Livrable OpenStack S9 2025

NOM prénom : TOSSOU BOCO Elvire

NOM prénom : ZINSOU-PLY Ornel

Quelques parties et codes ont été corrigés avec l'IA

Consignes :

Toutes les réponses doivent être argumentées.

- Le document est donné à titre indicatif. Charge à vous de le modifier pour le rendre le plus précis.
- Agencez votre document de façon propre pour faciliter la lecture. (Pris en compte dans la note)
- Certaines parties de ce document demandent la création ou la remise de fichiers, vous remettrez donc, sur le campus, une archive avec vos deux noms contenant le document rempli ainsi que ces fichiers.

I. Déploiement d'une infrastructure Openstack

A. Kolla et Kolla-ansible

Expliquez en quelques paragraphes la méthode de déploiement utilisée pour déployez vos labs Openstack (qu'est-ce que Kolla, Kolla-ansible, comment ça fonctionne, avec quels outils technologiques et quelle(s) commande(s) permet(tent) de lancer le déploiement)

Pour déployer les labs Openstack, on utilise la méthode de conteneurisation et d'automatisation. Cette méthode facilite l'isolation, la maintenance et la mise à jour grâce à deux outils :

- *Kolla est un outil dont l'objectif est de fournir les services d'Openstack tels que Nova, Glance, Neutron et etc mais sous forme de conteneurs. En effet, les différents services sont donc sous forme d'images Docker. Ainsi, Kolla fournit les services sous forme d'images Docker mais elle ne fait pas le déploiement.*

- *Quant à Kolla-ansible, il automatise le déploiement d'Openstack avec des playbooks ansible à partir des images Docker de Kolla.*

Les différents outils technologiques utilisés sont :
- *Docker pour la conteneurisation*
- *Ansible pour l'automatisation*
- *MariaDB pour le stockage des données*
- *HAProxy pour la haute disponibilité*

Les commandes permettant le déploiement sont et se basent sur le fichier d'inventaire appelé multinode :
- *Kolla-ansible -i multinode bootstrap-servers pour l'installation des paquets sur les hôtes*
- *Kolla-ansible -i multinode prechecks pour la vérification de la configuration*
- *Kolla-ansible -i multinode deploy pour le déploiement d'Openstack*

B. Configuration de votre lab Openstack

Vous déposerez sur le campus avec ce livrable le fichier globals.yml utilisé en TP après activation des services cinder.

C. Evolution de l'infrastructure : point de vue physique

On voudrait augmenter le nombre de compute node sur notre infrastructure Openstack. En supposant que l'on est déjà à disposition trois nouveaux serveurs physiques, expliquez en quelques paragraphes la démarche pour intégrer ces nouvelles machines en tant que compute node.

(On considère que les réseaux Host-only de notre lab Openstack sont accessibles par les autres serveurs physiques.)

Pour ce fait, il faut s'assurer que les serveurs sont prêts :

- *Les serveurs doivent être allumés et accessibles via SSH.*
- *Ils doivent avoir des adresses IP fixes et doivent communiquer avec le reste du lab.*
- *Installer une clé SSH pour que le nœud de déploiement puisse se connecter aux nouveaux serveurs.*

Ensuite, il faut modifier le fichier multinode en ajoutant les adresses IP et les noms des nouvelles machines dans la section compute.

Enfin on va lancer les commandes :

- *kolla-ansible -i multinode bootstrap-servers --limit [adresses IP des serveurs] pour l'installation des outils de base.*
- *kolla-ansible -i multinode prechecks --limit [adresses IP des serveurs] pour la vérification de la configuration.*
- *kolla-ansible -i multinode deploy --limit [adresses IP des serveurs] pour le déploiement.*

D. Evolution de l'infrastructure : point de vue services

On voudrait ajouter à notre infrastructure Openstack un service de systèmes de fichiers partagé. Expliquez en quelques paragraphes quel service il faudrait sélectionner, son fonctionnement, ses prérequis, ainsi que la démarche et les actions techniques à mener pour le déploiement de ce service.

Le service à installer est Manila. Manila est le service "Shared File Systems" d'Openstack. Il fonctionne comme un dossier partagé sur le réseau (NFS) que plusieurs VMs peuvent utiliser en même temps pour s'échanger des fichiers.

Fonctionnement
Manila ne stocke pas les fichiers lui-même, il commande un "serveur de stockage" (le backend).L'utiisateur demande un partage via Manila, ensuite Manila crée automatiquement ce dossier sur le serveur de stockage et donne les autorisations aux VMs.

Prérequis
Il faut avoir un serveur de stockage disponible comme le NFS et un réseau configuré pour que les VMs puissent voir le serveur de stockage.

Démarche technique
Pour l'ajouter il faut :

- *Activer le service dans le fichier globals.yml:*

  enable_manila : "yes"
  enable_manila_backend_generic: "yes"

- *Utiliser la commande reconfigure pour mettre à jour la configuration :*

  kolla-ansible -i multinode prechecks
  kolla-ansible -i multinode reconfigure

- *Créer un modèle de partage pour que les utilisateurs puissent cliquer sur "Créer un partage" dans l'interface :*

  openstack share type create default_share False

II. Déploiement et Orchestration

On s'interesse dans cette partie aux déploiements d'instances et de diverses ressources Openstack. Vous déposerez sur le campus les fichiers ou archives correspondants aux déploiements ci-dessous.

A. Stack de création d'une instance simple avec des volumes persistants

On veut à terme une stack qui permettent de lancer une instance avec une adresse IP accessible depuis l'extérieur de votre infrastructure Openstack. On veut que cette instance soit uniquement accessible depuis l'extérieur en http et https car elle servira de serveur web.
Pour cette première stack, on cherche simplement à déployer une instance sur un réseau interne avec deux volumes de données.
Le volume systeme devra porter le nom « system-<nom_de_l'instance> ». Le volume de données persistantes devra porter le nom « data-<nom_de_l'instance> » et devra être monté dans /srv sur l'instance.
On devrait donc se retrouver avec une instance liée à deux volumes, le premier qui sera l'image système customisée, le second contiendra les données de l'application hébergée dans /srv (par exemple pour un serveur web, le volume système aura apache d'installé, mais le documentroot pointera vers /srv/web/html, les pages web seront donc stockées dans /srv/web/html, qui est un sous-répertoire du point de montage du volume de données).

De cette façon, on devrait pouvoir garder les volumes de données et les rattachés à d'autres socle système au besoin.

Cette stack ne devra pas déployer un serveur web, mais juste une instance contenant deux volumes, l'un système, l'autre données persistantes. La stack devra prendre en paramètre :
- le nom de l'instance à déployer
- l'image à partir de laquelle déployer l'instance (avec une valeur par défaut sur « debian-11 »)
- le gabarit
- la clé ssh à déployer sur l'instance
- la taille de chaque volumes (système + données)
- le pointeur du volume de données dans le système(avec une valeur par défaut /dev/vdb)
- le réseau sur lequel déployer l'instance (on considère qu'il doit déjà être existant, en d'autres termes pas besoin de créer le réseau)

Cette stack constitue un point de départ vers des stacks plus customisables.

B. Stack de création d'une instance simple avec des volumes persistants, une ip flottante et un groupe de sécurité

Modifiez la stack précédente pour ajouter la possibilité à l'utilisateur de customiser le contenu de son instance via un script shell qu'il donnera en paramètre via user-data et qui pourra servir à installer une application. Cette nouvelle stack devra :
- Associer une ip flottante à l'instance pour qu'elle soit accessible de l'extérieur
- Associer un groupe de sécurité passé en paramètre avec pour valeur par défaut le groupe de sécurité « default-web-secgroup ».
- Associer une ip flottante provenant du réseau externe nommé « Provider »

La stack devra afficher en sortie l'adresse ip privée qui lui à été attribuée, ainsi que l'adresse ip flottante avec laquelle elle est joignable.

C. Stack de création d'une instance bastion

On cherche ici à créer une stack qui permette la création d'un environnment type bastion.
Cette stack devra déployer une instance contenant deux volumes, l'un système, l'autre données persistantes.
Elle devra également créer :
- Un réseau privé auquel elle sera liée (avec son sous-réseau)
- Une ip flottante associée en provenance du réseau externe « Provider »
- Un groupe de sécurité authorisant le ssh pour tous à associer à l'instance
- Une nouvelle clé ssh

La stack devra prendre en paramètre à minima :
- le nom de l'instance à déployer
- l'image à partir de laquelle déployer l'instance (avec une valeur par défaut sur « debian-11 »)
- le gabarit
- la clé ssh à créer et déployer sur l'instance
- la taille de chaque volumes (système + données)
- le pointeur du volume de données dans le système(avec une valeur par défaut /dev/vdb)
- le nom du réseau privé à créer
- l'adressage du sous-réseau privé (pour son nom, on prendra le « nomDuRéseau_subnet »

D. Stack de création d'un environnement d'instances scalable

Proposez une stack permettant de créer un environnement composé de plusieurs instances identiques. On voudrait pouvoir donner en paramètre un nombre d'instances à déployer pour pouvoir eventuellement modifier ce paramètre et faire de la mise à l'échelle (du scaling).

On pourrait à terme se servir de la stack bastion pour initier une infrastructure privée permettant l'administration en ssh d'un environnement scalable constitué de machines web comme avec l'instance 2c déployée avec cette stack.