqu’à l’ajout des nœuds workers. Grâce à Kubernetes, il est possible de gérer efficacement des applications conteneurisées avec une grande flexibilité et résilience. Cette technologie s’impose aujourd’hui comme un standard dans les environnements DevOps et les déploiements cloud-native.