Kubernetes  
Ecole by Capgemini AWS #2

Exercice pratique

**Introduction**

Amazon Elastic Kubernetes Service (Amazon EKS) est un service managé que vous pouvez utiliser pour exécuter Kubernetes sur AWS sans avoir à installer, à utiliser et à maintenir votre control pane (master nodes). Kubernetes est un système open source conçu pour l'automatisation du déploiement, la mise à l'échelle et la gestion d'applications conteneurisées.

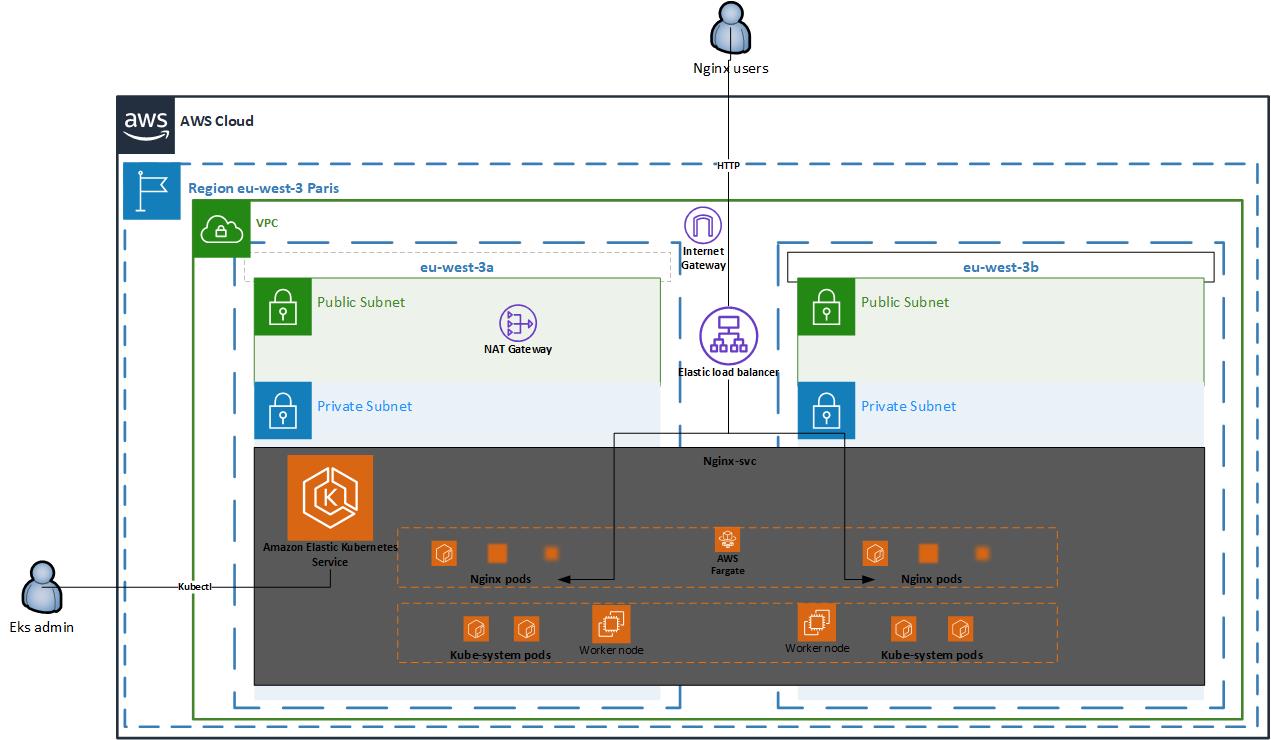
**Comment fonctionne Amazon EKS ?**


    Fonctionnement d'Amazon EKS
   

**Objectif**

Dans ce lab vous allez :

* Déployer les ressources nécessaires à la création d’un cluster EKS (VPC, subnets, IGW, NGW, route table…)
* Configurer une instance EC2 et installer les outils nécessaires pour l’administration d’un cluster EKS et la réalisation du TP :
  + Kubectl : Outil générique d’administration des clusters K8S
  + Eksctl : Outil propriétaire d’AWS pour l’administration des clusters EKS
  + AWS CLI : Outil CLI d’administration AWS, y compris EKS (complémentaire à eksctl)
  + GIT : afin de récupérer les manifest K8S (yaml) et les appliquer sur votre cluster EKS.
* Déployer un cluster EKS avec 2 worker nodes.
* Déployer une application NGINX containérisée.
* Upgrade un cluster EKS & configurer un fargate profile

**HLD**

LAB

# Networking

Vous aurez besoin des ressources suivantes :

1. 1 VPC
2. 4 subnets (2 pub, 2 priv)
3. 1 internet gateway
4. 1 NAT gateway
5. 2 route table (1 pub, 1 priv)

Configurez les RT, la publique doit pointer sur l’IGW, la privée vers la NAT GW.

Assignez les RT aux subnets correspondants (pub / priv).

Lancer une instance EC2 et configurer les outils CLI

1. Créer une instance EC2 Amazon linux 2 T2.micro (public, SSH open)
2. Check la version des AWS CLI, si < 2.X, mettre à jour
3. Configurer le CLI : **aws configure**
4. Download **kubectl**

Check de l’installation : **kubectl version --short --client**

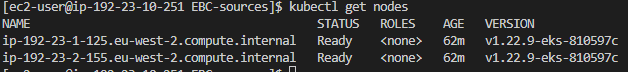
1. Download **eksctl**

Check de l’installation : **eksctl version**

1. Installer GIT : **sudo yum install -y git**
2. Télécharger les fichiers du cours : <https://github.com/jalbalad/EBC-sources.git>

# Provisionner un cluster EKS

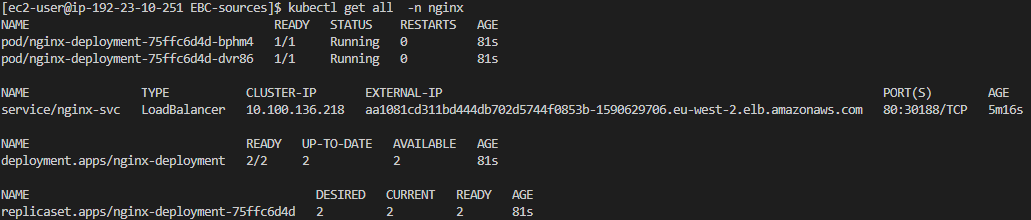
1. Utiliser **eksctl** pour déployer le cluster en appliquant la configuration du fichier **eks.yaml**, adapter le fichier en renseignant les ID de votre VPS et subnets. Eksctl utilise les templates **cloudformation** pour déployer le cluster EKS, vous pouvez suivre la création et les différentes stacks via cloudformation.
2. Confirmer la bonne installation du cluster et le status de vos nodes :



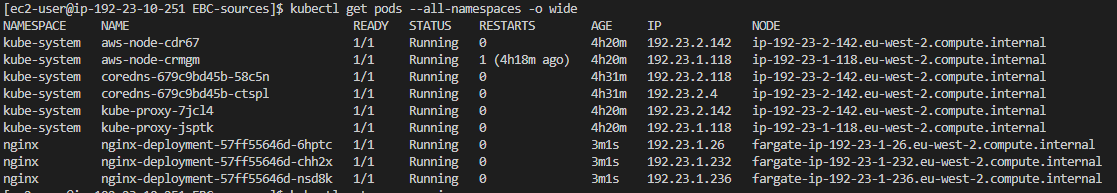
1. Créer un namespace **«** **nginx »**

# Manipulation des objets Kubernetes

1. Utiliser kubectl pour créer un service de type load balancer (**svc-nginx.yaml**)
2. Check le status de votre loadbalancer via kubectl et la console AWS
3. Utiliser kubectl pour créer un déploiement (**deployment-nginx.yaml)**
4. Confirmer la création des différents objets au sein du namepsace « nginx »



1. Lorsque le déploiement est opérationnel, vérifiez que vous pouvez accéder à votre application en utilisant le nom DNS de l’ELB.
2. Scaling horizontal du déploiement nginx pour passer à 3 pods.
3. Modifier la page index.html de nginx via un objet de type **ConfigMap**
4. Déployer un profil fargate pour l’execution de votre namespace nginx, résultat attendu :



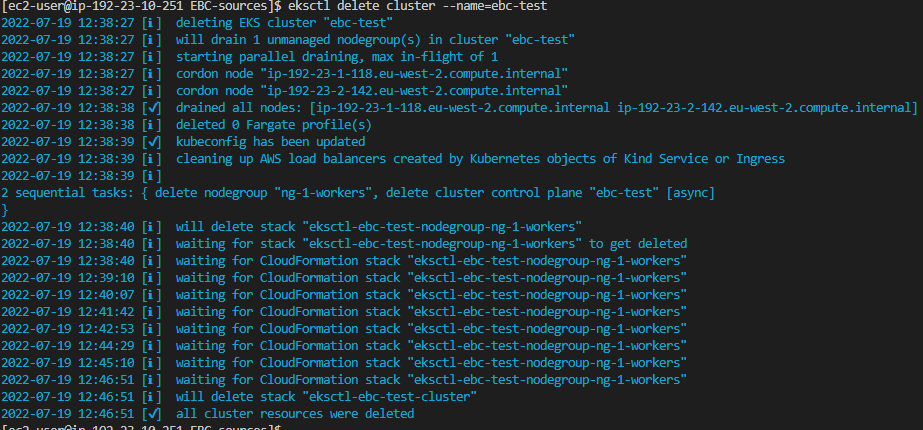
1. Mettre à jour le cluster EKS en version 1.22
2. Déployer un profil fargate pour l’execution du déploiement coredns.

# Tester la haute disponibilité du cluster

1. En supprimant les instances worker.
2. En supprimant les pods nginx.
3. S’assurer que le cluster reprovisionne les ressources supprimées.

# Suppression de l’environnement

1. Supprimer les différents objets créés dans votre cluster (svc, deployment, configmap, namespace…..)
2. Suppression du cluster EKS :



# BONUS

1. Utiliser Terraform pour déployer l’infrastructure.
2. Automatiser le déploiement de l’infrastructure via un workflow github action / gitlab ci
3. Créer un cluster K8S vanilla via le site pédagogique de Kasten : <https://learning.kasten.io/kubernetes/labs/first-kubernetes-cluster/>

# Ressources utiles

K8S : <https://kubernetes.io/fr/docs/home/>

Kubectl : <https://docs.aws.amazon.com/eks/latest/userguide/install-kubectl.html>

Eksctl : <https://docs.aws.amazon.com/eks/latest/userguide/eksctl.html>

AWS CLI : <https://docs.aws.amazon.com/cli/latest/userguide/getting-started-install.html>

Fargate : <https://docs.aws.amazon.com/eks/latest/userguide/fargate-getting-started.html>

# Aide mémoire

## Glossaire

* Cluster - Group of physical or virtual servers wherein Kubernetes is installed
* Node (Master) - Physical or virtual server that controls the Kubernetes cluster (Control Pane)
* Node (Worker) - Physical or virtual servers where workloads run in a given container technology
* Pods - Group of containers and volumes which share the same network namespace
* Labels - User defined Key:Value pair associated to Pods
* Master - Control plane components which provide access point for admins to manage cluster workloads
* Service - An abstraction which serves as a proxy for a group of Pods performing a “service”

## Commande

### Création d'objets

* kubectl apply -f ./my-manifest.yaml # Crée une ou plusieurs ressources
* kubectl apply -f ./my1.yaml -f ./my2.yaml # Crée depuis plusieurs fichiers
* kubectl apply -f ./dir # Crée une ou plusieurs ressources depuis tous les manifests dans dir
* kubectl apply -f https://git.io/vPieo # Crée une ou plusieurs ressources depuis une url
* kubectl create deployment nginx --image=nginx # Démarre une instance unique de nginx
* kubectl explain pods # Affiche la documentation pour les manifests pod

### Visualisation et Recherche

* kubectl get services # Liste tous les services d'un namespace
* kubectl get pods --all-namespaces # Liste tous les Pods de tous les namespaces
* kubectl get pods -o wide # Liste tous les Pods du namespace courant, avec plus de détails
* kubectl get deployment my-dep # Liste un déploiement particulier
* kubectl get pods # Liste tous les Pods dans un namespace
* kubectl get pod my-pod -o yaml # Affiche le YAML du Pod
* kubectl get ns/svc/pv/deployment/… -**n NAMESPACE** # Recherche d’objet dans un namespace
* kubectl describe nodes my-node # Affiche le descriptif d’un node
* kubectl describe pods my-pod # Affiche le descriptif d’un pod

### Suppression de ressources

* kubectl delete -f ./pod.json # Supprime un pod en utilisant le type et le nom spécifiés dans pod.json
* kubectl delete pod,service baz foo # Supprime les pods et services ayant les mêmes noms "baz" et "foo"
* kubectl delete pods,services -l name=myLabel # Supprime les pods et services ayant le label name=myLabel
* kubectl -n my-ns delete pod,svc --all # Supprime tous les pods et services dans le namespace my-ns

### Troubleshooting

* kubectl logs my-pod # Affiche les logs du pod (stdout)
* kubectl logs -l name=myLabel # Affiche les logs des pods ayant le label name=myLabel (stdout)
* kubectl logs my-pod --previous # Affiche les logs du pod (stdout) pour une instance précédente du conteneur
* kubectl logs my-pod -c my-container # Affiche les logs d'un conteneur particulier du pod (stdout, cas d'un pod multi-conteneurs)
* kubectl logs -l name=myLabel -c my-container # Affiche les logs des pods avec le label name=myLabel (stdout, cas d'un pod multi-conteneurs)