

Déploiement Cluster K8S sur RHEL8

30 MARS 2022

Mibu Corp

Créé par : Romain GROSOS



Table des matières

| | | |
|----|-----------------------------------|---|
| 1) | Contexte | 3 |
| 2) | Installation de Docker | 3 |
| 3) | Installation de Kubernetes | 4 |
| A. | Prérequis | 4 |
| B. | K8S | 6 |
| 4) | Configuration de Kubernetes | 6 |
| A. | Initialisation du Cluster | 7 |
| B. | Configuration des workers | 8 |

1) Contexte

Dans ce document, nous verrons l'installation d'un cluster K8S sur des VM RHEL 8.

Pour avoir un cluster fonctionnel, il nous faut à minima 3 VM :

- 1 Master
- 2 Worker

Voici les caractéristiques minimales :

- Master : 2vCPU et 2Go de vRAM
- Worker : 2vCPU et 4Go de vRAM

RHEL 8 est installé dans la version « hôte de virtualisation » sur chacune des VM.

L'OS a été mis à jour :

```
dnf update -y
```

2) Installation de Docker

Nous allons activer le repository de Docker :

```
dnf config-manager --add-repo=https://download.docker.com/linux/centos/docker-ce.repo
```

Puis, on lance l'installation de Docker :

```
dnf install docker-ce -y
```

Pour finalement activer et démarrer le daemon :

```
systemctl enable --now docker
```

Nous allons ensuite modifier la configuration de Docker pour qu'il démarre avec systemd :

```
nano /etc/docker/daemon.json
```

Ajouter les informations suivantes :

```
{  
  "exec-opts": ["native.cgroupdriver=systemd"]  
}
```

Valider la modification avec « CTRL + O » puis quitter avec « CTRL + X »

Afin de prendre en compte la modification, il est nécessaire de redémarrer le daemon de Docker :

```
systemctl restart docker
```

3) Installation de Kubernetes

L'ensemble des éléments contenus dans cette partie sont à réaliser sur l'ensemble des hôtes (master comme worker).

A. Prérequis

Pour avoir les meilleures performances, il est nécessaire de désactiver le swap :

```
swapoff -a
```

Pour rendre définitif cette modification, il faut modifier le fstab :

```
nano /etc/fstab
```

Commentez la ligne correspondant au swap en ajoutant un « # » au début :

```
# /dev/mapper/rhel_kubernetes--m-swap none swap defaults 0 0
```

Un redémarrage de la machine est nécessaire pour prise en compte :

```
reboot
```

Il faut également rendre SELinux permissif :

```
sed -i 's/^SELINUX=enforcing$/SELINUX=permissive/' /etc/selinux/config
```

Si votre environnement ne dispose pas de serveur DNS pour résoudre le nom du master et des worker, il est nécessaire de modifier le fichier hosts pour prendre en compte ce point (remplacez les IP décrites ici par les IP de vos VM) :

```
nano /etc/hosts
```

Et ajoutez les informations suivantes :

```
192.168.1.2      k8s-master
192.168.1.3      k8s-w1
192.168.1.4      k8s-w2
```

Sauvegardez avec « CTRL + O » et quittez avec « CTRL + X »

Il faut également modifier les règles du firewall pour prendre en compte les communications entre les nœuds, et vers les services de K8S.

Côté **master**, il faut ouvrir les ports suivants :

```
firewall-cmd --permanent --add-port=6443/tcp
firewall-cmd --permanent --add-port=2379-2380/tcp
firewall-cmd --permanent --add-port=10250/tcp
firewall-cmd --permanent --add-port=10251/tcp
firewall-cmd --permanent --add-port=10252/tcp
firewall-cmd --permanent --add-port=10255/tcp
firewall-cmd --permanent --add-port=30000-32767/tcp
firewall-cmd --add-masquerade --permanent
firewall-cmd --reload
```

Côté **workers**, il faut ouvrir les ports suivants :

```
firewall-cmd --permanent --add-port=10250/tcp
firewall-cmd --permanent --add-port=10255/tcp
firewall-cmd --permanent --add-port=8472/udp
firewall-cmd --permanent --add-port=30000-32767/tcp
firewall-cmd --add-masquerade --permanent
firewall-cmd --reload
```

B. K8S

Nous passons ensuite à l'installation de Kubernetes en elle-même.

Il nous faut ajouter le repository de Kubernetes :

```
nano /etc/yum.repos.d/kubernetes.repo
```

Ajouter les informations suivantes :

```
[kubernetes]
name=Kubernetes
baseurl=https://packages.cloud.google.com/yum/repos/kubernetes-el7-x86_64
enabled=1
gpgcheck=1
repo_gpgcheck=1
gpgkey=https://packages.cloud.google.com/yum/doc/yum-key.gpg
https://packages.cloud.google.com/yum/doc/rpm-package-key.gpg
exclude=kubelet kubeadm kubectl
```

Validez avec « CTRL + O » et quittez avec « CTRL + X »

Et nous installons l'ensemble des applications K8S :

```
dnf install -y kubelet kubeadm kubectl --disableexcludes=kubernetes
```

Puis finalement activer et démarrer le daemon :

```
systemctl enable --now kubelet
```

Nos 3 hôtes sont prêts à être configurés.

Docker est installé, l'ensemble du nécessaire côté Kubernetes également.

4) Configuration de Kubernetes

Les éléments contenus dans cette partie sont à réaliser sur des hôtes spécifiques (soit master, soit workers).

A. Initialisation du Cluster

Sur le master :

Nous allons initialiser le cluster Kubernetes :

```
kubeadm init --pod-network-cidr=10.244.0.0/16
```

Le process se déroule, et nous renvoie plusieurs informations :

```
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 10.128.15.228:6443 --token cqb8vy.iicmmqrblm8u9cob \
--discovery-token-ca-cert-hash sha256:79748a56f603e6cc57f67bf90b7db5aeb090107d540d6cc8a8f65b785de7543
[linuxtechi@master-node-k8 ~]$
```



Dans l'encadré, les commandes à entrer pour finaliser l'installation (à partir d'un utilisateur non root).

Les voici :

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

Alternativement, si vous utilisez root :

```
Export KUBECONFIG=/etc/kubernetes/admin.conf
```

Sous la flèche verte, la commande à entrer sur les worker pour rejoindre le cluster.

Attention, ce token est valable 48h.

Notez-la et gardez la commande de côté.

Nous allons ensuite installer Flannel pour gérer la partie interne du réseau K8S :

```
kubectl apply -f
https://github.com/coreos/flannel/raw/master/Documentation/kube-flannel.yml
```

Une fois réalisé, nous pouvons consulter les nœuds du cluster (seul le master est présent à cette étape) :

```
kubectl get nodes -o wide
```

```
[root@kubernetes-m ~]# kubectl get nodes -o wide
NAME                                STATUS    ROLES    AGE      VERSION    INTERNAL-IP    EXTERNAL-IP    OS-IMAGE
kubernetes-m.mibu-corp.fr          Ready    control-plane,master    6d7h    v1.23.5    192.168.1.26    <none>         Red Hat Enterprise Linux 8.5 (Ootpa)
4.18.0-348.20.1.el8_5.x86_64    docker://20.10.14
```

Le nœud « master » apparaît comme « Ready ».

B. Configuration des workers

Sur les workers :

Sur chacun des worker, nous allons lancer la commande, mise de côté à l'étape précédente, pour rejoindre le cluster :

```
kubeadm join 10.128.15.228:6443 --token cqb8vy.iicmmqrb1m8u9cob --discovery-
token-ca-cert-hash
sha256:79748a56f603e6cc57f67bf90b7db5aeb090107d540d6cc8a8f65b785de7543
```

Sur le master :

Nous allons assigner le rôle « worker » aux nœuds correspondants :

```
kubectl label node k8s-w1 node-role.kubernetes.io/worker=worker
kubectl label node k8s-w2 node-role.kubernetes.io/worker=worker
```

Nous vérifions à nouveau l'état des nœuds :


```
kubectl get nodes -o wide
```

Les 2 workers doivent désormais apparaître, en statut « ready » et apparaître avec le rôle « worker » :

```
[root@kubernetes-m ~]# kubectl get nodes -o wide
```

| NAME | KERNEL-VERSION | STATUS | ROLES | CONTAINER-RUNTIME | AGE | VERSION | INTERNAL-IP | EXTERNAL-IP | OS-IMAGE |
|----------------------------|----------------|--------|----------------------|------------------------------|------|---------|--------------|-------------|-------------------------|
| kubernetes-m.mibu-corp.fr | x 8.5 (Ootpa) | Ready | control-plane,master | 4.18.0-348.20.1.el8_5.x86_64 | 6d7h | v1.23.5 | 192.168.1.26 | <none> | Red Hat Enterprise Linu |
| kubernetes-w1.mibu-corp.fr | x 8.5 (Ootpa) | Ready | worker | 4.18.0-348.20.1.el8_5.x86_64 | 6d7h | v1.23.5 | 192.168.1.27 | <none> | Red Hat Enterprise Linu |
| kubernetes-w2.mibu-corp.fr | x 8.5 (Ootpa) | Ready | worker | 4.18.0-348.20.1.el8_5.x86_64 | 6d7h | v1.23.5 | 192.168.1.28 | <none> | Red Hat Enterprise Linu |

Votre cluster Kubernetes est prêt à être utilisé.