



# Rapport du Projet de Cloud Computing

## Mise en place d'une infrastructure cloud privé avec OpenStack

**Filière :** Cybersécurité et Confiance Numérique

**Encadré par :** M.KANDOUSSI EL MEHDI

**Réalisé par :** Zahra KASMOUTI et Imane TOUIBA

**Année Universitaire :** 2024-2025

# Table des matières

<b>Introduction</b>	<b>7</b>
<b>1 Préparation de la machine</b>	<b>8</b>
1.1 Configuration de la machine virtuelle . . . . .	8
1.1.1 Les paramètres de la machine virtuelle . . . . .	8
1.1.2 Configuration de l'interface réseau . . . . .	9
1.2 Installation des services nécessaires . . . . .	11
1.2.1 Installation et configuration de Chrony . . . . .	11
1.2.2 Installation et configuration de MariaDB . . . . .	11
1.2.3 Installation et configuration de RabbitMQ, Memcached, et Nginx . . . . .	11
<b>2 Keystone</b>	<b>13</b>
2.1 Installation et configuration de keystone . . . . .	13
2.1.1 Configuration de MariaDB pour keystone . . . . .	13
2.1.2 Installation de keystone . . . . .	14
2.1.3 Configuration de Keystone . . . . .	14
2.1.4 Configuration du service Apache . . . . .	15
2.2 Ajout des projets sur Keystone . . . . .	15
2.2.1 Création et chargement du fichier des variables d'environnement . . . . .	15
2.2.2 Création des projets . . . . .	16
<b>3 Glance</b>	<b>18</b>
3.1 Préparation de Keystone et MariaDB pour Glance . . . . .	18
3.1.1 Configuration de Keystone pour Glance . . . . .	18
3.1.2 Configuration de MariaDB pour Glance . . . . .	20
3.2 Installation et Configuration de Glance . . . . .	21
3.2.1 Installation de Glance . . . . .	21
3.2.2 Configuration de Glance . . . . .	21
3.2.3 Configuration de Nginx pour les paramètres du proxy .	24

<b>4 Ajout des images des machines virtuelles</b>	<b>25</b>
4.1 Ajout de l'image virtuelle à Glance . . . . .	25
<b>5 Nova</b>	<b>27</b>
5.1 Préparation de Keystone et MariaDB pour Nova . . . . .	27
5.1.1 Configuration de Keystone pour Nova . . . . .	27
5.1.2 Configuration de MariaDB pour Nova . . . . .	32
5.2 Installation et configuration de Nova . . . . .	34
5.2.1 Installation des services de Nova . . . . .	34
5.2.2 Configuration de Nova . . . . .	34
5.2.3 Configuration de Nginx et activation des services de Nova . . . . .	40
5.2.4 Installation et configuration de Nova Compute . . . . .	42
<b>6 Neutron</b>	<b>45</b>
6.1 Préparation de Keystone et MariaDB pour Neutron . . . . .	45
6.1.1 Configuration Keystone pour Neutron . . . . .	45
6.1.2 Configuration MariaDB pour Neutron . . . . .	47
6.2 Installation et configuration des services de Neutron . . . . .	48
6.2.1 Installation des services de Neutron . . . . .	48
6.2.2 Configuration des services de Neutron . . . . .	49
6.2.3 Configuration Nginx . . . . .	52
6.2.4 Démarrage des services de Neutron . . . . .	52
6.3 Configuration du réseau Neutron . . . . .	54
6.3.1 Configuration des services Neutron . . . . .	54
6.3.2 Création d'un réseau virtuel . . . . .	55
<b>7 Ajout d'utilisateurs Openstack et ajout des Flavors</b>	<b>57</b>
<b>8 Crédit et Démarrage des instances de machine virtuelle</b>	<b>62</b>
8.1 Préparation de l'environnement . . . . .	62
8.1.1 Configuration du groupe de sécurité . . . . .	62
8.1.2 Configuration des clés SSH . . . . .	63
8.2 Crédit et démarrage des instances . . . . .	64
8.2.1 Configuration du réseau . . . . .	64
8.2.2 Lancement des instances . . . . .	64
8.2.3 Configuration des règles de sécurité pour SSH et ICMP	66
8.3 Accès aux instances . . . . .	68
8.3.1 Accès via la Console VNC . . . . .	68
8.3.2 Tests de connectivité . . . . .	69

**Conclusion** **73**

**Bibliographie** **73**

# Table des figures

1.1	La configuration de la machine virtuelle . . . . .	9
1.2	La configuration du réseau . . . . .	10
1.3	La configuration de DHCP . . . . .	10
2.1	Contenu du fichier <code>~/keystonerc</code> . . . . .	16
2.2	Création d'un projet service . . . . .	16
2.3	vérification de la création du projet service . . . . .	17
3.1	Ajout d'un utilisateur glance dans le projet service . . . . .	18
3.2	Création d'un service associé à glance . . . . .	19
3.3	Création d'un endpoint public pour Glance . . . . .	19
3.4	Création d'un endpoint interne pour Glance . . . . .	20
3.5	Création d'un endpoint admin pour Glance . . . . .	20
3.6	Liste des endpoints . . . . .	20
3.7	Configuration de MariaDB pour Glance . . . . .	21
3.8	fichier <code>/etc/glance/glance-api.conf</code> partie 1 . . . . .	22
3.9	fichier <code>/etc/glance/glance-api.conf</code> partie 2 . . . . .	23
3.10	Configuration du fichier <code>/etc/nginx/nginx.conf</code> . . . . .	24
4.1	Ajout de l'image virtuelle à Glance . . . . .	26
4.2	Vérification de la création de l'image cirros . . . . .	26
5.1	Ajout des utilisateurs pour Nova dans Keystone . . . . .	28
5.2	création de l'utilisateur placement dans le projet service . . . . .	28
5.3	création d'une entrée de service pour Nova . . . . .	29
5.4	création d'une entrée de service pour placement . . . . .	29
5.5	création d'un endpoint public pour Nova . . . . .	30
5.6	création d'un endpoint interne pour nova . . . . .	30
5.7	création d'un endpoint admin pour Nova . . . . .	31
5.8	création d'un endpoint public pour placement . . . . .	31
5.9	création d'un endpoint interne pour placement . . . . .	32
5.10	création d'un endpoint admin pour placement . . . . .	32

5.11	Configuration de MariaDB pour Nova . . . . .	33
5.12	Suite de la configuration de MariaDB pour Nova . . . . .	33
5.13	Contenu du fichier nova.conf partie1 . . . . .	36
5.14	Contenu du fichier nova.conf partie2 . . . . .	37
5.15	Contenu du fichier nova.conf partie3 . . . . .	38
5.16	Contenu du fichier palcement.conf . . . . .	39
5.17	contenu du fichier nginx.conf . . . . .	41
5.18	Affichage des services Nova configurés et leurs états . . . . .	42
5.19	Activation de la virtualisation matérielle au niveau de VMWare	42
5.20	vérification de l'installation et l'activation de la virtualisation	43
5.21	Affichage des services Nova configurés et leurs états . . . . .	43
6.1	Ajout de l'utilisateur neutron dans le projet service . . . . .	45
6.2	Création d'une entrée de service pour neutron . . . . .	46
6.3	création d'un endpoint public pour neutron . . . . .	46
6.4	création d'une endpoint intenre pour neutron . . . . .	47
6.5	création d'une endpoint admin pour neutron . . . . .	47
6.6	configuration MariaDB pour Neutron . . . . .	48
6.7	Affichage de l'état des services Neutron . . . . .	53
6.8	contenu du fichier /etc/systemd/network/ens38.network . . . . .	54
6.9	l'interface ens38 est active . . . . .	54
6.10	Création d'un réseau virtuel . . . . .	55
6.11	Création de sous-réseau . . . . .	56
6.12	Affichage de la liste des réseaux . . . . .	56
6.13	Affichage de la liste des sous réseaux . . . . .	56
7.1	Ajout d'un Projet . . . . .	57
7.2	Ajout d'un utilisateur . . . . .	58
7.3	Affichage de la liste des rôles . . . . .	58
7.4	contenu du fichier /keystonerc . . . . .	59
7.5	Affichage de la liste des projets . . . . .	60
7.6	Création de flavor m1.iccn . . . . .	60
7.8	Echec de la création d'un flavor par un utilisateur normal . . . . .	61
7.7	liste des flavors . . . . .	61
8.1	création d'un groupe de sécurité pour les instances . . . . .	63
8.2	Affichage des groupes de sécurité disponibles . . . . .	63
8.3	Création d'une paire de clés SSH pour se connecter aux instances et ajout d'une clé publique . . . . .	64
8.4	Création de la première instance nommée cirros . . . . .	65
8.5	Création de la première instance nommée cirros062 . . . . .	65

8.6	État des instances créées . . . . .	66
8.7	Configuration des paramètres de sécurité pour le groupe de sécurité pour accéder à ICMP . . . . .	66
8.8	Configuration des paramètres de sécurité pour le groupe de sécurité pour accéder à SSH . . . . .	67
8.9	Affichage des règles de sécurité associées au groupe de sécurité secgroup01 . . . . .	67
8.10	les URL de la console VNC des deux instances . . . . .	68
8.12	connexion avec le nom d'utilisateur et le mot de passe configurés par défaut . . . . .	69
8.11	interface de connexion VNC . . . . .	69
8.13	ping de l'instance cirros depuis cirros062 . . . . .	70
8.14	ping de l'instance cirros062 à depuis cirros . . . . .	71
8.15	Connexion SSH à l'instance cirros062 depuis l'instance cirros .	71
8.16	Connexion SSH à l'instance cirros depuis l'instance nommée cirros062 . . . . .	72

# Introduction

OpenStack est une plateforme open source qui permet la création et la gestion de clouds privés. Elle regroupe un ensemble de services modulaires et interopérables, chacun ayant un rôle spécifique dans le déploiement d'une infrastructure cloud. Ces services, bien que distincts, fonctionnent ensemble pour garantir la disponibilité, la scalabilité et la sécurité de l'écosystème.

Le service Keystone, par exemple, assure l'authentification et l'autorisation. Il gère les identités et les autorisations via des jetons, garantissant ainsi un accès sécurisé et contrôlé aux ressources. Glance, quant à lui, est dédié à la gestion des images de machines virtuelles. Il permet de stocker, partager et récupérer des modèles de systèmes d'exploitation, facilitant la création d'instances à partir de ces images.

Nova, le service de calcul, est au cœur de l'infrastructure OpenStack. Il offre un accès à des ressources de calcul extensibles et s'intègre à différentes technologies comme KVM, permettant ainsi une gestion flexible des ressources. Enfin, Neutron, le service réseau, simplifie la mise en réseau des instances en fournissant une connectivité avancée et en déployant des processus autonomes pour garantir une interaction fluide entre les composants et autres services OpenStack.

Ce rapport se concentre sur l'installation et la configuration de ces services, offrant ainsi une vue d'ensemble pratique pour déployer une infrastructure cloud privé avec OpenStack en se référant à la documentation [3].

# **Chapitre 1**

## **Préparation de la machine**

Dans ce premier chapitre, nous allons préparer notre machine virtuelle pour déployer notre cloud privé. Cela comprend la configuration du réseau, en établissant une connexion NAT avec la machine hôte pour permettre l'accès à Internet, ainsi qu'un réseau interne dédié à OpenStack, facilitant les communications internes et fournissant les services nécessaires aux utilisateurs du cloud. Par la suite, nous procéderons à l'installation d'un serveur NTP pour synchroniser automatiquement l'heure, ainsi qu'un serveur MariaDB, essentiel pour la persistance des données d'authentification et des fonctions requises par les services d'OpenStack.

### **1.1 Configuration de la machine virtuelle**

#### **1.1.1 Les paramètres de la machine virtuelle**

Nous avons commencé par récupérer l'image ISO d'Ubuntu 24.04, puis avons ajusté les paramètres de la machine virtuelle dans VMware Workstation 17 Pro, comme illustré dans la figure 4.2.

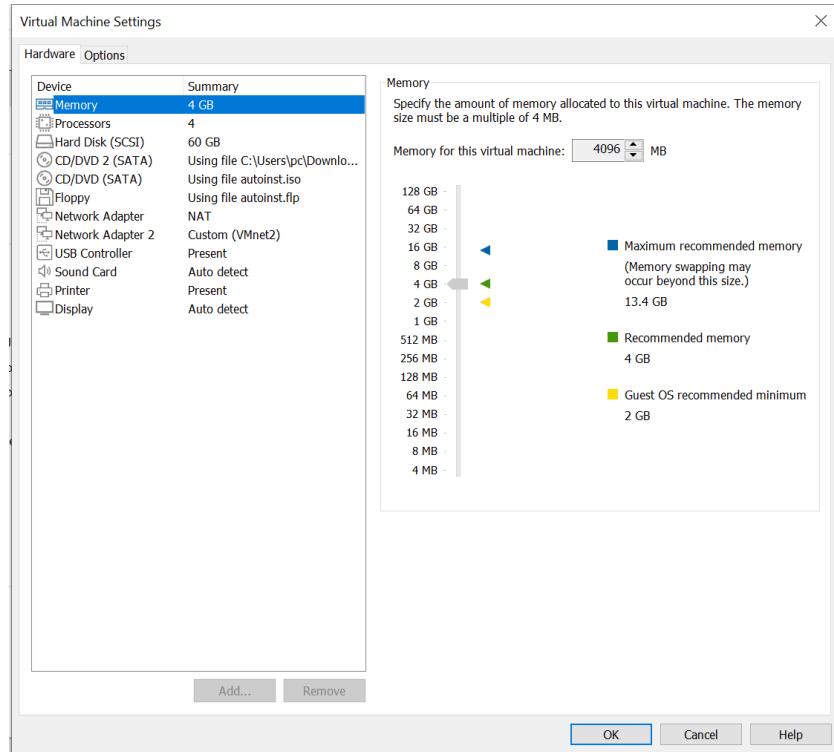


FIGURE 1.1 – La configuration de la machine virtuelle

### 1.1.2 Configuration de l'interface réseau

Pour la configuration du réseau, il est nécessaire de créer une interface virtuelle en mode NAT, permettant ainsi l'accès à Internet. Par la suite, nous configurerons une deuxième interface de type "host-only" dédiée au réseau privé, en utilisant les paramètres suivants (figures 3.10 et 1.3).

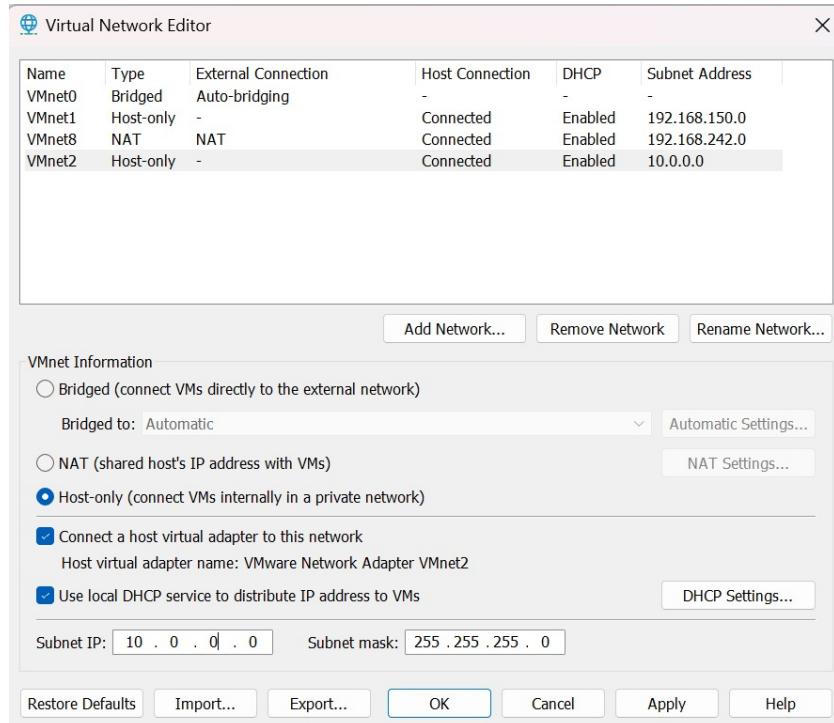


FIGURE 1.2 – La configuration du réseau

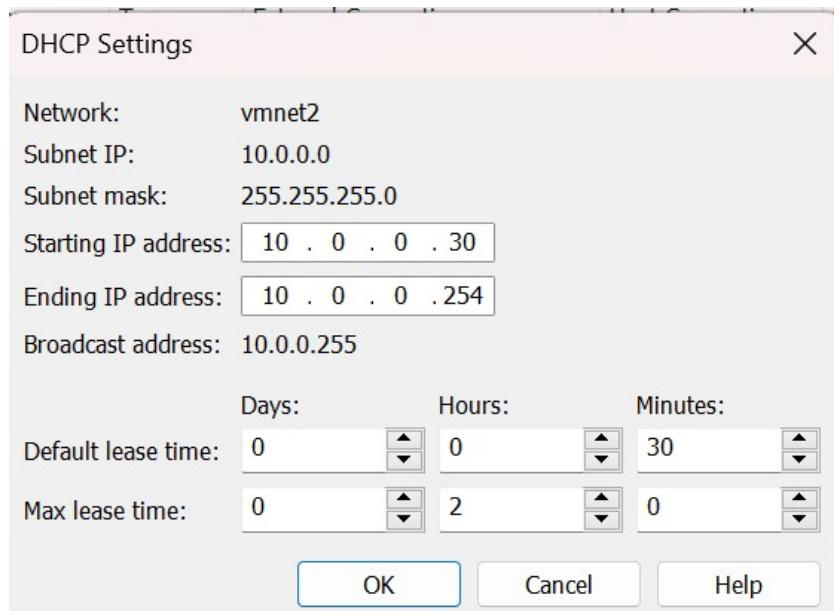


FIGURE 1.3 – La configuration de DHCP

## 1.2 Installation des services nécessaires

### 1.2.1 Installation et configuration de Chrony

Nous installons Chrony pour configurer le serveur NTP et assurer la synchronisation de l'heure. Les étapes sont les suivantes :

1. Installation de Chrony :

```
apt -y install chrony
```

2. Configuration de Chrony :

```
vi /etc/chrony/chrony.conf
```

3. Redémarrage de Chrony :

```
systemctl restart chrony
```

### 1.2.2 Installation et configuration de MariaDB

Nous installons MariaDB pour configurer le serveur de base de données, indispensable au fonctionnement des services d'OpenStack.

1. Installation de MariaDB :

```
apt -y install mariadb-server
```

2. Configuration de MariaDB :

```
vi /etc/mysql/mariadb.conf.d/50-server.cnf
```

3. Redémarrage de MariaDB :

```
systemctl restart mariadb
```

### 1.2.3 Installation et configuration de RabbitMQ, Memcached, et Nginx

Nous installons RabbitMQ, Memcached et Nginx pour compléter l'environnement de travail.

1. Installation des paquets nécessaires :

```
apt -y install rabbitmq-server memcached python3-pymysql nginx  
libnginx-mod-stream
```

2. Ajout et configuration d'un utilisateur RabbitMQ :

```
rabbitmqctl add_user openstack password  
rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

3. Configuration de Memcached :

```
vi /etc/memcached.conf
```

4. Désactivation du site par défaut de Nginx :

```
unlink /etc/nginx/sites-enabled/default
```

5. Redémarrage des services :

```
systemctl restart mariadb rabbitmq-server memcached nginx
```

# Chapitre 2

# Keystone

Keystone, considéré comme le service central d’OpenStack, gère les identités et les autorisations d'accès. Il offre un système complet pour administrer les utilisateurs, les tokens et le catalogue des droits d'accès pour l'ensemble des composants d’OpenStack. De plus, il prend en charge divers formats d'authentification, tels que les mots de passe et d'autres méthodes.

## 2.1 Installation et configuration de keystone

### 2.1.1 Configuration de MariaDB pour keystone

Dans cette section, nous allons créer une base de données pour Keystone, puis ajouter un utilisateur à cette base, en lui attribuant tous les priviléges nécessaires.

1. Se connecter à MariaDB :

```
mysql;
```

2. Créer de la base de données pour Keystone :

```
create database keystone;
```

3. Attribuer les priviléges pour un utilisateur local :

```
GRANT ALL PRIVILEGES ON keystone.*  
to 'keystone'@'localhost' IDENTIFIED BY 'password';
```

4. Attribuer les priviléges pour un utilisateur distant :

```
grant all privileges on keystone.*  
to keystone@'%' identified by 'password';
```

## 2.1.2 Installation de keystone

La commande suivante permet d'installer Keystone et d'autres services nécessaires

```
apt -y install keystone python3-openstackclient apache2  
libapache2-mod-wsgi-py3 python3-oauth2client
```

## 2.1.3 Configuration de Keystone

Nous avons défini l'adresse du serveur Memcache pour améliorer les performances en utilisant le cache, configuré la connexion à la base de données MariaDB où les données de Keystone seront stockées, et activé l'utilisation du mécanisme de tokens Fernet pour garantir la sécurité des tokens. Pour ce faire, nous avons accéder au fichier de configuration de Keystone avec la commande suivante :

```
vi /etc/keystone/keystone.conf
```

Après la synchronisation de Keystone, nous procéderons à l'initialisation des clés, à la configuration de l'hôte de Keystone, puis nous exécuterons la commande ‘keystone-manage’ pour créer un utilisateur, un projet et un rôle, et enfin attribuer ces rôles.

1. Création des tables dans la base de données :

```
su -s /bin/bash keystone -c "keystone-manage db_sync"
```

2. Initialisation de la clé Fernet :

```
keystone-manage fernet_setup --keystone-user keystone  
--keystone-group keystone
```

3. Initialisation des informations d'identification de Keystone :

```
keystone-manage credential_setup --keystone-user keystone  
--keystone-group keystone}
```

4. Définition de l'hôte de l'API Keystone :

```
export controller=controller
```

5. Initialisation de Keystone (bootstrap) :

```
keystone-manage bootstrap --bootstrap-password adminpassword \  
--bootstrap-admin-url http://$controller:5000/v3/ \  
--bootstrap-internal-url http://$controller:5000/v3/ \  
--bootstrap-public-url http://$controller:5000/v3/ \  
--bootstrap-region-id RegionOne
```

### 2.1.4 Configuration du service Apache

Dans le fichier de configuration d'Apache (`/etc/apache2/apache2.conf`), nous avons ajouté la ligne suivante : `ServerName controller`. Cette directive permet de spécifier le nom du serveur utilisé par Apache pour le service Keystone. Elle est essentielle pour garantir que les requêtes HTTP sont correctement dirigées vers Keystone, et elle évite les erreurs de configuration lors du démarrage d'Apache. En définissant ce nom de domaine, nous assurons que Keystone peut être accessible via des URLs adaptées à ce serveur, facilitant ainsi l'accès aux API de gestion des identités et des autorisations.

La commande utilisée est :

```
vi /etc/apache2/apache2.conf
```

## 2.2 Ajout des projets sur Keystone

### 2.2.1 Création et chargement du fichier des variables d'environnement

Nous allons définir des variables d'environnement dans le fichier `~/keystonerc`, les exécuter, puis ajouter leur contenu au fichier `~/.bash_profile` afin qu'elles soient chargées automatiquement à chaque session.

```

GNU nano 7.2                               /root/kestonerc
export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=adminpassword
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
export PS1='\u@\h \W(keystone)\$ '

```

FIGURE 2.1 – Contenu du fichier `~/kestonerc`

Les commandes utilisées :

```

# chmod 600 ~/kestonerc
# source ~/kestonerc
(keystone)# echo "source ~/kestonerc" >> ~/.bashrc

```

## 2.2.2 Crédation des projets

Dans cette dernière étape, nous allons créer un projet nommé service en utilisant la commande :

```

openstack project create --domain default --description
"Service Project" service

```

Field	Value
description	Service Project
domain_id	default
enabled	true
id	7fa4c9352ca741338f0728b5a507a127
is_domain	false
name	service
options	{}
parent_id	default
tags	[]

FIGURE 2.2 – Crédation d'un projet service

Ensuite, en exécutant la commande suivante : `openstack project list`, nous vérifierons la présence de notre projet, comme illustré dans la figure 2.2

```
root@controller ~ (keystone) # openstack project list
+-----+-----+
| ID      | Name   |
+-----+-----+
| 7f43f45a142b4c8ab33cf716dbeb6bcd | admin   |
| 7fa4c9352ca741338f0728b5a507a127 | service |
+-----+-----+
```

FIGURE 2.3 – vérification de la création du projet service

# Chapitre 3

## Glance

Glance, le service d’images, offre aux utilisateurs la possibilité de télécharger, gérer et explorer des ressources de données conçues pour interagir avec d’autres services. Il permet ainsi de découvrir, stocker et récupérer des images de machines virtuelles de manière efficace.

### 3.1 Préparation de Keystone et MariaDB pour Glance

#### 3.1.1 Configuration de Keystone pour Glance

1. création de l’utilisateur glance dans le projet service :

```
openstack user create --domain default --project service  
--password servicepassword glance
```

```
root@controller -(keystone)# openstack user create --domain default --project service --password servicepassword glance  
+-----+  
| Field | Value |  
+-----+  
| default_project_id | 7fe4c9352ca741338f0728b5a507a127 |  
| domain_id | default |  
| enabled | True |  
| id | 284f14ba3e324e8bade1ede9e6d5bc6e |  
| name | glance |  
| options | {} |  
| password_expires_at | None |  
+-----+
```

FIGURE 3.1 – Ajout d’un utilisateur glance dans le projet service

2. Ajout de l’utilisateur glance au rôle admin :

```
openstack role add --project service --user glance admin
```

### 3. Création d'un service associé à glance :

```
openstack service create --name glance --description  
"OpenStack Image service" image
```

```
root@controller ~(keystone)# openstack role add --project service --user glance admin  
root@controller ~(keystone)# openstack service create --name glance --description "OpenStack Image service" image  
+-----+-----+  
| Field | Value |  
+-----+-----+  
| description | OpenStack Image service |  
| enabled | True |  
| id | 5c0dd656sec244d3a542fffb0f0503497 |  
| name | glance |  
| type | image |  
+-----+-----+  
root@controller ~(keystone)# export controller=controller
```

FIGURE 3.2 – Création d'un service associé à glance

### 4. Définition de l'API hôte de Glance :

```
export controller=dlp.srv.world
```

### 5. Crédation d'un endpoint public pour Glance :

```
openstack endpoint create --region RegionOne image public  
http://$controller:9292
```

```
root@controller ~(keystone)# openstack endpoint create --region RegionOne image public https://$controller:9292  
+-----+-----+  
| Field | Value |  
+-----+-----+  
| enabled | True |  
| id | ad48dbe741404f2bbaf3ff0014156c0e |  
| interface | public |  
| region | RegionOne |  
| region_id | RegionOne |  
| service_id | 5c0dd656sec244d3a542fffb0f0503497 |  
| service_name | glance |  
| service_type | image |  
| url | https://controller:9292 |  
+-----+-----+
```

FIGURE 3.3 – Crédation d'un endpoint public pour Glance

### 6. Crédation d'un endpoint interne pour Glance :

```
openstack endpoint create --region RegionOne image internal  
http://$controller:9292
```

openstack endpoint create --region RegionOne image internal http://\$controller:9292	
Field	Value
enabled	True
id	6a85a6a15ecf4b6097f01dfa35c67083
interface	internal
region	RegionOne
region_id	RegionOne
service_id	5c0dds5e5ec244d3a542ffb8f0503497
service_name	glance
service_type	image
url	http://controller:9292

FIGURE 3.4 – Création d'un endpoint interne pour Glance

#### 7. Création d'un endpoint admin pour Glance :

```
openstack endpoint create --region RegionOne image admin
http://$controller:9292
```

openstack endpoint create --region RegionOne image admin http://\$controller:9292	
Field	Value
enabled	True
id	42d94c03dda6455d92ab91981e6790f1
interface	admin
region	RegionOne
region_id	RegionOne
service_id	5c0dds5e5ec244d3a542ffb8f0503497
service_name	glance
service_type	image
url	http://controller:9292

FIGURE 3.5 – Création d'un endpoint admin pour Glance

#### 8. Liste des endpoints :

```
openstack endpoint list
```

ID	Region	Service Name	Service Type	Enabled	Interface	URL
42d94c03dda6455d92ab91981e6790f1	RegionOne	glance	image	True	admin	http://controller:9292
6a85a6a15ecf4b6097f01dfa35c67083	RegionOne	glance	image	True	internal	http://controller:9292
6adb8bf99f0043e88a20a086	RegionOne	keystone	identity	True	internal	http://controller:5000/v3
6fccaa966	RegionOne	keystone	identity	True	admin	http://controller:5000/v3
c577035a126541928160f658	RegionOne	glance	image	True	public	http://controller:9292
5b5f958e	RegionOne	glance	image	True	public	http://controller:9292
fbb36d2b7c284578a07fa3d6	RegionOne	keystone	identity	True	public	http://controller:5000/v3
839e81b6	RegionOne	keystone	identity	True	public	/
fdb8cdf0287a478fb6e23834	RegionOne	keystone	identity	True	public	/
838b4a6a	RegionOne	keystone	identity	True	public	/

FIGURE 3.6 – Liste des endpoints

#### 3.1.2 Configuration de MariaDB pour Glance

Nous allons créer un utilisateur et une base de données dans MariaDB pour Glance en exécutant les commandes suivantes.

1. Se connecter à MariaDB :

```
mysql;
```

2. Créer de la base de données pour glance :

```
create database glance;
```

3. Attribuer les privilèges pour un utilisateur local :

```
GRANT ALL PRIVILEGES ON glance.*  
to glance@'localhost' IDENTIFIED BY 'password';
```

4. Attribuer les privilèges pour un utilisateur distant :

```
grant all privileges on glance.*  
to glance@'%' identified by 'password';
```

```
root@controller ~ (keystone)# mysql  
Welcome to the MariaDB monitor. Commands end with ; or \g.  
Your MariaDB connection id is 47  
Server version: 10.11.8-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04  
  
Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.  
  
Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.  
  
MariaDB [(none)]> create database glance;  
Query OK, 1 row affected (0.044 sec)  
  
MariaDB [(none)]> grant all privileges on glance.* to glance@'localhost' identified by 'password';  
Query OK, 0 rows affected (0.047 sec)  
  
MariaDB [(none)]> grant all privileges on glance.* to glance@'%' identified by 'password';  
Query OK, 0 rows affected (0.008 sec)  
  
MariaDB [(none)]> exit  
Bye
```

FIGURE 3.7 – Configuration de MariaDB pour Glance

## 3.2 Installation et Configuration de Glance

### 3.2.1 Installation de Glance

Nous allons installer Glance avec la commande suivante :

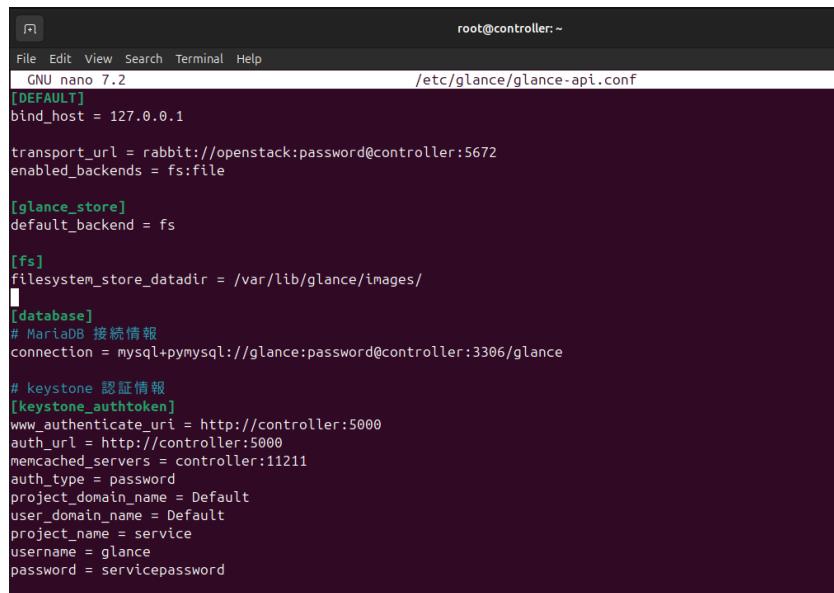
```
apt -y install glance
```

### 3.2.2 Configuration de Glance

La configuration s'effectue en apportant des modifications à certaines lignes du fichier ‘/etc/glance/glance-api.conf’.

1. Modifier le fichier `/etc/glance/glance-api.conf` :

```
nano /etc/glance/glance-api.conf
```



The screenshot shows a terminal window titled "root@controller:~". The title bar also displays "GNU nano 7.2" and the file path "/etc/glance/glance-api.conf". The main area of the terminal shows the configuration file content:

```
[DEFAULT]
bind_host = 127.0.0.1

transport_url = rabbit://openstack:password@controller:5672
enabled_backends = fs:file

[glance_store]
default_backend = fs

[fs]
filesystem_store_datadir = /var/lib/glance/images/
|_
[database]
# MariaDB 接続情報
connection = mysql+pymysql://glance:password@controller:3306/glance

# keystone 認証情報
[keystone_auth_token]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = servicepassword
```

FIGURE 3.8 – fichier `/etc/glance/glance-api.conf` partie 1

```
[fs]
filesystem_store_datadir = /var/lib/glance/images/

[database]
# MariaDB 接続情報
connection = mysql+pymysql://glance:password@controller:3306/glance

# keystone 認証情報
[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = glance
password = servicepassword
# Apache2 Keystone で自己署名の証明書を使用の場合は [true]
insecure = true

[paste_deploy]
flavor = keystone
[oslo_policy]
enforce_new_defaults = true
```

FIGURE 3.9 – fichier /etc/glance/glance-api.conf partie 2

2. Modifier les permissions du fichier de configuration de Glance :

```
chmod 640 /etc/glance/glance-api.conf
```

3. Changer le propriétaire et le groupe du fichier de configuration de Glance :

```
chown root:glance /etc/glance/glance-api.conf
```

4. Synchroniser de la base de données de Glance :

```
su -s /bin/bash glance -c "glance-manage db_sync"
```

5. Redémarrer le service Glance API :

```
systemctl restart glance-api
```

6. Activer le service Glance API pour qu'il démarre automatiquement au démarrage du système :

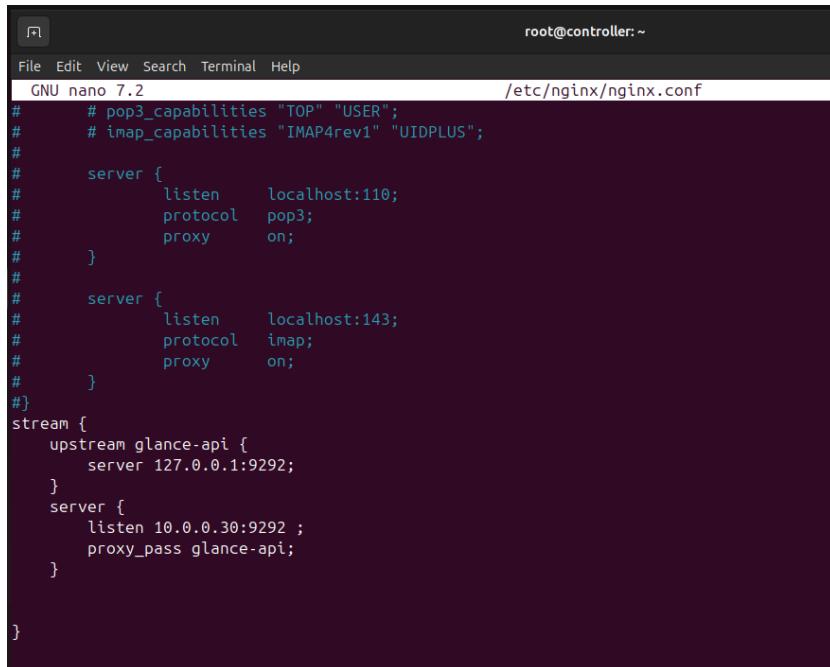
```
systemctl enable glance-api
```

### 3.2.3 Configuration de Nginx pour les paramètres du proxy

Enfin, nous procéderons à la configuration du service Nginx pour héberger notre API Glance, en ajustant les paramètres dans le fichier ‘/etc/nginx/nginx.conf’, puis nous redémarrerons le service pour appliquer les modifications.

1. Ajouter à la fin du fichier /etc/nginx/nginx.conf le contenu de la figure suivante (figure 3.10 ).

```
nano /etc/nginx/nginx.conf
```



The screenshot shows a terminal window titled 'root@controller: ~'. The window title bar also displays 'GNU nano 7.2' and the file path '/etc/nginx/nginx.conf'. The terminal content is the configuration code for Nginx:

```
#      # pop3_capabilities "TOP" "USER";
#      # imap_capabilities "IMAP4rev1" "UIDPLUS";
#
#      server {
#          listen      localhost:110;
#          protocol   pop3;
#          proxy      on;
#      }
#
#      server {
#          listen      localhost:143;
#          protocol   imap;
#          proxy      on;
#      }
#
#}
stream {
    upstream glance-api {
        server 127.0.0.1:9292;
    }
    server {
        listen 10.0.0.30:9292 ;
        proxy_pass glance-api;
    }
}
```

FIGURE 3.10 – Configuration du fichier /etc/nginx/nginx.conf

2. Redémarrer le service nginx : `systemctl restart nginx`

# Chapitre 4

## Ajout des images des machines virtuelles

How the design was Ce chapitre est consacré à la création d'une image, suivie de son intégration dans Glance, que nous avons déjà configuré lors des étapes précédentes.

### 4.1 Ajout de l'image virtuelle à Glance

Tout d'abord, nous avons installé l'image ISO de Cirros à partir du site <https://docs.openstack.org/image-guide/obtain-images.html#id7>.

Ensuite, nous avons ajouté l'image virtuelle à Glance avec la commande suivante :

```
openstack image create "cirros" --file  
/home/imanezahra/Downloads/cirros-0.6.2-x86_64-disk.img  
--disk-format qcow2 --container-format bare --public
```

```

root@controller ~ (keystone) # openstack image create "cirros" --file /home/inanezahra/Downloads/cirros-0.6.2-x86_64-disk.img --disk-format qcow2 --container-format bare --public
+-----+
| Field      | Value
+-----+
| container_format | bare
| created_at     | 2024-12-13T22:24:32Z
| disk_format    | qcow2
| file          | /v2/images/4dcbf12d-ac06-464a-acd7-04d34942807f/file
| id            | 4dcbf12d-ac06-464a-acd7-04d34942807f
| min_disk       | 0
| min_ram        | 0
| name          | cirros
| owner          | 7f43f45a142b4c8ab33cf716dbeb6bcd
| properties     | os_hidden='False', owner_specified.openstack.md5='', owner_specified.openstack.object='images/cirros', owner_specified.openstack.sha256=''
| protected      | False
| schema         | /v2/schemas/image
| status         | queued
| tags           |
| updated_at     | 2024-12-13T22:24:32Z
| visibility     | public
+-----+
root@controller ~ (keystone) #

```

FIGURE 4.1 – Ajout de l'image virtuelle à Glance

Pour voir la liste des images disponibles nous avons exécuté la commande suivante :

```
openstack image list
```

```

root@controller ~ (keystone) # openstack image list
+-----+-----+-----+
| ID      | Name   | Status |
+-----+-----+-----+
| 4dcbf12d-ac06-464a-acd7-04d34942807f | cirros | active |
+-----+-----+-----+

```

FIGURE 4.2 – Vérification de la création de l'image cirros

# **Chapitre 5**

## **Nova**

Nova est le projet OpenStack qui fournit un moyen de fournir des instances de calcul (ou serveurs virtuels), qui est utilisé pour héberger et gérer des systèmes informatiques en cloud. C'est un contrôleur d'analyse d'informatique en cloud, qui est la partie principale d'un système IaaS. Il fournit une capacité de calcul resizable dans OpenStack Nova permet un contrôle complet des ressources informatiques et s'exécute directement dans l'environnement OpenStack. Cela réduit à quelques minutes le temps nécessaire pour obtenir et démarrer de nouvelles instances de serveur.[6]

### **5.1 Préparation de Keystone et MariaDB pour Nova**

#### **5.1.1 Configuration de Keystone pour Nova**

Dans cette sous-section, nous configurons Keystone pour permettre à Nova de s'authentifier et d'interagir avec les autres services d'OpenStack. Dans cette étape nous allons ajouter des utilisateurs pour Nova sur Keystone. Pour ce faire nous avons exécuter un ensemble de commandes :

- Pour ajouter les utilisateurs et autres pour Nova dans Keystone:  
~(keystone)# openstack user create --domain default --project service  
--password servicepassword nova

```
root@controller ~ (keystone) # openstack user create --domain default --project service --password servicepassword neutron
+-----+-----+
| Field | Value |
+-----+-----+
| default_project_id | 7fa4c9352ca741338f0728b5a507a127 |
| domain_id | default |
| enabled | True |
| id | 7a717fafb629499d8b238600b526b82c |
| name | neutron |
| options | {} |
| password_expires_at | None |
+-----+
```

FIGURE 5.1 – Ajout des utilisateurs pour Nova dans Keystone

- Pour ajouter Nova dans le rôle admin :

```
~(keystone) # openstack role add --project service --user nova admin
```

- Pour créer l'utilisateur placement dans le projet service :

```
~(keystone) # openstack user create --domain default --project service
--password servicepassword placement
```

```
root@controller ~ (keystone) # openstack user create --domain default --project service --password servicepassword placement
+-----+-----+
| Field | Value |
+-----+-----+
| default_project_id | 7fa4c9352ca741338f0728b5a507a127 |
| domain_id | default |
| enabled | True |
| id | 8ea2c8d9ec434844a7b02fb5f78259dd |
| name | placement |
| options | {} |
| password_expires_at | None |
+-----+
root@controller ~ (keystone) #
```

FIGURE 5.2 – création de l'utilisateur placement dans le projet service

- Pour ajouter l'utilisateur placement dans le rôle admin :

```
# openstack role add --project service --user placement admin
```

- Pour créer une entrée de service pour nova :

```
~(keystone) # openstack service create --name nova --description
"OpenStack Compute service" compute
```

```

root@controller -(keystone)# openstack service create --name nova --description "OpenStack Compute service" compute
+-----+-----+
| Field | Value |
+-----+-----+
| description | OpenStack Compute service |
| enabled | True |
| id | da77e8f446ce4943b568704a095c0adb |
| name | nova |
| type | compute |
+-----+
root@controller -(keystone)#

```

FIGURE 5.3 – création d'une entrée de service pour Nova

- Pour créer une entrée de service pour placement :
- ```

~(keystone)# openstack service create --name placement --description
"OpenStack Compute Placement service" placement

```

```

root@controller -(keystone)# openstack service create --name placement --description "OpenStack Compute Placement service"
placement
+-----+-----+
| Field | Value |
+-----+-----+
description	OpenStack Compute Placement service
enabled	True
id	280afdbb7ca148b5b3d8cf6d60789e22
name	placement
type	placement
+-----+
root@controller -(keystone)#

```

FIGURE 5.4 – création d'une entrée de service pour placement

- Pour définir l'API hôte de Nova :
- ```
# export controller=controller
```

- Pour créer un endpoint public pour Nova :

```

~(keystone)# openstack endpoint create --region RegionOne compute
public http://$controller:8774/v2.1/%\$(tenant_id)\$s

```

```

root@controller ~(keystone)# openstack endpoint create --region RegionOne compute public http://$controller:8774/v2.1/\%(tenant_id)s
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | b3839e656ab54a5b9500308110aedcc6 |
| interface | public |
| region | RegionOne |
| region_id | RegionOne |
| service_id | da77e8f446ce4943b568704a095c0adb |
| service_name | nova |
| service_type | compute |
| url | http://controller:8774/v2.1/\%(tenant_id)s |
+-----+-----+
root@controller ~(keystone)#

```

FIGURE 5.5 – création d'un endpoint public pour Nova

- Pour créer un endpoint interne pour Nova :

```

~(keystone)# openstack endpoint create --region RegionOne compute internal http://$controller:8774/v2.1/\%(tenant_id)s

```

```

root@controller ~(keystone)# openstack endpoint create --region RegionOne compute internal http://$controller:8774/v2.1/\%(tenant_id)s
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | ibc15d3607f64db3a114e0b59edae7c5 |
| interface | internal |
| region | RegionOne |
| region_id | RegionOne |
| service_id | da77e8f446ce4943b568704a095c0adb |
| service_name | nova |
| service_type | compute |
| url | http://controller:8774/v2.1/\%(tenant_id)s |
+-----+-----+
root@controller ~(keystone)#

```

FIGURE 5.6 – création d'un endpoint interne pour nova

- Pour créer un endpoint admin pour Nova :

```

~(keystone)# openstack endpoint create --region RegionOne compute admin http://$controller:8774/v2.1/\%(tenant_id)s

```

```

root@controller -(keystone)# openstack endpoint create --region RegionOne compute admin http://$controller:8774/v2.1/%(tenant_id)s
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | 2262881050f43bf959632d3bd490f50 |
| interface | admin |
| region | RegionOne |
| region_id | RegionOne |
| service_id | da77e8f446ce4943b568704a095c0adb |
| service_name | nova |
| service_type | compute |
| url | http://controller:8774/v2.1/%(tenant_id)s |
+-----+-----+
root@controller -(keystone)#

```

FIGURE 5.7 – création d'un endpoint admin pour Nova

- Pour créer un endpoint public pour placement :

```

~(keystone)# openstack endpoint create --region RegionOne placement
public http://$controller:8778

```

```

root@controller -(keystone)# openstack endpoint create --region RegionOne placement public http://$controller:8778
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | 17648d749bf949dc8fe2e069ed707190 |
| interface | public |
| region | RegionOne |
| region_id | RegionOne |
| service_id | 280afdbb7ca148b5b3d8cf6d60789e22 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+-----+
root@controller -(keystone)#

```

FIGURE 5.8 – création d'un endpoint public pour placement

- Pour créer un endpoint interne pour placement :

```

~(keystone)# openstack endpoint create --region RegionOne placement
internal http://$controller:8778

```

```

root@controller -(keystone)# openstack endpoint create --region RegionOne placement internal http://$controller:8778
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | f892e4f9d10d4254a5437c6f1835cadc |
| interface | internal |
| region | RegionOne |
| region_id | RegionOne |
| service_id | 280afdbb7ca148b5b3d8cf6d60789e22 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+
root@controller -(keystone)#

```

FIGURE 5.9 – création d'un endpoint interne pour placement

- Pour créer un endpoint admin pour placement :

```

~(keystone)# openstack endpoint create --region RegionOne placement
admin http://$controller:8778

```

```

root@controller -(keystone)# openstack endpoint create --region RegionOne placement admin http://$controller:8778
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | c1a0def1b91245238e8f5ad05d270505 |
| interface | admin |
| region | RegionOne |
| region_id | RegionOne |
| service_id | 280afdbb7ca148b5b3d8cf6d60789e22 |
| service_name | placement |
| service_type | placement |
| url | http://controller:8778 |
+-----+
root@controller -(keystone)#

```

FIGURE 5.10 – création d'un endpoint admin pour placement

### 5.1.2 Configuration de MariaDB pour Nova

Cette sous-section décrit les étapes pour configurer MariaDB afin de répondre aux besoins spécifiques de Nova, notamment la création de bases de données dédiées, la configuration des utilisateurs et l'ajustement des paramètres.

```

root@controller ~ (keystone) # mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 43
Server version: 10.11.8-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database nova;
Query OK, 1 row affected (0.042 sec)

MariaDB [(none)]> grant all privileges on nova.* to nova@'localhost' identified by 'password';
Query OK, 0 rows affected (0.054 sec)

MariaDB [(none)]> grant all privileges on nova.* to nova@'%' identified by 'password';
Query OK, 0 rows affected (0.006 sec)

MariaDB [(none)]> create database nova_api;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> grant all privileges on nova_api.* to nova@'localhost' identified by 'password';
Query OK, 0 rows affected (0.017 sec)

MariaDB [(none)]> grant all privileges on nova_api.* to nova@'%' identified by 'password';
Query OK, 0 rows affected (0.008 sec)

MariaDB [(none)]> create database placement;
Query OK, 1 row affected (0.005 sec)

MariaDB [(none)]>
```

FIGURE 5.11 – Configuration de MariaDB pour Nova

```

MariaDB [(none)]> grant all privileges on placement.* to placement@'localhost' identified by 'password';
Query OK, 0 rows affected (0.039 sec)

MariaDB [(none)]> grant all privileges on placement.* to placement@'%' identified by 'password';
Query OK, 0 rows affected (0.009 sec)

MariaDB [(none)]> create database nova_cell0;
Query OK, 1 row affected (0.002 sec)

MariaDB [(none)]> grant all privileges on nova_cell0.* to nova@'localhost' identified by 'password'
    ->;
Query OK, 0 rows affected (0.011 sec)

MariaDB [(none)]> grant all privileges on nova_cell0.* to nova@'%' identified by 'password';
Query OK, 0 rows affected (0.008 sec)

MariaDB [(none)]> exit
Bye
root@controller ~ (keystone) #
```

FIGURE 5.12 – Suite de la configuration de MariaDB pour Nova

## 5.2 Installation et configuration de Nova

### 5.2.1 Installation des services de Nova

Dans cette sous-section nous avons installer les services de Nova avec la commande :

```
~(keystone)# apt -y install nova-api nova-conductor nova-scheduler  
nova-novncproxy placement-api python3-novaclient
```

### 5.2.2 Configuration de Nova

Après l'installation des services, Nova doit être configuré pour répondre aux besoins spécifiques de l'infrastructure OpenStack. Cette sous-section détaille la configuration des fichiers principaux, notamment pour définir les connexions aux autres services comme Keystone et MariaDB. Pour configurer Nova nous avons modifié les fichiers /etc/nova/nova.conf, /etc/placement/placement.conf ainsi que /etc/httpd/conf.d/00-placement-api.conf et on change aussi les priviléges et les propriétaires de ces fichiers. Nous avons renommé le fichier /etc/nova/nova.conf au /etc/nova/nova.conf.org et nous avons crée un nouveau fichier avec le contenu suivant :

```
[DEFAULT]  
osapi_compute_listen = 127.0.0.1  
osapi_compute_listen_port = 8774  
metadata_listen = 127.0.0.1  
metadata_listen_port = 8775  
state_path = /var/lib/nova  
enabled_apis = osapi_compute,metadata  
log_dir = /var/log/nova  
# RabbitMQ connection info  
transport_url = rabbit://openstack:password@controller:5672  
  
[api]  
auth_strategy = keystone  
  
[vnc]  
enabled = True  
novncproxy_host = 127.0.0.1  
novncproxy_port = 6080  
novncproxy_base_url = http://controller:6080/vnc_auto.html
```

```

# Glance connection info
[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = $state_path/tmp

# MariaDB connection info
[api_database]
connection = mysql+pymysql://nova:password@controller:3306/nova_api

[database]
connection = mysql+pymysql://nova:password@controller:3306/nova

# Keystone auth info
[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = nova
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[placement]
auth_url = http://controller:5000
os_region_name = RegionOne
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = placement
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[wsgi]

```

```
api_paste_config = /etc/nova/api-paste.ini

[oslo_policy]
enforce_new_defaults = true
```

```
GNU nano 7.2                               imanezahra@controller: ~
/etc/nova/nova.conf

[DEFAULT]
osapi_compute_listen = 127.0.0.1
osapi_compute_listen_port = 8774
metadata_listen = 127.0.0.1
metadata_listen_port = 8775
state_path = /var/lib/nova
enabled_apis = osapi_compute,metadata
log_dir = /var/log/nova
# RabbitMQ connection info
transport_url = rabbit://openstack:password@controller:5672

[api]
auth_strategy = keystone

[vnc]
enabled = True
server_listen = 10.0.0.30
server_proxyclient_address = 10.0.0.30
novncproxy_host = 127.0.0.1
novncproxy_port = 6080
novncproxy_base_url = http://controller:6080/vnc_auto.html

# Glance connection info
[glance]
api_servers = http://controller:9292

[oslo_concurrency]
lock_path = $state_path/tmp
[ Read 67 lines ]
```

FIGURE 5.13 – Contenu du fichier nova.conf partie1

```
imanezahra@controller:~  
/etc/nova/nova.conf  
GNU nano 7.2  
  
# MariaDB connection info  
[api_database]  
connection = mysql+pymysql://nova:password@controller:3306/nova_api  
  
[database]  
connection = mysql+pymysql://nova:password@controller:3306/nova  
  
# Keystone auth info  
[keystone_authtoken]  
www_authenticate_uri = http://controller:5000  
auth_url = http://controller:5000  
memcached_servers = controller:11211  
auth_type = password  
project_domain_name = Default  
user_domain_name = Default  
project_name = service  
username = nova  
password = servicepassword  
# if using self-signed certs on Apache2 Keystone, turn to [true]  
insecure = true  
  
[placement]  
auth_url = http://controller:5000  
os_region_name = RegionOne  
auth_type = password  
project_domain_name = Default  
user_domain_name = Default
```

FIGURE 5.14 – Contenu du fichier nova.conf partie2

```

GNU nano 7.2                               imanezahra@controller: ~
/etc/nova/nova.conf

memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = nova
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[placement]
auth_url = http://controller:5000
os_region_name = RegionOne
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = placement
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[wsgi]
api_paste_config = /etc/nova/api-paste.ini

[oslo_policy]
enforce_new_defaults = true

```

FIGURE 5.15 – Contenu du fichier nova.conf partie3

Nous avons changé les droits d'accès de ce fichier et nous avons défini nova comme groupe propriétaire du fichier :

```

~(keystone)# chmod 640 /etc/nova/nova.conf
~(keystone)# chgrp nova /etc/nova/nova.conf

```

De même pour le fichier /etc/placement/placement.conf nous l'avons renommé au /etc/placement/placement.conf.org et nous avons créé un nouveau fichier /etc/placement/placement.conf avec le contenu suivant :

```

[DEFAULT]
debug = false

[api]
auth_strategy = keystone

[keystone_authhtoken]
www_authenticate_uri = http://controller:5000

```

```

auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = placement
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[placement_database]
connection = mysql+pymysql://placement:password@controller:3306/placement

```

The screenshot shows a terminal window titled "imanezahra@controller: ~". The command "GNU nano 7.2" is displayed at the top. The file path "/etc/placement/placement.conf" is shown in the title bar. The content of the file is displayed in the terminal window:

```

[DEFAULT]
debug = false

[api]
auth_strategy = keystone

[keystone_auth_token]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = placement
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[placement_database]
connection = mysql+pymysql://placement:password@controller:3306/placement

```

FIGURE 5.16 – Contenu du fichier placement.conf

Ensuite nous avons modifier la première ligne du fichier /etc/apache2/sites-

enabled/placement-api.conf pour contenir : Listen 127.0.0.1 :8778 Et nous avons changé les droits d'accès de ce fichier et le groupe propriétaire du fichier au groupe placement :

```
~(keystone)# chmod 640 /etc/placement/placement.conf
~(keystone)# chgrp placement /etc/placement/placement.conf
```

### 5.2.3 Configuration de Nginx et activation des services de Nova

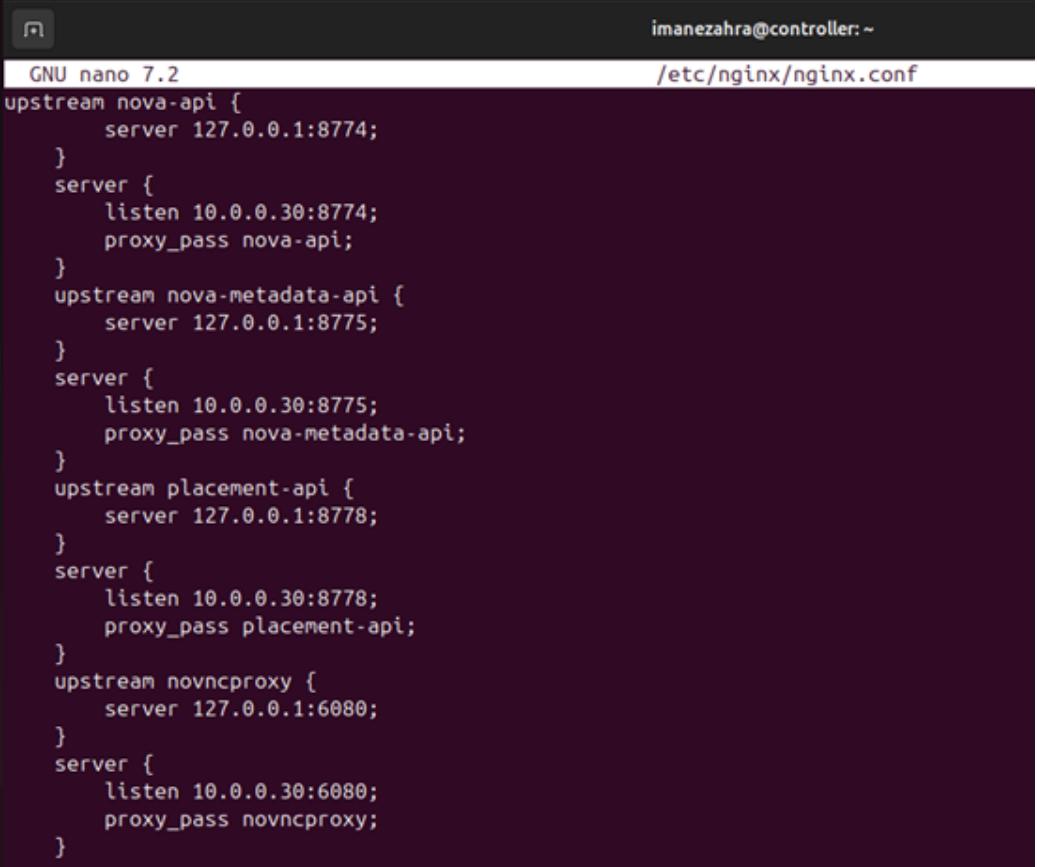
Pour optimiser les performances et la sécurité, Nova est souvent associé à un serveur web comme Nginx. Cette sous-section décrit la configuration de Nginx pour gérer les requêtes liées à Nova et l'activation des services Nova afin de garantir leur bon fonctionnement dans l'environnement OpenStack. Après avoir configuré Nova, nous avons passé aux configurations proxy de nginx en modifiant le fichier /etc/nginx/nginx.conf : Nous avons ajouté les lignes suivantes à la section stream du fichier :

```
upstream nova-api {
    server 127.0.0.1:8774;
}
server {
    listen 10.0.0.30:8774 ;
    proxy_pass nova-api;
}
upstream nova-metadata-api {
    server 127.0.0.1:8775;
}
server {
    listen 10.0.0.30:8775 ;
    proxy_pass nova-metadata-api;
}
upstream placement-api {
    server 127.0.0.1:8778;
}
server {
    listen 10.0.0.30:8778 ;
    proxy_pass placement-api;
}
upstream novncproxy {
    server 127.0.0.1:6080;
```

```

}
server {
    listen 10.0.0.30:6080;
    proxy_pass novncproxy;
}

```



The screenshot shows a terminal window titled 'GNU nano 7.2' with the file path '/etc/nginx/nginx.conf'. The content of the file is as follows:

```

upstream nova-api {
    server 127.0.0.1:8774;
}
server {
    listen 10.0.0.30:8774;
    proxy_pass nova-api;
}
upstream nova-metadata-api {
    server 127.0.0.1:8775;
}
server {
    listen 10.0.0.30:8775;
    proxy_pass nova-metadata-api;
}
upstream placement-api {
    server 127.0.0.1:8778;
}
server {
    listen 10.0.0.30:8778;
    proxy_pass placement-api;
}
upstream novncproxy {
    server 127.0.0.1:6080;
}
server {
    listen 10.0.0.30:6080;
    proxy_pass novncproxy;
}

```

FIGURE 5.17 – contenu du fichier nginx.conf

Et nous avons supprimé ces deux lignes situées à la fin de la section stream :

```

ssl_certificate "/etc/letsencrypt/live/dlp.srv.world/fullchain.pem";
ssl_certificate_key "/etc/letsencrypt/live/dlp.srv.world/privkey.pem";

```

Ensuite, nous avons ajouté des données à la base de données et puis lancé les services de Nova . Et pour vérifier que les services Nova sont correctement installés et fonctionnent comme prévu nous avons lancé la commande :

```
~(keystone)# openstack compute service list
```

Cette commande permet de lister les services Nova configurés, leur état comme elle est montrée sur la figure suivante.

ID	Binary	Host	Zone	Status	State	Updated At
8e800473-a30c-49d3-9078-108fa335e42b	nova-conductor	controller	internal	enabled	up	2024-12-14T02:40:30.000000
9ace4446-72bb-47d8-b39c-db3d6de6d91	nova-scheduler	controller	internal	enabled	up	2024-12-14T02:40:30.000000

FIGURE 5.18 – Affichage des services Nova configurés et leurs états

### 5.2.4 Installation et configuration de Nova Compute

Nova Compute est le composant responsable de la gestion des instances de calcul sur les nœuds physiques ou virtuels. Cette sous-section détaille les étapes pour installer et configurer Nova Compute. Au début nous avons installé l'hyperviseur KVM pour la virtualisation, ainsi que les outils nécessaires pour gérer les machines virtuelles, configurer le réseau via des ponts, et faciliter l'installation des systèmes invités sur notre machine ubuntu :

```
# apt -y install qemu-kvm libvirt-daemon-system libvirt-daemon virtinst bridge-utils
```

Ensuite nous avons activé la virtualisation matérielle pour la machine virtuelle au niveau de VMware.

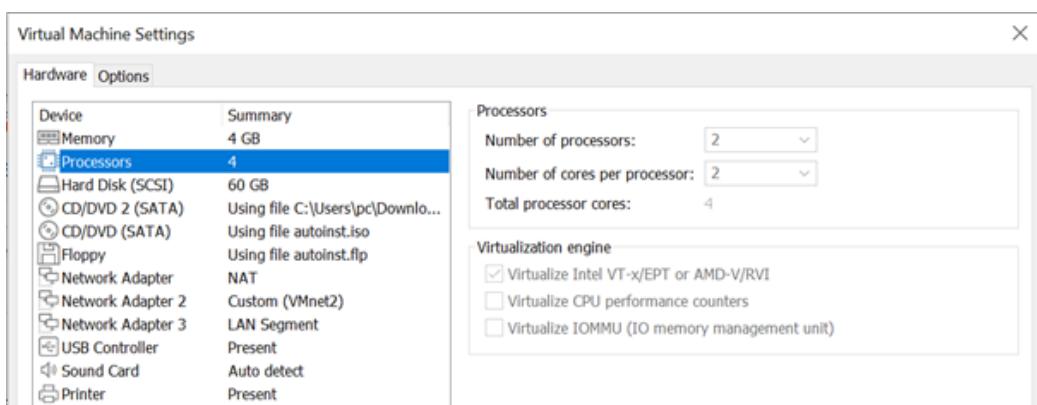


FIGURE 5.19 – Activation de la virtualisation matérielle au niveau de VMware

Et nous avons executé la commande suivante pour vérifier si les modules liés à KVM sont chargés dans le noyau Linux, indiquant la disponibilité de la virtualisation matérielle.

```
imanezahra@controller -(keystone)$ lsmod | grep kvm
kvm_intel           487424  3
kvm                 1404928  2 kvm_intel
irqbypass          12288   1 kvm
```

FIGURE 5.20 – vérification de l’installation et l’activation de la virtualisation

Le fait de voir des résultats dans la sortie de cette commande indique que la virtualisation est disponible et que les modules nécessaires sont actifs dans le noyau. Si aucun résultat n’apparaît, cela peut signifier que KVM n’est pas activé ou que la virtualisation matérielle n’est pas supportée ou activée sur le processeur. Après la vérification du bon fonctionnement de la virtualisation nous avons installé les packages nova-compute et nova-compute-kvm en utilisant la commande suivante :

```
~(keystone)# apt -y install nova-compute nova-compute-kvm
```

Et en plus des configurations que nous avons fait précédemment nous avons ajouté ces deux lignes dans la section [vnc] le fichier /etc/nova/nova.conf puis nous avons démarré le service Nova Compute :

```
server_listen = 10.0.0.30
server_proxyclient_address = 10.0.0.30
```

Pour vérifier le bon fonctionnement des services, nous utilisons la commande :

```
~(keystone)# openstack compute service list
```

ID	Binary	Host	Zone	Status	State	Updated At
8e800473-a30c-49d3-9078-108fa335e42b	nova-conductor	controller	internal	enabled	up	2024-12-15T15:13:43.000000
9ace4446-72bb-47d8-b39c-dbb3d6de6d91	nova-scheduler	controller	internal	enabled	up	2024-12-15T15:13:42.000000
2ca75e71-470c-4b1c-91cd-4b1339a39bf0	nova-compute	controller	nova	enabled	up	2024-12-15T15:13:38.000000

FIGURE 5.21 – Affichage des services Nova configurés et leurs états

En affichant les services Nova nous avons nova-compute qui est ajouté à la liste des services.

# Chapitre 6

## Neutron

Neutron est un projet OpenStack destiné à fournir une « connectivité réseau en tant que service (NaaS)». Le service OpenStack Networking (neutron) fournit une API qui permet aux utilisateurs de construire des topologies réseau riches, de définir la connectivité réseau, configurer les politiques de réseau avancées et l'adressage dans le cloud.[7]

### 6.1 Préparation de Keystone et MariaDB pour Neutron

#### 6.1.1 Configuration Keystone pour Neutron

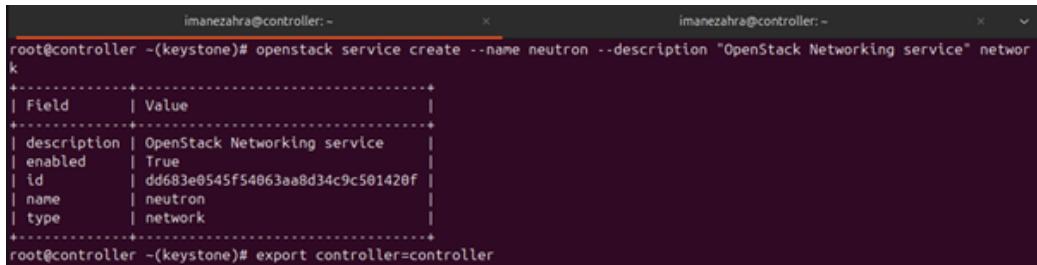
Nous avons ajouté des utilisateurs ou services pour Neutron sur Keystone avec les commandes suivantes : - Pour créer l'utilisateur neutron dans le projet service :

```
~(keystone)# openstack user create --domain default --project service  
--password servicepassword neutron
```

Field	Value
default_project_id	7fa4c9352ca741338f0728b5a507a127
domain_id	default
enabled	True
id	7a717fafb629499d8b238600b526b82c
name	neutron
options	{}
password_expires_at	None

FIGURE 6.1 – Ajout de l'utilisateur neutron dans le projet service

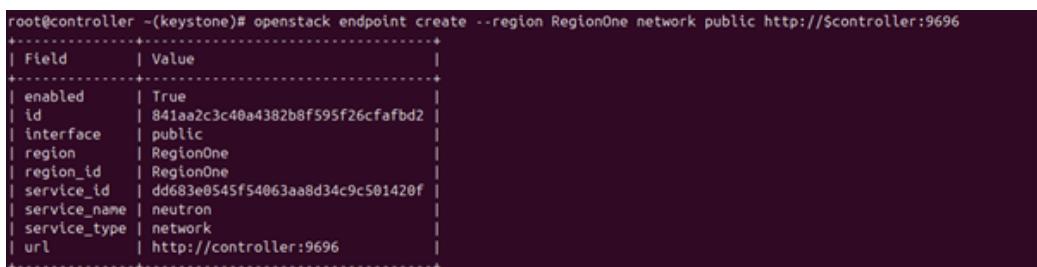
- Pour ajouter neutron dans le rôle admin :
- ```
~(keystone)# openstack role add --project service --user neutron admin
```
- Pour créer une entrée de service pour neutron :
- ```
~(keystone)# openstack service create --name neutron --description "OpenStack Networking service" network
```



```
root@controller -(keystone)# openstack service create --name neutron --description "OpenStack Networking service" network
+-----+-----+
| Field | Value |
+-----+-----+
| description | OpenStack Networking service |
| enabled | True |
| id | dd683e0545f54063aa8d34c9c501420f |
| name | neutron |
| type | network |
+-----+-----+
root@controller -(keystone)# export controller=controller
```

FIGURE 6.2 – Création d'une entrée de service pour neutron

- Pour définir l'API hôte de neutron :
- ```
# export controller=controller
```
- Pour créer un endpoint public pour neutron :
- ```
~(keystone)# openstack endpoint create --region RegionOne network public http://$controller:9696
```



```
root@controller -(keystone)# openstack endpoint create --region RegionOne network public http://$controller:9696
+-----+-----+
| Field | Value |
+-----+-----+
| enabled | True |
| id | 841aa2c3c40a4382b8f595f26cfafbd2 |
| interface | public |
| region | RegionOne |
| region_id | RegionOne |
| service_id | dd683e0545f54063aa8d34c9c501420f |
| service_name | neutron |
| service_type | network |
| url | http://controller:9696 |
+-----+-----+
```

FIGURE 6.3 – création d'un endpoint public pour neutron

- Pour créer un endpoint interne pour neutron :
- ```
~(keystone)# openstack endpoint create --region RegionOne network internal http://$controller:9696
```

| root@controller ~ (keystone)# openstack endpoint create --region RegionOne network internal http://\$controller:9696 |                                   |
|----------------------------------------------------------------------------------------------------------------------|-----------------------------------|
| Field                                                                                                                | Value                             |
| enabled                                                                                                              | True                              |
| id                                                                                                                   | 0ae15068e89742dfaef16306e42ebbed4 |
| interface                                                                                                            | internal                          |
| region                                                                                                               | RegionOne                         |
| region_id                                                                                                            | RegionOne                         |
| service_id                                                                                                           | dd683e0545f54063aa8d34c9c501420f  |
| service_name                                                                                                         | neutron                           |
| service_type                                                                                                         | network                           |
| url                                                                                                                  | http://controller:9696            |

FIGURE 6.4 – création d'une endpoint intenre pour neutron

- Pour créer un endpoint admin pour neutron :

```
~(keystone)# openstack endpoint create --region RegionOne network
admin http://$controller:9696
```

| root@controller ~ (keystone)# openstack endpoint create --region RegionOne network admin http://\$controller:9696 |                                  |
|-------------------------------------------------------------------------------------------------------------------|----------------------------------|
| Field                                                                                                             | Value                            |
| enabled                                                                                                           | True                             |
| id                                                                                                                | 81742dd0010b4ea0a44f7b63ed3eb4cf |
| interface                                                                                                         | admin                            |
| region                                                                                                            | RegionOne                        |
| region_id                                                                                                         | RegionOne                        |
| service_id                                                                                                        | dd683e0545f54063aa8d34c9c501420f |
| service_name                                                                                                      | neutron                          |
| service_type                                                                                                      | network                          |
| url                                                                                                               | http://controller:9696           |

FIGURE 6.5 – création d'une endpoint admin pour neutron

### 6.1.2 Configuration MariaDB pour Neutron

Nous avons utilisé les commandes suivantes pour ajouter un utilisateur et une base de données dans MariaDB pour Neutron :

```
~(keystone)# mysql
MariaDB [(none)]> create database neutron_ml2;
MariaDB [(none)]> grant all privileges on neutron_ml2.* to
neutron@'localhost'
identified by 'password';
MariaDB [(none)]> grant all privileges on neutron_ml2.* to neutron@'%'
identified by 'password';
MariaDB [(none)]> exit
```

```

root@controller ~(keystone)# mysql
Welcome to the MariaDB monitor. Commands end with ; or \g.
Your MariaDB connection id is 953
Server version: 10.11.8-MariaDB-0ubuntu0.24.04.1 Ubuntu 24.04

Copyright (c) 2000, 2018, Oracle, MariaDB Corporation Ab and others.

Type 'help;' or '\h' for help. Type '\c' to clear the current input statement.

MariaDB [(none)]> create database neutron_ml2;
Query OK, 1 row affected (0.028 sec)

MariaDB [(none)]> grant all privileges on neutron_ml2.* to neutron@'localhost' identified by 'password';
Query OK, 0 rows affected (0.052 sec)

MariaDB [(none)]> grant all privileges on neutron_ml2.* to neutron@'%' identified by 'password';
Query OK, 0 rows affected (0.004 sec)

MariaDB [(none)]> exit
Bye

```

FIGURE 6.6 – configuration MariaDB pour Neutron

## 6.2 Installation et configuration des services de Neutron

### 6.2.1 Installation des services de Neutron

Pour installer les services nécessaires à la configuration d'un environnement réseau avancé avec OpenStack, en particulier pour la gestion de la virtualisation du réseau avec OVN et Open vSwitch nous avons lancé la commande suivante :

```

~(keystone)# apt -y install neutron-server neutron-plugin-ml2
neutron-ovn-metadata-agent python3-neutronclient ovn-central ovn-host
openvswitch-switch

```

Cette commande installe plusieurs composants nécessaires à la configuration de Neutron, les éléments installés par cette commande :

- **neutron-server** : Le serveur principal du service Neutron, responsable de la gestion des configurations réseau dans OpenStack [4].
- **neutron-plugin-ml2** : Un plugin de gestion de réseaux pour Neutron qui permet d'intégrer différents types de réseaux [5].
- **neutron-ovn-metadata-agent** : Un agent permettant de gérer les métadonnées OVN (Open Virtual Network) dans Neutron, nécessaire pour l'intégration des services de réseau virtuel [2].
- **python3-neutronclient** : Le client en Python pour interagir avec Neutron via la ligne de commande [4].

- **ovn-central** : Le composant central d’OVN (Open Virtual Network), utilisé pour la gestion des réseaux virtuels [4].
- **ovn-host** : Composant OVN installé sur les hôtes pour gérer les réseaux virtuels et les connexions [4].
- **openvswitch-switch** : Le paquet pour Open vSwitch (OVS), un commutateur logiciel permettant de gérer les réseaux virtuels entre les machines physiques et virtuelles [1].

### 6.2.2 Configuration des services de Neutron

Pour configurer les services de neutron nous avons modifié les fichiers suivants :

```
— /etc/neutron/neutron.conf
Au niveau de ce fichier :
[DEFAULT]
bind_host = 127.0.0.1
bind_port = 9696
core_plugin = ml2
service_plugins = ovn-router
auth_strategy = keystone
state_path = /var/lib/neutron
allow_overlapping_ips = True
notify_nova_on_port_status_changes = True
notify_nova_on_port_data_changes = True
# RabbitMQ connection info
transport_url = rabbit://openstack:password@controller:5672

# Keystone auth info
[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_name = Default
user_domain_name = Default
project_name = service
username = neutron
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true
```

```

[database]
connection = mysql+pymysql://neutron:password@controller:3306/neutron_ml2

[nova]
auth_url = http://controller:5000
auth_type = password
project_domain_name = Default
user_domain_name = Default
region_name = RegionOne
project_name = service
username = nova
password = servicepassword
# if using self-signed certs on Apache2 Keystone, turn to [true]
insecure = true

[oslo_concurrency]
lock_path = $state_path/tmp

[oslo_policy]
enforce_new_defaults = true

— /etc/neutron/plugins/ml2/ml2_conf.ini
Au niveau du fichier /etc/neutron/plugins/ml2/ml2_conf.ini :
[DEFAULT]
debug = false

[ml2]
type_drivers = flat,geneve
tenant_network_types = geneve
mechanism_drivers = ovn
extension_drivers = port_security
overlay_ip_version = 4

[ml2_type_geneve]
vni_ranges = 1:65536
max_header_size = 38

[ml2_type_flat]
flat_networks = *

```

```

[securitygroup]
enable_security_group = True
firewall_driver = neutron.agent.linux.iptables_firewall
    .OVSHybridIptablesFirewallDriver

[ovn]
ovn_nb_connection = tcp:10.0.0.30:6641
ovn_sb_connection = tcp:10.0.0.30:6642
ovn_l3_scheduler = leastloaded
ovn_metadata_enabled = True

— /etc/neutron/neutron_ovn_metadata_agent.ini
[DEFAULT]

Nous avons ajouté ces lignes à partir de la 2ème ligne du fichier
nova_metadata_host = controller
nova_metadata_protocol = http

metadata_proxy_shared_secret = metadata_secret

Nous avons modifié la ligne 263
[ovs]
ovsdb_connection = tcp:127.0.0.1:6640

Nous avons ajouté ces lignes à la fin du fichier
[agent]
root_helper = sudo neutron-rootwrap /etc/neutron/rootwrap.conf

[ovn]
ovn_sb_connection = tcp:10.0.0.30:6642

— /etc/sysconfig/openvswitch
    Au niveau /etc/sysconfig/openvswitch :
    Nous avons dé commenter la ligne 8
    OVS_CTL_OPTS="--ovsdb-server-options='--remote=ptcp:6640:127.0.0.1'""

— Ainsi que /etc/nova/nova.conf

```

Nous avons ajouté ces deux lignes à la section [DEFAULT]

```
vif_plugging_is_fatal = True  
vif_plugging_timeout = 300
```

Nous avons ajouté à la fin du fichier ce qui suit :

```
[metadata_agent.ini]  
[neutron]  
auth_url = http://controller:5000  
auth_type = password  
project_domain_name = Default  
user_domain_name = Default  
region_name = RegionOne  
project_name = service  
username = neutron  
password = servicepassword  
service_metadata_proxy = True  
metadata_proxy_shared_secret = metadata_secret  
insecure = true
```

### 6.2.3 Configuration Nginx

Pour la configuration de nginx pour les paramètres de proxy nous avons ajouté dans la section stream dans le fichier /etc/nginx/nginx.conf ces deux lignes :

```
upstream neutron-api {  
    server 127.0.0.1:9696;  
}  
server {  
    listen 10.0.0.30:9696;  
    proxy_pass neutron-api;  
}
```

### 6.2.4 Démarrage des services de Neutron

Pour lancer les services de Neutron nous avons utilisé l'ensemble des commandes suivantes :

```

~(keystone)# systemctl restart openvswitch-switch

~(keystone)# ln -s /etc/neutron/plugins/ml2/ml2_conf.ini
/etc/neutron/plugin.ini

~(keystone)#su -s /bin/bash neutron -c "neutron-db-manage --config-file
/etc/neutron/neutron.conf --config-file /etc/neutron/plugin.ini upgrade head"
~(keystone)# systemctl restart ovn-central ovn-northd ovn-controller ovn-host

~(keystone)#ovn-nbctl set-connection ptcp:6641:10.0.0.30 --set connection
. inactivity_probe=60000

~(keystone)# ovn-sbctl set-connection ptcp:6642:10.0.0.30 {set connection
. inactivity_probe=60000

~(keystone)# ovs-vsctl set open . external-ids:ovn-remote=tcp:10.0.0.30:6642

~(keystone)# ovs-vsctl set open . external-ids:ovn-encap-type=geneve

~(keystone)# ovs-vsctl set open . external-ids:ovn-encap-ip=10.0.0.30

~(keystone)# ovs-vsctl set open . external-ids:ovn-cms-options=enable-chassis
-as-gw

~(keystone)# systemctl restart neutron-server neutron-ovn-metadata-agent
nova-api nova-compute nginx

```

Et à la fin, nous avons afficher la liste des agents de réseau configurés et gérés par le service Neutron dans OpenStack et leur état à l'aide de la commande :

```
~(keystone)# openstack network agent list
```

Le résultat de cette commande est illustré dans la figure suivante :

| ID                  | Agent Type         | Host       | Availability Zone | Alive | State | Binary                |
|---------------------|--------------------|------------|-------------------|-------|-------|-----------------------|
| cb46b860-ef73-5c4f- | OVN Metadata agent | controller |                   | (::)  | UP    | neutron-ovn-metadata- |
| 84c7-50686301e66f   |                    |            |                   |       | agent |                       |
| d341c324-e0d1-45b0- | OVN Controller     | controller |                   | (::)  | UP    | ovn-controller        |
| 8a39-a6887059d923   | Gateway agent      |            |                   |       |       |                       |

FIGURE 6.7 – Affichage de l'état des services Neutron

## 6.3 Configuration du réseau Neutron

### 6.3.1 Configuration des services Neutron

Un fichier de configuration système /etc/systemd/network/ens38.network est créé pour l’interface réseau utilisée par Neutron dans notre cas le nom de l’interface est ens38 avec le contenu suivant :

```
[Match]
Name=ens38
[Network]
LinkLocalAddressing=no
IPv6AcceptRA=no
```

```
root@controller ~ (keystone) # cat /etc/systemd/network/ens38.network
[Match]
Name=ens38

[Network]
LinkLocalAddressing=no
IPv6AcceptRA=no
```

FIGURE 6.8 – contenu du fichier /etc/systemd/network/ens38.network

Cette interface n’a pas d’adresse IP. Pour activer l’interface nous avons lancé la commande :

```
~(keystone) # ip link set ens38 up
```

Nous avons lancé la commande ip a pour afficher les interfaces réseaux :

```
7: ens38: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast master ovs-system state UP group default qlen 1000
    link/ether 00:0c:29:b7:2f:af brd ff:ff:ff:ff:ff:ff
    altname enp2s0
    inet6 fe80::3c6b:79cf:20ba:62a2/64 scope link noprefixroute
        valid_lft forever preferred_lft forever
```

FIGURE 6.9 – l’interface ens38 est active

Nous avons ensuite créé un pont, ajouté un port à ce pont et nous avons fait le mappage du réseau physique au pont virtuel via les commandes suivantes :

```

~(keystone)# ovs-vsctl add-br br-ens38
~(keystone)# ovs-vsctl add-port br-ens38 ens38
~(keystone)# ovs-vsctl set open . external-ids:ovn-bridge-mappings=physnet1
:br-ens38

```

### 6.3.2 Crédation d'un réseau virtuel

- Crédation d'un réseau virtuel nommé sharednet1 :

| Field                     | Value                                |
|---------------------------|--------------------------------------|
| admin_state_up            | UP                                   |
| availability_zone_hints   |                                      |
| availability_zones        |                                      |
| created_at                | 2024-12-15T16:59:58Z                 |
| description               |                                      |
| dns_domain                |                                      |
| id                        | 0cbafeea-efc7-47e0-b004-d13bcf3f8a5c |
| ipv4_address_scope        |                                      |
| ipv6_address_scope        |                                      |
| is_default                | False                                |
| is_vlan_transparent       |                                      |
| mtu                       | 1500                                 |
| name                      | sharednet1                           |
| port_security_enabled     | True                                 |
| project_id                | 7fa4c9352ca741338f0728b5a507a127     |
| provider:network_type     | flat                                 |
| provider:physical_network | physnet1                             |
| provider:segmentation_id  |                                      |
| qos_policy_id             |                                      |
| revision_number           | 1                                    |
| router:external           | Internal                             |
| segments                  |                                      |
| shared                    | True                                 |
| status                    | ACTIVE                               |

FIGURE 6.10 – Crédation d'un réseau virtuel

- Crédation de sous-réseau 10.0.0.0/24 pour le réseau sharednet1 :

```

imanezahra@controller:~ imanezahra@controller:~ 
root@controller ~(keystone)# openstack subnet create subnet1 --network sharednet1 \
--project $projectID --subnet-range 10.0.0.0/24 \
--allocation-pool start=10.0.0.200,end=10.0.0.254 \
--gateway 10.0.0.1 --dns-nameserver 10.0.0.10
+-----+-----+
| Field | Value |
+-----+-----+
allocation_pools	10.0.0.200-10.0.0.254
cidr	10.0.0.0/24
created_at	2024-12-15T17:00:37Z
description	
dns_nameservers	10.0.0.10
dns_publish_fixed_ip	None
enable_dhcp	True
gateway_ip	10.0.0.1
host_routes	
id	9c6388a2-281b-4ac0-8a85-5140d87e4983
ip_version	4
ipv6_address_mode	None
ipv6_ra_mode	None
name	subnet1
network_id	0cbafeea-efc7-47e0-b004-d13bcf3f8a5c
project_id	7fa4c9352ca741338f0728b5a507a127
revision_number	0
segment_id	None
service_types	
subnetpool_id	None
tags	
updated_at	2024-12-15T17:00:37Z
+-----+-----+

```

FIGURE 6.11 – Création de sous-réseau

- Vérification de la création du réseau et du sous-réseau:
- ```
~(keystone)# openstack network list
```

```

imanezahra@controller ~(keystone)$ openstack network list
+-----+-----+-----+
| ID | Name | Subnets |
+-----+-----+-----+
| 0cbafeea-efc7-47e0-b004-d13bcf3f8a5c | sharednet1 | 9c6388a2-281b-4ac0-8a85-5140d87e4983 |
+-----+-----+-----+

```

FIGURE 6.12 – Affichage de la liste des réseaux

```
~(keystone)# openstack subnet list
```

```

root@controller ~(keystone)# openstack subnet list
+-----+-----+-----+-----+
| ID | Name | Network | Subnet |
+-----+-----+-----+-----+
| 9c6388a2-281b-4ac0-8a85-5140d87e4983 | subnet1 | 0cbafeea-efc7-47e0-b004-d13bcf3f8a5c | 10.0.0.0/24 |
+-----+-----+-----+-----+

```

FIGURE 6.13 – Affichage de la liste des sous réseaux

# Chapitre 7

## Ajout d'utilisateurs Openstack et ajout des Flavors

Dans une infrastructure cloud basée sur OpenStack, la gestion des utilisateurs et des flavors est essentielle pour garantir une organisation et une utilisation optimale des ressources. Les utilisateurs représentent les entités (personnes ou applications) ayant accès au système, tandis que les flavors définissent les configurations matérielles standardisées pour les machines virtuelles, incluant les ressources telles que le processeur, la mémoire et le stockage. Nous avons ajouté des comptes d'utilisateurs dans Keystone qui peuvent utiliser le système Openstack en suivant ces étapes :

- Ajout d'un Projet via la commande :

```
~(keystone)# openstack project create --domain default --description "Hiroshima Project" hiroshima
```

Field	Value
description	Hiroshima Project
domain_id	default
enabled	True
id	4ff544a7d91c4c19b7c660c58b6a5f07
is_domain	False
name	hiroshima
options	{}
parent_id	default
tags	[]

FIGURE 7.1 – Ajout d'un Projet

- Ajout d'un utilisateur en utilisant la commande :

```
~(keystone)# openstack user create --domain default --project hiroshima  
--password userpassword serverworld
```

```
root@controller -(keystone)# openstack user create --domain default --project hiroshima --password userpassword serverworld
+-----+-----+
| Field | Value |
+-----+-----+
| default_project_id | 4ff544a7d91c4c19b7c660c58b6a5f07 |
| domain_id | default |
| enabled | True |
| id | cc67f7fc0da142da82beebfa9928de34 |
| name | serverworld |
| options | {} |
| password_expires_at | None |
+-----+-----+
```

FIGURE 7.2 – Ajout d'un utilisateur

- Affichage de la liste des rôles disponibles dans le système par la commande :  
~(keystone)# openstack role list

```
root@controller -(keystone)# openstack role list
+-----+-----+
| ID | Name |
+-----+-----+
| 2981a03efc46477a89f5d56d96afbdec | member |
| 62b694bfa75343a79e6afa5a7f2a612a | admin |
| 65f98c9bdd994bb3b2c8695de8881ead | service |
| 6c76a8c016764d629578d29dad52fd8 | manager |
| 9aabe111c32b4f2f952d6590b6e8a1e2 | reader |
+-----+-----+
```

FIGURE 7.3 – Affichage de la liste des rôles

- Ajout d'un utilisateur au rôle member dans le projet hiroshima :  
~(keystone)# openstack role add --project hiroshima --user serverworld member

Ensuite nous avons configuré les variables d'environnement pour OpenStack pour un utilisateur du système d'exploitation afin qu'il puisse utiliser le système OpenStack .Les variables d'environnement nécessaires sont définies dans le fichier /keystonerc

```

imanezahra@controller:~          imanezahra@controller:~
GNU nano 7.2                      /root/keystonerc

export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=adminpassword
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
export PS1='\u@\h \W(keystone)\$ '

```

FIGURE 7.4 – contenu du fichier /keystonerc

Nous avons changé les permissions du fichier /keystonerc à ce que seul le propriétaire peut lire et écrire le fichier. Les autres utilisateurs n'ont aucun accès :

```
~$ chmod 600 ~/keystonerc
```

Pour charger et activer les variables d'environnement définies dans le fichier nous avons lancé la commande :

```
~$ source ~/keystonerc
```

Et pour que ces variables soient automatiquement chargées à chaque ouverture de session, nous avons ajouté la commande

```
"source ~/keystonerc "
```

au fichier

```
~/.bash_profile
```

```
~(keystone)$ echo "source ~/keystonerc " >> ~/.bash_profile
```

Nous avons utilisé la commande suivante pour afficher la liste des projets disponibles dans OpenStack :

```
~(keystone)]$ openstack project list
```

```
imanezahra@controller ~(keystone)$ openstack project list
+-----+-----+
| ID      | Name   |
+-----+-----+
| 4ff544a7d91c4c19b7c660c58b6a5f07 | hiroshima |
+-----+-----+
```

FIGURE 7.5 – Affichage de la liste des projets

- Ajout des flavors :

```
~(keystone)# openstack flavor create --id 6 --vcpus 1 --ram 256 --disk 1 m1.iccn
~(keystone)# openstack flavor create --id 2 --vcpus 2 --ram 4096 --disk 10 m1.small
~(keystone)# openstack flavor create --id 3 --vcpus 4 --ram 8192 --disk 10 m1.medium
~(keystone)# openstack flavor create --id 4 --vcpus 8 --ram 16384 --disk 10 m1.large
~(keystone)# openstack flavor create --id 5 --vcpus 4 --ram 8192 --disk 10 --ephemeral 10 m2.medium
```

```
root@controller ~(keystone)# openstack flavor create --id 6 --vcpus 1 --ram 256 --disk 1 m1.iccn
+-----+-----+
| Field      | Value   |
+-----+-----+
| OS-FLV-DISABLED:disabled | False |
| OS-FLV-EXT-DATA:ephemeral | 0    |
| description      | None  |
| disk            | 1    |
| id              | 6    |
| name            | m1.iccn |
| os-flavor-access:is_public | True |
| properties      |      |
| ram             | 256   |
| rxtx_factor     | 1.0   |
| swap            | 0    |
| vcpus           | 1    |
+-----+-----+
```

FIGURE 7.6 – Création de flavor m1.iccn

- Affichage des flavors créés

```
~(keystone)# openstack flavor list
```

```

imanezahra@controller:~$ cd
imanezahra@controller:~$ nano ~/kestonerc
imanezahra@controller:~$ chmod 600 ~/kestonerc
imanezahra@controller:~$ source ~/kestonerc
imanezahra@controller ~(keystone)$ echo "source ~/kestonerc" >> ~/.bash_profile
imanezahra@controller ~(keystone)$ openstack project list
+-----+-----+
| ID | Name |
+-----+-----+
| 4ff544a7d91c4c19b7c660c58b6a5f07 | hiroshima |
+-----+-----+
imanezahra@controller ~(keystone)$ openstack flavor create --id 1 --vcpus 1 --ram 2048 --disk 10 m1.tiny
ForbiddenException: 403: Client Error for url: http://controller:8774/v2.1/4ff544a7d91c4c19b7c660c58b6a5f07/flavors, Policy doesn't allow os_compute_api:os-flavor-manage:create to be performed.
imanezahra@controller ~(keystone)$ openstack flavor list
+-----+-----+-----+-----+-----+
| ID | Name | RAM | Disk | Ephemeral | VCPUs | Is Public |
+-----+-----+-----+-----+-----+
| 1 | m1.tiny | 2048 | 10 | 0 | 1 | True |
| 2 | m1.small | 4096 | 10 | 0 | 2 | True |
| 3 | m1.medium | 8192 | 10 | 0 | 4 | True |
| 4 | m1.large | 16384 | 10 | 0 | 8 | True |
| 5 | m2.medium | 8192 | 10 | 10 | 4 | True |
| 6 | m1.iccn | 256 | 1 | 0 | 1 | True |
+-----+-----+-----+-----+-----+

```

FIGURE 7.8 – Echec de la création d'un flavor par un utilisateur normal

```

imanezahra@controller ~(keystone)$ openstack flavor list
+-----+-----+-----+-----+-----+
| ID | Name | RAM | Disk | Ephemeral | VCPUs | Is Public |
+-----+-----+-----+-----+-----+
| 1 | m1.tiny | 2048 | 10 | 0 | 1 | True |
| 2 | m1.small | 4096 | 10 | 0 | 2 | True |
| 3 | m1.medium | 8192 | 10 | 0 | 4 | True |
| 4 | m1.large | 16384 | 10 | 0 | 8 | True |
| 5 | m2.medium | 8192 | 10 | 10 | 4 | True |
| 6 | m1.iccn | 256 | 1 | 0 | 1 | True |
+-----+-----+-----+-----+-----+

```

FIGURE 7.7 – liste des flavors

Après avoir créer les flavors en tant que l'utilisateur root, nous avons ouvert un nouveau terminal et nous sommes connectés en tant qu'utilisateur normal et nous avons essayé de créer un flavor . L'objectif était de tester si un utilisateur sans droits administratifs pouvait également créer une flavor.

Une erreur a été affichée, cela est dû au fait que la création de flavors est une opération restreinte nécessitant des priviléges administratifs.

# **Chapitre 8**

## **Création et Démarrage des instances de machine virtuelle**

Dans un environnement cloud, les instances de machines virtuelles (VMs) représentent des serveurs virtuels qui permettent aux utilisateurs de déployer et de gérer leurs applications sans avoir besoin d'infrastructure matérielle physique. Dans cette partie nous allons expliquer les étapes de création et de démarrage des instances dans OpenStack depuis la configuration des ressources (images, réseaux, et clés SSH) jusqu'au lancement des instances.

### **8.1 Préparation de l'environnement**

Avant de créer et démarrer une instance de machine virtuelle, nous avons nous connecté en tant qu'utilisateur que nous avons défini dans les variables d'environnement pour Openstack au niveau du fichier /keystonerc dans la partie précédente.

#### **8.1.1 Configuration du groupe de sécurité**

Ensuite nous avons créer un groupe de sécurité pour les instances en utilisant les commandes suivantes :

```
~(keystone)$ openstack security group create secgroup01
```

imanezahra@controller -(keystone)\$ openstack security group create secgroup01	
Field	Value
created_at	2024-12-15T17:33:00Z
description	secgroup01
id	e4fd582b-6b8a-4d91-8b70-792e745a9539
name	secgroup01
project_id	4ff544a7d91c4c19b7c660c58b6a5f07
revision_number	1
rules	created_at='2024-12-15T17:33:00Z', direction='egress', ethertype='IPv6', id='6454e669-8483-4aa4-b0a4-b6d9a750861f', standard_attr_id='20', updated_at='2024-12-15T17:33:00Z' created_at='2024-12-15T17:33:00Z', direction='egress', ethertype='IPv4', id='9dc6f8c-9d38-4b73-a0bc-eaac19c4ac4', standard_attr_id='21', updated_at='2024-12-15T17:33:00Z'
shared	False
stateful	True
tags	[]
updated_at	2024-12-15T17:33:00Z

FIGURE 8.1 – création d'un groupe de sécurité pour les instances

```
~(keystone)$ openstack security group list
```

imanezahra@controller -(keystone)\$ openstack security group list				
ID	Name	Description	Project	Tags
65fd9e79-afc9-4397-95ac-3d03bb75cbcd	default	Default security group	4ff544a7d91c4c19b7c660c58b6a5f07	[]
e4fd582b-6b8a-4d91-8b70-792e745a9539	secgroup01	secgroup01	4ff544a7d91c4c19b7c660c58b6a5f07	[]

FIGURE 8.2 – Affichage des groupes de sécurité disponibles

### 8.1.2 Configuration des clés SSH

Pour réaliser cette partie nous avons executé les commades suivantes :

- Création d'une paire de clés SSH pour se connecter aux instances

```
~(keystone)$ ssh-keygen -q -N ""
```

- Ajout d'une clé publique :

```
~(keystone)$ openstack keypair create {public-key / .ssh/id_rsa.pub mykey
```

- Affichage de la liste des paires de clés :

```
~(keystone)$ openstack keypair list
```

Les trois commandes sont illustrées dans la figure suivante :

```

imanezahra@controller ~(keystone)$ ssh-keygen -q -N ""
Enter file in which to save the key (/home/imanezahra/.ssh/id_ed25519):
imanezahra@controller ~(keystone)$ openstack keypair create --public-key ~/.ssh/id_ed25519.pub mykey
+-----+-----+
| Field | Value |
+-----+-----+
| created_at | None |
| fingerprint | 24:83:71:73:15:cc:26:b4:db:5a:53:66:5d:2c:61:37 |
| id | mykey |
| is_deleted | None |
| name | mykey |
| type | ssh |
| user_id | cc67f7fc0da142da82beebfa9928de34 |
+-----+-----+
imanezahra@controller ~(keystone)$ openstack keypair list
+-----+-----+-----+
| Name | Fingerprint | Type |
+-----+-----+-----+
| mykey | 24:83:71:73:15:cc:26:b4:db:5a:53:66:5d:2c:61:37 | ssh |
+-----+-----+-----+

```

FIGURE 8.3 – Création d'une paire de clés SSH pour se connecter aux instances et ajout d'une clé publique

## 8.2 Création et démarrage des instances

### 8.2.1 Configuration du réseau

–Récupération de l'ID du réseau nommé sharednet1 dans OpenStack et Attribution de cet ID à une variable nommée netID :  
`~(keystone)\$ netID=\$(openstack network list | grep sharednet1 | awk '{ print \$2 }')`

### 8.2.2 Lancement des instances

Création et démarrage de deux instances :

lmanezahra@controller -(keystone)\$ openstack server create --flavor m1.iccn --image cirros --security-group secgroup01 - -nic net-id=\$netID --key-name mykey cirros	
Field	Value
OS-DCF:diskConfig	MANUAL
OS-EXT-AZ:availability_zone	NOSTATE
OS-EXT-STS:power_state	scheduling
OS-EXT-STS:task_state	building
OS-EXT-STS:vn_state	None
OS-SRV-USG:launched_at	None
OS-SRV-USG:terminated_at	None
accessIPv4	
accessIPv6	
addresses	
adminPass	RtEyssz88byr
config_drive	
created	2024-12-15T17:38:29Z
flavor	m1.iccn (6)
hostId	
id	391a8ccb-22b4-4f59-ae75-22409fd42266
image	cirros (4dcbf12d-ac06-464a-acd7-04d34942807f)
key_name	mykey
name	cirros
os-extended-volumes:volumes_attached	[]
progress	0
project_id	4ff544a7d91c4c19b7c660c58b6a5f07
properties	
security_groups	name='e4fd582b-6b8a-4d91-8b70-792e745a9539'
status	BUILD

FIGURE 8.4 – Création de la première instance nommée cirros

lmanezahra@controller -(keystone)\$ openstack server create --flavor m1.iccn --image cirros --security-group secgroup01 - -nic net-id=\$netID --key-name mykey cirros062	
Field	Value
OS-DCF:diskConfig	MANUAL
OS-EXT-AZ:availability_zone	NOSTATE
OS-EXT-STS:power_state	scheduling
OS-EXT-STS:task_state	building
OS-SRV-USG:launched_at	None
OS-SRV-USG:terminated_at	None
accessIPv4	
accessIPv6	
addresses	
adminPass	J2gkAz8Yfmxp
config_drive	
created	2024-12-15T18:35:47Z
flavor	m1.iccn (6)
hostId	
id	cdbd8dfc-e493-4212-9551-7482f6c67530
image	cirros (4dcbf12d-ac06-464a-acd7-04d34942807f)
key_name	mykey
name	cirros062
os-extended-volumes:volumes_attached	[]
progress	0
project_id	4ff544a7d91c4c19b7c660c58b6a5f07
properties	
security_groups	name='e4fd582b-6b8a-4d91-8b70-792e745a9539'

FIGURE 8.5 – Création de la première instance nommée cirros062

-Affichage de l'état des instances créées :  
~(keystone)\$ openstack server list

openstack server list						
ID	Name	Status	Networks	Image	Flavor	
cdbd8dfc-e493-4212-9551-7482f6c67530	cirros062	ACTIVE	sharednet1=10.0.0.233	cirros	m1.iccn	
3b3f8254-31b1-4cf7-8741-fa1ef45484b0	cirros	ACTIVE	sharednet1=10.0.0.201	cirros	m1.iccn	

FIGURE 8.6 – État des instances créées

### 8.2.3 Configuration des règles de sécurité pour SSH et ICMP

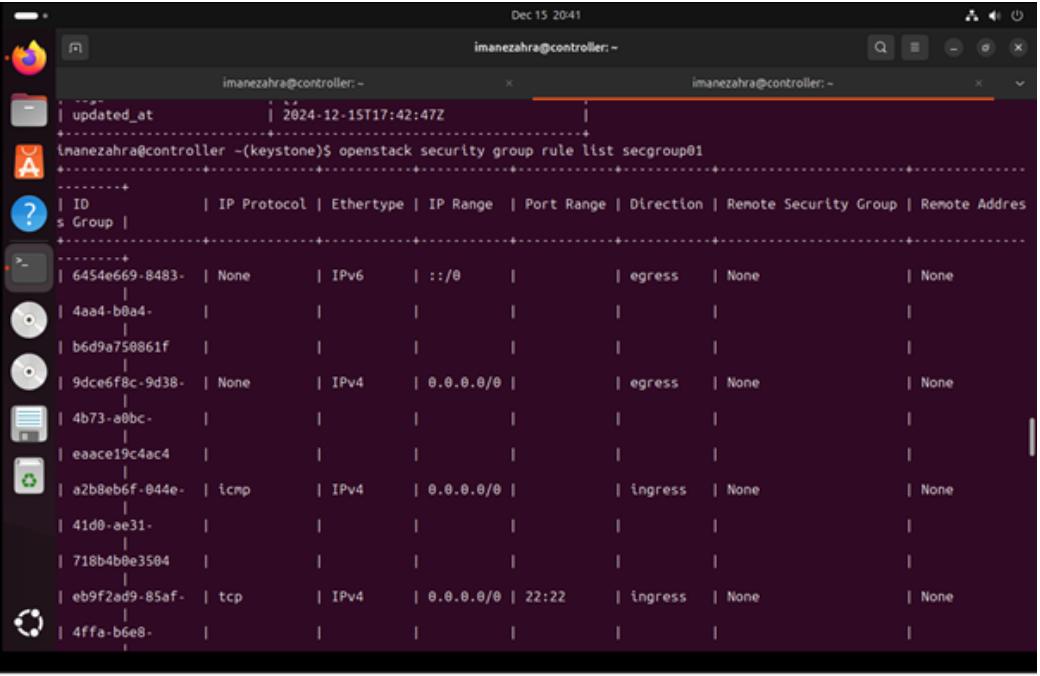
-Configuration des paramètres de sécurité pour le groupe de sécurité que nous avons déjà créé pour accéder à SSH et ICMP :

openstack security group rule create --protocol icmp --ingress secgroup01	
Field	Value
belongs_to_default_sg	False
created_at	2024-12-15T17:42:28Z
description	
direction	ingress
ether_type	IPv4
id	a2b8eb6f-044e-41d0-ae31-718b4b0e3504
name	None
normalized_cidr	0.0.0.0/0
port_range_max	None
port_range_min	None
project_id	4ff544a7d91c4c19b7c660c58b6a5f07
protocol	icmp
remote_address_group_id	None
remote_group_id	None
remote_ip_prefix	0.0.0.0/0
revision_number	0
security_group_id	e4fd582b-6b8a-4d91-8b70-792e745a9539
tags	[]
updated_at	2024-12-15T17:42:28Z

FIGURE 8.7 – Configuration des paramètres de sécurité pour le groupe de sécurité pour accéder à ICMP

imanezahra@controller -(keystone)\$ openstack security group rule create --protocol tcp --dst-port 22:22 secgroup01	
Field	Value
belongs_to_default_sg	False
created_at	2024-12-15T17:42:47Z
description	
direction	ingress
ether_type	IPv4
id	eb9f2ad9-85af-4ffa-b6e8-429a7b9ff532
name	None
normalized_cidr	0.0.0.0/0
port_range_max	22
port_range_min	22
project_id	4ff544a7d91c4c19b7c660c58b6a5f07
protocol	tcp
remote_address_group_id	None
remote_group_id	None
remote_ip_prefix	0.0.0.0/0
revision_number	0
security_group_id	e4fd582b-6b8a-4d91-8b70-792e745a9539
tags	[]
updated_at	2024-12-15T17:42:47Z

FIGURE 8.8 – Configuration des paramètres de sécurité pour le groupe de sécurité pour accéder à SSH



ID	IP Protocol	Ethertype	IP Range	Port Range	Direction	Remote Security Group	Remote Address Group
6454e669-8483-4aa4-b0a4-48d9a758861f	None	IPv6	::/0		egress	None	None
9dce6f8c-9d38-4b73-a0bc-eaac19c4ac4	None	IPv4	0.0.0.0/0		egress	None	None
a2b8eb6f-044e-41d0-ae31-718b4b0e3504	icmp	IPv4	0.0.0.0/0		ingress	None	None
eb9f2ad9-85af-4ffa-b6e8-429a7b9ff532	tcp	IPv4	0.0.0.0/0	22:22	ingress	None	None
4ffa-b6e8-429a7b9ff532							

FIGURE 8.9 – Affichage des règles de sécurité associées au groupe de sécurité secgroup01

## 8.3 Accès aux instances

### 8.3.1 Accès via la Console VNC

Finalement pour accéder à la console graphique de nos instances dans OpenStack, nous avons utilisé la commande suivante :

```
imanezahra@controller -(keystone)$ openstack console url show cirros
+-----+
| Field | Value
+-----+
| protocol | vnc
| type | novnc
| url | http://controller:6080/vnc_auto.html?path=%3Ftoken%3D2c421adf-9f12-411d-a74c-8f2a63debc70
+-----+
imanezahra@controller -(keystone)$ openstack console url show cirros062
+-----+
| Field | Value
+-----+
| protocol | vnc
| type | novnc
| url | http://controller:6080/vnc_auto.html?path=%3Ftoken%3D458d5920-e9d4-4b42-a1b4-184f24149eb7
+-----+
imanezahra@controller -(keystone)$
```

FIGURE 8.10 – les URL de la console VNC des deux instances

Cette commande permet d'afficher l'URL de la console graphique de type VNC pour chaque instance. Nous avons copié et collé L'URL obtenue dans la barre d'adresse du navigateur et nous avons été redirigé vers la console VNC où nous avons pu nous connecter à l'aide du nom d'utilisateur 'cirros' et du mot de passe par défaut.

```
login as 'cirros' user. default password: 'gocubsgo'. use 'sudo' for root.  
cirros062 login: cirros  
Password:
```

FIGURE 8.12 – connexion avec le nom d’utilisateur et le mot de passe configurés par défaut



FIGURE 8.11 – interface de connexion VNC

### 8.3.2 Tests de connectivité

Une fois connecté, nous avons affiché les addresses ip au niveau de chaque instance pour ensuite tester le ping entre les deux instances.

```

Dec 15 20:26
Ubuntu 24.04 : OpenStack X QEMU (instance-000000) X QEMU (instance-000000) X Get Images — Virtual Ma ...
controller:6080/vnc_auto.html?path=%3Ftoken%3D45@d5920-e9d4-4b42-a1b4-184f24149eb7

File: /var/log/cirros-0.6.2/blk.log
[ 0.000000] x86_64: Checked W-X mappings: passed, no W-X pages found.
[ 0.641452] x86_64: Checking user space page tables
[ 0.895183] x86_64: Checked W-X mappings: passed, no W-X pages found.
[ 0.895493] km: /init as init process

[ 0.923500] further output written to </dev/ttys00
[ 0.923500] 9.179669] virtio_blk virtio2: (total 2097152 512-byte logical blocks (1.07 G
[ 0.923500] 9.220067] GPT:Primary header thinks alt. header is not at the end of the di
sk.
[ 0.923500] 9.243766] GPT:229375 t= 2097151
[ 0.923500] 9.255034] GPT:Alternate GPT header not at the end of the disk.
[ 0.923500] 9.273436] GPT:229375 t= 2097151
[ 0.923500] 9.284305] GPT: Use GNU Parted to correct GPT errors.
[ 0.923500] 9.678632] virtio_gpu virtio6: (drm) drm_plane_enable_fb_damage_clips() not called

Login as 'cirros' user, default password: 'goodshgo'. Use 'sudo' for root.
cirros062 login: cirros
Password:
$ ip add
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 brd 127.0.0.1 scope host lo
        valid_lft forever preferred_lft forever
    inet6 ::1/128 brd :: scope host
        valid_lft forever preferred_lft forever
2: eth0: <NO-CARRIER,BROADCAST,MULTICAST,UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:24:b0:70 brd ff:ff:ff:ff:ff:ff
    inet 10.0.0.233/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
        valid_lft 40677sec preferred_lft 35277sec
    inet6 fe80::fb16:3eff:fe41:9cf8/64 scope link
        valid_lft forever preferred_lft forever
$ ping 10.0.0.201
PING 10.0.0.201 (10.0.0.201) 56(84) bytes of data.
64 bytes from 10.0.0.201: icmp_seq=1 ttl=64 time=0.00 ms
64 bytes from 10.0.0.201: icmp_seq=2 ttl=64 time=27.2 ms
64 bytes from 10.0.0.201: icmp_seq=3 ttl=64 time=4.10 ms
--- 10.0.0.201 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2006ms
rtt min/avg/max/mdev = 4.096/13.107/27.221/10.106 ms

```

FIGURE 8.13 – ping de l’instance cirros depuis cirros062

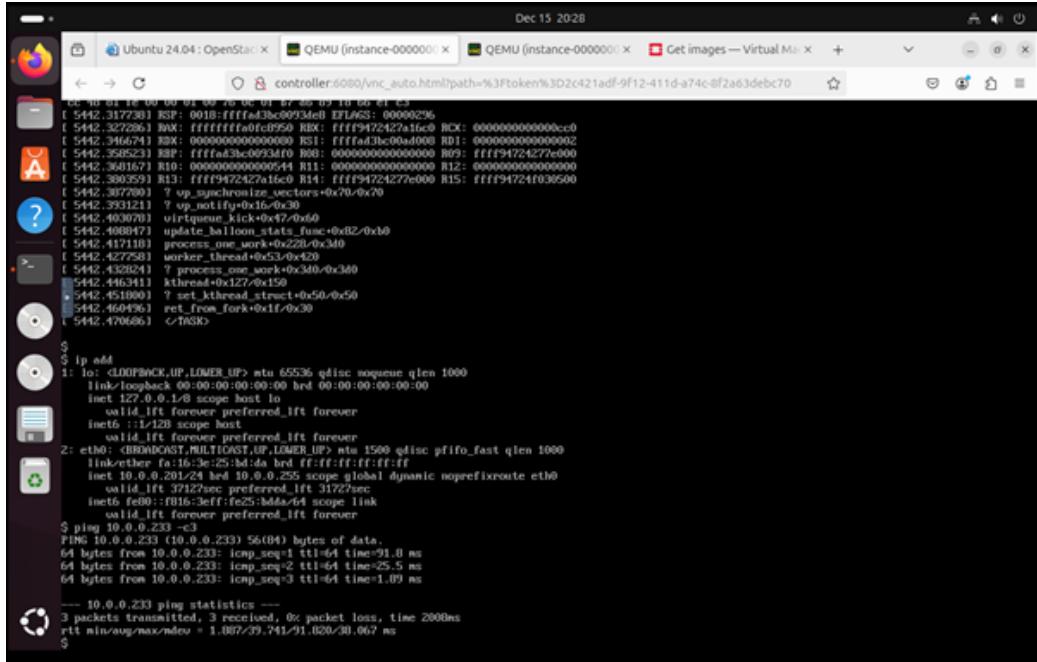


FIGURE 8.14 – ping de l’instance cirros062 à depuis cirros

Nous avons aussi tester la connectivité par SSH entre les deux instances,et nous avons pu se connecter d’une instance à l’autre .La figure suivante représente la connexion SSH à cirros062 depuis l’autre instance cirros.

```

$ ssh -l root cirros062
root@cirros062:~# ping 10.0.0.233 -c3
PING 10.0.0.233 (10.0.0.233) 56(84) bytes of data.
64 bytes from 10.0.0.233: icmp_seq=1 ttl=64 time=91.8 ms
64 bytes from 10.0.0.233: icmp_seq=2 ttl=64 time=25.5 ms
64 bytes from 10.0.0.233: icmp_seq=3 ttl=64 time=1.09 ms
--- 10.0.0.233 ping statistics ---
3 packets transmitted, 3 received, 0% packet loss, time 2000ms
rtt min/avg/max/mdev = 1.087/39.741/91.820/38.067 ms
$ ssh cirros@10.0.0.233
Host '10.0.0.233' is not in the trusted hosts file.
(sshd-ed25519 fingerprint sha1!! 63:8f:09:bb:45:f3:13:61:16:1e:72:83:23:33:5f:25:0a:ba:75:34)
Do you want to continue connecting? (y/n) y
cirros@10.0.0.233's password:
$ whoami
cirros
$
```

FIGURE 8.15 – Connexion SSH à l’instance cirros062 depuis l’instance cirros

```
$ ssh cirros@10.0.0.201
Host '10.0.0.201' is not in the trusted hosts file.
(sshd-25519 fingerprint sha1!! 56:c6:73:8f:72:96:47:65:ab:79:93:e5:a9:7b:8f:7b:92:25:dd:74)
Do you want to continue connecting? (y/n) y
cirros@10.0.0.201's password:
$ 
$ ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
        inet 127.0.0.1/8 scope host lo
            valid_lft forever preferred_lft forever
        inet6 ::1/128 scope host
            valid_lft forever preferred_lft forever
2: eth0: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc pfifo_fast qlen 1000
    link/ether fa:16:3e:25:b0:da brd ff:ff:ff:ff:ff:ff
        inet 10.0.0.201/24 brd 10.0.0.255 scope global dynamic noprefixroute eth0
            valid_lft 32921sec preferred_lft 27521sec
        inet6 fe80::f816:3eff:fe25:b0da/64 scope link
            valid_lft forever preferred_lft forever
$
```

FIGURE 8.16 – Connexion SSH à l’instance cirros depuis l’instance nommée cirros062

# Conclusion

Ce projet a montré la puissance d'OpenStack en tant que solution d'infrastructure Cloud. L'intégration des différents services tels que Nova pour la gestion des instances de calcul, Keystone pour l'authentification, MariaDB pour la gestion des bases de données, et Nginx pour le proxy, nous avons mis en évidence la complexité et l'interdépendance des services nécessaires au bon fonctionnement d'une infrastructure cloud.

D'après ce que nous avons observé, la qualité et l'agrément d'OpenStack dépendent d'un bon nombre de facteurs à savoir : une bonne compréhension des concepts sous-jacents, une configuration minutieuse des services, et une gestion rigoureuse des ressources. Chaque étape de déploiement, de la configuration initiale à l'activation des services, a mis en lumière les défis techniques, mais aussi les avantages d'une infrastructure cloud bien conçue.

Ainsi, ce projet constitue une excellente arborescence pour les travaux futurs à venir et nous avons appris également des recommandations pertinentes concernant la prise de snapshots au fur et à mesure qu'on avance dans un projet, la configuration ne se fait sur un fichier de configuration qu'après avoir une copie de ce fichier, la vérification de l'état des services installés, etc.

# Bibliographie

- [1] Chapter 2. working with ml2/ovn. [https://docs.redhat.com/en/documentation/red\\_hat\\_openstack\\_platform/17.1/html/configuring\\_red\\_hat\\_openstack\\_platform\\_networking/assembly-work-with-ovn\\_rhosp-network](https://docs.redhat.com/en/documentation/red_hat_openstack_platform/17.1/html/configuring_red_hat_openstack_platform_networking/assembly-work-with-ovn_rhosp-network).
- [2] Red hat openstack platform 17.1 configuring red hat openstack platform networking managing the openstack networking service (neutron). [https://docs.redhat.com/en-us/documentation/red\\_hat\\_openstack\\_platform/17.1/pdf/configuring\\_red\\_hat\\_openstack\\_platform\\_networking/Red\\_Hat\\_OpenStack\\_Platform-17.1-Configuring\\_Red\\_Hat\\_OpenStack\\_Platform\\_networking-en-US.pdf](https://docs.redhat.com/en-us/documentation/red_hat_openstack_platform/17.1/pdf/configuring_red_hat_openstack_platform_networking/Red_Hat_OpenStack_Platform-17.1-Configuring_Red_Hat_OpenStack_Platform_networking-en-US.pdf), 2024.
- [3] Ubuntu 24.04 : Openstack. [https://www.server-world.info/en/note?os=Ubuntu\\_24.04&p=openstack\\_caracal&f=1](https://www.server-world.info/en/note?os=Ubuntu_24.04&p=openstack_caracal&f=1), 2024.
- [4] OpenStack Documentation. Openstack docs. [https://docs.openstack.org/neutron/latest/install/ovn/manual\\_install.html](https://docs.openstack.org/neutron/latest/install/ovn/manual_install.html).
- [5] OpenStack Documentation. Openstack docs. [https://docs.openstack.org/neutron/latest/admin/config\\_ml2.html](https://docs.openstack.org/neutron/latest/admin/config_ml2.html).
- [6] University of Cambridge. Computing service : Openstack nova. <https://docs.hpc.cam.ac.uk/cloud/userguide/03-nova.html>, 2018.
- [7] University of Cambridge. Networking service : Openstack neutron. <https://docs.hpc.cam.ac.uk/cloud/userguide/02-neutron.html>, 2018.