NGINX INGRESS

Objetivos:

- ✓ Despliegue de un cluster de k8s.
- ✓ Despliegue de la herramienta de NGINX Ingress.

PROCEDIMIENTO

1. **Lo primero que necesitamos son 4 instancias que servirán para instalar el clúster de k8s que tendrá un master y dos nodos. Y la cuarta instancia que nos ayudara con ansible.**

2. **Instalamos las dependencias sobre el servidor que funcionara como ansible.**

```
# yum install epel-release -y
# yum install git ansible -y
```

3. **Clonamos los playbooks de instalación para el cluster que se encuentran en el repositorio de git.**

```
# git clone https://gitlab.com/jartmontes/ngxexpert-k8s.git
```

4. **Modificamos el archivo de variables ya que este contiene las ips para poder instalar el cluster.**

```
# cd ~/ngxexpert-k8s/playbooks
# vim k8s_vars
...
#INGRESAR LA IP DEL MASTER
ad_addr: 192.168.1.X

#INGRESAR LAS IPS DEL CLUSTER
ip_master: 192.168.1.X
ip_nodo1: 192.168.1.Y
ip_nodo2: 192.168.1.Z
...
```

5. Creamos la llave y la compartimos con los nodos que serán miembros del cluster.

```
# echo "192.168.1.212 master.ose.pe master" >> /etc/hosts
# echo "192.168.1.213 nodo1.ose.pe nodo1" >> /etc/hosts
# echo "192.168.1.214 nodo2.ose.pe nodo2" >> /etc/hosts
# ssh-keygen
# ssh-copy-id root@master
# ssh-copy-id root@nodo1
# ssh-copy-id root@nodo2
```

6. Con los pre requisitos ya listo, procedemos con la instalación del cluster.

```
# cd ~/ngxexpert-k8s/
# ansible-playbook -i hosts install-cluster.yaml -v
```

7. La instalación del cluster tomará por lo menos 8 minutos.

8. Una vez el cluster haya sido instalado, validamos con el siguiente comando en el servidor master.

```
# kubectl get nodes
```

```
[root@master ~]# kubectl get nodes
NAME           STATUS   ROLES    AGE   VERSION
master.ose.pe  Ready    master   47m   v1.18.18
nodo1.ose.pe   Ready    <none>   45m   v1.18.18
nodo2.ose.pe   Ready    <none>   45m   v1.18.18
[root@master ~]#
```

9. Para poder instalar NGINX Ingress, necesitamos clonar el siguiente repositorio desde el master.

```
# git clone https://github.com/nginxinc/kubernetes-ingress/
# cd ~/kubernetes-ingress/deployments
# git checkout v1.11.1
```

10. La versión que se va a instalar solamente es admisible en un cluster de k8s 1.18 o superior.

11. Para iniciar con la instalación se necesita crear diferentes recursos desde el namepsace y secrets. Para ello ejecutamos los siguientes yamls en el servidor master.

```
# cd ~/kubernetes-ingress/deployments
# kubectl apply -f common/ns-and-sa.yaml
# kubectl apply -f rbac/rbac.yaml
# kubectl apply -f rbac/ap-rbac.yaml
# kubectl apply -f common/default-server-secret.yaml
#kubectl apply -f common/nginx-config.yaml
```

```
# kubectl apply -f common/ingress-class.yaml
# kubectl apply -f common/crds/k8s.nginx.org_virtualservers.yaml
# kubectl apply -f common/crds/k8s.nginx.org_virtualserverroutes.yaml
# kubectl apply -f common/crds/k8s.nginx.org_transportservers.yaml
# kubectl apply -f common/crds/k8s.nginx.org_policies.yaml
```

12. Antes de proceder con la creación de los pods, modificamos el archivo de deployment el cual contiene la cantidad de réplicas y también el origen de la imagen.

```
# cd ~/kubernetes-ingress/deployments
# vi deployment/nginx-ingress.yaml
```

```
apiVersion: apps/v1
kind: Deployment
metadata:
  name: nginx-ingress
  namespace: nginx-ingress
spec:
  replicas: 2
  selector:
    matchLabels:
      app: nginx-ingress
  template:
    metadata:
      labels:
        app: nginx-ingress
     #annotations:
       #prometheus.io/scrape: "true"
       #prometheus.io/port: "9113"
    spec:
      serviceAccountName: nginx-ingress
      containers:
      - image: jartmontes/ngxexp-ingress:1.11.1
        imagePullPolicy: IfNotPresent
        name: nginx-ingress
        ports:
        - name: http
```

13. Ya modificado el archivo YAML lo ejecutamos.

```
# cd ~/kubernetes-ingress/deployments
# kubectl apply -f deployment/nginx-ingress.yaml
```

14. Esperamos a que se termine de ejecutar con el siguiente comando.

```
# kubectl get pods -n nginx-ingress
```

```
[root@master deployments]# kubectl get pods -n nginx-ingress
NAME                             READY   STATUS    RESTARTS
nginx-ingress-5976cd8fd5-2gr4d   1/1     Running   0
nginx-ingress-5976cd8fd5-86lqp   1/1     Running   0
```

15. **Ya con los pods listos, se exponen utilizando el YAML de service**

```
# cd ~/kubernetes-ingress/deployments
# kubectl create -f service/nodeport.yaml
```

16. **Con esta configuración se tiene listo NGINX Ingress Controller para poder publicar los servicios desde un solo puerto utilizando múltiples host o contextos.**