République **A**lgérienne **D**émocratique et **P**opulaire

Ministère de l'**E**nseignement **S**upérieur et de la **R**echerche **S**cientifique

UNIVERSITE **M**OULOUD **M**AMMERI DE **T**IZI-**O**UZOU

FACULTE DU GENIE ELECTRIQUE ET D'INFORMATIQUE
DEPARTEMENT D'INFORMATIQUE

Mémoire de Fin d'Etudes

De MASTER ACADEMIQUE

Domaine : Mathématiques et Informatique
Filière : Informatique

Spécialité : systèmes informatiques

Présenté par

MEHDI Fariza

OUANDJELI Nawal

Thème

Techniques de tolérance aux pannes dans le Cloud Computing

Mémoire soutenu publiquement le 18/09/2019 devant le jury composé de:

Président : Mr DAOUI Mehammed
Encadreur : Mme AOUDJIT Rachida
Examinateur : Mme BELKADI Malika
Examinateur: Mme BELATTAF Samia

Remerciements

Nous tenons tout d’abord à remercier Dieu le tout puissant et miséricordieux, qui nous a donné la force et la patience d’accomplir ce modeste travail.

Nous tenons à exprimer nos plus grands remerciements à nos très chers parents pour leur soutien moral et leurs encouragements.

Mme AOUDJIT pour son encadrement, Nous la remercions pour l’orientation et l’aide qui ont constitué un apport considérable sans lequel ce travail n’aurait pas pu aboutir.

Nous tenons à exprimer nos sincères remerciements à tous les enseignants qui nous ont transmis leur savoir et qui par leurs compétences nous ont soutenu dans la poursuite de nos études.

Nous tenons à remercier les membres du jury qui vont évaluer cet humble travail.

Enfin, nous remercions toutes les personnes qui ont participé de près ou de loin à la réalisation de ce travail.

Dédicace

Je dédie ce modeste travail
À mes parents pour tous leurs sacrifices, leur tendresse, leur soutien et leurs prières tout au long de mes études. Que dieu leur procure bonne santé et longue vie.

À ma sœur Linda et son mari **Meziene** *et sa fille* **Ilina,**

À mes sœurs **Sarah, Lilia** *et mon petit frères* **Omar-Amine,**

À mon ami **Mohand,**

À ceux qui nous ont quittés très tôt,
Qui sont toujours dans mon esprit et dans mon cœur,
Que Dieu, le miséricordieux, vous accueille dans son éternel paradis.

À ma chère binôme **Farizaqui** *m'a aidé et m'a supporté dans les moments difficiles tout au long de mes années d'études à l'UMMTO,*

Je la remercie pour sa patience infinie et je lui souhaite plus de succès,
À tous ceux que j'aime,
Merci.

Nawal

Dédicace

Je dédie ce modeste travail et ma profonde gratitude

À ma mère, à qui je dois la réussite, pour l'éducation qu'elle m'a prodigué, avec tous les moyens et au prix de tous les sacrifices qu'elle a consentis à mon égard, pour le sens du devoir qu'elle m'a enseigné depuis mon enfance.

À l'âme de mon père et de mes grands-pères

À mon cher frère et ma très chère sœur

À mes deux oncles, merci pour vos précieux conseils et pour votre soutien tout au long de mon cursus scolaire.

À mes tentes, mes cousins et mes grands-mères.

À ma binôme **Nawal** *avec qui j'ai partagé cette expérience et cette partie de ma vie, je lui souhaite plus de succès.*

À tous mes amis et être chers spécialement **Minou** *et tous ceux qui ont contribué à cette réussite.*

Je vous remercie.

Faiza

Table des matières

Liste des figures :

Liste des acronymes :

API : application programming interface.

CISCO : Commercial &Industrial Security Corporation.

BaaS : Backend as a Service.

CLI : Command-line interface.

CPU : Central Processing Unit.

DaaS : Desktop as a Service ou Data as a service.

DHCP : Dynamic Host Configuration Protocol.

DNS : Domain Name System.

GID : Group ID.

HTTP : Hypertext Transfer Protocol

IaaS : Interface as a Service.

ID : Identifier.

IP : Interface Protocol.

IT : Information Technologie.

KVM :Kernel-based Virtual Machine.

LUN : Logiciel Unit Number.

LTS : Long-Term Support.

MTR : Matt'sTRaceroute.

MQ : Message Queuing .

NAS : Network Attached Storage.

NAT : Network Address Translation.

NFS : Network File system.

NIST : National Institut of Standard and Technologie.

NTP : Network Time Protocole.

OS : Operating System.

OVS :Open Virtual Switch.

PAAS : Plateform as a Service.

QoS : Qualité de Service.

RAM : Random Access Memory.

SaaS : Software as a Service.

SCP : Secure Copy.

SSH : Secure Shell.

TCP : Transport Control Protocol.

TLS : Transport Layer Security .

UID : User Identifier.

VLAN : Virtual Local Network.

VM : Virtual Machine.

VNC : Virtual Network Computing.

VPN : Virtual Private Network.

XML : Extensible Markup Language

Introduction générale :

L'année 2008 a connu l'émergence d'un nouveau terme informatique « **Cloud Computing** » ou « l'informatique des nuages » ou encore « l'infonuagique ». Ce nouveau concept est apparu pour certains comme une révolution et pour d'autres comme un simple terme de marketing qui ne fait que rassembler des services et des technologies qui existent depuis longtemps[5].

Le principe de base du Cloud s'appuie sur le concept de virtualisation. Ce dernier c'est un ensemble de techniques logiciels qui permettent d'organiser les infrastructures informatiques de manière à être plus efficaces et rentables. En effet il permet de faire fonctionner sur une seule machine plusieurs systèmes d'exploitation et/ou applications isolées les unes des autres.

Le Cloud est donc un ensemble de machines virtuelle qui utilise la même infrastructure physique, mais lorsque le nombre de machines augmente la probabilité de voir une panne intervenir durant l'exécution devient de plus en plus fort. Dans un tel contexte la sureté de fonctionnement, la stabilité des applications sont des critères importants. C'est pourquoi un mécanisme de tolérance aux pannes devient nécessaire pour en assurer certains aspects comme la fiabilité et la qualité (Qos) de service.

La tolérance aux pannes a toujours été le problème majeur des chercheurs dans différents domaines de l'informatique, de même le Cloud ne s'est pas échappé de cette réalité, la tolérance aux fautes reste encore un grand défi pour lui. Plusieurs mécanismes et manières de tolérance aux pannes ont été proposés.

On distingue plusieurs solutions de tolérance aux pannes : les protocoles par point de reprise, les protocoles par journalisation et la tolérance aux pannes par duplication, ou encore la migration des machines virtuelles, chacune de ces catégories présente des propriétés différentes en termes de performance.

La migration des machines consiste à déplacer une machine virtuelle d'un hôte physique à un autre, cette technologie facilite la gestion des pannes, répartir de manière uniforme et automatique une charge de travail donnée.

Ce mémoire présente une étude sur la tolérance aux pannes dans le Cloud Computing en utilisant la migration. Pour cela nous avons dans un premier temps créé un environnement de Cloud Computing sur OpenStack en utilisant une architecture multi-nœuds constituée d'un nœud contrôleur et de deux nœuds de traitement. Ensuite nous avons fait toutes les configurations requises pour activer la migration en direct qui n'est pas activée par défaut sur OpenStack. Enfin nous avons testé la migration (non directe et directe).

Pour mener à bien notre travail, nous avons opté pour une démarche qui s'étale sur quatre (04) chapitres :

- Le premier a pour objectif d'introduire la notion de Cloud Computing, la virtualisation et présente les concepts, les services, les caractéristiques, et le modèle de déploiement des services de Cloud Computing.
- Le second fournit une vue sur les concepts et problème associés aux notions de la tolérance aux pannes dans les environnements du Cloud Computing : définition de la panne, types de panne et leur classifications, la sureté de fonctionnement. Nous présenterons aussi dans ce même chapitre les techniques, les approches et les politiques de tolérances aux pannes. Et nous exposerons quelques travaux à la tolérance aux pannes dans le Cloud Computing trouvés dans la littérature.
- Le troisième présentera en détail la solution proposée qui est la migration, il présentera aussi les outils utilisés.
- Le quatrième et dernier chapitre nous présenterons dedans les étapes suivies pour l'expérimentation de l'approche proposée et nous discuterons des résultats obtenus.

Nous clôturons ce mémoire par une conclusion générale sur le travail accompli ainsi que ses perspectives.

CHAPITRE 1

LE CLOUD COMPUTING

Chapitre1: Cloud Computing

1.1 Introduction :

La génération actuelle d'internet nous permet grâce au Cloud Computing d'acheter des services informatiques à partir d'un portail web. Nous sommes en mesure de louer à partir d'une vitrine virtuelle la base nécessaire pour construire un centre des données virtuel, tels que le processeur, la mémoire et le stockage, et ajouter au-dessus le middleware nécessaire tels que les serveurs d'applications Web, bases de données et bus serveur d'entreprise, etc.

Au niveau de ce chapitre nous allons présenter la définition du Cloud Computing, son évolution, ses caractéristiques et ses modèles de dépoilement et de service, son architecture.

Nous décrivons aussi la virtualisation qui est une partie essentielle dans l'informatique en nuages, ainsi que les avantages, les inconvénients, et les défis auxquels il fait face.

1.2 Définition :

Plusieurs définitions ont été données à ce terme, parmi les plus utilisées

CISCO [2] : « Le Cloud Computing est une plateforme de mutualisation informatique fournissant aux entreprises des services à la demande avec l'illusion d'une infinité des ressources».

NIST [3] : « Le Cloud Computing est un modèle qui permet un accès simple et pratique, à la demande, à un Pol partagé de ressources informatiques configurables (telles que réseaux, serveurs, stockage, application et services).Ces ressources peuvent être provisionné rapidement et distribuées avec un minimum de gestion ou d'interaction avec le fournisseur de services ».

L'infonuagique est l'exploitation de la puissance de calcul ou de stockage de serveurs informatiques distants par l'intermédiaire d'un réseau généralement internet, ces serveurs sont loués à la demande, le plus souvent par tranche d'utilisation selon les critères technique (Puissance, bande passante,..). [4]

La figure ci-dessous illustre le Cloud

Figure 1-1: Cloud Computing

1.3 Historique :

L'idée du Cloud est né en 1960 lorsque John McCarthy affirma dans une conférence au MIT que la puissance de traitement informatique serait accessible au public dans le futur. [52]

Mais ce fut qu'en 1990et surtout en 1991 que ce concept commence à prendre de l'importance avec la naissance d'Internet et la mise sur le marché du logiciel CERN qui a été le premier logiciel accessible par le Web. Ensuite, l'idée a progressé avec l'apparition de nouvelles solutions IT le lancement du navigateur Mosaic en 1993 et celui du navigateur Netscape en 1994. [53]

En 1990 Salesforce.com a gravé son nom en lançant le concept de fourniture d'applications d'entreprise via un simple site Web. Le développement suivant été en 2002 lorsque Amazon proposa un ensemble d'hébergeur d'application. EC2 / S3 est devenu le 1 ^{er} service d'infrastructure de technologie Cloud accessible. [6]

En 2009, une autre étape importante a marqué Go6ogle du Web 2.0. Google et d'autres ont commencé à proposer des applications basées sur un navigateur via Google Apps (qui permets aux utilisateurs de stocker des documents sur le Cloud) et d'autres applications. Viennent ensuite Azure de Microsoft: Microsoft et Google fournissent tous deux des services fiables et faciles à consommer. [6]

En 2010, salesforce.com lance une base de données Cloud avec Database.com pour les développeurs, marquant ainsi le développement des services de Cloud Computing

utilisables sur n'importe quel terminal, exécutables sur n'importe quelle plateforme écrite en n'importe quel langage de programmation.

Le Cloud évolue en permanence, en 2013 les dépenses mondiales en service du Cloud étaient estimées à 47 milliards de dollars. Ce chiffre ne cesse de croitre, selon les prévisions, d'ici 2020, le marché du Cloud Computing devrait dépasser les 241 milliards de dollars. [50]

1.4 Les modèles de déploiement du Cloud Computing [19] :

L'approche Cloud se décline en différents modes d'hébergement. Chaque mode d'hébergement va avoir un impact sur les caractéristiques essentielles du Cloud.

1.4.1 Le Cloud public :

Le Cloud public est un ensemble des services et des ressources accessibles par Internet et géré par un prestataire externe. Ces services sont partagés par le grand public qui les utilise à la demande.

Ce type d'accès est moins sécurisé en raison de son ouverture. L'un des principaux avantages de ce type de Cloud est que la mise en place est facile et peu couteuse parce que le matériel, l'application et le coût de bande passante sont couverts par le fournisseur. Dans le Cloud public, l'environnement est géré par les entreprises qui met à disposition les services du Cloud et c'est le fournisseur qui conçoit, maintient et fait évoluer les ressources en fonction des besoins des clients et donc la sécurité est de la responsabilité du fournisseur qui en assure la gestion.

1.4.2 Le Cloud privé :

Le Cloud privé est un modèle de Cloud Computing qui fournit des services à un seul client ou à une seule organisation, Le service peut être administré par l'organisation elle-même ou par un fournisseur externe. Ce modèle de Cloud est moins vulnérable aux problèmes de sécurité car l'accès aux services et aux réseaux étant restreint, de plus il permet aux fournisseurs et aux clients de mieux contrôler leur infrastructure. Ce haut degré de contrôle et de transparence permet au propriétaire d'un Cloud privé de se conformer plus facilement à des normes, politiques de sécurités qui peuvent être requises dans certains domaines.

1.4.3 Le Cloud hybride :

Le Cloud hybride est une infrastructure combinant le Cloud public et le Cloud privé pour offrir les meilleurs avantages des deux mondes**.** Ce modèle est souvent utilisé pour supporter des pics de charges exceptionnels (évènementiels), par exemple pendant les fêtes de Noël**.** Les différents Cloud qui la composent restent des entités indépendantes, mais sont reliés entre elles par une technologie normalisée ou propriétaire qui permet la portabilité des données et des applications.

1.5 Les modèles de services de Cloud Computing :

Les modèles de service sont les modèles de référence sur lesquels le Cloud Computing est basé. Il y a 3 modèles et chacun de ces derniers joue un rôle spécifique.

1.5.1 Infrastructure as a Service (IaaS): [20]

En Français "L'infrastructure en tant que service", est un service principal dans le Cloud, il fournit l'accès aux ressources fondamentales telles que les machines physique, machines virtuelles, le stockage Virtual, etc. Ils fournissent aux utilisateurs une capacité de traitement, de stockage, de bande passante, de réseaux, de serveurs virtuels et d'autres ressources de calcul. IaaS est dédié aux ingénieurs réseau.

Les utilisateurs de l'infrastructure en tant que service peuvent donc exécuter et héberger leurs applications informatiques dans le nuage et ne paient que les ressources qu'ils consomment.

Ils n'ont aucun contrôle sur l'infrastructure sous-jacente du provider de Cloud mais ils ont le contrôle sur le système d'exploitation, le stockage et les applications hébergées sur le serveur virtuels. Exemples de service IasS gratuit: Google drive et DropBox

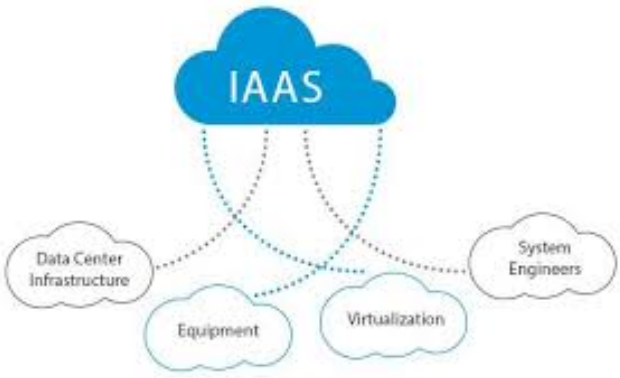


Figure 1-2: Plateforme IaaS

1.5.2 Plateforme as a Service (PaaS) :

PaaS en Français " plate-forme en tant que service", fournit l'environnement d'exécution pour les applications, les outils de développement et de déploiement.Ces plateformes peuvent être utilisés pour exécuter des sites web, des SaaS ou tout développement issu de l'entreprise. Ces développements doivent respecter le langage de développement et l'architecture de la plateforme PaaS c'est-à-dire les applications déployées sur l'infrastructure sont basé sur des langages de programmation et les outils supports par ce dernier sont spécifiques à un langage et à une base de données.PaaS est dédié aux développeurs (DotNet et J2EE).L'utilisateur va utiliser cette plate-forme via son navigateur et il n'a aucun contrôle sur l'infrastructure sous-jacente du provider de Cloud (serveurs, stockage, système d'exploitation, réseau….Etc.) Mais il a le contrôle sur les applications déployées et l'environnement d'hébergement (paramétrage des services Web, structure des bases de données…).Le système d'exploitation et les outils d'infrastructure sont sous la responsabilité du fournisseur. Exemple de cas d'usage :

- ✓ Mise à disposition d'instances de serveur d'application/ base de données.
- ✓ Ajout ou suppression de serveurs d'une manière dynamique.

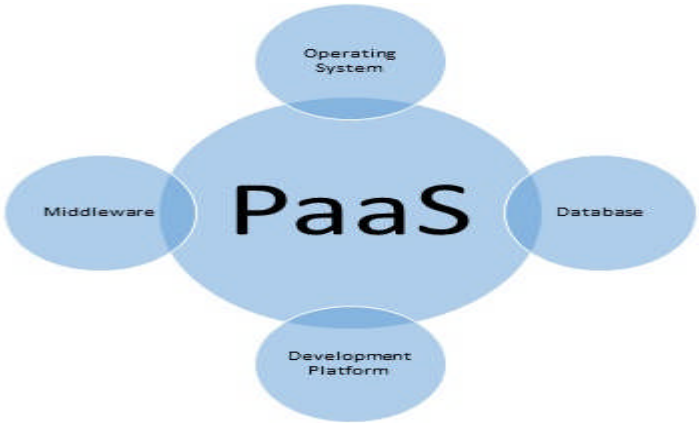


Figure 1-3: Plateforme PaaS

1.5.3 Software as a Service (SaaS) : [20] [21]

En français " Logiciel en tant que service", ce modèle fournit un logiciel sous forme de service. La différence entre un logiciel et SaaS c'est que ce dernier proposent des logiciels opérationnels, prêt à l'emploi, sans les installer et sans aucune tâche de maintenance. L'entreprise n'hébergera pas ses applications et ne stockera pas ses données en interne. Il n'est donc pas nécessaire d'acquérir directement ces applications et posséder des serveurs pour les héberger. De plus, la maintenance et les mises à jour des applications seront gérées en externe par le prestataire. L'accès à ces applications se fait à distance par le navigateur web. SaaS n'est pas dédié à un type d'utilisateur spécifique. Les utilisateurs n'ont aucun contrôle sur l'infrastructure du Cloud que ce soit sur l'aspect technique (Datacenter, réseau, stockage…etc.) ou sur les aspects fonctionnels (version de l'application, fonctions disponibles).Les utilisateurs de l'entreprise devront simplement disposer d'un ordinateur et des codes d'accès au service en ligne pour pouvoir travailler. Exemples de cas d'usage :

- ✓ Accès à une messagerie en ligne ou bureautique.

Figure 1-4: Plateforme SaaS

De nombreux autres acronymes en ‘asS’ émergent de jour en jours, et sont associés au Cloud Computing qui s’affiche désormais comme un mode de consommation des services IT. Nous citerons certains d’entre eux :

- Desktop as a Service (DaaS) : Il met en disposition un bureau partagé où on trouve des logiciels installés et des espaces de stockages.
- Data as a Service (DaaS) : Il simplifie le stockage des données et confie au fournisseur le soin de supporter les variations de volume de données stockées. Ce stockage peut être permanent ou temporaire. [51]
- Backend as a Service (Saas) : Fournit un back end pour les applications (principalement mobile), Il fournit des API et des outils permettant à différent langages de s’intégrer à leur back-end. Il fournit aussi des services supplémentaires comme le stockage, les tableaux de bord…

1.6 Caractéristiques du Cloud :

- **Service à la demande :**

Un client peut faire une demande à tout instant et pour n’importe quel type de ressources informatiques et dans ce cas, la réponse à une demande est immédiate et tout ceci sans avoir besoin d’intervention du fournisseur de Cloud puisque tout est automatisé.

- **service mesurable :**

Comme le Cloud amène la capacité d’allouer dynamiquement des ressources, le client et le fournisseur vont avoir besoin de mesurer l’usage fait des ressources, l’utilisateur n’utilise que ce dont il a besoin et ne paye que ce qu’il utilise. Il est important pour le client d’avoir un suivi du consommé. Le suivi des ressources consommé est important pour la gestion et l’optimisation des ressources mais aussi pour des questions de facturation.

- **Mutualisation des ressources :**

Un hébergeur Cloud possède d’énormes ressources et ces dernières sont partagées entre l’ensemble de ses clients en fonction de la demande. Et tout ça sans qu’aucun ne sache où se situe la ressources consommée.

- **l’élasticité des ressources :**

L'élasticité des ressources est la capacité d'allouer dynamiquement des ressources en fonction des besoins qu'ils soient temporaires ou durables.

1.7 La virtualisation [7] [8] :

L'évolutivité est l'idée la plus importante derrière le Cloud et la technologie qui le rend possible est le Cloud.

La virtualisation est une technologie qui permet de créer des services informatiques utiles à l'aide de ressources qui sont liées au matériel. Elle permet d'exploiter toute la capacité d'une machine physique en la répartissant entre de nombreux utilisateurs ou environnements différents. En d'autres termes, elle permet de faire fonctionner sur une seule machine plusieurs systèmes d'exploitation, plusieurs instances différentes ainsi que plusieurs applications séparément les uns des autres, comme si ils fonctionnaient sur des machines physique distinctes.

Les intérêts de la virtualisation sont : l'utilisation optimale des ressources, l'économie sur le matériel par mutualisation, l'allocation dynamique de la puissance de calcul, la facilité d'installation, de déploiement et de migration des machines virtuelles. La virtualisation est utilisée dans plusieurs domaines de l'infrastructure informatique d'une entreprise, on peut citer les domaines d'applications suivants :

- ✓ **Virtualisation des serveurs** : La virtualisation des serveurs permet d'exécuter plusieurs systèmes d'exploitation sur le même serveur physique en tant que machines virtuelles ultraperformantes. [9]

- ✓ **Virtualisation de réseau** : En reproduisant entièrement un réseau physique, la virtualisation de réseau permet d'exécuter des applications sur un réseau virtuel comme sur un réseau physique, mais avec les avantages opérationnels et l'indépendance vis-à-vis du matériel assurés par la virtualisation. (Elle présente les périphériques et services réseau logiques (ports, commutateurs, routeurs, pare-feu, équilibreurs de charge, VPN et autres) aux charges de travail connectées.)[9]

- ✓ **Virtualisation du stockage** : c'est la mise en commun de stockage physique de multiples périphériques de stockage réseau dans ce qui semble être un dispositif de stockage unique qui est gérer depuis une console centrale .

1.8 Architecture global du Cloud Computing :

L'architecture globale du Cloud Computing comporte essentiellement:

- ✓ **Des clients:** personne, entreprise, groupes qui accèdent aux différents services offerts par le Cloud.
- ✓ **Des Services** : Un service Cloud comprend des produits, des serviceset des solutions livrés et consommés en temps réel sur Internet. Par exemple, les services Web accessibles par d'autres composants et logiciels de Cloud Computing.
- ✓ **Un réseau:** Intermédiaire entre le client et le fournisseur, qui permet de transiter les services c'est le réseau Internet.
- ✓ **Des fournisseurs:** Ce sont des entités chez les quelles on alloue les services Cloud.
- ✓ **Des serveurs**: Où sont stockés les services Cloud ils ont éparpillé partout dans le monde qui constitue le cœur hardware de Cloud Computing.

1.9 Avantage du Cloud Computing :

L'avantage du Cloud Computing réside dans le sens où au lieu d'acheter cher des serveurs et des logiciels, qui ne sont pas utilisés à 100%, les entreprises les louent et ne paient que pour l'usage en font .elles peuvent aussi, auxquelles minutes, accéder à des capacités de stockages et de calcul importantes. En plus de cela, Cloud offre plusieurs avantages aux utilisateurs [3].

- ✓ **Flexibilité :** Les utilisateurs peuvent dimensionner les services en fonction de leurs besoins, personnaliser les applications et accéder aux services Cloud depuis n'importe où avec une connexion Internet. [22]
- ✓ **La réduction des coûts :** La réduction du cout du à l'efficacité opérationnelle et déploiement plus rapide de nouveaux services aux entreprises.
- ✓ **L'accessibilité :** Les services de Cloud Computing sont accessible à tout moment, sur tous les supports, via une connexion internet. [22]
- ✓ **La sécurité des données :** les fournisseurs garantissent aux utilisateurs un très haut degré de sécurité des données avec le chiffrement des données, la surveillance logicielle et la sécurisation des lieux de stockage. [22]
- ✓ **Pas de dépenses de capital** : Plus besoin de locaux pour élargir vos infra-structures informatiques.

- ✓ **La virtualisation:** c'est à dire pas besoin d'investir pour concevoir une plateforme de Cloud Computing.

1.10 Inconvénient du Cloud Computing [23] :

Les grandes catégories de risques liés à l'adoption du Cloud se déclinent selon les points suivants :

- ✓ **La sécurité :**

La plateforme Cloud, si elle est externe (non installée sur le réseau interne ou avec une ouverture extérieure) doit être suffisamment sécurisée pour éviter le risque d'intrusion, de vol des données par piratage. Néanmoins, les fournisseurs de Cloud garantissent un important niveau de sécurité grâce à des moyens de grande envergure : logiciels de détection d'intrusion poussés et réplication des données en temps réel. L'autre risque est qu'un utilisateur oublie de se déconnecter sur un appareil accessible par des éléments externes à l'organisation. Il faut dans ce cas prévoir une déconnexion automatique en cas de non-activité du compte et bien segmenter les droits utilisateurs afin que ces derniers ne puissent accéder qu'aux données des projets dans lesquels ils sont impliqués.

- ✓ **Connexion :**

Si l'utilisateur n'a pas de connexion internet, ou une connexion insuffisante, il ne pourra accéder à sa plateforme de travail. L'idée dans ce cas est de permettre le travail sur une application locale qui synchronise ensuite les données avec le serveur dès que l'utilisateur a à nouveau.

- ✓ **Consommation d'électricité :**

Le grand problème du Cloud Computing c'est la consommation élevé de l'électricité et l'énergie dépensé par les Datacenter.

- ✓ **Localisation des données :**

La dématérialisation des données sur différents sites physique de stockage peut conduire à un éclatement des données et leur répartition sur différentes zones géographiques.

1.11 Les grands défis relatifs à l'adoption du Cloud Computing :

Le Cloud Computing prend de plus en plus sa place dans le monde du Business des systèmes d'information notamment du fait qu'il permet aux utilisateurs de réaliser une économie d'échelle importante en payant uniquement pour les ressources utilisées. Cependant, il existe encore un nombre important de défis liés à l'adoption du Cloud Computing

- **Sécurité :**

C'est la plus grande préoccupation concernant le Cloud Computing. Certains des défis de sécurité Cloud courants sont les suivants: [24]

- Interfaces non sécurisées.
- Gestion des identités et des informations d'identification.
- Détournement du compte et de trafic.
- Les violations de données.
- Perte de données.

- **L'équilibrage de charge : [32]**

L'équilibrage de la charge est une discipline informatique qui implique la répartition des requêtes vers plusieurs serveurs, qu'ils soient regroupés dans un centre de données ou géographiquement dispersés, afin d'éviter les situations dans lesquelles un serveur unique est saturé par la charge de trafic Web.

- **Tolérance aux fautes et disponibilité : [10]**

Le fournisseur Cloud est censé offrir à ses clients un environnement tolérant aux fautes où le client ne risque ni de perdre ses données, ni de voir l'exécution de ses applications perturbée. La tolérance aux pannes est une méthode qui permet à un système de rester fonctionnel, de manière réduite au lieu de tomber complètement en panne avec une réduction du débit ou une augmentation du temps de réponse.

1.12 Conclusion :

Dans cette partie, nous avons pu développer le vaste sujet du Cloud et nous avons pu identifier les challenges du Cloud aux quels la communauté scientifique doit faire face.

Le Cloud Computing marque une réelle avancée vers l'infrastructure informatique dématérialisée. Il fournit des ressources informatiques, logicielles ou matérielles, accessibles à distance en tant que service. L'adoption de ce modèle soulève un certain nombre de défis, notamment au sujet de la tolérance aux pannes. Dans le prochain chapitre nous introduirons la notion de tolérance aux pannes ainsi que l'équilibrage de charge dans les environnements Cloud.

CHAPITRE 2

LA TOLÉRANCE AUX PANNES

Chapitre 2: La tolérance aux pannes

2.1 Introduction :

Au cours de ces dernières années, les recherches et les évolutions autour des technologies de réseaux ont fortement contribué à l'émergence de l'informatique distribuée, avec cette évolution les systèmes sont devenus de plus en plus vulnérables et complexes, de nombreuses pannes surgissent et la continuité de leurs fonctionnements n'est plus garantie.

La tolérance aux pannes est donc devenue une préoccupation majeure, divers politiques et techniques tolérantes aux pannes y sont implémentées.

La migration est l'un des mécanismes les plus célèbres de tolérance fautes, il permet de garantir la continuité de l'exécution de l'application et évite toute perte due à une défaillance matérielle de manière transparente et efficace.

Dans ce chapitre nous allons étudier la tolérance aux pannes. Nous présentons les concepts de panne et leurs classifications et nous exposons quelques techniques de tolérance aux pannes.

2.2 Pannes

2.2.1 Définition des concepts :

Panne : La panne est l'arrêt accidentel ou l'interruption du fonctionnement d'un système informatique, c'est l'état anormal d'un système ou de l'un de ses composantes matérielles ou logiciels le mettant dans l'impossibilité d'accomplir des fonctions sous des calculs requis.

Faute et erreur : « Une faille (ou panne) du système se produit lorsque son comportement devient inconsistant et ne fournit pas le résultat voulu. La panne est une conséquence d'une ou plusieurs erreurs. Une erreur représente un état invalide du système due à une faute (défaut). La faute est donc la première cause de l'erreur, cette dernière provoque la faille du système. Le but de la tolérance aux pannes est d'éviter la faille totale du système malgré la présence de fautes dans un sous ensemble de ses composants élémentaires » [54].

2.2.2 Classification des pannes :

On peut classifier les pannes selon différent critères, le schéma suivant montre la classification selon l'origine de la panne et selon son degré soit de gravité ou de permanence :

Selon son origine ***: Les pannes peuvent être d'origine matérielle, logicielle, humaine ou environnementale :***

- **Humaine** : elles constituent 15 à 30% de l'ensemble des pannes [9]. Ces pannes sont dues à l'intervention humaine, lors du placement d'un composant, d'une reconfiguration ou lors d'une mise à jour logicielle. Elles sont souvent intentionnelles, ou causées par le manque de compétences.

- **Matérielle** : Les pannes d'origine matérielle sont dues à un défaut dans un ou plusieurs composants matériels du système. Le plus souvent, ce sont des pannes franches qui causent l'arrêt du fonctionnement du système. Elles sont donc facilement détectables, mais requièrent le remplacement des composants défaillants [9].

- **Logicielle** : elles sont de loin les plus fréquentes, ces bugs sont dues à des fautes lors de la conception ou le développement du logiciel à cause d'un manque de budget ou d'une mauvaise planification ou d'une mauvaise compréhension des fonctionnalités du projet. L'activation de ces fautes peut survenir à n'importe quel moment lors du fonctionnement du logiciel. En particulier, le manque de ressources, une configuration nouvelle de l'environnement d'exécution, l'interaction avec d'autres systèmes ou encore le vieillissement du logiciel peuvent constituer des facteurs d'activation [9]. Bien souvent, le redémarrage du logiciel fautif peut suffire à éradiquer la panne et à restaurer un état sain.

Selon le degré : on distingue le degré de gravité de défaillance et le degré de permanence

Degré de gravité de défaillance :

- **Pannes franches** : le composant fonctionne correctement puis panne et cesse immédiatement et d'une façon indéfini de fonctionner, il ne répond plus à toute sollicitation et il ne génère plus de nouvelles requêtes jusqu'à sa réparation.

- **Pannes transitoires ou par omission** : Dans ce type de pannes, le système peut perdre des messages en entrée ou en sortie ou les deux, cette panne est considérée comme temporelle de durée infinie. Ce modèle de panne est utilisé pour représenter des défaillances du réseau [10].

- **Pannes temporelles :** le temps de réponse du système dépasse les exigences De spécification ou son délai de garde expire.

- **Panne byzantine ou arbitraire :** inclut tous les types de pannes, y compris le fait de délivrer un résultat ou un message erroné (intentionnellement ou non), le processus peut alors faire «n'importe quoi », y compris avoir un comportement malveillant. « Les fautes byzantines sont dues à des processus byzantins qui ont un comportement arbitraire, ne suivant plus (nécessairement) le code de leurs algorithmes locaux. Cela peut être du a une erreur matérielle, un virus, ou la corruption du code de l'algorithme» [10].

Selon le degré de permanence :

- **Panne transitoire :** Comportement erroné des composants pendant une certaine période, pour reprendre ensuite à la fin de cette période son fonctionnement correctement.
- **Panne intermittent :** elle se produit d'une manière aléatoire, elle panne et reprend son comportement normal dans un intervalle irrégulier.
- **Panne permanente :** continue et stable dans le temps, la panne permanente persiste tant qu'il n'y a pas d'intervention externe pour l'éliminer. Un changement physique dans un composant provoque une panne matérielle permanente.

- **Panne accidentelle :** Elle se produit de manière accidentelle, soit par une fausse manœuvre, en exécutant du code erroné ou en installant un composant mal configuré

ou moins conformant [10]. Dans ce cas le composant soit, s'arrête complètement de fonctionner ou bien continue mais sans retourner à un état stable (valide).

La figure ci-dessus résume la classification des pannes

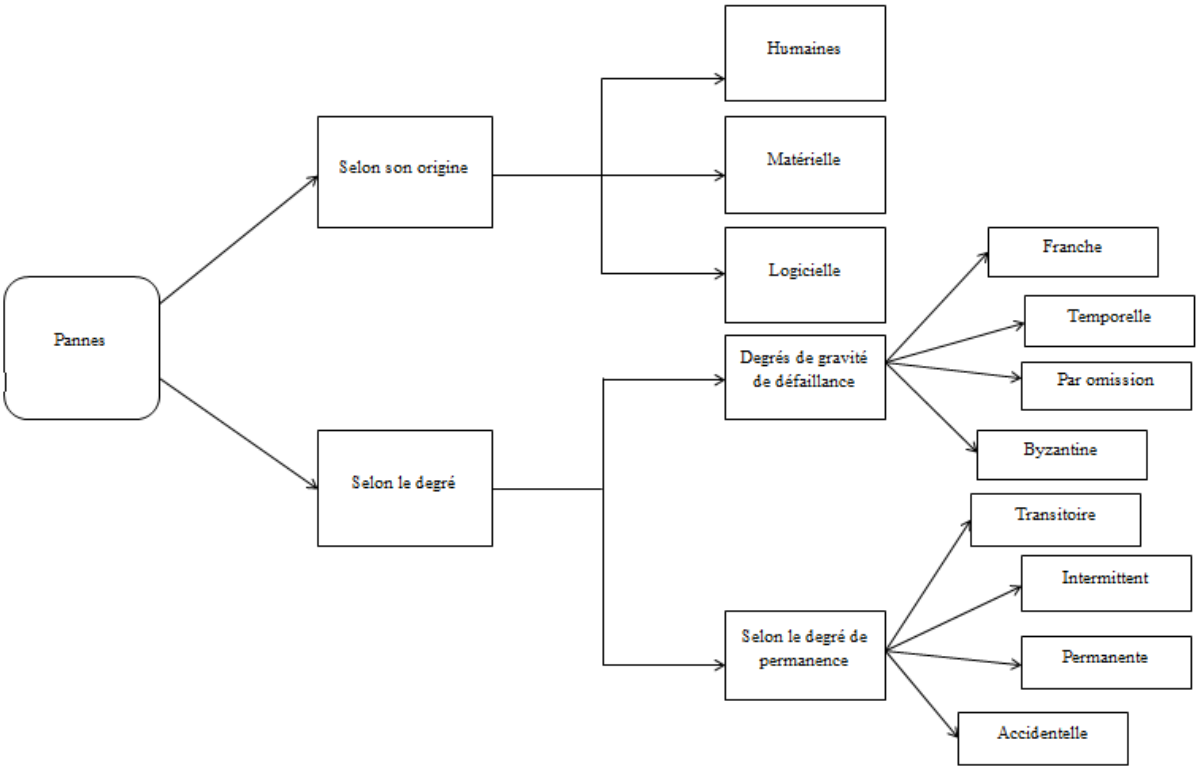


Figure 2-1: classification des pannes

2.3 Sureté de fonctionnement d'un système :

La tolérance aux pannes s'inscrit dans le contexte plus large de la sureté de fonctionnement. La sureté de fonctionnement est définit comme étant la propriété qui permets aux utilisateurs du système de placer une confiance justifiée dans le service qu'il leurs délivre [11]. La sûreté de fonctionnement s'intéresse entre autre à améliorer la survivance des systèmes, c'est-à-dire leur capacité à continuer de fonctionner pendant et après une perturbation rencontrée durant leur vie opérationnelle. Cette perturbation peut provenir de l'environnement du système (ce qui lui est extérieur) ou du système lui-même (ressources, fautes, ce qui lui est propre) [10].

La figure ci-dessous présente les trois notions La sûreté de fonctionnement

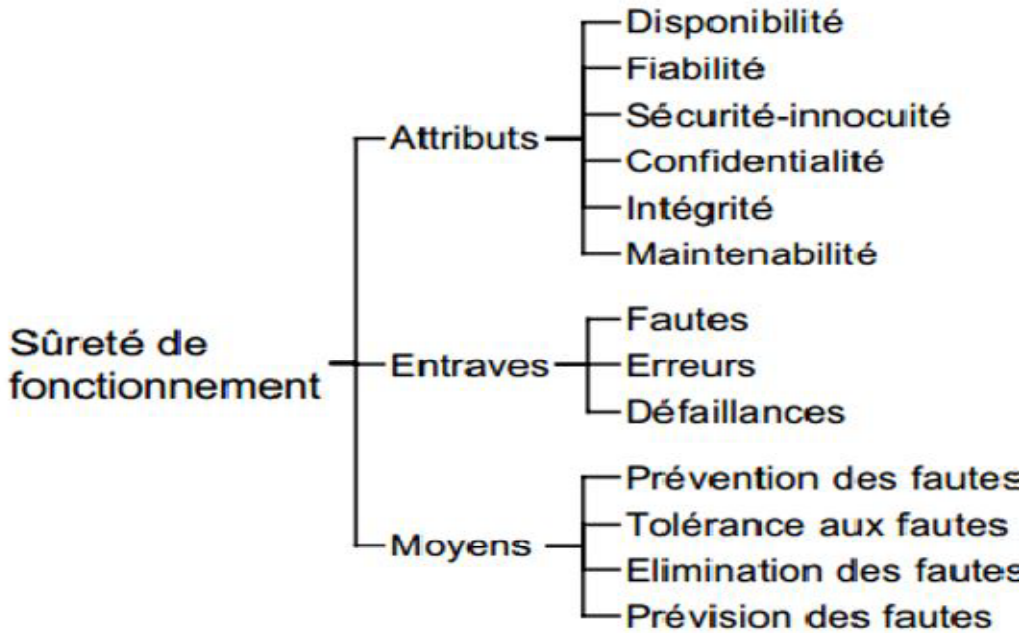


Figure 2-2: l'arbre de sureté de fonctionnement d'un système

2.3.1 Attributs :

Les attributs sont les propriétés que le système doit vérifier à fin d'évaluer la qualité du service qu'il offre.IL existe six attributs de la sureté de fonctionnement :

a. ***Disponibilité*** : Il s'agit du temps durant lequel le système est disponible. En d'autre terme c'est la capacité du système à interagir avec les autres sans interruption et dans le délai attendu.

b. **Fiabilité** : c'est la capacité du système à se retourner dans un état cohérent après panne et la continuité du service.

c. **Sécurité** : c'est l'absence de conséquences catastrophiques sur l'utilisateur ou son environnement avec la préservation de la confidentialité et de l'intégrité des informations [10].

d. **Maintenabilité** : la capacité du système à être maintenu, de manière cohérente et à moindre coût, en état de fonctionnement.

e. **Confidentialité** : cet attribut évalue la capacité du système à fonctionner en dépit de fautes intentionnelles et d'intrusions illégales.

f. **L'intégrité** : définit son aptitude à assurer des altérations approuvées des données.

2.3.2 Les entraves :

Les entraves à la sûreté de fonctionnement sont un ensemble d'états dans un système qui peuvent influencer négativement sur l'un des attributs précédemment décrits, une combinaison de ces attributs, ou même sur l'ensemble de ces attributs.

L'activation d'une faute du système provoque la propagation d'erreurs dans le système. Une défaillance a lieu lorsqu'une erreur est observée à la frontière du système.

Les entraves sont représentées par la trilogie faute, erreur et défaillance [12].

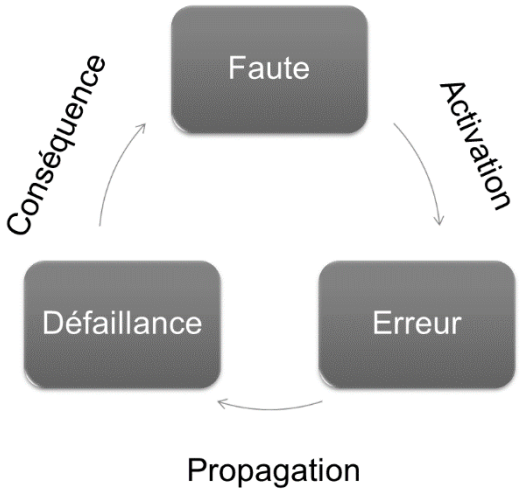


Figure 2-3: Trilogie des entraves

a. Faute : La Faute est la cause interne de la défaillance. Elle peut être introduite par le concepteur, l'utilisateur ou l'environnement. Elle est inévitable, naturelle, tolérable [12].

b. Erreur : L'erreur est une manifestation interne, un signal ou un état incorrect [12].

c. Défaillance : c'est la cessation de l'aptitude d'une entité à accomplir une fonction requise [12].

2.3.3 Moyens d'assurer le bon fonctionnement d'un système :

C'est les méthodes et techniques éprouvés pour casser les enchaînements dans la trilogie faute, erreur et défaillance. Le développement d'un système sur de fonctionnement passe par l'utilisation combinée de ces méthodes qui peuvent être classées en quatre catégories [13].

a. **La prévention des fautes** : elle se base sur des méthodes de développement et des techniques d'implémentation pour empêcher que les fautes s'introduisent lors de la phase du développement.

b. **L'élimination des fautes** : peut être divisée en 2 catégories

- Élimination pendant la phase de développement : en utilisant des techniques de vérification de façon à détecter les fautes et les enlever.
- Élimination pendant la phase d'utilisation : tenir à jour les défaillances rencontrées et les retirer pendant les cycles de maintenance [10].

c. La prévision des fautes : anticipe ces dernières pour ensuite pouvoir appliquer des moyens de l'élimination ou de la tolérance aux fautes.

d. La tolérance aux fautes : elle est intégrée pour corriger et éliminer les erreurs du système.

2.4 Tolérance aux pannes :

La tolérance aux pannes est une méthode qui vise à éviter la défaillance en détectant les erreurs et contribuer au rétablissement du système, lui permettant ainsi de rester cohérant. La propriété de tolérance aux pannes est définie par l'aptitude du réseau à maintenir ses fonctionnalités, en cas de panne de certains de ses nœuds. Elle vise donc à minimiser l'influence de ces pannes sur la tâche globale du réseau [10].La tolérance aux fautes est mise en œuvre selon la procédure de la figure ci-dessus

Figure 2-4: Techniques de tolérance aux pannes

2.4.1 Détection de pannes :

La détection de panne est résolue selon le modèle de communication du système. Dans les modèles synchrones où le temps de transmission d'un message est borné. Les pannes de franches et de temporisation peuvent alors être détectées lors de la communication à l'aide du délai de garde « watchdog » : lorsqu'un processus communiquant ne reçoit pas de confirmation après le délai de garde de la part de l'autre, ce dernier est alors considéré défaillant.

2.4.2 Détention de pannes :

Cette phase établit des limites des effets de la panne sur une zone particulière afin d'empêcher la contamination des autres régions. En cas de détection d'intrusion, par exemple, l'isolation des composants compromis minimise le risque d'attaques des composants encore fonctionnels.

2.4.3 Recouvrement d'erreur :

Cette phase vise à transformer l'état erroné en un état exempt d'erreur, on procède à l'élimination des erreurs détectées, en utilisant un point de reprise ou par poursuite sur un état prédéfini. Ce qui est sauvegardé lors de la génération d'un point de reprise n'est généralement pas un cliché de l'état de l'ensembledu système, mais seulement l'état d'une partie du système **[15].**

Le recouvrement d'erreurs se base sur 3 technique : la reprise, la poursuite, la compensation.

- **La reprise** : dans cette technique l'état du système est régulièrement sauvegardé appelé point de reprise où le système est ramené en cas d'occurrence d'erreur.
- **La poursuite** : consiste à rechercher dans le disque un état exempt d'erreur (généralement un état dégradé) et ramené le système vers cette état.
- **La compensation** : Il faut que le système ait suffisamment de redondance pour permettre sa transformation en un état fonctionnel, elle est transparente vis-à-vis car elle ne nécessite pas de ré-exécuter une partie del'application (reprise), ni d'exécuter une procédure dédiée (poursuite)

Elle peut par exemple être réalisée en répliquant des composants et en effectuant un vote majoritaire sur les résultats. Cette méthode ne nécessite pas une détection d'erreur spécifique puisqu'elle effectue elle-même la détection d'erreur [18].

2.4.4 Traitement de panne :

Elle consiste effectuer un diagnostic sur la panne pour déterminer les causes, la localisation et la nature d'erreur puis passer à la réparation de la panne en isolant les composants erronés. La procédure de réparation dépend du type de panne. Les pannes permanentes exigent une substitution du composant avec un autre composant fonctionnel [15].

2.5 Politiques de tolérance aux pannes :

Il y a principalement deux politiques standards de tolérance aux fautes: Politique de tolérance aux pannes proactive et Politique de tolérance aux pannes réactive. Basé sur ces politiques de tolérance de pannes, différentes techniques sont utilisées pour fournir la tolérance de panne. La figure suivante résume ces techniques :

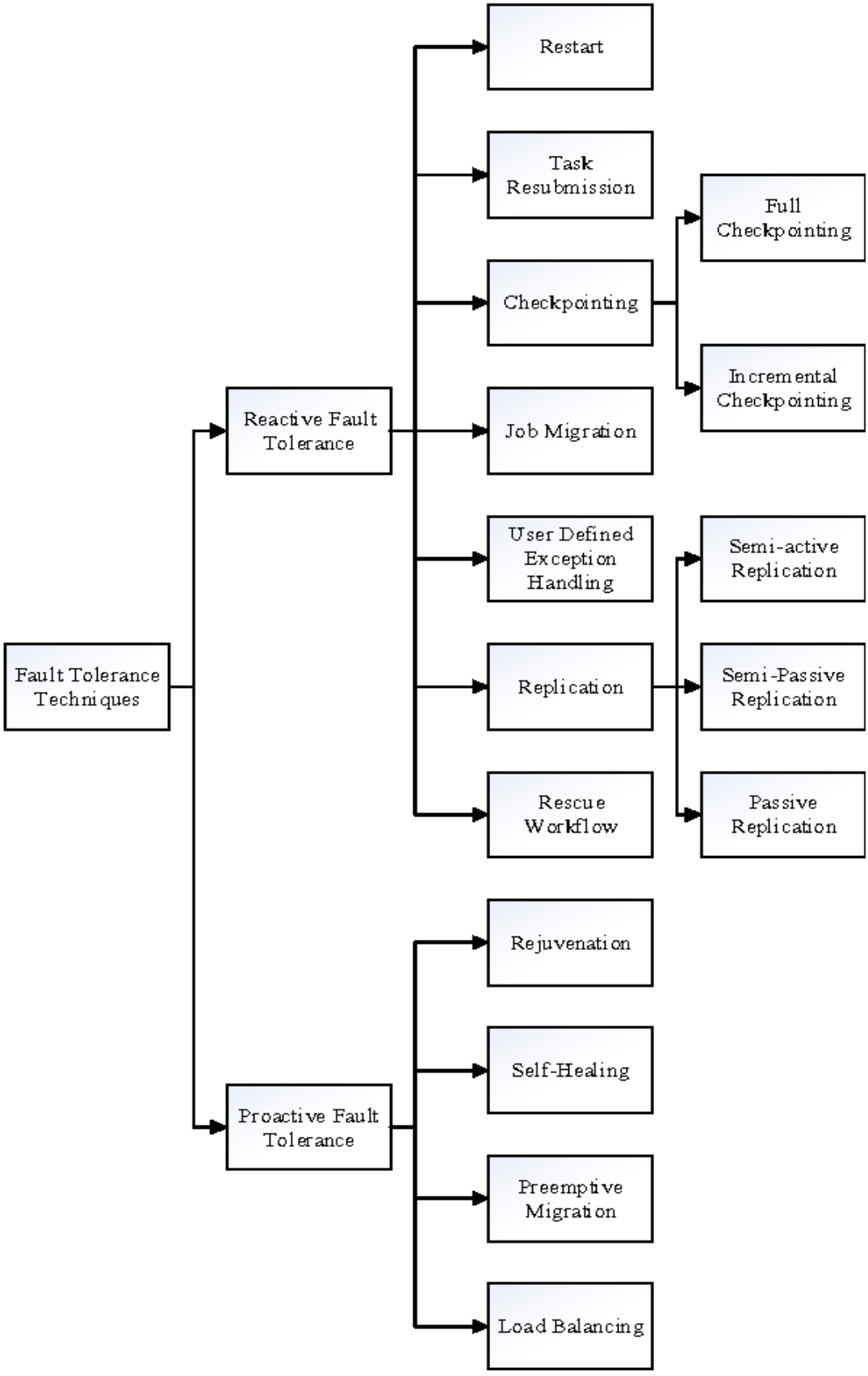


Figure 2-5: Politique de tolérance aux pannes

2.5.1 La tolérance aux pannes réactive

Systèmes de tolérance agissent sur le processus de récupération après occurrence de l'échec. Ces régimes prennent moins de quantité de ressources, plus de temps pour la récupération, comparé aux régimes proactifs [16]. Parmi les techniques réactives on trouve :

A. Checkpointing [15] [26] :

C'est une technique qui est utilisée pour réduire temps d'exécution des programmes de longue durée dans la présence d'échecs. Elle permet l'enregistrement périodique des états du système pendant l'exécution sans défaillance. En cas de panne il fait un retour en arrière et recharge les derniers point de reprise cohérant, reprend ensuite l'exécution à partir de ces points.

La création d'un point de sauvegarde nécessite souvent la suspension de l'exécution de l'application considérée. Le Checkpointing donc un cout sur l'exécution des applications.

De même, pour garantir la cohérence des applications, leurs communications réseau peuvent être journalisées ou mises en tampon mémoire à certain moment lors du processus du Checkpointing. Ainsi, ce processus a également un cout sur la communication des applications.

B. Duplication des données :

Consiste en la création des copies multiples des composants sur des processus différents. Cette approche rend le traitement masquant, le processus ne peut pas s'apercevoir qu'une défaillance a eu lieu. La réplication met en œuvre un processus qui est chargé de la création, du placement et de la gestion de copies d'entités physiques et/ou logicielles. Les entités répliquées peuvent être des données, du code, des objets, des composants physiques ou une combinaison de tous ces éléments.

Les techniques de réplication

Une stratégie de création de réplique doit répondre au problème suivant : le choix de la réplique, l'emplacement des nouvelles réplique sur les centres de donnés, quand créer la réplique et comment répliquer.

Trois principaux protocoles sont utilisés pour la gestion des répliques dans les systèmes distribués [26] [28] :

Protocoles de réplication passive :

La réplication passive distingue deux comportements d'un composant répliqué : la copie primaire (primary copy) et les copies secondaires (backups). La copie primaire est la seule à effectuer tous les traitements. Les copies secondaires ne traitent pas de messages, elles surveillent uniquement la copie primaire. En cas de défaillance de la copie primaire, la copie secondaire doit détecter cette défaillance et devient la nouvelle copie primaire. Cette technique nécessite un mécanisme de détection d'erreur.

Protocoles de réplication active :

Dans ce protocole de réplication, chaque copie joue un rôle identique à celui des autres copies. Toutes les copies reçoivent la même séquence, totalement ordonnées, des requêtes des clients, les exécutent puis renvoient la même s´séquence, totalement ordonnées, des réponses.

Protocoles de réplication semi-active :

C'est un protocole hybride qui se situe entre les deux protocoles précèdent où toutes les copies exécutent en même temps la requête du client, mais une seule copie (leader) d'entre elles émets la réponse, les autres copies mettent à jour leur état interne et sont donc ´étroitement synchronisées avec le leader.

2.5.2 Politique de tolérance aux pannes proactive :

A. L'équilibrage en charge :

Le principe de base de l'équilibrage de charge (LoadBalancing) consiste à effectuer une distribution des taches à des machines de façon intelligente. Les objectifs du load balancing est comme suit :

- Améliorer des temps de réponse des services.
- Capacité à pallier la défaillance d'une ou de plusieurs machines.
- Ajout de nouveaux serveurs sans interruption de service. [17]

B. La migration :

Un composant, serveur ou service est susceptible de tomber en panne lors de son exécution, pour éviter l'arrêt du système on migre vers un composant, serveur ou service plus robuste. A titre d'exemple si une baie de stockage tombe en panne, la baie redondante prend le relai avec un temps d'arrêt minimal du système même chose en cas d'arrêt d'un serveur ou service [10].Cette technique sera détaillé dans le prochain chapitre.

2.6 Travaux connexe : La tolérance aux pannes dans le Cloud Computing :

Le Cloud Computing est devenu une technologie révolutionnaire avec une tarification à l'utilisation, une évolutivité et une disponibilité à la demande des ressources informatiques en tant que caractéristiques principales. Mais en même temps, la fiabilité et la sécurité font partie des problèmes actuels de cette technologie. La tolérance aux pannes peut être utilisée pour résoudre et gérer des problèmes tels que la fiabilité et la disponibilité. Différents types de pannes et techniques de tolérance aux pannes dans le Cloud Computing ont été discutés Dans ce contexte on va exposer quelques une des recherches qui ont été élaborés.

Dans leur article « Improving Fault Tolerance in Virtual Machine Based Cloud Infrastructure » [29] Rajesh.S, KannigaDevi.R, proposent une méthode qui utilise deux ensembles différents de nœuds, l'un est l'ensemble des VM et l'autre est le nœud d'arbitrage principal (nœud serveur). La machine virtuelle utilise le teste d'acceptation pour validité sa logique. Le serveur contient le temps checker qui est l'évaluateur de la fiabilité et le mécanisme de décision pour trouver la VM fiable qui s'occupera pour traiter la demande du client. Pour fournir la tolérance aux pannes, les données peuvent être stockées sur de multiples Cloud utilisant la technique de virtualisation.

Zhao et al. Dans leur article « Fault Tolerance Middleware for Cloud Computing» [30] ont proposé un middleware tolérant aux pannes composé de trois composants: 1) un protocole de messagerie de bas niveau.

2) Un protocole d'affiliation à la détermination du chef pour que chaque membre d'un groupe a une vue cohérente du membre principal et les autres membres appartiennent à son groupe.

3) Un cadre de détermination virtuel qui transforme chaque opération non déterministe d'un membre principal en une opération pratiquement déterministe tout en garantissant que toutes les sauvegardes reçoivent les résultats dans le même ordre que celui du membre primaire.

Une autre approche différente de tolérance aux pannes dans le Cloud Computing est proposé par Anjali D.Meshram, A.S.Sambare, S.D.Zade [31], dans leur article ils ont proposé un modèle FTMC (FaultTolerance Model for Cloud) qui tolère les pannes sur une base de fiabilité de chaque nœud, ce dernier peut être supprimé si les applications qu'il héberge ne fonctionnent pas bien.

2.7 Conclusion :

Dans ce chapitre nous avons présenté un bref aperçu sur les notions de la tolérance aux pannes, nous avons commencé par définir qu'est-ce que c'est qu'une panne, donné sa classification. Nous avons ensuite traité d'un point de vue général la sureté de fonctionnement d'un système, les principaux concepts et techniques conçu pour tolérer les pannes.

Pour conclure nous avons montré quelques travaux sur la tolérance aux pannes dans le Cloud Computing. Ce dernier est sujet à divers types de pannes causées par de nombreux facteurs. Dans le prochain chapitre, nous exposons une stratégie de tolérance aux pannes basées sur la migration qui permet de garantir et d'assurer le bon fonctionnement d'un environnement Cloud Computing.

CHAPITRE 3

SOLUTION PROPOSÉE ET OUTILS UTILISÉS

Chapitre 3: Solution proposée et outils utilisés

3.1 Introduction :

Le Cloud Computing est sujet à divers types de pannes, Parmi les causes de ces pannes, on peut citer les problèmes techniques, les catastrophes naturelles, les pannes de courant, les problèmes de connexion au réseau, le sur chauffage du matériels, les systèmes de stockage défectueux et les erreurs humaines …etc. Pour éviter qu'un système tombe complètement en panne nous avons proposé la solution de migration qui permet d'améliorer la tolérance aux pannes et qui peut être utilisée aussi pour l'équilibrage de charge, afin d'optimiser l'utilisation des ressources CPU.

Le présent chapitre présentera la méthode de la migration d'une manière détaillée. Puis il décrit l'outil de travail éventuellement la plateforme d'OpenStack. Ensuite il décrit la migration dans l'environnement OpenStack.

3.2 Définition de la migration des machines virtuelles :

La migration de processus est le déplacement d'un processus d'une machine source vers une machine destination, ces deux machines étant reliées par un réseau de communication et n'utilisant pas de mémoire partagée. [26]

3.3 Techniques de migration de VM :

Les techniques de la migration sont classifiées globalement en deux types :

- Migration à chaud (live Migration).
- Migration à froid (non live Migration).

3.3.1 Migration à froid ‘’Stop and Copy migration ‘’ [44] :

La migration à froid (méthode d'arrêt et de copie) est une technique où la VM cesse de fonctionner sur la machine source et son exécution avec sa mémoire sont migrées vers la machine destination. Une fois toutes les pages de mémoires transférées, la VM est activée dans la machine destination. La VM reste en état désactivé jusqu'à ce que toutes les pages soient transférées à la machine destination. Dans ce type de migration, le temps total de migration est égal au temps d'arrêt.

La figure 3.1 illustre les étapes de migration Stop and Copy. Nous prenons comme exemple les étapes de migration de la VM3 de la machine 1 vers la machine 2 :

1. L'exécution de la VM3 est arrêtée dans la machine cible.
2. L'état d'exécution et les pages mémoires de la VM3 sont transférés vers la deuxième machine (machine2).
3. La VM3 est activée dans la machine destination.

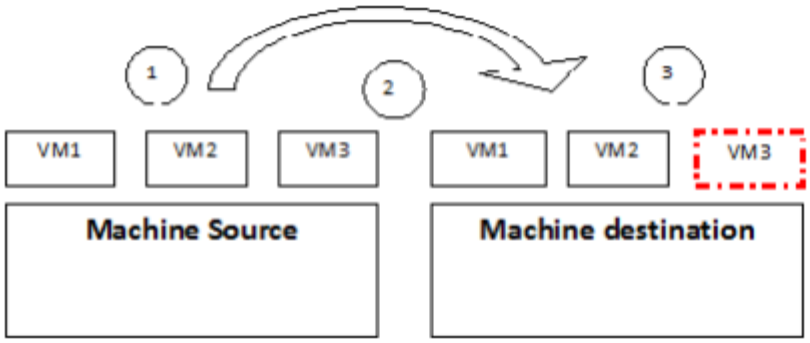


Figure 3-1: Processus de migration à froid

3.3.2 La migration à chaud [44]:

La migration à chaud (Live Migration) est un outil pour la migration des systèmes d'exploitation complets ainsi que leurs applications entre les machines physiques éloignées des Data Center et des Clusters. C'est une technologie qui facilite l'équilibrage de charge, la gestion des pannes, la maintenance du système et la réduction de la consommation d'énergie.

La migration à chaud des machines virtuelles est le processus de transférer une ou plusieurs machines virtuelles d'une machine physique ou d'un espace de stockage vers un autre sans interrompre les autres VMs.

Cette technique est utile dans plusieurs scénarios, parmi lesquels nous avons :

- ✓ Une VM dans un nœud physique défaillant peut être migrée vers un autre nœud non défaillant.
- ✓ Les VMs au mode repos sur une machine physique peuvent être migrées vers d'autres hôtes pour optimiser l'utilisation des ressources.

✓ Les VMs sur un nœud physique chargé peuvent être migrés vers différents nœuds pour équilibrer les charges.

❖ Les techniques de migration à chaud sont caractérisées par 2 approches basées sur le processus de copie de mémoire. Ces technique sont “Post-Copy“ et “Pre-Copy“.

A. Live Migration ‘’post-copy’’ :

Dans cette technique, dans un premier temps la VM est suspendu dans une machine source et son état d’exécution (CPU, les registres et les pages mémoires nécessaire pour activer la VM dans la machine destination) est transféré et copié vers la machine destination. Ensuite, les pages mémoire sont copiées. La VM commence son exécution sur la machine destination même si les pages mémoires n’ont pas été copiées entièrement. La post-copie est plus optimisée pour les applications nécessitant beaucoup d'écritures en mémoire. L’inconvénient de cette méthode est l’apparition des défauts de pages mémoires de la machine destination. [52]

La figure 3-2 illustre le principe de fonctionnement de la migration Post-Copy. Nous prenons par exemple la migration de la VM3 :

1. La VM3 est suspendue dans la mémoire source.
2. L’état d’exécution de la VM3 est transféré vers la machine destination.
3. La VM3 est activée dans la machine destination.
4. Générer les défauts de pages mémoires.
5. Transférer les pages à défaut correspondantes.
6. Les étapes 4 et 5 sont répétées jusqu’à ce que toutes les pages mémoires sont transférées vers la machine destination et la VM3 s’exécute.

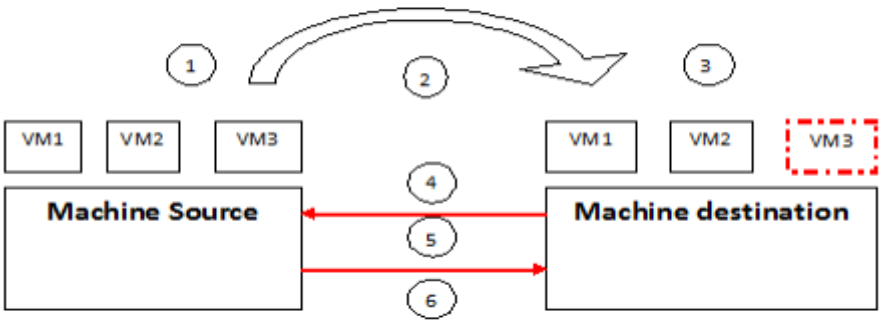


Figure 3-2: Processus de migration post-copy

B. Live Migration ‘’pre-copy’’ :

C’est une technique de la migration à chaud qui est adopté par des environnements de virtualisation. Cette technique commence par copier l’état complet de la machine virtuelle source vers la destination système. Pendant la copie le système source reste réactif, et continue à progresser toutes les applications. Les pages peuvent être mise à jour après même qu’elle soit copié dans la destination. L’approche emploie des mécanismes pour surveiller les pages mises à jour puis la recopie dans la destination.

La Figure 3-3 illustre le principe de la migration Pre-Copy de la VM3 en 5 étapes :

1. Envoyer toutes les pages mémoire de la VM3 de la machine source vers la machine destination.
2. Transférer les pages mémoire modifiées d’une manière itérative.
3. Arrêter la VM3 dans la machine source.
4. Envoyer les états restants de la VM3.
5. Activer la VM3 dans la machine destination.

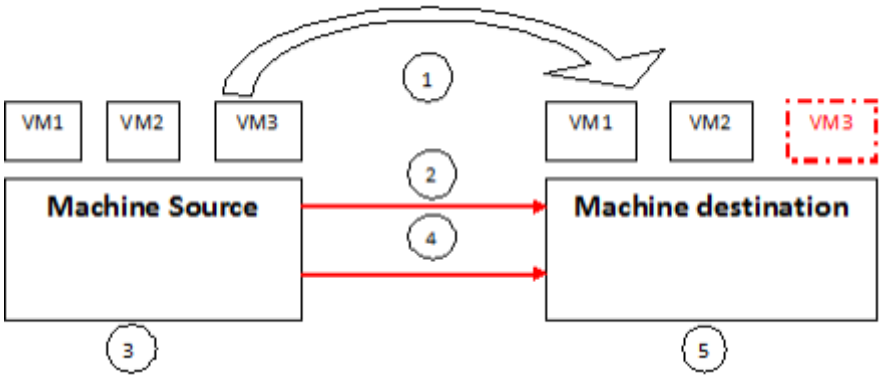


Figure 3-3: Processus de la migration Pre-Copy

En cas de panne de la machine destination, la technique Pre-copy est bien meilleure que la Post-copy. La récupération des VM3 est possible par la méthode Pre-copy car la dernière copie de mémoire est conservée dans un lieu sûr sur la machine source y compris l'état du CPU.

❖ Les migrations à chaud peuvent être aussi classées plus loin en fonction de leur traitement du stockage d'instance [45]:

- **Live migration basée sur le stockage partagé** : L'instance contient des disques éphémères situés sur un stockage partagé entre les hôtes source et cible.
- **Block live migration** : ou simplement la migration. L'instance contient des disques éphémères qui ne sont pas partagés entre les hôtes source et cible. La migration de bloc est incompatible avec les périphériques en lecture seule tels que les CD-ROM et Configuration Drive (config_drive).
- **Live migration basée sur le volume** : Les instances utilisent des volumes plutôt que des disques éphémères.

Bloquer la migration en direct nécessite la copie de disques de la source vers l'hôte de destination. Cela prend plus de temps et met plus de charge sur le réseau. La migration dynamique à stockage partagé et à sauvegarde sur volume ne copie pas les disques.

❖ Les mesures suivantes sont généralement utilisées pour mesurer la performance de la migration à chaud :

✓ **Préparation :** Dans cette phase, les ressources appropriées sont réservées sur la machine destination et diverses opérations sont effectuées sur la machine source.

✓ **Temps d'arrêt :** c'est le temps où la machine virtuelle est suspendue sur l'hôte source.

✓ **Temps résumé :** cette étape implique l'installation de la VM sur l'hôte destination avec le même état que la VM suspendue. Le temps d'achèvement de cette phase varie selon différentes approches.

✓ **Temps total de migration :** le temps total après l'accomplissement de toutes ces phases est appelé temps de migration totale.

3.3.3 Avantage de la Migration :

Les avantages de la migration des machines virtuelles sont les suivants :

- **Tolérance aux pannes** : la tolérance aux pannes permet à la machine virtuelle de continuer ses tâches même si une partie quelconque de système échoue.

La technique de migration à tolérances aux pannes permet d'améliorer la disponibilité du serveur physique et évite la dégradation des performances des applications.

- **Equilibrage de charge** : la technique de migration d'équilibrage de charge vise à répartir la charge de travail sur les serveurs physiques pour améliorer l'évolution des serveurs physique dans un environnement du Cloud.
- **Réduction de la consommation d'énergie :** la consommation d'énergie des Data center est principalement basée sur l'utilisation des serveurs et leurs systèmes de refroidissement. Ces serveurs utilise jusqu'à 70% de leur consommation d'énergie maximale, même avec une faible utilisation des ressources. Par conséquent, il faut utiliser des techniques de migration qui conservent l'énergie consommée par les serveurs par une utilisation optimale des ressources.

3.4 Outils utilisés :

3.4.1 OpenStack :

3.4.1.1 Définition :

OpenStack est un ensemble d'outils logiciels pour la construction et la gestion de plates-formes de Cloud Computing pour les nuages publics et privés. Soutenu par les grandes entreprises dans le développement et l'hébergement de logiciels, ainsi que des milliers de membres de communauté, beaucoup pense qu'OpenStack est l'avenir du Cloud Computing.

Les objectifs d'OpenStack consistent à aider les organisations à mettre en œuvre un système de serveur et de stockage virtuel et prendre en charge l'interopérabilité entre différents services Cloud.

Et surtout OpenStack est un logiciel open source, ce qui signifie que toute personne qui choisit d'accéder au code source, effectue des modifications dont il a besoin et partage librement ces changements dans la communauté. [36]

3.4.1.2 composants d'OpenStack :

A. NOVA :

Nova est le principal moteur informatique derrière OpenStack. Il est utilisé pour déployer et gérer un grand nombre de machines virtuelles et d'autres instances afin de gérer des tâches informatiques.

Architecture du service nova :

Un système Nova de base comprend les composants suivants:

- **Le service nova-api** qui fournit aux utilisateurs l'API REST externe.
- **Nova-scheduler** responsables du suivi des ressources et décide quel hôte reçoit chaque instance.
- Une «**base de données API**» utilisée principalement par nova-api et nova-scheduler pour suivre les informations d'emplacement relatives aux instances, ainsi qu'un emplacement temporaire pour les instances en cours de création mais non encore planifiées.
- **Le service nova-conductor** qui décharge les tâches de longue durée pour le service de niveau et joue le rôle de proxy de base de données ou gère les conversions d'objet.
- **Le service nova-compute** qui gère le pilote virtuel et l'hôte de l'hyperviseur.
- Une «base de données de cellules» utilisée par les services de l'API, du conducteur et du calcul, et contenant la majorité des informations sur les instances.
- Une «**base de données cell0**» qui ressemble à la base de données de cellules, mais ne contient que les instances dont la planification a échoué.
- **Une file de messages** qui permet aux services de communiquer entre eux via RPC.

B. Cinder [42] :

C'est un service de stockage persistant en mode bloc, accessible via une API en libre-service.

Architecture de cinder :

Architecture de cinder est composée des elements suivants :

- **Cinder-api :** c'est une interface pour cendre.
- **Cinder-scheduler** :

  Prend la demande du service api.

  Fonctionne avec le service de volume pour satisfaire la demande.

  Fonctionne généralement sur le nœud du contrôleur.
- **Cinder-volume driver** :

Interagir avec les systèmes de stockage des fournisseurs.

Peut fonctionner sur un nœud de contrôleur

Parfois déchargé sur un hôte différent

- **Cinder-backup :**

Interface pour les volumes de sauvegarde sur le stockage comme TSM, Google Cloud Storage.

Capable de passer à plusieurs nœuds pour une opération de simulation.

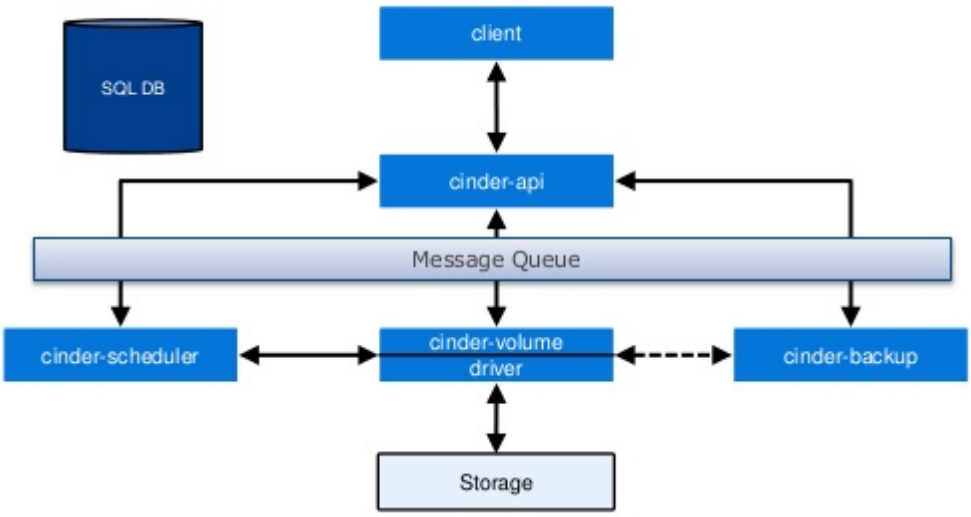


Figure 3-4 Architecture de cinder.

C. Keystone :

Keystone est le principal outil d'authentification des utilisateurs et de contrôle d'accès basé sur les rôles. Il agit comme un répertoire central de tous les utilisateurs et de leurs autorisations respectives, il prend également en charge les connexions par nom d'utilisateur et mot de passe traditionnels.

Architecture de keystone :

Keystone est organisé en groupe de services exposés aux ordinateurs d'extrémité. Il y a 6 services:

- Identity : valide les informations d'identification des utilisateurs.
- Ressource: fournit des données sur les projets.
- Assignment: fournit des données sur les rôles et les attributions de rôles entre les objets de Services d'identité et de ressources.
- Token: valide et gère les Tokens.
- Catalog: fournit un registre de points de terminaison.

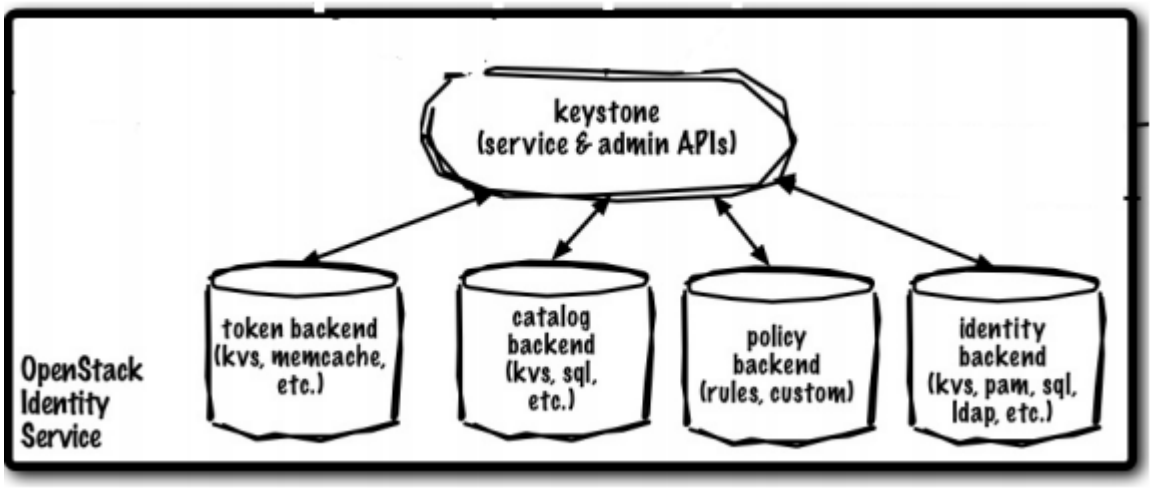


Figure 3-5: Architecture de Keystone

D. Glance :

Fournit des services d'image à OpenStack. Dans ce cas, "images" fait référence aux images (ou copies virtuelles) de disques durs. Glance permet à ces images d'être utilisées comme modèles lors du déploiement de nouvelles instances de machine virtuelle.

Architecture de Glance :

Glance a les composants suivants:**[43]**

- **Glance-api ,** qui accepte les appels d'API pour la découverte, la récupération et le stockage d'images.
- **Glance-registry** , qui stocke, traite et récupère les informations de métadonnées pour les images.
- **Base de données**, qui stocke le**référentiel de stockage des** métadonnées de l'image, qui s'intègre à divers composants externes à OpenStack tels que les systèmes de fichiers classiques, Amazon S3 et HTTP pour le stockage des images.

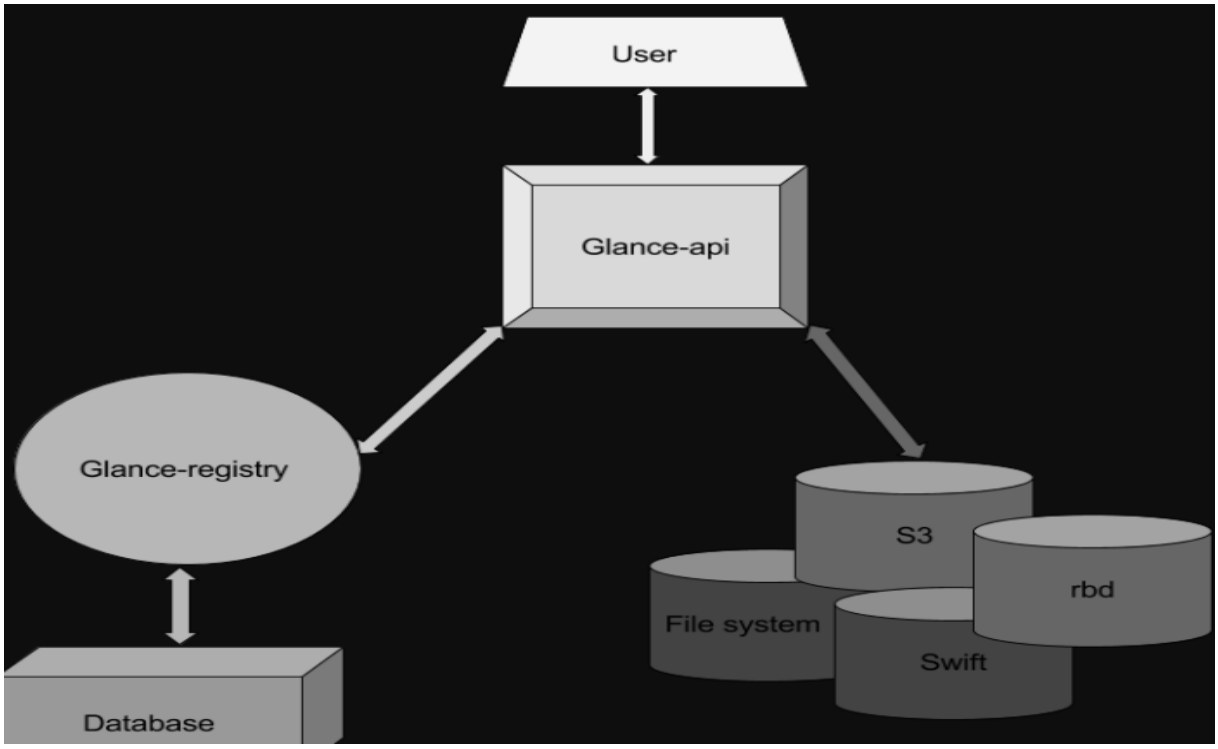


Figure 3-6 Architecture de glance.

D. Neutron :

Gère tous les réseaux associés aux Clouds OpenStack. Il comprend des API qui permettent aux administrateurs de personnaliser et de déployer divers types de réseaux tels que les réseaux plats, les VLAN ou les VPN. Neutron donne le choix à l'utilisateur d'OpenStack de créer au choix deux options de réseau :

- Option de mise en réseau 1: réseaux de fournisseur

L'option de réseau de fournisseur déploie le service de réseau OpenStack de la manière simple Il relie essentiellement les réseaux virtuels aux réseaux physiques et s'appuie sur l'infrastructure de réseau physique pour les services de couche 3 (routage). En outre, un service DHCP fournit des informations d'adresse IP aux instances.

- Option de mise en réseau 2: réseaux en libre-service

L'option de réseaux en libre-service complète l'option de réseaux de fournisseurs avec les services de couche 3 (routage) qui activent les réseaux en libre-service à l' aide de méthodes de segmentation en superposition telles que VXLAN. En plus clair on peut créer des réseaux personnalise ensuite il route virtuels vers les réseaux physiques en utilisant NAT .

Architecture de neutron : [41]

L'architecture Neutron a été présentée dans le schéma suivant:

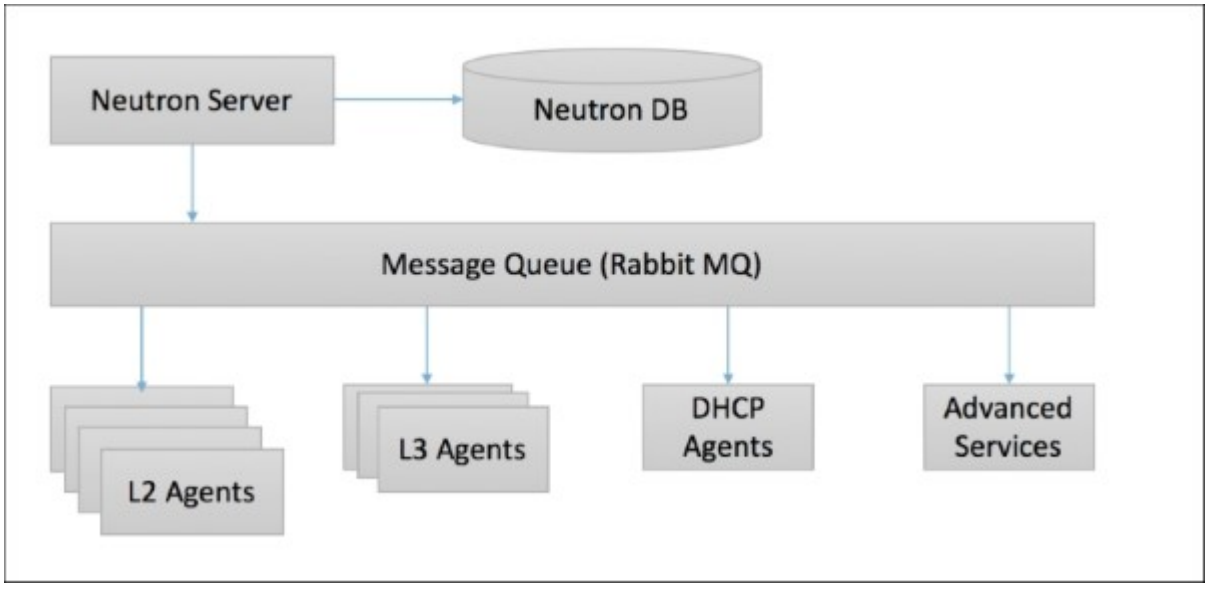


Figure 3-7: Architecture de neutron.

Le serveur Neutron

La fonction de ce composant est d'être le visage de l'environnement Neutron dans son ensemble. Il est essentiellement composé de trois modules:

Service REST : le service REST accepte les demandes d'API des autres composants et expose tous les détails de fonctionnement internes en termes de réseaux, sous-réseaux, ports, etc. C'est une application WSGI écrite en Python qui utilise le port 9696 pour la communication.

Service RPC : le service RPC communique avec le bus de messagerie et sa fonction est d'activer une communication d'agent bidirectionnel.

Plugin : un plugin se décrit mieux comme une collection de modules Python qui implémente une interface standard, qui accepte et reçoit certains appels d'API standard, puis se connecte aux périphériques en aval. Ils peuvent être de simples plugins ou implémenter des pilotes pour plusieurs classes de périphériques.

- *Agent L2*

L'agent L2 s'exécute sur l'hyperviseur (nœuds de calcul) et sa fonction consiste simplement à connecter de nouveaux périphériques, ce qui signifie qu'il fournit des connexions à de nouveaux serveurs dans des segments de réseau appropriés et qu'il envoie également des notifications lorsqu'un périphérique est connecté ou supprimé. Dans notre installation, nous utiliserons l'agent OVS.

- *Agent L3*

Les agents L3 s'exécutent sur le nœud de réseau et sont responsables du routage statique, du transfert IP et d'autres fonctionnalités de L3, telles que DHCP.

Processus de base de Neutron :

Les étapes de lancement de nouvelle instance avec neutron sont la suivante :

a) Démarrage de la machine virtuelle.
b) Créez un port et informez le DHCP du nouveau port.
c) Créez un nouveau périphérique (bibliothèque de virtualisation - libvirt).
d) Port fil (connectez la machine virtuelle au nouveau port).
e) Démarrage complet.

f) Horizon : C'est le seul composant graphique d'OpenStack. Il s'agit essentiellement d'un outil Web utilisé par les utilisateurs et les administrateurs pour accéder aux autres composants. Beaucoup l'appellent le tableau de bord d'OpenStack.

La figure suivante illustre les services de bases d'OpenStack :

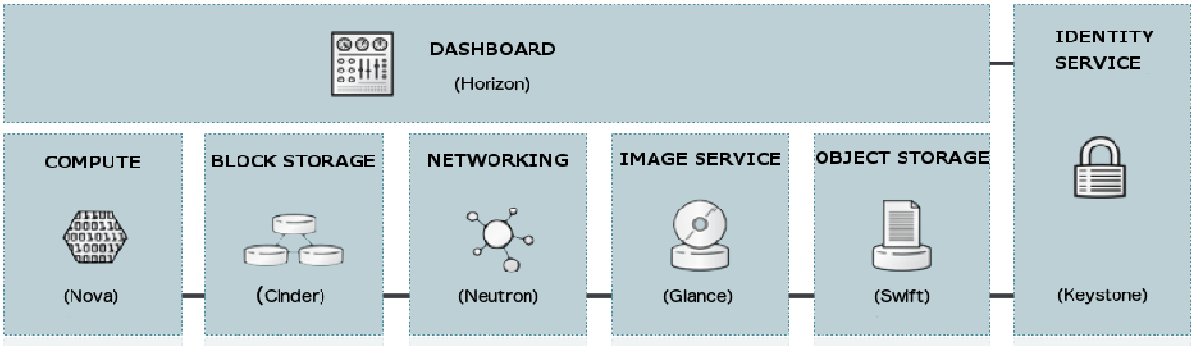


Figure 3-8 : Architecture des services de base d'OpenStack.

3.4.1.3 Architecture minimale :

L'architecture minimale requiert au moins deux nœuds (hôtes) pour lancer une machine virtuelle ou une instance de base. Ces hôtes sont un nœud de contrôle et un nœud de calcul. Les services optionnels tels que le stockage de blocs et le stockage d'objets nécessitent des nœuds supplémentaires. [39]

a. Controller (contrôleur) :

Le nœud du contrôleur exécute le service d'identité, le service d'image, les partie de gestion de Compute, la partie de gestion du réseau, les différents agents de mise en réseau et le tableau de bord. Il inclut également des services de support tels qu'une base de données SQL, une file d'attente de messages et NTP.

En option, le nœud du contrôleur exécute des portions des services de stockage de blocs, de stockage d'objets, d'orchestration.

Le nœud du contrôleur nécessite au minimum deux interfaces réseau.

b. Compute (calcule) :

Le nœud Compute exécute la partie hyperviseur de Compute qui fait fonctionner les instances. Par défaut, Compute utilise l'hyperviseur KVM ou QEMU. Le nœud Compute héberge également un agent du service Réseau qui connecte les instances aux réseaux virtuels et fournit des services de firewall aux instances via les groupes de sécurité. Vous pouvez déployer plus d'un nœud Compute. Chaque nœud nécessite au minimum deux interfaces réseau.

c. Stockage par Bloc (cinder)

Le nœud optionnel de Stockage par Blocs contient les disques que les services de Stockage par Blocs et de Systèmes de Fichiers Partagés provisionnent pour les instances.

Pour simplifier, le trafic du service entre les nœuds compute et ce nœud utilise le réseau de management. Les environnements de production devraient implémenter un réseau de stockage séparé pour accroitre la performance et la sécurité.

Vous pouvez déployer plus d'un nœud stockage. Chaque nœud nécessite au minimum une interface réseau.

d. Stockage Objet (swift)

Le nœud optionnel de Stockage Objet contient les disques que le service de Stockage Objet utilise pour stocker les comptes, les conteneurs et les objets.

Pour simplifier, le trafic du service entre les nœuds compute et ce nœud utilise le réseau de management. Les environnements de production devraient implémenter un réseau de stockage séparé pour accroitre la performance et la sécurité.

Ce service nécessite deux nœuds. Chaque nœud doit avoir au minimum une interface réseau. Vous pouvez déployer plus de deux nœuds de stockage objet.

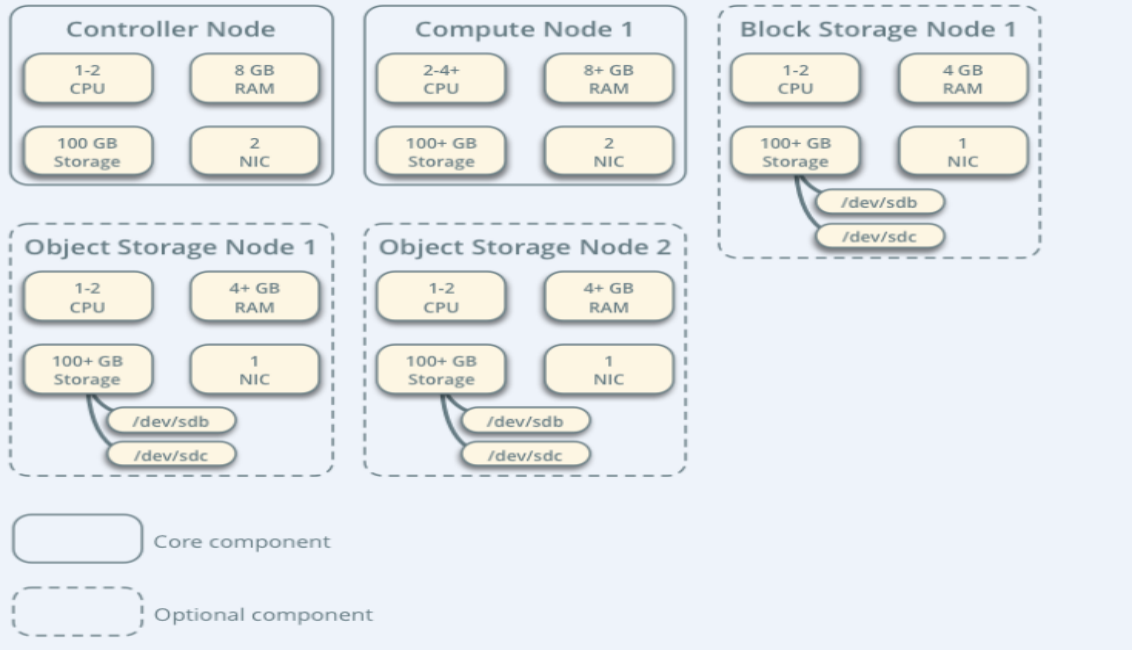


Figure 3-9 Architecture minimale openstack.

3.4.2 MTR :

C'est est un logiciel de réseau qui combine les fonctionnalités du traceroute et du ping, il est utilisé pour le diagnostic réseau.

MTR sonde les routeurs sur le chemin de la route en limitant le nombre de sauts de paquets individuel en écoutant les réponses de leur expiration. Habituellement, il est répété une fois par seconde, et garde la trace des temps de réponse des sauts le long du chemin. Pour un cas d'utilisation de migration en direct, le paramètre le plus important peut être mesuré par cet outil et les paquets perdus, ce paramètre sera augmenté en cas de panne de la machine virtuelle lors de la migration. [49]

3.5 Lancement d'une instance sur OpenStack :

La mise en service d'une nouvelle instance implique l'interaction de plusieurs composants dans OpenStack:

- **Interpréteur de ligne de commande CLI** pour la soumission de commandes à OpenStack Compute.
- **Le tableau de bord (Horizon)** fournit l'interface pour tous les services OpenStack.

- **Compute (Nova)** récupère les images de disques virtuels («Glance»), attache une saveur et les métadonnées associées et transforme les demandes d'API de l'utilisateur final en instances en cours d'exécution.
- **Neutron (Quantum)** fournit à Compute un réseau virtuel qui permet aux utilisateurs de créer leurs propres réseaux, puis de les lier aux instances.
- **Block Storage (Cinder)** fournit des volumes de stockage persistants pour les instances Compute.
- **Image (Glance)** peut stocker les fichiers de disque virtuel dans le magasin d'images.
- **Identity (Keystone)** fournit une authentification et une autorisation pour tous les services OpenStack.
- **Message Queue (RabbitMQ)** gère la communication interne au sein de composants Openstack tels que Nova, Quantum et Cinder.

Le processus de Création d'instance OpenStack :

Jusqu'à présent nous n'avons pas encore expliqué le mécanisme de la création d'une machine virtuelle à l'aide d'OpenStack. La partie suivante met en évidence les étapes nécessaires pour approvisionner une instance.

Le processus d'approvisionnement, comprennent de nombreuses étapes afin de construire une instance (voir Figure 3-10). Pendant le processus, le nœud du contrôleur

envoie différentes demandes dans un ordre chronologique, commençant par la vérification des identifiants, terminant par la création des instances sur le nœud compute et passant par la communication entre les composants (nova, Glance et Neutron) via le protocole de communication/messagerie RabbitMQ.

Dans ce qui suit, nous allons décrire et expliquer chaque transaction (numérotées de 1 à 28) dans Figure 3-10.Ci-dessus:[55]

Etape1 : Le tableau de bord ou la CLI obtient les informations d'identification de l'utilisateur et effectue l'appel REST vers Keystone pour l'authentification.

Etape2 : Keystone authentifie les informations d'identification et génère & renvoie des jetons d'authentification qui seront utilisés pour envoyer une demande à d'autres composants via un appel REST.

Etape3 : Dashboard ou CLI convertit la demande de nouvelle instance spécifiée dans le formulaire 'instance de lancement' ou 'nova-boot' en demande d'API REST et l'envoie à nova-api.

Etape4 : nova-api reçoit la demande et envoie la demande de validation, jeton automatique et autorisation d'accès à Keystone .

Etape5 : Keystone valide le jeton et envoie les en-têtes d'authentification mis à jour avec les rôles et les autorisations.

Etape6 : nova-api interagit avec nova-database .

Etape7 : Crée une entrée de base de données initiale pour la nouvelle instance.

Etape8 : nova-api envoie la demande rpc.call à nova-scheduler pour obtenir une entrée d'instance mise à jour avec l'ID d'hôte spécifié.

Etape9 : nova-scheduler sélectionne la demande dans la file d'attente.

Etape10 : nova-scheduler interagit avec nova-database pour trouver un hôte approprié via le filtrage et la pesée.

Etape11 : Renvoie l'entrée d'instance mise à jour avec l'ID d'hôte approprié après filtrage et pesage.

Etape12 : nova-scheduler envoie la demande rpc.cast à nova-compute pour "l'instance de lancement" sur l'hôte approprié.

Etape13 : nova-compute sélectionne la demande dans la file d'attente.

Etape14 : nova-compute envoie la demande rpc.call à nova-conductor afin de récupérer les informations sur l'instance, telles que l' ID d'hôte et la variante (Ram, CPU, Disk).

Etape15 : nova-conductor sélectionne la demande dans la file d'attente.

Etape16 : nova-conductor interagit avec nova-database.

Etape17 : Renvoie les informations sur l'instance.

Etape18 : nova-compute sélectionne les informations d'instance dans la file d'attente.

Etape19 : nova-compute effectue l'appel REST en passant auth-token à glance-api pour obtenir l'URI Image par ID d'œil et télécharger l'image à partir du stockage d'images.

Etape20 : Glance-Api valide l'authentification-jeton avec Keystone.

Etape21 : nova-compute récupère les métadonnées de l'image.

Etape22 : nova-compute effectue l'appel REST en transmettant l'authentification par jeton à l'API réseau pour allouer et configurer le réseau de sorte que l'instance obtienne l'adresse IP.

Etape23 : quantum-server valide l'authentification -token avec keystone.

Etape24 : nova-compute récupère les informations du réseau.

Etape25 : nova-compute effectue l'appel REST en transmettant auth -token à l'API Volume pour attacher des volumes à l'instance.

Etape26 :cinder-api valide l'authentification-jeton avec keystone .

Etape27 : nova-compute obtient les informations de stockage de bloc.

Etape28 : nova-compute génère des données pour le pilote d'hyperviseur et exécute la requête sur l'hyperviseur (via libvirt ou api).

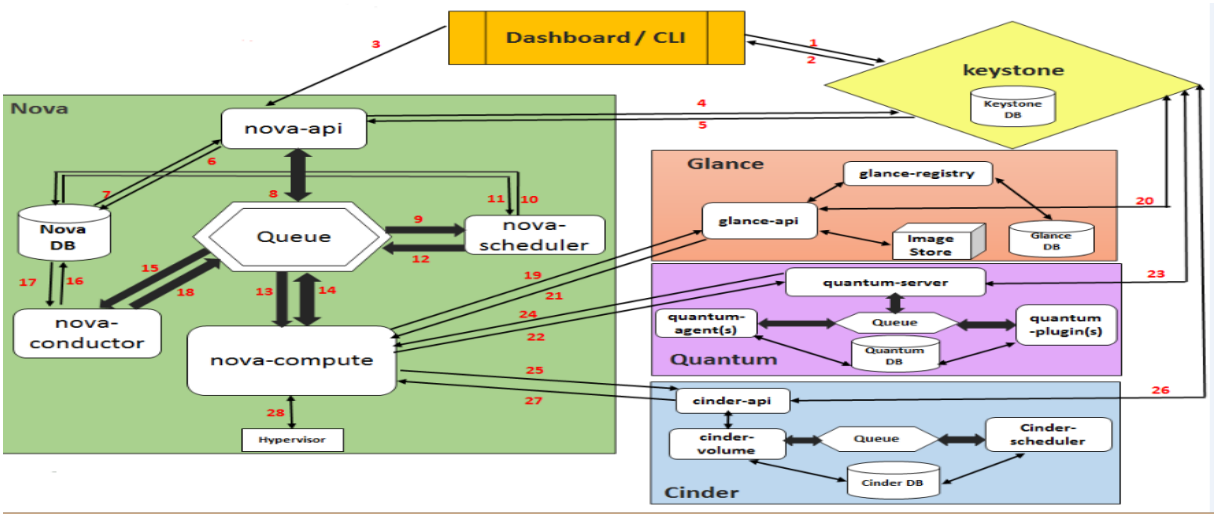


Figure 3-10 : Les étapes de provisionnement d'une machine virtuelle.

3.6 Principe de QEMU/ KVM dans OpenStack [47] :

La migration inclut la migration du système dans son ensemble et la migration d'une charge de travail. La défragmentation du système consiste à copier complètement tous les logiciels du système, y compris le système d'exploitation, sur une autre machine matérielle physique. La migration dans un environnement vitalisé peut être divisée en migration statique (migration froide ou migration hors ligne froide) et en migration dynamique (migration en direct, migration à chaud ou migration en ligne). La plus grande différence entre migration statique et migration dynamique réside dans le fait que la migration statique

a une période de temps considérable pendant laquelle le service dans le client est indisponible, tandis que la migration dynamique n'a pas de temps de pause de service évident.

3.6.1 La migration à froid :

La migration à froid ne nécessite pas que les nœuds source et de destination partagent le stockage, bien que le stockage partagé soit également possible. Une condition doit être remplie avant de migrer: les utilisateurs nova doivent être configurés sans accès par mot de passe entre les nœuds de calcul.

Ci-dessous, l'organigramme de l'instance Migrate [1] :

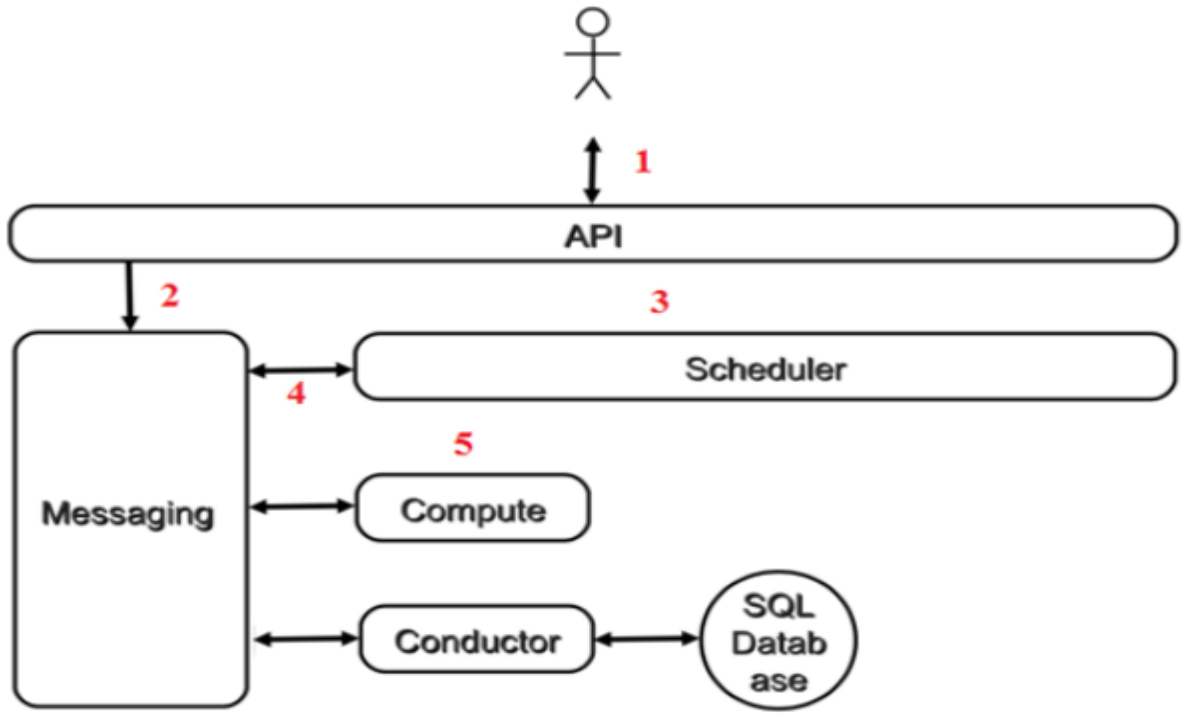


Figure 3-11 : Le processus de la migration à froid.

Les étapes de la migration à froid sont :

1. Envoyez une demande à nova-api.
2. Nova-api envoyer un message.
3. Planification de l'exécution du planificateur Nova.
4. Nova-scheduler Envoyer un message.

5. Nova-Compute.

❖ **Envoyez une demande à nova-api :**

Le client (utilisateur final OpenStack ou autre programme) envoie une requête à l'API (nova-api): "Migrer cette instance pour moi" .L'opération de migration est une opération privilégiée et ne peut être exécutée que dans l'instance d'Admin.

❖ **Nova-api envoyer un message :**

Nova-api a envoyé un message à Messaging (RabbitMQ): "Migrer cette instance".

❖ **Planification de l'exécution du planificateur Nova :**

Lorsque nova-scheduler reçoit le message, il sélectionne le nœud de traitement cible approprié pour l'instance.

❖ **Nova-scheduler Envoyer un message :**

Nova-scheduler envoie un message informant le nœud de calcul que l'instance peut être migrée.

❖ **Nova-Compute** :

Nova-compute effectue des opérations sur le nœud de calcul source et le nœud de calcul cible.

❖ **Nœud de calcul source compute 1 :**

Nova-compute du nœud source accède au nœud destinataire via une connexion SSH. Ensuite, l'étape suivante consiste à créer une instance du répertoire sur le nœud cible. Après avoir créé l'instance dans le nœud cible elle sera fermée dans le nœud source, ensuite le fichier d'image d'instance sera transmet au nœud cible via **SCP**. Enfin l'instance va être démarrée sur le nœud cible, qui est très similaire à l'instance de lancement.

Le lancement de cette instance passera par les étapes suivantes:

1. Préparez les ressources processeur, mémoire et disque pour l'instance.
2. Créer un fichier image d'instance.
3. Créez un fichier de définition XML par exemple.
4. Créez un réseau virtuel et démarrez une instance.

Apres l'instance passe à l'état "Confirmer ou annuler le redimensionnement / la migration", obligeant l'utilisateur à confirmer ou à annuler l'opération de migration en cours.

Si la décision de confirmation est prise, les événements suivants se produisent :

1. Nova-api a reçu un message de confirmation.
2. Le nœud de calcul source supprime le répertoire d'instance et supprime l'instance sur l'hyperviseur.
3. Le nœud de calcul cible n'a rien à faire.

Sinon la décision « annuler » est prise, alors événements suivants se produisent :

1. Nova-api a reçu un message pour revenir.
2. Fermer l'instance sur le nœud de traitement cible, supprimer le répertoire d'instance et supprimer l'instance sur l'hyperviseur.
3. Démarrer l'instance sur le nœud de calcul source, étant donné que la migration précédente ne fermait que l'instance sur le nœud source, l'opération de réinitialisation n'a besoin que de redémarrer l'instance.

3.6.2 La migration à chaud :

Les nœuds source et de destination doivent remplir certaines conditions pour prendre en charge Live Migration:

- Les nœuds source et de destination doivent avoir le même type de CPU.
- La version Libvirt des nœuds source et de destination doit être identique.

Les utilisateurs nova doivent être configurés sans accès par mot de passe entre les nœuds de calcul en utilisant SSH.

Processus de la migration à chaud sur OpenStack :

Les principales étapes de la migration en direct sont illustré dans ce diagramme de séquence : [56]

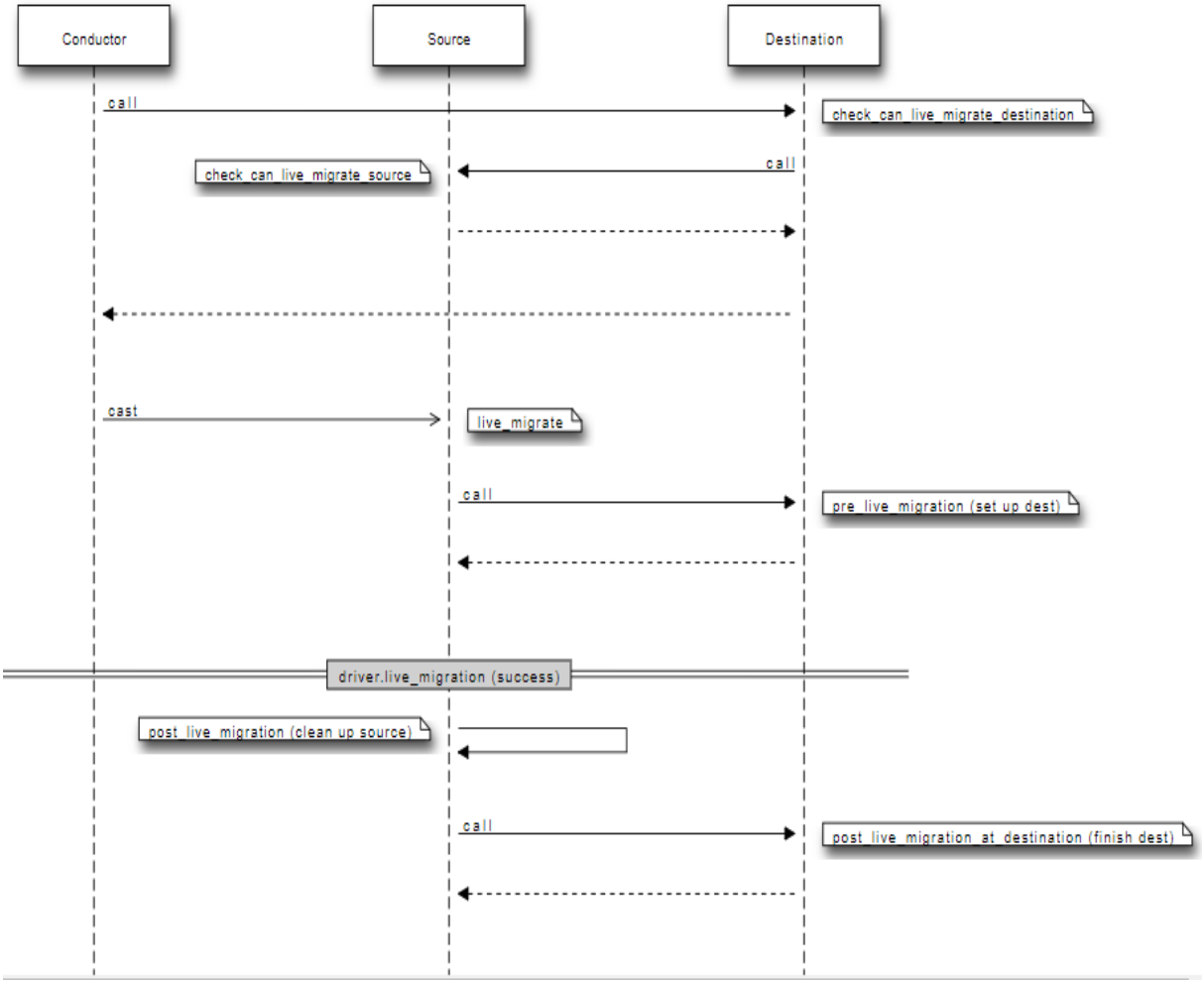


Figure 3-12 : Processus de migration à chaud.

1. Le nœud contrôleur demande au nœud de destination s'il peut faire la migration.
2. Le nœud de destination interroge le nœud source (sur les caractéristiques de la source).
3. Le nœud source répond au nœud destinataire.
4. Le nœud destinataire répond au nœud contrôleur (service nova-conductor)
5. Le nœud contrôleur demande au nœud source de commencer la migration.
6. Nova compute du nœud source appelle la méthode pre_live migration du gestionnaire nova Compute sur l'hôte de destination :

- Préparer le répertoire d'instance partagé.
- Etablir la connexion entre les nœuds de calcul via SSH.

7. Les opérations de post live migration sont effectuées sur l'hôte source :

- Démarrer la migration de l'hôte source vers l'hôte destination.
- Une fois la migration terminée : déconnecter les volumes (déconnecter le domaine et toute connexion de volume, supprimer les fichiers de domaine)
- Supprimer les réseaux sur l'hôte source.

8. La méthode post-live-migration de nova sur l'hôte destination est appelée
    - Configurer le réseau (gestion du VPN, du DHCP, de la passerelle)
    - Appeler la méthode migrate-instance-finish.
    - Mettre à jour les nouvelles données de domaine dans la base de données
9. A ce stade la migration en direct est terminée.

3.7 Conclusion :

Dans ce chapitre nous avons expliqué la méthode de la migration des machines virtuelles d'une manière générale et nous avons présenté l'outil choisi (OpenStack), et son fonctionnement et nous avons expliqué d'une manière détaillée la migration à froid et à chaud dans l'environnement OpenStack.

Dans le chapitre suivant, nous présenterons l'implémentation de notre solution proposée dans l'environnement OpenStack ainsi que quelques résultats de notre travail.

CHAPITRE 4

RÉALISATION

Chapitre4: Réalisation

4.1 Introduction :

Dans le chapitre précédent nous avons expliqué en détail le processus de migration des machines virtuelles, on a donné aussi un bref aperçu sur les outils utilisés.

Dans ce chapitre nous allons introduire les phases de réalisation de notre étude : la mise en place de l'environnement de travail, la migration des machines virtuelles sur OpenStack. Et enfin voir le résultat.

4.2 Installation d'OpenStack :

Voir annexe A page 91.

4.3 Création d'une instance OpenStack :

Se référer à l'annexe B page 121.

4.4 Migration des machines virtuelles :

4.4.1 Migration à froid :

Pour lancer une migration non direct il faut que les nœuds source et destination puissent effectuer une connexion secrète SSH. (Les étapes de la configuration de la clé secrète sont décrites en détail dans le point suivant.)

Avant de lancer une migration, il faut en premier lieu connaitre les instances qu'on a.

La figure 4.1 donne une liste des instances de tout notre cluster

```
root@controller:~# openstack server list
+--------------------------------------+-----------+--------+--------------------+--------+--------
-+
| ID                                   | Name      | Status | Networks           | Image  | Flavor
 |
+--------------------------------------+-----------+--------+--------------------+--------+--------
-+
| 8dc3411c-1ca2-448e-bdb3-4090d87d621b | instance1 | ACTIVE | provider=10.0.0.102 | cirros | m1.nano
 |
+--------------------------------------+-----------+--------+--------------------+--------+--------
```

Figure 4-1: Liste des instances du Cloud

On affiche ensuite, l'hôte sur lequel réside notre instance

```
root@controller:~# openstack server show 8dc3411c-1ca2-448e-bdb3-4090d87d621b
+-------------------------------------+-----------------------------------------------------------+
| Field                               | Value                                                     |
+-------------------------------------+-----------------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                                    |
| OS-EXT-AZ:availability_zone         | nova                                                      |
| OS-EXT-SRV-ATTR:host                | compute1                                                  |
| OS-EXT-SRV-ATTR:hypervisor_hostname | compute1                                                  |
| OS-EXT-SRV-ATTR:instance_name       | instance-0000002a                                         |
| OS-EXT-STS:power_state              | Running                                                   |
| OS-EXT-STS:task_state               | None                                                      |
| OS-EXT-STS:vm_state                 | active                                                    |
| OS-SRV-USG:launched_at              | 2019-08-31T17:36:07.000000                                |
| OS-SRV-USG:terminated_at            | None                                                      |
| accessIPv4                          |                                                           |
| accessIPv6                          |                                                           |
| addresses                           | provider=10.0.0.102                                       |
| config_drive                        |                                                           |
| created                             | 2019-08-31T17:34:40Z                                      |
| flavor                              | m1.nano (0)                                               |
| hostId                              | 15490061853220bba501219fa703d0d1db59218601fa845ee47fbbd8  |
| id                                  | 8dc3411c-1ca2-448e-bdb3-4090d87d621b                      |
| image                               | cirros (4a6cf4da-8c01-430a-86ab-c0c05f8471a2)             |
| key_name                            | mykey                                                     |
| name                                | instance1                                                 |
| progress                            | 0                                                         |
| project_id                          | bec9bd23182447399f3b6a013d613031                          |
| properties                          |                                                           |
| security_groups                     | name='default'                                            |
| status                              | ACTIVE                                                    |
| updated                             | 2019-08-31T17:36:08Z                                      |
| user_id                             | 08c43dcba2024b6aa5cd021e908def71                          |
| volumes_attached                    |                                                           |
+-------------------------------------+-----------------------------------------------------------+
```

Figure 4-2: Tableau d'information de l'instance à migrer

Comme cette figure 4.2 l'indique l'hôte réside sur le nœud compute1.

On va maintenant lancer la migration.

```
root@controller:~# openstack server migrate 8dc3411c-1ca2-448e-bdb3-4090d87d621b
```

Figure 4-3 : Commande de la migration à froid

Dans le nœud source on suit les fichiers log du service nova-compute

```
2019-08-31 17:42:21.472 2648 INFO nova.virt.libvirt.driver [req-67091e0c-10d6-4621-8a03-1f485e7c840e
 b714762d35854e4eb3daba3bd50b87ae 12ce3aeb6e824ddfb65bf79cecdd7fd7 - default default] [instance: 8dc
3411c-1ca2-448e-bdb3-4090d87d621b] Instance shutdown successfully after 3 seconds.
2019-08-31 17:42:21.479 2648 INFO nova.virt.libvirt.driver [-] [instance: 8dc3411c-1ca2-448e-bdb3-40
90d87d621b] Instance destroyed successfully.
2019-08-31 17:42:29.017 2648 WARNING nova.compute.manager [req-c05b586b-d3f7-456e-8311-cedcd5bc067c
0e300b5c2c824934b5404021a9c4945c ca1c277fc3ee4759893407352d527789 - default default] [instance: 8dc3
411c-1ca2-448e-bdb3-4090d87d621b] Received unexpected event network-vif-unplugged-04e30d17-8680-4a0c
-a067-1b4cb6604994 for instance with vm_state active and task_state resize_migrated.
```

Figure 4-4: Le fichier log de nova-compute de l'hôte source (migration à froid).

Dans le résultat de la commande tail du fichier log du service nova-compute, on peut repérer les points far qui indiquent que la migration s'est bien déroulée (ou non).

A la 3ème ligne on aperçoit que l'instance est arrêtée, après que toutes les pages mémoires soient copiées, l'instance est détruite et lancée dans l'autre nœud compute2. Après la migration l'état de la machine est VERIFY_RESIZE , si la migration échoue , on l'annule avec **nova resize-revert instance-id,** si la migration réussit, on la confirme avec **nova resize-confirm instance-id.**

La figure suivante montre que l'instance a bien changé d'hôte et son état est "active", donc on peut dire que la migration à froid est réussie.

```
root@controller:~# openstack server show 8dc3411c-1ca2-448e-bdb3-4090d87d621b
+-------------------------------------+-----------------------------------------------------------+
| Field                               | Value                                                     |
+-------------------------------------+-----------------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                                    |
| OS-EXT-AZ:availability_zone         | nova                                                      |
| OS-EXT-SRV-ATTR:host                | compute2                                                  |
| OS-EXT-SRV-ATTR:hypervisor_hostname | compute2                                                  |
| OS-EXT-SRV-ATTR:instance_name       | instance-0000002a                                         |
| OS-EXT-STS:power_state              | Running                                                   |
| OS-EXT-STS:task_state               | None                                                      |
| OS-EXT-STS:vm_state                 | active                                                    |
| OS-SRV-USG:launched_at              | 2019-08-31T17:42:36.000000                                |
| OS-SRV-USG:terminated_at            | None                                                      |
| accessIPv4                          |                                                           |
| accessIPv6                          |                                                           |
| addresses                           | provider=10.0.0.102                                       |
| config_drive                        |                                                           |
| created                             | 2019-08-31T17:34:40Z                                      |
| flavor                              | m1.nano (0)                                               |
| hostId                              | 7c1385eda2303f8c63e0e7cdbf6b559133233b4d0dbc1fe5b85e4f97  |
| id                                  | 8dc3411c-1ca2-448e-bdb3-4090d87d621b                      |
| image                               | cirros (4a6cf4da-8c01-430a-86ab-c0c05f8471a2)             |
| key_name                            | mykey                                                     |
| name                                | instance1                                                 |
| progress                            | 0                                                         |
| project_id                          | bec9bd23182447399f3b6a013d613031                          |
| properties                          |                                                           |
| security_groups                     | name='default'                                            |
| status                              | ACTIVE                                                    |
| updated                             | 2019-08-31T17:46:48Z                                      |
| user_id                             | 08c43dcba2024b6aa5cd021e908def71                          |
| volumes_attached                    |                                                           |
+-------------------------------------+-----------------------------------------------------------+
```

Figure 4-5: Confirmation du déplacement de l'instance.

4.4.2 Migration à chaud :

Avant de déplacer d'une manière directe une instance d'un ordinateur hôte à un autre, il est impératif de faire les configurations requises pour activer la migration en direct.

4.4.2.1 configurations la migration dynamique :

A. Créer une clé SSH entre les nœuds de traitement :

Chaque nœud doit être configuré avec une authentification SSH afin que le service de traitement puisse utiliser SSH pour déplacer des disques vers d'autre nœud.

1. Créer une clé SSH dans chacun des nœuds de traitement

```
root@compute1:~# ssh-keygen
Generating public/private rsa key pair.
Enter file in which to save the key (/root/.ssh/id_rsa):
Enter passphrase (empty for no passphrase):
Enter same passphrase again:
Your identification has been saved in /root/.ssh/id_rsa.
Your public key has been saved in /root/.ssh/id_rsa.pub.
The key fingerprint is:
SHA256:Rd3FLjV39wZUmMG9aji+trUgRkW4SOT8ILYnmVcqsms root@compute1
The key's randomart image is:
+---[RSA 2048]----+
|      ..  oo +oOo|
|      o. o. . *.B|
|     o.+..o.   +B|
|    . =.=o.   ..+|
|   . * +So  . .o |
|    o = .  o o   |
|   .     o..o.   |
|  E.    . .oo .  |
|  ..      .oo.   |
+----[SHA256]-----+
```

Figure 4-6: Clé SSH entre les nœuds de calcul.

2. Accéder au fichier **/etc/ssh/sshd_config**et ajouter **PermitRootLoginyes** , cette commande permet un accès à ce nœud. (refaire cette étape sur l'autre nœud de calcul).

```
  GNU nano 2.9.3                                  sshd_config

#ListenAddress ::

#HostKey /etc/ssh/ssh_host_rsa_key
#HostKey /etc/ssh/ssh_host_ecdsa_key
#HostKey /etc/ssh/ssh_host_ed25519_key

# Ciphers and keying
#RekeyLimit default none

# Logging
#SyslogFacility AUTH
#LogLevel INFO

# Authentication:

#LoginGraceTime 2m
PermitRootLogin yes
```

3. A partir de l'autre nœud de traitement, on tape la commande sshroot@ip du nœud, pour vérifier si on peut accéder à ce nœud.

```
root@compute2:~# ssh root@10.0.0.32
root@10.0.0.32's password:
Welcome to Ubuntu 18.04.2 LTS (GNU/Linux 4.15.0-55-generic x86_64)

 * Documentation:  https://help.ubuntu.com
 * Management:     https://landscape.canonical.com
 * Support:        https://ubuntu.com/advantage

 System information disabled due to load higher than 1.0


22 packages can be updated.
0 updates are security updates.

Failed to connect to https://changelogs.ubuntu.com/meta-release-lts. Check your Internet connection
or proxy settings


Last login: Tue Aug 27 18:17:34 2019 from 10.0.0.33
root@compute1:~# _
```

Du nœud nommé compute2 on a eu accès au nœud compute1 (refaire l'étape sur le nœud compute1)

4. Créer une autre clé SSH pour l'utilisateur nova du nœud source et destination car c'est le service nova qui s'occupe de l'exécution des machines virtuelles sur OpenStack et c'est lui qui les gère, donc on active en premier lieu la capacité de connexion pour l'utilisateur nova.

```
root@compute1:~# usermod -s / bin / bash nova
```

5. Basculer sur le compte nova

```
root@compute1:~# su - nova
nova@compute1:~$ _
```

6. Sur le compte nova, on va créer un dossier SSH où on va copier les clés SSH

```
nova@compute1:~$ mkdir -p -m 700 .ssh
nova@compute1:~$  cat > .ssh/config <<EOF
>
> Host *
>
> StrictHostKeyChecking no
>
> UserKnownHostsFile=/dev/null
>
> EOF
```

7. On génère une clé SSH

```
nova@compute1:~$  ssh-keygen -f id_rsa -b 1024 -P ''
Generating public/private rsa key pair.
id_rsa already exists.
Overwrite (y/n)? y
Your identification has been saved in id_rsa.
Your public key has been saved in id_rsa.pub.
The key fingerprint is:
SHA256:WrOli6G4YTU9sgOfO+pJQLhC5TXdep4Y6aDBVLEH3YE nova@compute1
The key's randomart image is:
+---[RSA 1024]----+
|    ..*+ +..     |
| .o.. +E o       |
|..+. . .o        |
|.o o o.+ .       |
|o . B * S o      |
|.  + B O B       |
|  o o B o        |
| . + + + .       |
|  oo= . .        |
+----[SHA256]-----+
```

Figure 4-7: Clé SSH entre les utilisateurs nova

8. On copie la clé dans le dossier authorized_keys du nœud compute1 et compute2

```
nova@compute1:~$ scp /var/lib/nova/.ssh/id_rsa.pub root@compute1:/var/lib/nova/.ssh/authorized_keys
Warning: Permanently added 'compute1,10.0.0.32' (ECDSA) to the list of known hosts.
root@compute1's password:
id_rsa.pub                                    100%  227    12.0KB/s   00:00
nova@compute1:~$ scp /var/lib/nova/.ssh/id_rsa.pub root@compute2:/var/lib/nova/.ssh/authorized_keys
Warning: Permanently added 'compute2,10.0.0.33' (ECDSA) to the list of known hosts.
root@compute2's password:
id_rsa.pub                                    100%  227    14.3KB/s   00:00
nova@compute1:~$ scp /var/lib/nova/.ssh/* root@compute2:/var/lib/nova/.ssh/
Warning: Permanently added 'compute2,10.0.0.33' (ECDSA) to the list of known hosts.
root@compute2's password:
authorized_keys                               100%  227   306.9KB/s   00:00
config                                        100%   65     1.1KB/s   00:00
id_rsa                                        100%  887    79.7KB/s   00:00
id_rsa.pub                                    100%  227   210.5KB/s   00:00
nova@compute1:~$
```

On répète le processus pour l'autre nœud

9. Retourner en tant que root, pour donner les permissions appropriées

```
root@compute1:~# sudo chown -R nova:nova /var/lib/nova/
root@compute1:~# sudo chmod 700 /var/lib/nova/.ssh
root@compute1:~# sudo chmod 600 /var/lib/nova/.ssh/authorized_keys
```

10. Tester

```
nova@compute1:~$ ssh nova@compute2 ls
Warning: Permanently added 'compute2,10.0.0.33' (ECDSA) to the list of known hosts.
buckets
CA
id_rsa
id_rsa.pub
images
instances
keys
networks
tmp
nova@compute1:~$ _
```

On voit qu'à partir du nœud compute1 ons peut lister les fichiers qui se trouvent dans nova du comoute2. (Refaire le même test sur l'autre nœud).

B. Stockage partagé :

Dans notre projet on a opté pour configurer une migration directe basée sur le stockage partagé, dans ce cas de migration l'instance contient des disques éphémères situés sur un stockage partagé entre les hôtes source et cible. Il existe plusieurs options pour partager le stockage par exemple NFS, LUN .. .

Dans ce qui suit nous allons configurer le système Linux standard en tant que serveur NFS.

1. S'assurer que le UID et le GID de l'utilisateur nova sont identiques sur les hôtes de calcul et le serveur NFS.
2. Créer un répertoire qu'on va partager, dans ce projet on a utilisé le répertoire existant **/var/lib/nova/instances.**
3. Sur le nœud contrôleur, on installe le serveur NFS avec la commande **aptinstallnfs-kernel-server** .
4. Définir le bit 'execute ' sur le répertoire partagé QEMU puisse utiliser les images contenues dans les répertoires lors de leur exportation vers les nœuds de calcul

```
root@controller:~# chmod o+x /var/lib/nova/instances
```

5. Ajouter la configuration suivante dans **/etc/exports** afin de partager /var/lib/nova/instance

```
/var/lib/nova/instances 10.0.0.0/24(rw,fsid=0,insecure,no_subtree_check,async,no_root_squash)
```

6. Redémarrer le serveur

```
root@controller:~# service nfs-kernel-server restart
```

7. Sur les nœuds de calcul, on installe le serveur NFS client grâce à la **commande aptinstall-nfs-common.**

8. Après avoir configuré le serveur NFS, on monte le système de fichier distant sur tous les hôtes de calcul, en ajoutant la ligne suivante dans /etc/fstab.

```
 GNU nano 2.9.3                                    /etc/fstab

UUID=900f38e4-a6ff-4381-8029-1602f1555b36 / ext4 defaults 0 0
/swap.img       none    swap    sw      0       0
```

Figure 4-8: Configuration du fichier fstab.

9. Tester NFS en montant le répertoire d'instances et vérifier les autorisations d'accès pour l'utilisateur nova.

```
root@compute2:~# sudo mount -a -v
/                        : ignored
none                     : ignored
/var/lib/nova/instances  : already mounted
root@compute2:~#  ls -ld /var/lib/nova/instances
drwxr-xr-x 4 nova nova 4096 Aug 27 15:28 /var/lib/nova/instances
```

C. Autres configurations :

1. Dans **nova.conf, dans** la section **VNC** on définit le **server_listen = 0.0.0.0,** c'est pour que notre serveur écoute n'importe quel adresse IP.

2. La résolution de noms sur tous les hôtes de calcul doit être identique, afin qu'ils puissent se connecter via leurs noms d'hôte.

3. Mettre à jour les configurations libvirt, dans le fichier

/etc/libvirt/libvirtd.conf on effectue les configurations suivantes (sur tous les nœuds) .

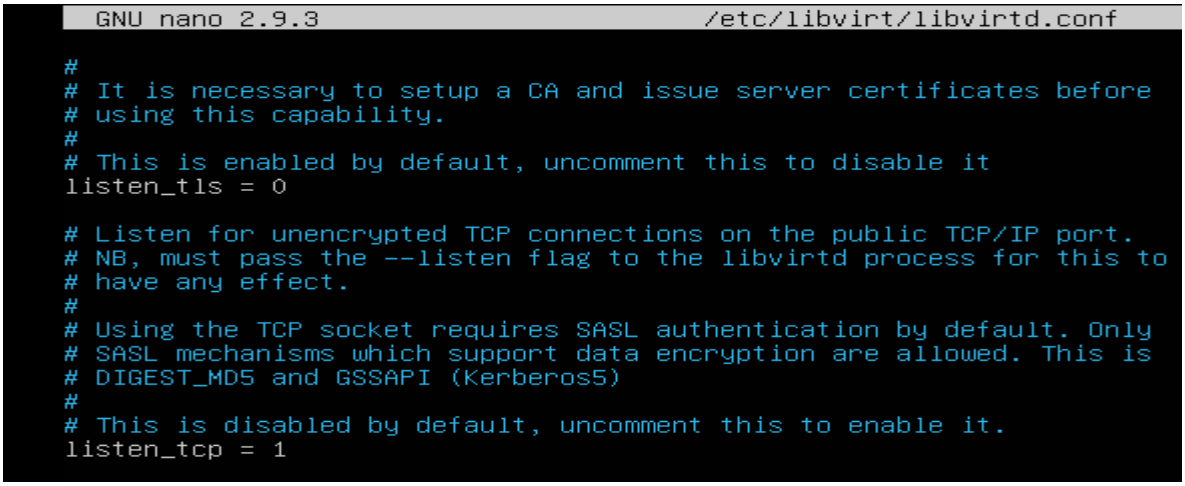


Figure 4-9: Configuration du fichier /etc/libvirt/libvirtd.

On met : listen_tls =0 pour désactiver l'option d'écoute des connexions sécurisées TLS sur le port TCP/IP

Listen_tcp =1 pour activer l'écoute des connexions non chiffrés

TCP sur le port TCP/IP

4. Ajouter l'option **libvrtd_opt = -l** dans le fichier **/etc/default/**libvirtd pour écouter le port TCP.

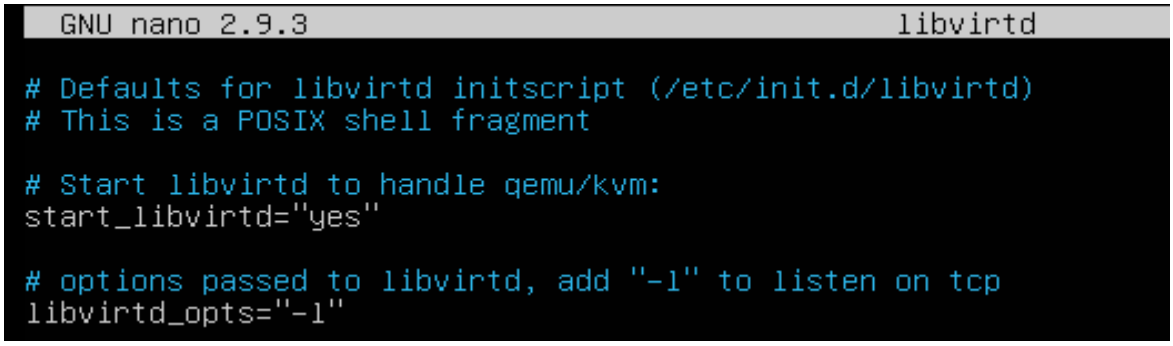


Figure 4-10: Configuration du fichier /etc/default/libvirtd.

5. Redémarrer le service libvirt.

4.4.2.2 Lancement d'une migration en direct :

Une fois que le cluster OpenStack et le système de fichier partagé ont été configurés, on peut lancer une migration en direct.

OpenStack donne le choix à l'administrateur du Cloud soit de choisir lui-même l'hôte de destination ou de laisser la sélection du nœud destinataire automatique.

Les étapes suivantes montre la migration en direct d'une instance en sélectionnant manuellement l'hôte de destination.

1. Obtenir l'ID de l'instance qu'on souhaite migrer

```
root@controller:~# openstack server list
+--------------------------------------+------------+--------+--------------------+--------+-------
--+
| ID                                   | Name       | Status | Networks           | Image  | Flavor
  |
+--------------------------------------+------------+--------+--------------------+--------+-------
--+
| 92f90394-0f5f-436f-9ee8-de877239ab61 | instance_A | ACTIVE | provider=10.0.0.114 | cirros | m1.nan
o |
| d224b6ae-ef9d-4873-96a9-efe503706085 | instance1  | ACTIVE | provider=10.0.0.102 | cirros | m1.nan
o |
| a495c5e4-538f-4274-9682-29d2ffe5311c | instance2  | ACTIVE | provider=10.0.0.106 | cirros | m1.nan
o |
+--------------------------------------+------------+--------+--------------------+--------+-------
```

Figure 4-11 : Liste des instances du cluster.

La commande **OpenStack server list** permet de lister et donner des informations de toutes les instances du Cloud.

2. Par exemple, on veut migrer l'instance nommé instance1, on vérifie alors que cette instance s'exécute sur le nœud à réparer. (l'administrateur du cloud est censé connaitre les instances qui existent sur le Cloud et sur quel hôte elles s'exécutent, cette étape c'est juste pour faire une vérification et s'assurer de ne pas faire une action sans intérêt surtout quand l'état est critique).

```
root@controller:~# openstack server show d224b6ae-ef9d-4873-96a9-efe503706085
+-------------------------------------+-----------------------------------------------------------+
| Field                               | Value                                                     |
+-------------------------------------+-----------------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                                    |
| OS-EXT-AZ:availability_zone         | nova                                                      |
| OS-EXT-SRV-ATTR:host                | compute2                                                  |
| OS-EXT-SRV-ATTR:hypervisor_hostname | compute2                                                  |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000025                                         |
| OS-EXT-STS:power_state              | Running                                                   |
| OS-EXT-STS:task_state               | None                                                      |
| OS-EXT-STS:vm_state                 | active                                                    |
| OS-SRV-USG:launched_at              | 2019-08-09T22:43:19.000000                                |
| OS-SRV-USG:terminated_at            | None                                                      |
| accessIPv4                          |                                                           |
| accessIPv6                          |                                                           |
| addresses                           | provider=10.0.0.102                                       |
| config_drive                        |                                                           |
| created                             | 2019-08-09T22:42:45Z                                      |
| flavor                              | m1.nano (0)                                               |
| hostId                              | 7c1385eda2303f8c63e0e7cdbf6b559133233b4d0dbc1fe5b85e4f97  |
| id                                  | d224b6ae-ef9d-4873-96a9-efe503706085                      |
| image                               | cirros (4a6cf4da-8c01-430a-86ab-c0c05f8471a2)             |
| key_name                            | mykey                                                     |
| name                                | instance1                                                 |
| progress                            | 0                                                         |
| project_id                          | bec9bd23182447399f3b6a013d613031                          |
| properties                          |                                                           |
| security_groups                     | name='default'                                            |
| status                              | ACTIVE                                                    |
| updated                             | 2019-08-28T16:04:53Z                                      |
| user_id                             | 08c43dcba2024b6aa5cd021e908def71                          |
| volumes_attached                    |                                                           |
+-------------------------------------+-----------------------------------------------------------+
```

Figure 4-12: Information de l'instance à migrer.

Comme on peut le constater, cette commande nous donne plusieurs informations sur l'instance notamment le nœud sur lequel elle s'exécute, son adresse IP, sa date de création… .

3. Lister les nœuds de calcul pour en choisir un destinataire.

```
root@controller:~# openstack compute service list
+----+------------------+------------+----------+---------+-------+----------------------------+
| ID | Binary           | Host       | Zone     | Status  | State | Updated At                 |
+----+------------------+------------+----------+---------+-------+----------------------------+
|  3 | nova-consoleauth | controller | internal | enabled | up    | 2019-08-28T16:40:15.000000 |
|  4 | nova-conductor   | controller | internal | enabled | up    | 2019-08-28T16:40:22.000000 |
|  5 | nova-compute     | compute    | nova     | enabled | down  | 2019-08-09T20:29:27.000000 |
|  6 | nova-scheduler   | controller | internal | enabled | up    | 2019-08-28T16:40:19.000000 |
|  7 | nova-compute     | compute1   | nova     | enabled | up    | 2019-08-28T16:40:16.000000 |
|  8 | nova-compute     | compute2   | nova     | enabled | up    | 2019-08-28T16:40:19.000000 |
+----+------------------+------------+----------+---------+-------+----------------------------+
```

La commande précédente démontre les services de nova , on a 3 nœuds de calcul compute, compute1 et compute2 pour des raisons de manque de ressources on a préféré utiliser que 2 nœuds de traitement , on a donc enlever un nœud de traitement qui est compute (son état est down car on l'a pas allumé) . On n'a pas d'autres choix que de choisir le nœud compute 1 comme destination.

4. Vérifier que le nœud destinataire a suffisamment de ressources pour la migration.

```
root@controller:~# openstack host show compute1
+----------+----------------------------------+-----+-----------+---------+
| Host     | Project                          | CPU | Memory MB | Disk GB |
+----------+----------------------------------+-----+-----------+---------+
| compute1 | (total)                          |   1 |      2609 |      19 |
| compute1 | (used_now)                       |   1 |       576 |       1 |
| compute1 | (used_max)                       |   2 |       128 |       2 |
| compute1 | bec9bd23182447399f3b6a013d613031 |   2 |       128 |       2 |
+----------+----------------------------------+-----+-----------+---------+
```

- **cpu**: Nombre de processeurs
- **memory_mb**: Quantité totale de mémoire, en Mo
- **disk_gb**: Quantité totale d'espace disponible pour NOVA-INST-DIR / instances, en Go

Dans ce tableau, la première ligne indique la quantité totale de ressources disponibles sur le serveur. La deuxième ligne montre les ressources actuellement utilisées. La troisième ligne indique le maximum de ressources utilisées.

5. Migrer l'instance

```
root@controller:~# openstack server migrate  d224b6ae-ef9d-4873-96a9-efe503706085 --live compute1
root@controller:~# _
```

Figure 4-13: commande de migration en direct.

Si la commande **OpenStack server migrate** ne donne aucune sortie ça veut dire qu'elle s'est exécutée correctement.

6. Confirmer que l'instance a été migrée avec succès

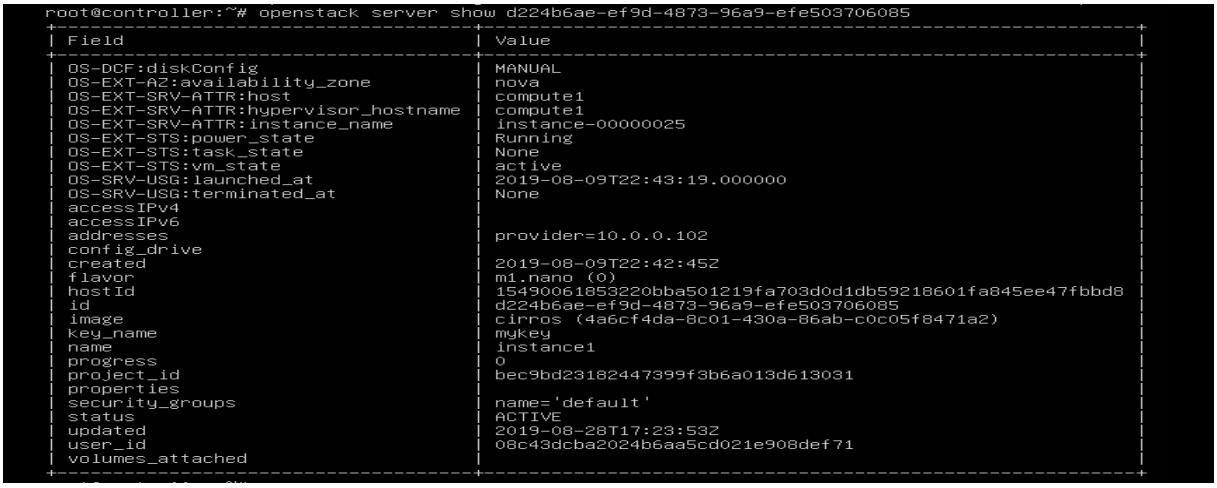


Figure 4-14: Instance à migrer.

Comme on le voit l'instance a bien changé de serveur, elle s'exécute maintenant sur compute1.

OpenStack ne donne pas la possibilité de suivre et de voir en détaille le processus de transfert de données, l'une des solutions qu'il propose pour surveiller et suivre le progrès de la migration (nous allons donner d'autre méthodes juste après) est : de lancer une commande tail du fichier **/var/log/nova/nova-compute.log** elle permet de suivre les fichiers log du service nova-compute sur les nœuds de traitement.

Analyses faite lors de la migration.

Après le lancement de la commande de migration, nous avons lancé la commande tail sur l'un des nœuds (l'hôte de destination).

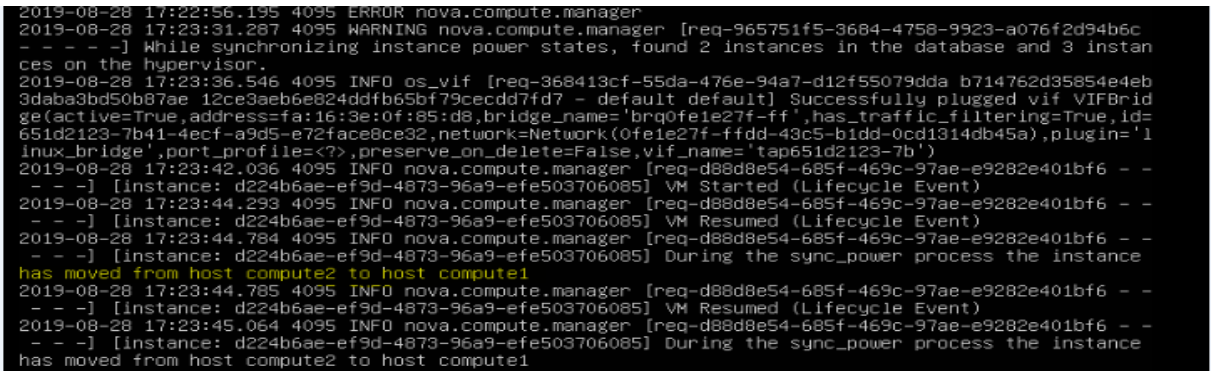


Figure 4-15: Fichier log de nova-compute du noeud destinataire.

Le résultat de la commande nous donne quelques informations comme le nombre d'instances, les états de la machine lors de la migration et nous montre bel et bien que l'instance d224b6ae-ef9d-4873-96a9-efe503706085 a été déplacée de l'hôte compute2 à l'hôte compute1.

4.4.2.3 Analyse et résultats :

Dans l'exemple de migration précédent, on a vu que l'hôte a été migré mais on n'a pas montré le concept de la live migration qui est de déplacer une machine virtuelle ou d'une application en cours d'exécution entre différentes machines physiques sans déconnecter le client ou l'application. Aussi comme on l'a dit précédemment OpenStack ne nous permet pas de voir comment la migration se fait réellement, mais elle a mis en place quelques commandes qui permettent de suivre le progrès de la migration comme la commande tail qu'on vu, et d'autre commandes que nous allons voir dans les exemples d'après. Pour voir un peu-près le temps de la migration (temps d'arrêt)on a penché à faire un diagnostic avec l'outil MTR.

Dans ce qui suit nous allons donner plusieurs exemples de migration afin d'éclaircir et d'analyser le maximum possible le processus de la migration à chaud sur OpenStack.

Exemple 1

Dans cet exemple nous allons essayer de faire la migration via le Dashboard horizon, et sur le nœud contrôleur on lance un tunnel SSH pour accéder à la console de l'instance qu'on souhaite déplacé.

La figure ci-dessus nous montre la liste des instances et des informations sur elles.

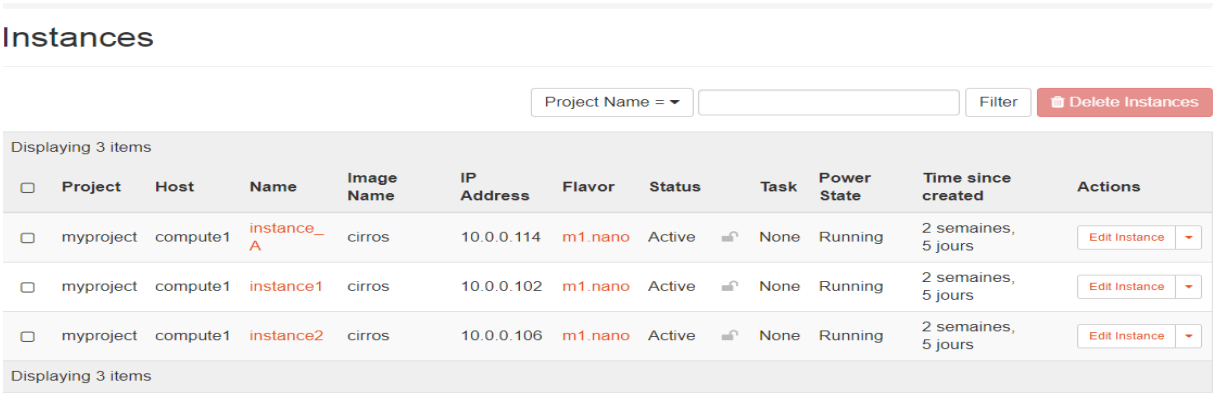


Figure 4-16: Liste des instances.

On choisit de migrer instance_A qui est sur le serveur compute1, donc on accède à sa console à travers un tunnel SSH.

```
root@controller:~# ssh cirros@10.0.0.114
The authenticity of host '10.0.0.114 (10.0.0.114)' can't be established.
ECDSA key fingerprint is SHA256:qXhDH6XcgbPuGZRtFeNiCF+ebDvNTwGc6HYh8VRG3Ns.
Are you sure you want to continue connecting (yes/no)? yes
Warning: Permanently added '10.0.0.114' (ECDSA) to the list of known hosts.
$
```

Figure 4-17: Accès SSH à la console de l'instance.

Une fois qu'on est sur la console de notre instance, on lance une opération continue (un ping) pour voir si l'action a été interrompue ou non lors de la migration.

Pour maigrir l'instance via le Dashboard, on fait comme suit

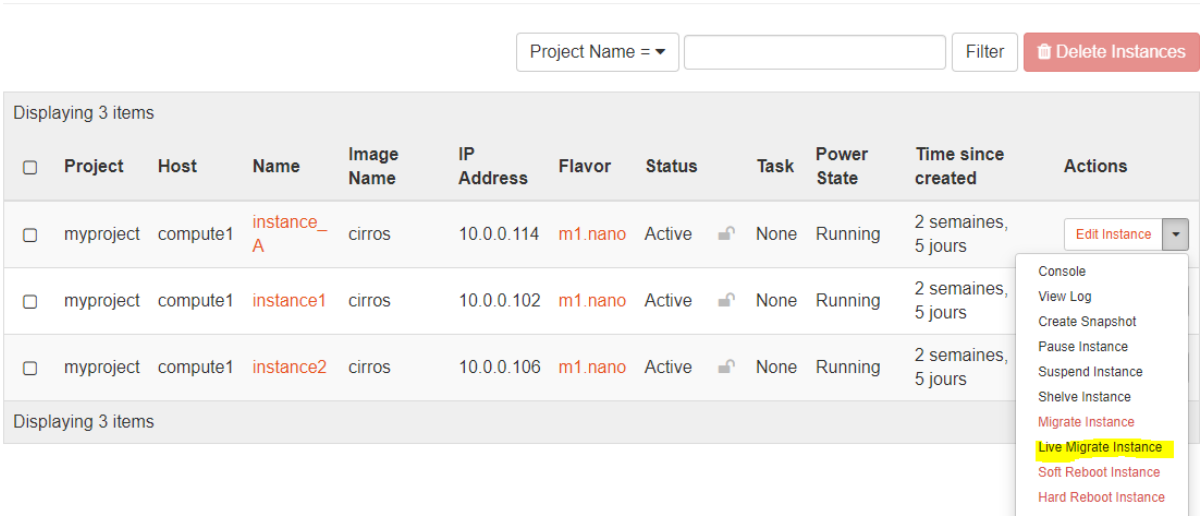


Figure 4-18: Lancer une migration via le dahboard.

Dans la flèche à droite on choisit live migrate instance, après y aura une boite de dialogue qui va s'afficher, pour choisir la destination et on lance la migration.

La figure ci-dessus, nous montre que la machine n'a pas été interrompue lors de la migration, elle continue toujours son action

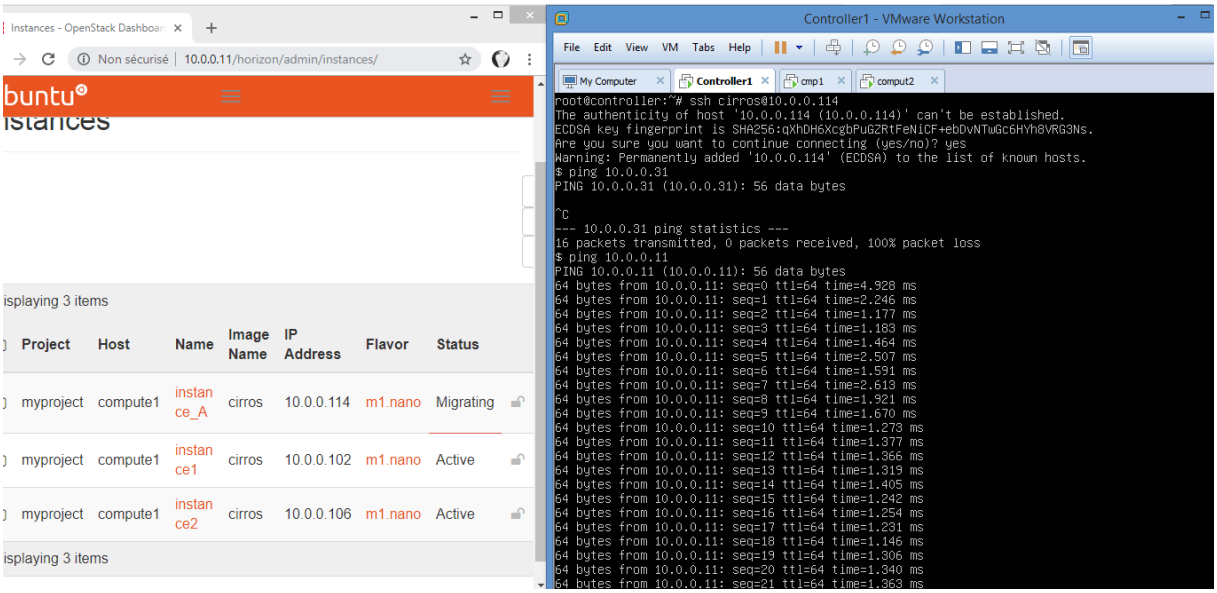


Figure 4-19: Concept de migration.

A la fin de la migration, le statut de l'instance se transforme de **migrating** à **active**

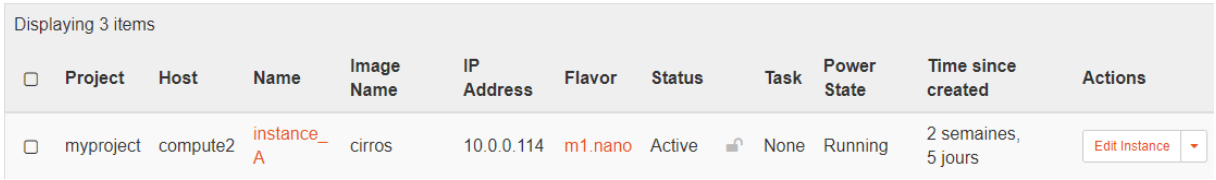
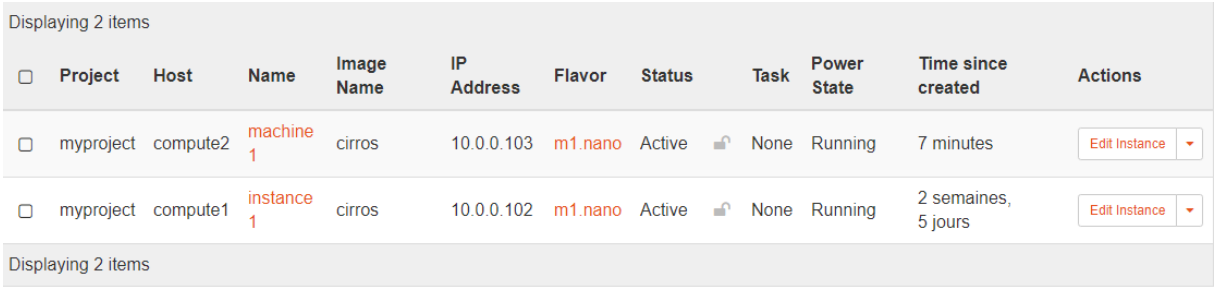


Figure 4-20: Confirmation de la migration.

Exemple 2

Dans cet exemple on utilise l'outil MTR pour calculer l'un des paramètres d'évolution de la migration à chaud qui est le **temps d'arrêt.**

On choisit l'instance à migrer



On va migrer machine1 qui se trouve sur le compute2.

On obtient son ID soit avec le Shell ou à partir du Dashboard.

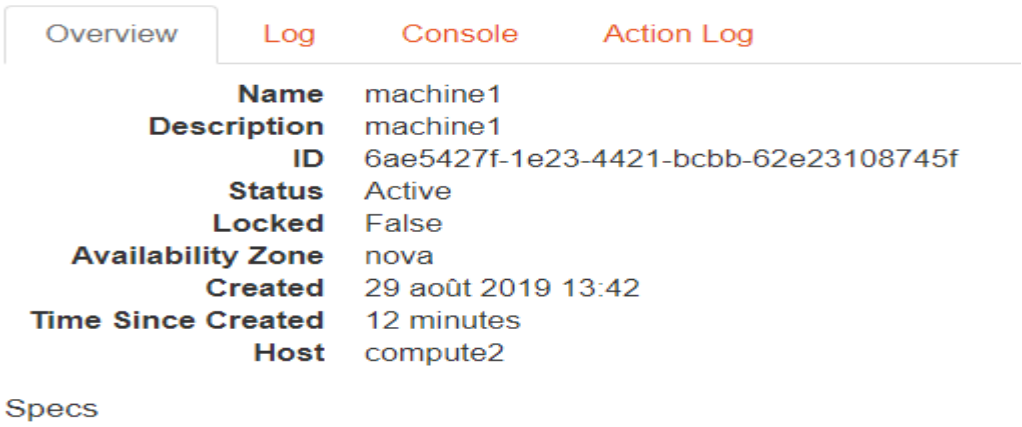


Figure 4-21: Vue d'ensemble de l'instance à migrer via le Dashboard

On lance la migration

```
root@controller:~# openstack server migrate 6ae5427f-1e23-4421-bcbb-62e23108745f --live compute1
```

Sur le nœud contrôleur où on a installé MTR , on tape la commande suivante (on a choisi un intervalle de 50)

```
root@controller:~# mtr --report-cycles 50 10.0.0.103
```

Durant de la migration

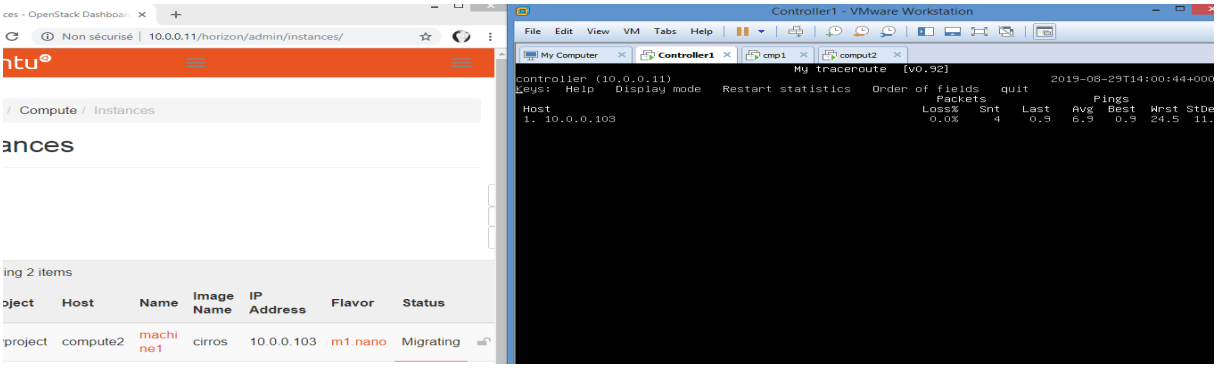


Figure 4-22: Diagnostic à l'aide de MTR.

A la fin de la migration dans le fichier log de nova-compute de **l'hôte source** on a eu les résultats suivants

```
root@compute2:~# tail /var/log/nova/nova-compute.log
2019-08-29 14:00:34.831 1789 INFO nova.virt.libvirt.driver [-] [instance: 6ae5427f-1e23-4421-bcbb-
e23108745f] Migration running for 0 secs, memory 100% remaining; (bytes processed=0, remaining=0,
tal=0)
2019-08-29 14:00:37.020 1789 INFO nova.compute.manager [req-e161ed98-2561-42cc-be77-d24e63c1e4ad -
- - -] [instance: 6ae5427f-1e23-4421-bcbb-62e23108745f] VM Paused (Lifecycle Event)
2019-08-29 14:00:37.584 1789 INFO nova.virt.libvirt.driver [-] [instance: 6ae5427f-1e23-4421-bcbb-
e23108745f] Migration operation has completed
2019-08-29 14:00:37.585 1789 INFO nova.compute.manager [-] [instance: 6ae5427f-1e23-4421-bcbb-62e2
08745f] _post_live_migration() is started..
2019-08-29 14:00:46.844 1789 WARNING nova.compute.manager [req-49c1a513-e904-41c5-b6b3-41ad5f0d8b7
0e300b5c2c824934b5404021a9c4945c ca1c277fc3ee475989340735d2527789 - default default] [instance: 6a
427f-1e23-4421-bcbb-62e23108745f] Received unexpected event network-vif-plugged-d82a996f-cfc2-43d6
bb8-642c86d05b3e for instance with vm_state active and task_state migrating.
2019-08-29 14:00:52.620 1789 INFO nova.compute.manager [-] [instance: 6ae5427f-1e23-4421-bcbb-62e2
08745f] VM Stopped (Lifecycle Event)
2019-08-29 14:00:52.661 1789 INFO os_vif [-] Successfully unplugged vif VIFBridge(active=True,addr
s=fa:16:3e:bd:2c:f7,bridge_name='brq0fe1e27f-ff',has_traffic_filtering=True,id=d82a996f-cfc2-43d6-
b8-642c86d05b3e,network=Network(0fe1e27f-ffdd-43c5-b1dd-0cd1314db45a),plugin='linux_bridge',port_p
file=<?>,preserve_on_delete=False,vif_name='tapd82a996f-cf')
2019-08-29 14:00:59.723 1789 WARNING nova.compute.resource_tracker [-] [instance: 6ae5427f-1e23-44
-bcbb-62e23108745f] Instance not resizing, skipping migration.
2019-08-29 14:00:59.756 1789 INFO nova.compute.resource_tracker [-] Final resource view: name=comp
e2 phys_ram=1969MB used_ram=512MB phys_disk=19GB used_disk=0GB total_vcpus=1 used_vcpus=0 pci_stat
[]
2019-08-29 14:00:59.970 1789 INFO nova.compute.manager [-] [instance: 6ae5427f-1e23-4421-bcbb-62e2
08745f] Migrating instance to compute1 finished successfully.
```

Figure 4-23: Fichier log de nova-compute du nœud source.

La commande tail nous indique les détails de la migration : le début de la migration, la mise an pause de l'instance, l'état de la VM est active et elle effectue une tache de migration, la VM s'est arrêté et téléchargé avec succès sur l'autre nœud et à la fin nous indique que la migration vers le compute1 est finit avec succès malgré que le warning nous a indiqué que la migration a été ignorée.

Displaying 2 items

Project	Host	Name	Image Name	IP Address	Flavor	Status	Task	Power State	Time since created	Actions
myproject	compute1	machine 1	cirros	10.0.0.103	m1.nano	Active	None	Running	18 minutes	Edit Instance
myproject	compute1	instance 1	cirros	10.0.0.102	m1.nano	Active	None	Running	2 semaines, 5 jours	Edit Instance

Sur le Dashboard on voit effectivement que l'instance a changé d'hôte.

L'image ci-dessus nous donne le résultat de la commande MTR

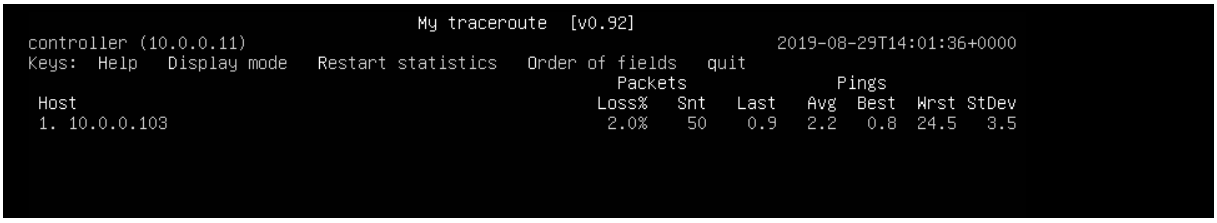


Figure 4-24: Résultats MTR.

Sur 50 paquets envoyés on a 2% de paquets perdus.

Le nombre de paquet perdu = % paquet perdu * le total des paquets envoyé [49]

Le nombre de paquet perdu = 2 * 50/100 = 1

Le temps d'arrêt de la machine lors de la migration = nombre de paquets perdu * le temps moyens pour chaque paquet [49]

Le temps d'arrêt = 1*2.2ms =2.2 ms

Sachant qu'on a choisi un intervalle de 50 largement grand pour faire la migration. On a calculé le temps d'arrêt de l'instance qui est l'un des paramètres d'évaluation de la migration à chaud car OpenStack ne donne pas la possibilité de calculer exactement le temps d'arrêt

Le temps d'arrêt est de 2.2ms il est inaperçu à l'utilisateur donc on conclue que la migration a réussi.

Rappelle : Le temps d'arrêt c'est le temps pendant laquelle la machine est suspendue à la fin de la migration pour pouvoir copier les quelques pages restantes sans interférence avec les écritures en mémoire de l'instance.

Exemple 3

Dans l'exemple suivant on va suivre le progrès de la migration avec les commandes dédiées pour cela sur OpenStack.

Displaying 2 items

Project	Host	Name	Image Name	IP Address	Flavor	Status	Task	Power State	Time since created	Actions
myproject	compute1	pc	cirros	10.0.0.101	m1.nano	Active	None	Running	2 minutes	Edit Instance
myproject	compute2	instance 2	cirros	10.0.0.106	m1.nano	Active	None	Running	2 semaines, 5 jours	Edit Instance

Displaying 2 items

On va migrer instance2 qui s'exécute sur le serveur compute2

Obtenir son ID

Après le lancement de la migration on va confirmer que l'instance est en cours de migration, comme le montre la figure ci-dessus le statut de l'instance est 'migrating'

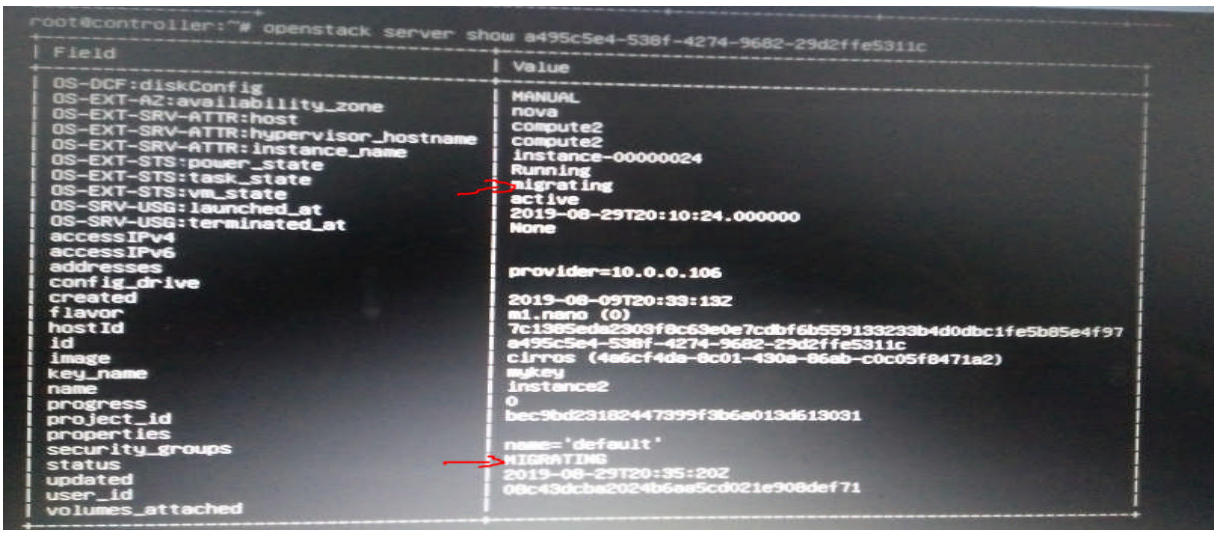


Figure 4-25: Suivit de l'état de l'instance durant la migration.

Surveiller le progrès de la migration

```
root@controller:~# nova server-migration-list a495c5e4-538f-4274-9682-29d2ffe5311c
+----+-------------+-----------+----------------+--------------+-----------+---------+--------------
----------------------+----------------------------+----------------------------+-----------------
---+-------------------------+-------------------------+-------------------+-----------------------+----
------------------+
| Id | Source Node | Dest Node | Source Compute | Dest Compute | Dest Host | Status  | Server UUID
                   | Created At                 | Updated At                 | Total Memory Byt
es | Processed Memory Bytes | Remaining Memory Bytes | Total Disk Bytes | Processed Disk Bytes | Rem
aining Disk Bytes |
+----+-------------+-----------+----------------+--------------+-----------+---------+--------------
----------------------+----------------------------+----------------------------+-----------------
---+-------------------------+-------------------------+-------------------+-----------------------+----
------------------+
| 58 | compute2    | compute1  | compute2       | compute1     | -         | running | a495c5e4-538f
-4274-9682-29d2ffe5311c | 2019-08-29T20:35:04.000000 | 2019-08-29T20:35:20.000000 | 0
   | 0                       | 0                       | 0                 | 0                     | 0
                |
+----+-------------+-----------+----------------+--------------+-----------+---------+--------------
----------------------+----------------------------+----------------------------+-----------------
---+-------------------------+-------------------------+-------------------+-----------------------+----
------------------+
```

Figure 4-26: Surveiller le progrès de la migration.

La commande **nova-server-migration-list** donne les migrations occurrentes et leurs ID, les autres colonnes de sortie ont été supprimées.

Après avoir eu l'ID de la migration on peut suivre l'état de la migration. (Tellement que la migration se fait rapidement on n'a pas eu le temps de prendre une capture pour l'état de la migration).

Confirmer que l'instance a bien été migrée.

```
root@controller:~# openstack server show a495c5e4-538f-4274-9682-29d2ffe5311c
+-------------------------------------+-----------------------------------------------------------------+
| Field                               | Value                                                           |
+-------------------------------------+-----------------------------------------------------------------+
| OS-DCF:diskConfig                   | MANUAL                                                          |
| OS-EXT-AZ:availability_zone         | nova                                                            |
| OS-EXT-SRV-ATTR:host                | compute1                                                        |
| OS-EXT-SRV-ATTR:hypervisor_hostname | compute1                                                        |
| OS-EXT-SRV-ATTR:instance_name       | instance-00000024                                               |
| OS-EXT-STS:power_state              | Running                                                         |
| OS-EXT-STS:task_state               | None                                                            |
| OS-EXT-STS:vm_state                 | active                                                          |
| OS-SRV-USG:launched_at              | 2019-08-29T20:10:24.000000                                      |
| OS-SRV-USG:terminated_at            | None                                                            |
| accessIPv4                          |                                                                 |
| accessIPv6                          |                                                                 |
| addresses                           | provider=10.0.0.106                                             |
| config_drive                        |                                                                 |
| created                             | 2019-08-09T20:33:13Z                                            |
| flavor                              | m1.nano (0)                                                     |
| hostId                              | 15490061853220bba501219fa703d0d1db59218601fa845ee47fbbd8        |
| id                                  | a495c5e4-538f-4274-9682-29d2ffe5311c                            |
| image                               | cirros (4a6cf4da-8c01-430a-86ab-c0c05f8471a2)                   |
| key_name                            | mykey                                                           |
| name                                | instance2                                                       |
| progress                            | 0                                                               |
| project_id                          | bec9bd23182447399f3b6a013d613031                                |
| properties                          |                                                                 |
| security_groups                     | name='default'                                                  |
| status                              | ACTIVE                                                          |
| updated                             | 2019-08-29T20:35:42Z                                            |
| user_id                             | 08c43dcba2024b6aa5cd021e908def71                                |
| volumes_attached                    |                                                                 |
+-------------------------------------+-----------------------------------------------------------------+
```

Figure 4-27: Confirmation de la migration.

4.5 Conclusion :

Dans ce chapitre, nous avons commencé de donner un petit exemple sur la migration à froid, puis nous avons activé les configurations nécessaires à la réalisation d'une migration en direct, puis nous avons simulé plusieurs exemples afin de pouvoir donner un aperçu sur les différentes façons de suivre le bon déroulement de la migration.

Durant cette réalisation, sur OpenStack on peut facilement réaliser la migration non direct, le seul paramètre qui devrai y être est une clé SSH entre les nœuds de calculs, par contre la migration en direct est un peu plus complexe car elle nécessite plus de configurations. De plus OpenStack ne permet pas la migration en direct si l'état de l'instance est arrêté (il retourne une erreur). Aussi il n'a pas un outil spécifique pour calculer le temps exact de la migration, par contre on peut surveiller l'état de la migration et son progrès et l'état de l'instance après la migration. Ce dernier peut être actif, arrêté, erroné ou autre alors le fait que l'instance soit en état actif après la migration est important.

Conclusion générale:

Le Cloud Computing est une plate-forme émergente et innovante, qui permet d'offrir des ressources informatiques sous forme de services à la demande accessible de n'importe où, n'importe quand et par n'importe qui. La fiabilité et la robustesse de ces services deviennent des éléments de premières importances surtout dans des environnements critiques, c'est pourquoi un mécanisme de tolérance aux pannes est devenu crucial.

Dans ce mémoire, nous avons procédé à une recherche sur les pannes auxquelles un système peut être confronté, puis sur les différentes techniques de tolérance aux pannes et des travaux connexes qui traitent la tolérance aux pannes dans le CloudComputing qui est l'objet de notre étude.

De ce fait, nous avons essayé d'appliquer dans ce mémoire une des stratégies de tolérance aux pannes qui est la migration. Cette stratégie consiste à transférer une machine virtuelle d'un serveur physique à un autre, sans perte de données, elle peut s'avérer efficace si on suspecte une panne dans notre serveur ou lorsque ce dernier a besoin d'une maintenance.

Pour mettre en œuvre cette solution, on s'est basé sur trois phases : la première consiste en la création d'un environnement Cloud sur OpenStack, cette étape était la plus longue et la plus complexe à cause du manque de ressources et surtout nécessite une grande connaissance en réseaux et en lignes de commandes linux. La deuxième consiste à configurer et activer la migration à chaud sur OpenStack, une étape indispensable. La troisième qui est le cœur de notre travail consiste à déplacer nos instances d'un serveur à un autre, et de suivre le progrès et le bon déroulement de la migration. Les résultats obtenus montrent l'efficacité de cette stratégie notamment lorsque le serveur source est planifié d'être maintenu.

Conclusion Générale

Un certain nombre de pistes peuvent être envisageables pour continuer notre travail, nous pouvons citer :

- Tester la performance de la migration avec des instances de plusieurs distributions et des gabarits différents.
- Trouver un moyen de suivre en détails le processus de transfert de données sur OpenStack, et comment calculer le temps total de la migration.
- Utiliser la migration pour assurer l'équilibrage de charge.
- utilisation de la migration post-copy.
- Tester la migration dans des conditions proches des conditions réelles, où l'état est critique et plusieurs machines doivent être déplacées.

Annexe A: installation OpenStack

Dans le cadre de ce projet, nous avons installé service par service une infrastructure OpenStack Rocky multi-nœuds (un nœud contrôleur et 2 nœuds traitements)

Nous avons utilisé le logiciel VMware comme hyperviseur et l'OS ubuntu 18.04 pour la création des 3 machines virtuelles.

I. Prérequis :

- Nœud contrôleur : 1 processeur, 4.5 GO de mémoire et 20 GO de stockage .
- Nœud compute : 1 processeur, 2 GO de mémoire, 30 GO de stockage (les mêmes performances pour les 2 computes.)
- Les 3 nœuds utilisent 2 cartes réseau de type NAT : 1 carte réseau ayant une ip statique pour gérer le réseau de gestion qui permet la communication entre les nœuds d'openstack et une autre avec une interface dynamique pour le réseau fournisseur qui permet aux instances d'openstack d'avoir un accès internet.

II. Configuration des cartes réseau :

La version 18.04 (Bionic Beaver) d'ubuntuutilisenetplan pour la gestion des interfaces réseau donc remplace /etc/network/interfaces qui est utilisé précédemment.

Pour pouvoir configurer les cartes réseau nous ouvrons le fichier YAML qui se trouve à l'emplacement /etc/netplan comme suit

```
root@controller:~# cd /etc/netplan
root@controller:/etc/netplan# nano 50*_
```

Ensuite le modifier selon la configuration souhaitée

```
  GNU nano 2.9.3                          50-cloud-init.yaml

# This file is generated from information provided by
# the datasource.  Changes to it will not persist across an instance.
# To disable cloud-init's network configuration capabilities, write a file
# /etc/cloud/cloud.cfg.d/99-disable-network-config.cfg with the following:
# network: {config: disabled}
network:
    ethernets:
        ens33:
            addresses: [10.0.0.11/24]
            gateway4: 10.0.0.2
            dhcp4: no
            nameservers:
               addresses: [8.8.8.8,8.8.4.4]
            optional: true
        ens34:
            dhcp4: true
    version: 2
```

Enfin nous exécutons la commande ci-dessus pour prendre en considération la modification effectuée.

```
root@controller:/etc/netplan# netplan apply
```

Dans notre architecture on a attribué les adresses ip comme suit : 10.0.0.11 pour le nœud controller et 10.0.0.32, 10.0.0.33 pour le compute1 et compute2 respectivement.

Après la configuration des cartes il faut configurer la résolution de nom et cela en définissons les noms de tous les nœuds et leurs adresses ip sur le fichier hosts de chaque nœud.

Ci-dessus /etc/hosts fichier pour le nœud controller

Pour vérifier la connectivité à partir de chaque nœud on ping les autres nœuds et on ping google pour voir si il a accès à internet.

```
root@compute1:~# ping c -4 controller
ping: c: Temporary failure in name resolution
root@compute1:~# ping -c 4 controller
PING controller (10.0.0.11) 56(84) bytes of data.
64 bytes from controller (10.0.0.11): icmp_seq=1 ttl=64 time=0.267 ms
64 bytes from controller (10.0.0.11): icmp_seq=2 ttl=64 time=11.5 ms
64 bytes from controller (10.0.0.11): icmp_seq=3 ttl=64 time=0.217 ms
64 bytes from controller (10.0.0.11): icmp_seq=4 ttl=64 time=0.201 ms

--- controller ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3042ms
rtt min/avg/max/mdev = 0.201/3.070/11.595/4.921 ms
root@compute1:~# ping -c 4 google.com
PING google.com (216.58.198.78) 56(84) bytes of data.
64 bytes from dub08s02-in-f78.1e100.net (216.58.198.78): icmp_seq=1 ttl=128 time=31.4 ms
64 bytes from dub08s02-in-f78.1e100.net (216.58.198.78): icmp_seq=2 ttl=128 time=32.3 ms
64 bytes from dub08s02-in-f78.1e100.net (216.58.198.78): icmp_seq=3 ttl=128 time=29.9 ms
64 bytes from dub08s02-in-f78.1e100.net (216.58.198.78): icmp_seq=4 ttl=128 time=42.2 ms

--- google.com ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3007ms
rtt min/avg/max/mdev = 29.997/34.027/42.289/4.847 ms
```

III. Environnement :

A. Protocole NTP :

Utilisé pour synchroniser correctement les services entre les nœuds , dans ce projet on a installé chrony une implémentation de ntp .

a. Sur le nœud contrôleur :

- Installation des paquets

```
root@controller:~# apt install chrony
Reading package lists... Done
Building dependency tree
Reading state information... Done
chrony is already the newest version (3.2-4ubuntu4.2).
0 upgraded, 0 newly installed, 0 to remove and 47 not upgraded.
```

- Editer le fichier /etc/chrony/chrony.conf et ajouter la ligne allow 10.0.0.0/24 (la description du sous-réseau) : c'est pour permettre aux autres noeuds de se connecter au démon chrony sur le nœud contrôleur.

```
  GNU nano 2.9.3                        /etc/chrony/chrony.conf

# Welcome to the chrony configuration file. See chrony.conf(5) for more
# information about usuable directives.
allow 10.0.0.0/24
# This will use (up to):
# - 4 sources from ntp.ubuntu.com which some are ipv6 enabled
# - 2 sources from 2.ubuntu.pool.ntp.org which is ipv6 enabled as well
# - 1 source from [01].ubuntu.pool.ntp.org each (ipv4 only atm)
# This means by default, up to 6 dual-stack and up to 2 additional IPv4-only
# sources will be used.
# At the same time it retains some protection against one of the entries being
# down (compare to just using one of the lines). See (LP: #1754358) for the
# discussion.
#
# About using servers from the NTP Pool Project in general see (LP: #104525).
# Approved by Ubuntu Technical Board on 2011-02-08.
# See http://www.pool.ntp.org/join.html for more information.
pool ntp.ubuntu.com        iburst maxsources 4
pool 0.ubuntu.pool.ntp.org iburst maxsources 1
pool 1.ubuntu.pool.ntp.org iburst maxsources 1
#pool 2.ubuntu.pool.ntp.org iburst maxsources 2
```

- Redémarrer le service NTP

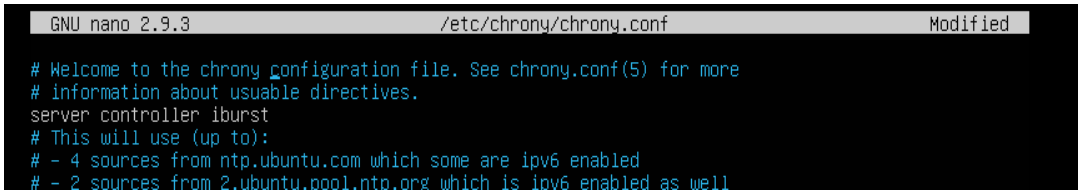
```
root@controller:~#  service chrony restart_
```

b. Sur les nœuds de calcul :

- Installation du paquet chrony

```
root@compute1:~# apt install chrony
```

- Editer le fichier /etc/chrony/chrony.conf et ajouter



- C’est pour synchroniser mes nœuds de calculs au contrôleur

NB : on met en commentaires les autres services universels.

- Redémarrer le service ntp pour en considération les modifications effectuées.

c. vérification :

- Sur le nœud contrôleur on la lance la commande suivante (elle nous affiche le serveur chrony sur lequel nous somme synchronisé : celui en * dans la colonne MS)

```
root@controller:~# chronyc sources
210 Number of sources = 8
MS Name/IP address         Stratum Poll Reach LastRx Last sample
===============================================================================
^+ chilipepper.canonical.com     2   6   377     3  +5889us[+5757us] +/-   79ms
^* pugot.canonical.com           2   6   377     1  +2668us[+2536us] +/-   72ms
^- golem.canonical.com           2   6     7    64    +48us[  -81us] +/-   64ms
^- alphyn.canonical.com          2   6   377     3  +4661us[+4529us] +/-  134ms
^- 160.119.238.133               2   6   377     1  +5877us[+5877us] +/-  136ms
^- stratum2-1.ntp.jnb01.za.>     2   6   377     1  -8814us[-8814us] +/-  195ms
^- ns5.btc.bw                    2   6   277     8  +9742us[+9610us] +/-  300ms
^- ns1.btc.bw                    2   6   377    66  +8320us[+8191us] +/-  332ms
```

- On lance la meme commande sur le nœud de traitement (elle doit obligatoirement nous afficher que le nœud est synchronisé sur le contrôleur)

B. Les paquets openstack :

On ajoute le référentiel de la branche qu’on veut, dans notre projet on a choisi la branche rocky pour sa stabilité

```
root@controller:~# add-apt-repository cloud-archive:rocky
```

Ensuite on met à niveau les packages

```
root@controller:~# apt update && apt dist-upgrade
```

Enfin on installe le client openstack

```
root@controller:~#  apt install python-openstackclient_
```

On refait les mêmes étapes pour les 2 nœuds de traitement.

C. base de données SQL :

S'installe uniquement sur le nœud contrôleur. On commence par installer les paquets

```
root@controller:~# apt install mariadb-server python-pymysql
```

Ensuite on crée un fichier **/etc/mysql/mariadb.conf.d/99-openstack.cnf**

```
root@controller:~# nano  /etc/mysql/mariadb.conf.d/99-openstack.cnf
```

On ajoute par la suite les configurations suivantes à ce fichier

```
  GNU nano 2.9.3                 /etc/mysql/mariadb.conf.d/99-openstack.cnf

 [mysqld]

 bind-address = 10.0.0.11
 default-storage-engine = innodb
 innodb_file_per_table = on
 max_connections = 4096
 collation-server = utf8_general_ci
 character-set-server = utf8
```

Tel que : bind-address c'est l'adresse ip de gestion du nœud contrôleur : c'est pour permettre l'accès par d'autres nœuds via ce réseau.

Enfin on redémarrer ce service.

```
root@controller:~# service mysql restart
```

D. Fil d'attente :

Utilisé pour la coordination des opérations et des informations d'état entre les services.

Openstack prend en charge plusieurs services de file d'attente de messages, notamment RabbitMQ , Qpid et ZeroMQ . Dans cette installation, on a implémenter RabbitMQ car c'est le plus utilisés et y a beaucoup de documentation sur .

Ce service s'installe uniquement sur le nœud contrôleur.

- Installer les paquets

```
root@controller:~# apt install rabbitmq-server
```

- Ajouter l'utilisateur openstack

```
root@controller:~#  rabbitmqctl add_user openstack RABBIT_PASS_
```

- Autorisez les utilisateurs à configurer, écrire et lire **openstack**:

```
root@controller:~# rabbitmqctl set_permissions openstack ".*" ".*" ".*"
```

E. Le service d'identité Memcached :

Memcached est utilisé par les services d'identité, et s'installe uniquement sur le nœud contrôleur

- Installer les packages

```
root@controller:~#  apt install memcached python-memcache
```

- Ouvrir le fichier /etc/memcached.conf pour configurer le service à utiliser l'adresse ip de traitement du nœud contrôleur.

```
  GNU nano 2.9.3                                /etc/memcached.conf

# Specify which IP address to listen on. The default is to listen o
# This parameter is one of the only security measures that memcache
# it's listening on a firewalled interface.
-l 10.0.0.11
```

- Redémarrer le service.

F. Le service ETCD :

Ce service est installé sur le nœud du contrôleur seulement.

- installer les paquets

```
root@controller:~# apt install etcd_
```

- ouvrir le fichiet /etc/default/etcd et effectuer les modifications suivantes

```
  GNU nano 2.9.3                              /etc/default/etcd

ETCD_NAME="controller"

ETCD_DATA_DIR="/var/lib/etcd"

ETCD_INITIAL_CLUSTER_STATE="new"

ETCD_INITIAL_CLUSTER_TOKEN="etcd-cluster-01"

ETCD_INITIAL_CLUSTER="controller=http://10.0.0.11:2380"

ETCD_INITIAL_ADVERTISE_PEER_URLS="http://10.0.0.11:2380"

ETCD_ADVERTISE_CLIENT_URLS="http://10.0.0.11:2379"

ETCD_LISTEN_PEER_URLS="http://0.0.0.0:2380"

ETCD_LISTEN_CLIENT_URLS="http://10.0.0.11:2379"
```

- Enfin redémarrer les services.

IV. Installation des services d'openstack

1) Le service d'identité keystone

Avant d'installer n'importe quel service, on commence d'abord par lui créer ses bases de données

- Accéder à la base de données mysql

```
root@controller:~# mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 52
Server version: 10.1.41-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

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

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

- Créer la base de données keystone

```
MariaDB [(none)]>  CREATE DATABASE keystone;
```

- Permettre l'accès pour les bases créer

```
MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'localhost' IDENTIFIED BY'opensta
ck';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON keystone.* TO 'keystone'@'%' IDENTIFIED BY'openstack';
Query OK, 0 rows affected (0.00 sec)
```

- Sortir de mysql avec la commande exit.

Après avoir créé la base de données de keystone ,

- On installe les paquets

```
root@controller:~# apt install keystone  apache2 libapache2-mod-wsgi
```

- Ensuite, éditer le fichier /etc/keystone/keysone.conf et modéfier les sections database et token

```
[database]
#connection = sqlite:////var/lib/keystone/keystone.db
connection = mysql+pymysql://keystone:openstack@controller/keystone
#
```

```
[token]
provider = fernet
#
```

- Remplir les tables de la base de données keystone

```
root@controller:~#  su -s /bin/sh -c "keystone-manage db_sync" keystone
```

- Initialiser les référentie

```
root@controller:~#  keystone-manage fernet_setup --keystone-user keystone --keystone-group keystone
root@controller:~#  keystone-manage credential_setup --keystone-user keystone --keystone-group keyst
one
```

- Bootstrap le service d'identité

```
root@controller:~#  keystone-manage bootstrap --bootstrap-password ADMIN_PASS \
>   --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
```

- Configurer le serveur http Apache et cela en éditant le fichier **/etc/apache2/apache2.conf** et configurer **le SERVERNAME**option pour référencer le nœud du contrôleur

```
  GNU nano 2.9.3                          /etc/apache2/apache2.conf

# Note that the use of %{X-Forwarded-For}i instead of %h is not recommended.
# Use mod_remoteip instead.
#
LogFormat "%v:%p %h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\""
LogFormat "%h %l %u %t \"%r\" %>s %O \"%{Referer}i\" \"%{User-Agent}i\"" comb
LogFormat "%h %l %u %t \"%r\" %>s %O" common
LogFormat "%{Referer}i -> %U" referer
LogFormat "%{User-agent}i" agent

# Include of directories ignores editors' and dpkg's backup files,
# see README.Debian for details.

# Include generic snippets of statements
IncludeOptional conf-enabled/*.conf

# Include the virtual host configurations:
IncludeOptional sites-enabled/*.conf

# vim: syntax=apache ts=4 sw=4 sts=4 sr noet
ServerName controller
```

- Redémarrer le service apache
- Configurer le compte administratif

```
root@ubuntu:/home/controller# $ export OS_USERNAME=admin
$: command not found
root@ubuntu:/home/controller#  export OS_USERNAME=admin
root@ubuntu:/home/controller# xport OS_PASSWORD=openstack
xport: command not found
root@ubuntu:/home/controller# export OS_PASSWORD=openstack
root@ubuntu:/home/controller# export OS_PROJECT_NAME=admin
root@ubuntu:/home/controller# export OS_USER_DOMAIN_NAME=Default
root@ubuntu:/home/controller# export OS_PROJECT_DOMAIN_NAME=Default
root@ubuntu:/home/controller# export OS_AUTH_URL=http://controller:5000/v3
root@ubuntu:/home/controller# export OS_IDENTITY_API_VERSION=3
```

Après avoir installé le service keystone , on crée un domaine, des projets, utilisateurs et des rôles.

- Le domaine « par défaut »existe déjà à partir de l'étape d'amorçage de keystone
- Créer un projet de service, à noter que chaque service ajoutée à l'environnement a un utilisateur unique

```
controller@controller:~$ openstack project create --domain default \
> --description "Service Project" service
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Service Project                  |
| domain_id   | default                          |
| enabled     | True                             |
| id          | 68bce9e726034e05bd16f2a054154d4f |
| is_domain   | False                            |
| name        | service                          |
| parent_id   | default                          |
| tags        | []                               |
+-------------+----------------------------------+
```

- Créer un projet pour les taches non administratives, ces dernières doivent utiliser un projet et un utilisateur non privilégiés.

```
controller@controller:~$ openstack project create --domain default \
> --description "Demo Project" myproject
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Demo Project                     |
| domain_id   | default                          |
| enabled     | True                             |
| id          | 500175ff9ea64b328e731640aa690265 |
| is_domain   | False                            |
| name        | myproject                        |
| parent_id   | default                          |
| tags        | []                               |
+-------------+----------------------------------+
```

- Créer un utilisateur non privilégié

```
controller@controller:~$ openstack user create --domain default \
> --password-prompt myuser
User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 11cacade5ace48919b884726c85581c8 |
| name                | myuser                           |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

- Créer un rôle pour l'utilisateur non privilégié

```
controller@controller:~$ openstack role create myrole
+-----------+----------------------------------+
| Field     | Value                            |
+-----------+----------------------------------+
| domain_id | None                             |
| id        | 02a6fe9bb6be403e9eef9c7809117151 |
| name      | myrole                           |
+-----------+----------------------------------+
```

- Ajouter le rôle à l'utilisateur

```
controller@controller:~$ openstack role add --project myproject --user myuser myrole
```

Le service d'identité est maintenant installer et configurer.

Vérification de son fonctionnement :

- Désactivez la variable temporaire **OS_AUTH_URL**et **OS_PASSWORD**la variable d'environnement, ensuite demander un jeton d'authentification

```
controller@controller:~$ unset OS_AUTH_URL OS_PASSWORD
controller@controller:~$  openstack --os-auth-url http://controller:5000/v3 \
> --os-project-domain-name Default --os-user-domain-name Default \
> --os-project-name admin --os-username admin token issue

Password:
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
| Field      | Value
                                                                                                           |
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
| expires    | 2019-07-04T01:07:11+0000
                                                                                                           |
| id         | gAAAAABdHUMvO3cRF1pF7bXA8Bpz3F0_FOnpcZ2L6kfCNHm_lkuE-2wzWTV5WG2UfKBKWrE3CdghmN85zcjJr
89w0Biai5WXXbkZlz-6XDgsO92FlebSe-gkA7LfV9Ia1GEY32hbXDwRIHsdu4tSaCAneE7fZx7HkeRdd2QEeASMvFvumhpUM34 |
| project_id | 949b8a5c544148799023bcc65c84b7be
                                                                                                           |
| user_id    | ef0ff787d26c41f896377b92b0efe9ac
                                                                                                           |
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
```

- **D**emander un jeton d'authentification pour l'utilisateur non privilégié **myuser** crée précédemment

```
controller@controller:~$  openstack --os-auth-url http://controller:5000/v3 \
>  --os-project-domain-name Default --os-user-domain-name Default \
> --os-project-name myproject --os-username myuser token issue

Password:
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
| Field      | Value                                                                                         |
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
| expires    | 2019-07-04T01:08:50+0000                                                                      |
| id         | gAAAAABdHUOSOViU1XWuCQ7oBzOSuftXDFYya9WRKDoOLtC6CsvGspNI7_Zrw6ZMH7seZ1EisAvaUm_4Se7PV
lzbrMnAGCAtEKf19Tjt6UkBZM2VQDip4a6hiOvKOYy4JWCp-dGF9OfzYGZedfk1qQoW8xzKCcjxzhvCFAv7yiW6K4vXUUYKNHA |
| project_id | 500175ff9ea64b328e731640aa690265                                                              |
| user_id    | 11cacade5ace48919b884726c85581c8                                                              |
+------------+-----------------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------------------+
controller@controller:~$ _
```

- **Créer** des scripts d'environnement client OpenStack, ces script sont utilisés comme référence pour charger les informations d'identification appropriées pour les opérations client

- Créer un fichier **admin-openrc** contenant les scripts de référence de l'utilisateur admin d'OpenStack

```
 GNU nano 2.9.3                                    admin-openrc

export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=admin
export OS_USERNAME=admin
export OS_PASSWORD=ADMIN_PASS
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

- Créer un fichier **demo-openrc** contenant les scripts de référence pour l'utilisateur non privilégié d'OpenStack

```
 GNU nano 2.9.3                                    demo-openrc

export OS_PROJECT_DOMAIN_NAME=Default
export OS_USER_DOMAIN_NAME=Default
export OS_PROJECT_NAME=myproject
export OS_USERNAME=myuser
export OS_PASSWORD=1
export OS_AUTH_URL=http://controller:5000/v3
export OS_IDENTITY_API_VERSION=3
export OS_IMAGE_API_VERSION=2
```

- Pour utiliser le client en tant que projet et utilisateur spécifiques, on charge le script l d'environnement client associé avant de les exécuter. Par exemple: demander un jeton d'authentification

```
root@controller:~#  openstack token issue
+------------+-------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------+
| Field      | Value
                                                                                               |
+------------+-------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------+
| expires    | 2019-08-27T17:18:29+0000
                                                                                               |
| id         | gAAAAABdZVfVNWKx-FzbGNC5y8MgDdYsy4xHc6A0G-wMjVD50Shle9Z58Q40y1Y8T0XMtR2y8nZRni1stgHN6
BWjD5n0oNfHEh9z46nADL09cRN2oNLXVV8So9scFKs1_Sl-Yp98JlRz2u3duPsz25Vy7MAC1wK8E6nCc1GcN5oZxUbArCF20mM |
| project_id | 12ce3aeb6e824ddfb65bf79cecdd7fd7
                                                                                               |
| user_id    | b714762d35854e4eb3daba3bd50b87ae
                                                                                               |
+------------+-------------------------------------------------------------------------------------
-----------------------------------------------------------------------------------------------+
```

2) Installation du service d'imagerie (glance) :

Comme on l'a dit précédemment, chaque service utilise une base de données propre à lui, donc avant d'installer n'importe quel service il faut lui créer sa base de données et lui accorder un accès.

```
root@controller:~# mysql
Welcome to the MariaDB monitor.  Commands end with ; or \g.
Your MariaDB connection id is 31
Server version: 10.1.40-MariaDB-0ubuntu0.18.04.1 Ubuntu 18.04

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 (0.00 sec)

MariaDB [(none)]>  GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'localhost' \
    ->
    ->   IDENTIFIED BY 'openstack';
Query OK, 0 rows affected (0.00 sec)

MariaDB [(none)]> GRANT ALL PRIVILEGES ON glance.* TO 'glance'@'%' \
    ->  IDENTIFIED BY 'openstack';
Query OK, 0 rows affected (0.00 sec)
```

- Tapper la commande .**admin-openrc**pour accéder aux commandes CLI pour l'administrateur uniquement.
- Créer les informations d'identification du service
- Créer l'utilisateur glance

```
controller@controller:~$ . admin-openrc
controller@controller:~$ openstack user create --domain default --password-prompt glance
User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 09ad924fd7384872a76abdf683fd1bc6 |
| name                | glance                           |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

- Ajouter le rôle d'administrateur à l'utilisateur glance et le projet de service

```
controller@controller:~$ openstack role add --project service --user glance admin
```

- Créer l'entité service de glance

```
controller@controller:~$ openstack service create --name glance \
>   --description "OpenStack Image" image
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Image                  |
| enabled     | True                             |
| id          | 1c6413f4c55a4b6abda11850d706acf6 |
| name        | glance                           |
| type        | image                            |
+-------------+----------------------------------+
```

- Créer les points de terminaisons de l'API du service image

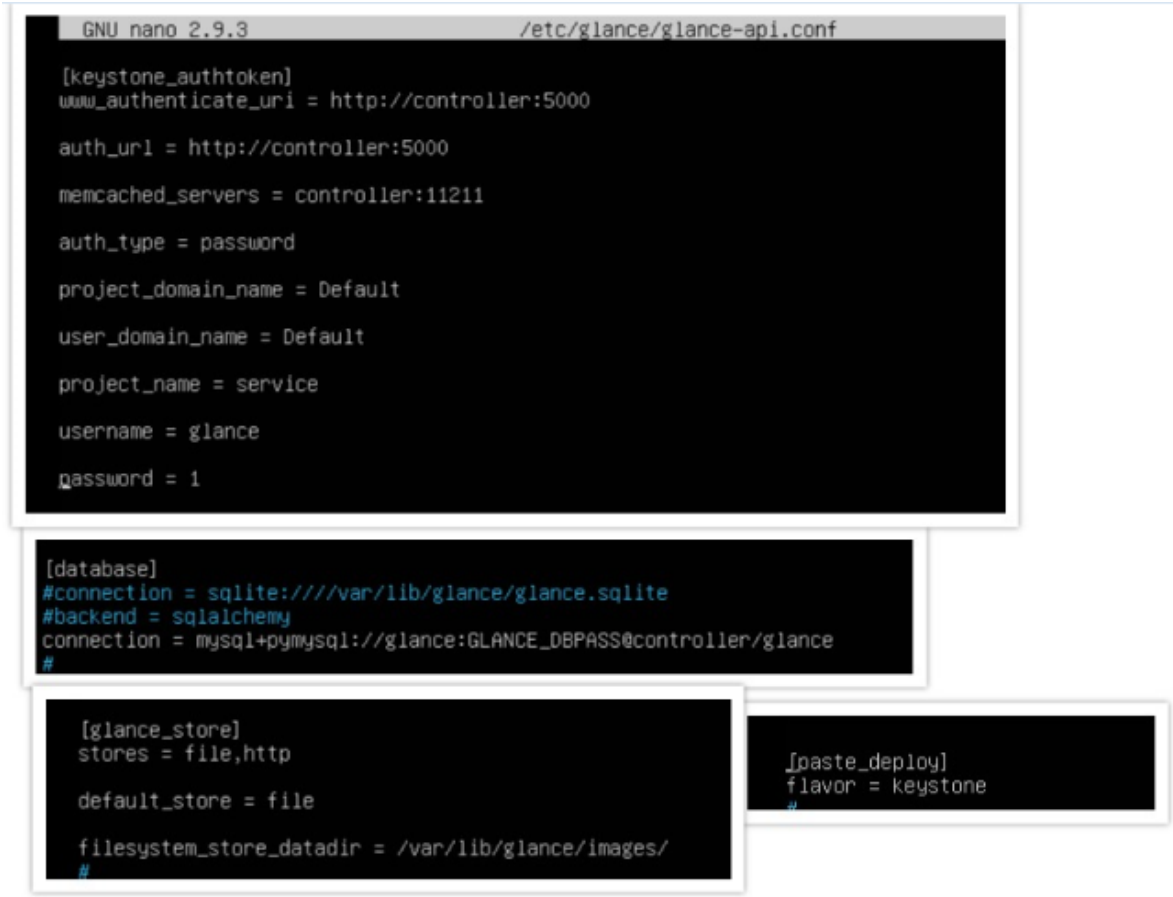
```
controller@controller:~$ openstack endpoint create --region RegionOne \
>  image internal http://controller:9292
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | c1200af814a747ff9b157cca3d2fa0ea |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 1c6413f4c55a4b6abda11850d706acf6 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+
controller@controller:~$  openstack endpoint create --region RegionOne \
>  image admin http://controller:9292
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 12168650c2c3466386f7935f3cef4af7 |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 1c6413f4c55a4b6abda11850d706acf6 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+
```

```
controller@controller:~$ openstack endpoint create --region RegionOne \
>   image public http://controller:9292
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | 505368752d3b4d55a2773d8dc872d720 |
| interface    | public                           |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | 1c6413f4c55a4b6abda11850d706acf6 |
| service_name | glance                           |
| service_type | image                            |
| url          | http://controller:9292           |
+--------------+----------------------------------+
```

- Installer les paquets

```
root@controller:~# apt install glance_
```

- Ouvrir le fichier **/etc/glance/glance-api.conf et** effectuer les configurations suivantes dans les sections [**database],[keystone_authtoken]**, **[paste_deploy]** et **[glance_store]**



- Ouvrir **/etc/glance/glance-registry.conf et** effectuer les configurations suivantes dans les sections **[database], [keystone_authtoken], [paste_deploy]**

- Remplir la base de données

```
root@controller:~# su -s /bin/sh -c "glance-manage db_sync" glance
```

- Redémarrer les services de glance .

Vérification du fonctionnement

- Télécharger l'image source

```
controller@controller:~$ . admin-openrc
controller@controller:~$ wget http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-disk.img
--2019-07-04 10:49:42--  http://download.cirros-cloud.net/0.4.0/cirros-0.4.0-x86_64-disk.img
Resolving download.cirros-cloud.net (download.cirros-cloud.net)... 64.90.42.85, 2607:f298:6:a036::bd
6:a72a
Connecting to download.cirros-cloud.net (download.cirros-cloud.net)|64.90.42.85|:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 12716032 (12M)
Saving to: 'cirros-0.4.0-x86_64-disk.img'

cirros-0.4.0-x86_64-disk 100%[=================================>]  12.13M   150KB/s    in 63s

2019-07-04 10:51:09 (197 KB/s) - 'cirros-0.4.0-x86_64-disk.img' saved [12716032/12716032]
```

- Confirmer le téléchargement

```
root@controller:~# openstack image list

+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 4a6cf4da-8c01-430a-86ab-c0c05f8471a2 | cirros | active |
```

3) Installation du service de calcul nova

a. Sur le nœud contrôleur :

Le service nova utilise 4 base de données : nova-api, nova, nova_cell0 et nova placement .

- Créer les bases de données et leur accorder l'accès approprié, ce sont les memes commandes que celle des autres service juste on change le nom de la table.
- Créer les informations d'identification du service de traitement

o L'utilisateur :

```
controller@controller:~$ . admin-openrc
controller@controller:~$  openstack user create --domain default --password-prompt nova
User Password:
Repeat User Password:
+---------------------+----------------------------------+
| Field               | Value                            |
+---------------------+----------------------------------+
| domain_id           | default                          |
| enabled             | True                             |
| id                  | 3550c3964d664544a12b5a0029891b96 |
| name                | nova                             |
| options             | {}                               |
| password_expires_at | None                             |
+---------------------+----------------------------------+
```

o Ajouter le rôle d'administrateur pour l'utilisateur nova

```
controller@controller:~$ openstack role add --project service --user nova admin
```

o Créer l'entité su service nova

```
controller@controller:~$ openstack service create --name nova \
> --description "OpenStack Compute" compute
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Compute                |
| enabled     | True                             |
| id          | 62cf00612de3444c8e07282fae0411fd |
| name        | nova                             |
| type        | compute                          |
+-------------+----------------------------------+
```

- Créer les points du terminaison des API

```
controller@controller:~$ openstack endpoint create --region RegionOne \
>  placement internal http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | a938788045e74388a570af8f50b3df76 |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | ed6214437a0a43e7912e0f8374aa8724 |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+
controller@controller:~$ openstack endpoint create --region RegionOne \
> placement admin http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | d10fea25fc5b45eeb32eb5eb05981806 |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | ed6214437a0a43e7912e0f8374aa8724 |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+
```

En réalité y a 3 tables des terminaisons d'API, on a oublié lors de l'installation de prendre une capture.

- Créer un utilisateur du service placement et lui accorder le rôle d'administrateur

```
controller@controller:~$  openstack role add --project service --user placement admin
controller@controller:~$ openstack service create --name placement \
> --description "Placement API" placement
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Placement API                    |
| enabled     | True                             |
| id          | ed6214437a0a43e7912e0f8374aa8724 |
| name        | placement                        |
| type        | placement                        |
+-------------+----------------------------------+
```

- Créer l'entrée API de placement dans le catalogue de services


```
controller@controller:~$  openstack role add --project service --user placement admin
controller@controller:~$ openstack service create --name placement \
> --description "Placement API" placement
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | Placement API                    |
| enabled     | True                             |
| id          | ed6214437a0a43e7912e0f8374aa8724 |
| name        | placement                        |
| type        | placement                        |
+-------------+----------------------------------+
```


- Créer les points de terminaison du service API de placement

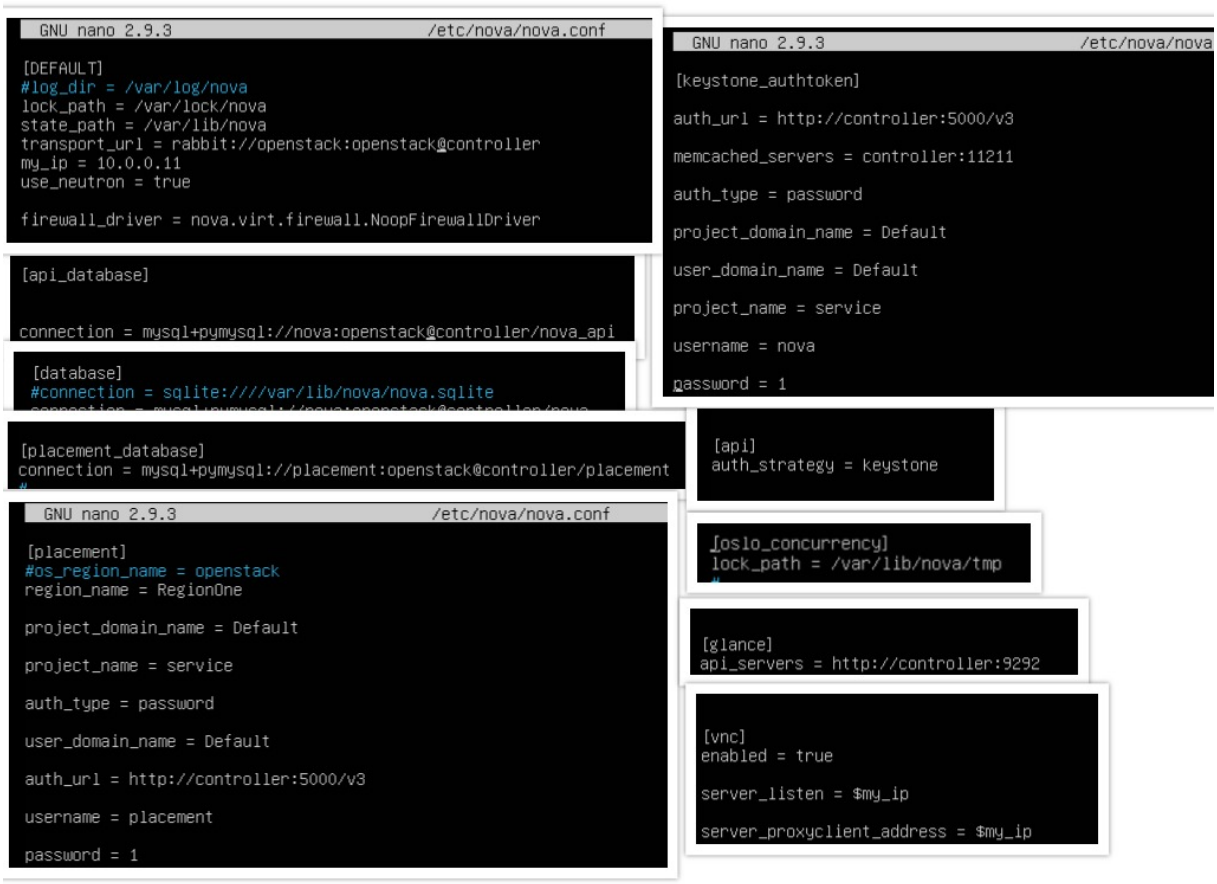
```
controller@controller:~$ openstack endpoint create --region RegionOne \
>  placement internal http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | a938788045e74388a570af8f50b3df76 |
| interface    | internal                         |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | ed6214437a0a43e7912e0f8374aa8724 |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+
controller@controller:~$ openstack endpoint create --region RegionOne \
> placement admin http://controller:8778
+--------------+----------------------------------+
| Field        | Value                            |
+--------------+----------------------------------+
| enabled      | True                             |
| id           | d10fea25fc5b45eeb32eb5eb05981806 |
| interface    | admin                            |
| region       | RegionOne                        |
| region_id    | RegionOne                        |
| service_id   | ed6214437a0a43e7912e0f8374aa8724 |
| service_name | placement                        |
| service_type | placement                        |
| url          | http://controller:8778           |
+--------------+----------------------------------+
```

- Installation et configuration des composants

o Installer les paquets

```
root@controller:~# apt install nova-api nova-conductor nova-consoleauth \
>  nova-novncproxy nova-scheduler nova-placement-api
```

o Dans /etc/nova/nova.conf effectuer les configurations suivantes



- Remplir la base de données nova-api et placement

```
root@controller:~# su -s /bin/sh -c "nova-manage api_db sync" nova
```

- Enregistrer la base de données cell0
- Créer la cellule cell1

```
root@controller:~#  /bin/sh -c "nova-manage cell_v2 create_cell --name=cell1 --verbose" nova
d33e4c5b-104b-474c-9c9b-c3c1b890fc6a
```

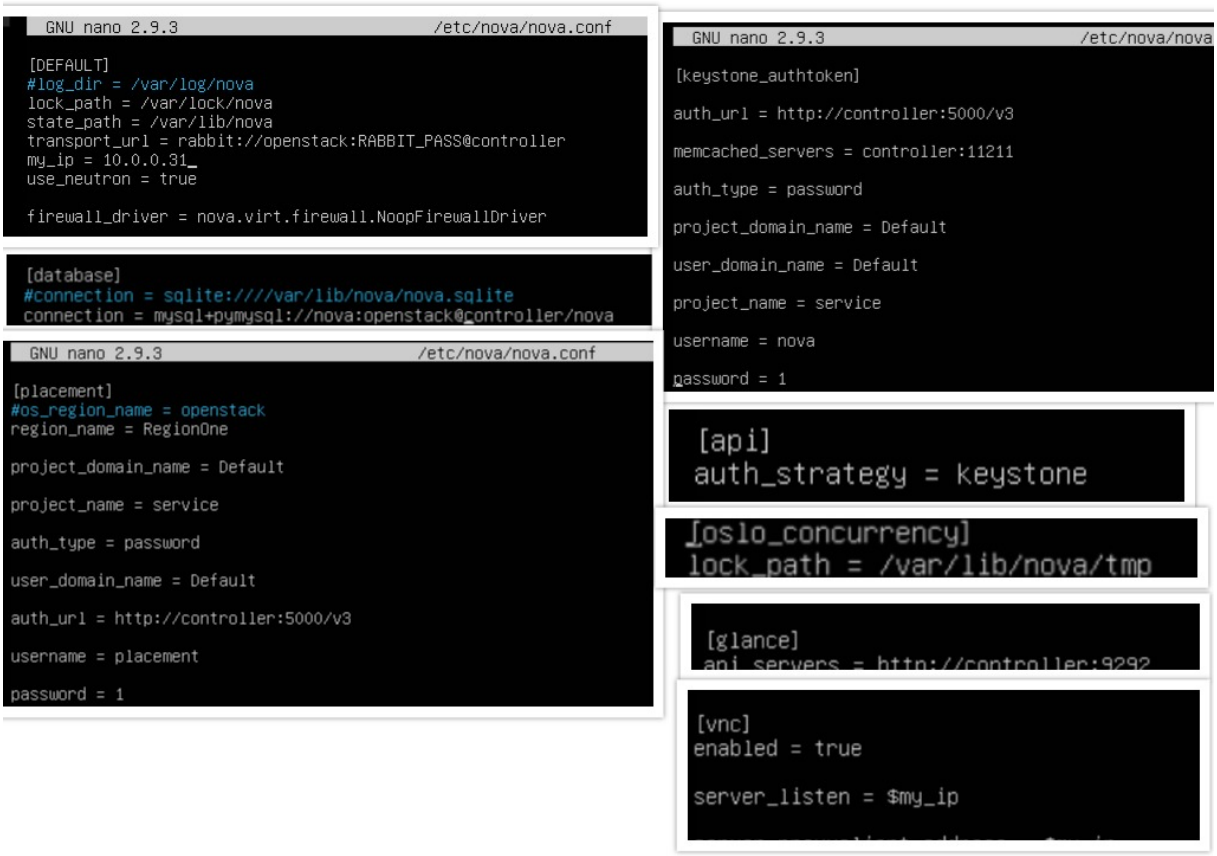
- Remplir la base de données nova
- Vérifier que nova cell0 et cell1 sont enregistré

```
root@controller:~# /bin/sh -c "nova-manage cell_v2 list_cells" nova
+-------+--------------------------------------+----------------------------------+-------------------------------------------------+----------+
|  Name |                 UUID                 |           Transport URL          |               Database Connection               | Disabled |
+-------+--------------------------------------+----------------------------------+-------------------------------------------------+----------+
| cell0 | 00000000-0000-0000-0000-000000000000 |              none:/              | mysql+pymysql://nova:****@controller/nova_cell0 |  False   |
| cell1 | d33e4c5b-104b-474c-9c9b-c3c1b890fc6a | rabbit://openstack:****@controller |    mysql+pymysql://nova:****@controller/nova    |  False   |
+-------+--------------------------------------+----------------------------------+-------------------------------------------------+----------+
```

- Redémarrer tous les services de calcul

b. Sur le nœud de traitement :

- Installer le paquet nova-compute
- Dans /etc/nova/nova.conf, on ajoute les configurations suivantes



- Déterminer si le nœud prend en charge l'accélération matérielle

```
root@compute1:~# egrep -c '(vmx|svm)' /proc/cpuinfo
0
```

- Dans notre cas elle ne le supporte pas car la recommande a retourné 0 donc on chande la section libvirt de /etc/nova/nova-compute.confvirt_type= kvm à virt_type= qemu .
- On redémarre le service nova-compute
- Après l'installation du nœud, on l'ajoute à la base de données de cellules

```
controller@controller:~$ . admin-openrc
controller@controller:~$ openstack compute service list --service nova-compute
+----+--------------+----------+------+---------+-------+----------------------------+
| ID | Binary       | Host     | Zone | Status  | State | Updated At                 |
+----+--------------+----------+------+---------+-------+----------------------------+
|  7 | nova-compute | compute2 | nova | enabled | up    | 2019-07-07T21:15:46.000000 |
+----+--------------+----------+------+---------+-------+----------------------------+
```

- Découvrir les hôtes de calculs

```
root@controller:~# /bin/sh -c "nova-manage cell_v2 discover_hosts --verbose" nova
Found 2 cell mappings.
Skipping cell0 since it does not contain hosts.
Getting computes from cell 'cell1': fdf3f172-0de2-46c4-8976-5116ca9514e9
Checking host mapping for compute host 'compute2': 40f06061-8c0a-4356-a9dc-f8ac5cd72738
Creating host mapping for compute host 'compute2': 40f06061-8c0a-4356-a9dc-f8ac5cd72738
Found 1 unmapped computes in cell: fdf3f172-0de2-46c4-8976-5116ca9514e9
root@controller:~# _
```

Vérifier le fonctionnement

- Lister les composants du service pour vérifier le lancement de chaque processus

```
controller@controller:~$ openstack compute service list
+----+------------------+------------+----------+---------+-------+----------------------------+
| ID | Binary           | Host       | Zone     | Status  | State | Updated At                 |
+----+------------------+------------+----------+---------+-------+----------------------------+
|  1 | nova-scheduler   | controller | internal | enabled | up    | 2019-07-07T21:20:09.000000 |
|  5 | nova-consoleauth | controller | internal | enabled | up    | 2019-07-07T21:20:08.000000 |
|  6 | nova-conductor   | controller | internal | enabled | up    | 2019-07-07T21:20:04.000000 |
|  7 | nova-compute     | compute2   | nova     | enabled | up    | 2019-07-07T21:20:06.000000 |
+----+------------------+------------+----------+---------+-------+----------------------------+
```

- Lister les points de terminaison des API des services installer
- Lister les images pour vérfier la connectivité avec le service image (glance)
- Vérifier que s cellules et l'API de positionnement fonctionnent correctement

```
| Upgrade Check Results                                                |
+----------------------------------------------------------------------+
| Check: Cells v2                                                      |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
| Check: Placement API                                                 |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
| Check: Resource Providers                                            |
| Result: Warning                                                      |
| Details: There are no compute resource providers in the Placement    |
|   service but there are 1 compute nodes in the deployment.           |
|   This means no compute nodes are reporting into the                 |
|   Placement service and need to be upgraded and/or fixed.            |
|   See                                                                |
|   https://docs.openstack.org/nova/latest/user/placement.html         |
|   for more details.                                                  |
+----------------------------------------------------------------------+
| Check: Ironic Flavor Migration                                       |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
| Check: API Service Version                                           |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
| Check: Request Spec Migration                                        |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
| Check: Console Auths                                                 |
| Result: Success                                                      |
| Details: None                                                        |
+----------------------------------------------------------------------+
```

4) Installation du service réseau (neutron)

a. Sur le nœud contrôleur :

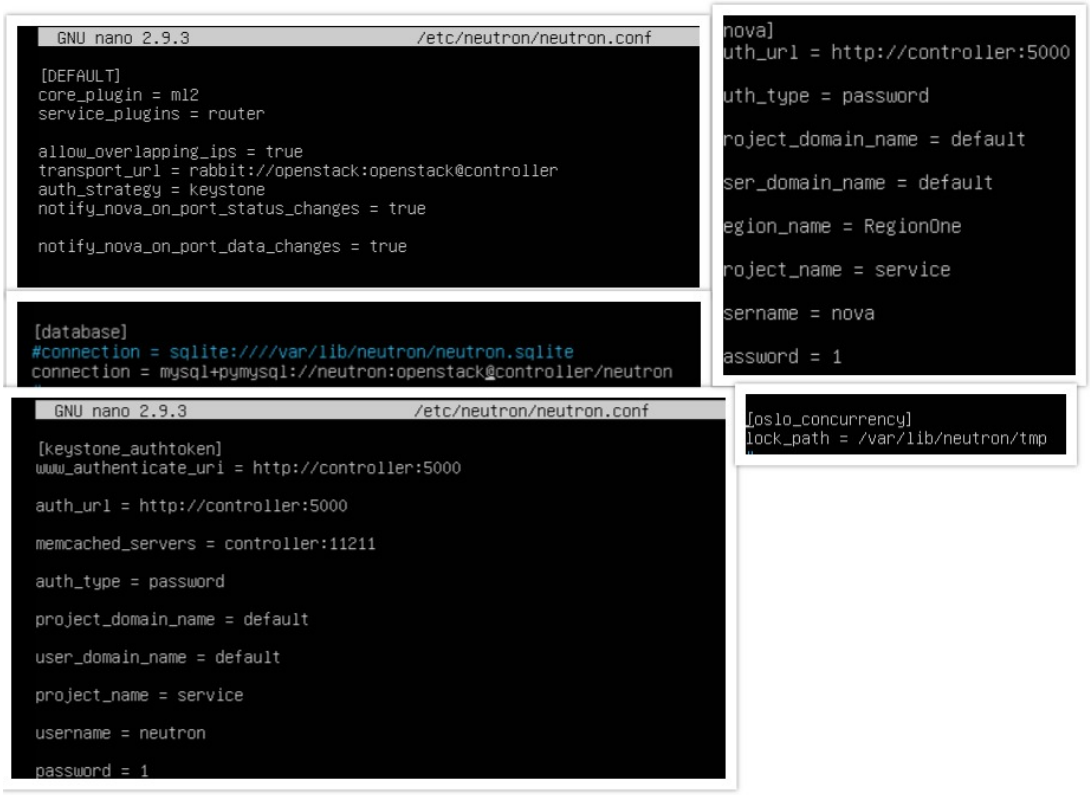
- Créer la base de donnée neutron et lui attribuer l'accès .
- Créer un utilisateur pour le service neutron et lui ajouter le rôle d'administrateur
- Créer l'entité de service neutron
- Créer les terminaisons de l'API du service.

Le service neutron utilise 2 architectures : réseau fournisseur et libre-service.Dans notre projet on a choisi l'architecture libre-service .

- Installer les composants

```
root@controller:~# apt install neutron-server neutron-plugin-ml2 \
>   neutron-linuxbridge-agent neutron-l3-agent neutron-dhcp-agent \
>   neutron-metadata-agent
```

- Editer le fichier **/etc/neutron/neutron.conf**et ajouter les configurations suivantes au sections [database],[default],[keystone_token],[nova] et [oslo_concurrency]



- Pour créer une infrastructure de réseau virtuel de couche 2, on configure Le plug-in ML2 qui tilise le mécanisme de pont Linux. Dans

- Configurer l’agent pont linux qui crée une infrastructure de réseau virtuel pour les instances de couche 2 (pontage et commutation) et gère les groupes de sécurité.
- Configurer l’agent de couche 3 (L3) qui fournit des services de routage et NAT pour les réseaux virtuels en libre-service.

- configurer l'agent DHCP pour fournir des services DHCP aux réseaux virtuels
- Configurer le service de calcul à utiliser le service de mise en réseau , dans /etc/nova/nova.conf on configure la section [neutron]

```
[neutron]
url = http://controller:9696

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 = 1

service_metadata_proxy = true

metadata_proxy_shared_secret = METADATA_SECRET
```

- Remplir la base de données
- Redémarrer le service nova API et le service de mise en réseau

```
root@controller:~# service nova-api restart
root@controller:~# service neutron-server restart
root@controller:~# service neutron-linuxbridge-agent restart
root@controller:~# service neutron-dhcp-agent restart
root@controller:~# service neutron-metadata-agent restart
root@controller:~# service neutron-l3-agent restart
```

b. Sur le nœud de traitement :

- Installer les composants neutron-linuxbridge-agent
- Configurer le service commun du service, donc on édite /etc/neutron/neutron.conf on configure [default],[keystone_authoken] et [oslo_conccurency]
- Configurer les composants de mise en réseau sur un nœud de *calcul*

- Configurer le service de calcul à utiliser le service de mise en réseau et cela en configurant la section [neutron] de /etc/nova/nova.conf
- Redémarrer le service de calcul et l'agent de pont Linux

```
root@compute1:~# service nova-compute restart
root@compute1:~# service neutron-linuxbridge-agent restart
```

Vérification du fonctionnement

Sur le nœud contrôleur, on lance la commande suivante

```
root@controller:~# openstack network agent list
+--------------------------------------+--------------------+------------+-------------------+------
-+-------+---------------------------+
| ID                                   | Agent Type         | Host       | Availability Zone | Alive
 | State | Binary                    |
+--------------------------------------+--------------------+------------+-------------------+------
-+-------+---------------------------+
| 1c52f1b6-ea56-4c01-bcce-ebc5e086370a | L3 agent           | controller | nova              | :-)
 | UP    | neutron-l3-agent          |
| 2b847dd4-00ab-4651-97a0-d5f137eb0c1d | Linux bridge agent | compute1   | None              | :-)
 | UP    | neutron-linuxbridge-agent |
| 4e0b4382-0419-4c16-8508-15754908dfae | Linux bridge agent | controller | None              | :-)
 | UP    | neutron-linuxbridge-agent |
| 90f04729-396c-42ee-9fb3-57f185a172de | DHCP agent         | controller | nova              | :-)
 | UP    | neutron-dhcp-agent        |
| a3bf2e61-9491-41d0-810d-da8762763e49 | Metadata agent     | controller | None              | :-)
 | UP    | neutron-metadata-agent    |
```

La sortie doit indiquer quatre agents sur le nœud du contrôleur et un agent sur chaque nœud de calcul.

5) installation du service de stockage en bloc (cinder)

- créer sa base de données et lui accorder un accès
- créer l'utilisateur du service et lui attribuer le rôle d'administrateur
- Créer les entités **cinderv2**et **cinderv3**s

```
controller@controller:~$ openstack role add --project service --user cinder admin
controller@controller:~$ openstack service create --name cinderv2 \
>  --description "OpenStack Block Storage" volumev2
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Block Storage          |
| enabled     | True                             |
| id          | f8077b4e4cf143ebbc6462b0fbf81d3c |
| name        | cinderv2                         |
| type        | volumev2                         |
+-------------+----------------------------------+
controller@controller:~$ openstack service create --name cinderv3 \
>   --description "OpenStack Block Storage" volumev3
+-------------+----------------------------------+
| Field       | Value                            |
+-------------+----------------------------------+
| description | OpenStack Block Storage          |
| enabled     | True                             |
| id          | dbf0d3c64158492ae52a4147449fc3c |
| name        | cinderv3                         |
| type        | volumev3                         |
+-------------+----------------------------------+
```

- Créer les points de terminaison de l'API du service Block Storage
- Installer les paquets cinder-api et cinder-scheduler
- Sur **/etc/cinder/cinder.conf,** on effectue les configurations suivantes

```
  GNU nano 2.9.3                          /etc/cinder/cinder.conf

[DEFAULT]
rootwrap_config = /etc/cinder/rootwrap.conf
api_paste_confg = /etc/cinder/api-paste.ini
iscsi_helper = tgtadm
volume_name_template = volume-%s
volume_group = cinder-volumes
verbose = True
auth_strategy = keystone
state_path = /var/lib/cinder
lock_path = /var/lock/cinder
volumes_dir = /var/lib/cinder/volumes
enabled_backends = lvm
transport_url = rabbit://openstack:RABBIT_PASS@controller
auth_strategy = keystone
my_ip = 10.0.0.11_
[database]
#connection = sqlite:////var/lib/cinder/cinder.sqlite
connection = mysql+pymysql://cinder:CINDER_DBPASS@controller/cinder
[keystone_authtoken]
www_authenticate_uri = http://controller:5000
auth_url = http://controller:5000
memcached_servers = controller:11211
auth_type = password
project_domain_id = default
user_domain_id = default
project_name = service
username = cinder
password = 1
[oslo_concurrency]
lock_path = /var/lib/cinder/tmp
```

- Redémarrer les service API de calcul et les services de stockage en bloc

```
root@controller:~# service nova-api restart
root@controller:~#  service cinder-scheduler restart
root@controller:~# service apache2 restart
```

6) installation du service horizon

Il existe deux façon d'installer le tableau de bord horizon : soit à partir de la source ou en l'installant à partir des paquets.

Dans notre cas on l'a installer à partir des paquets.

Nb :ce service n'utilise pas une base de données

- on commence par installer le paquet openstack-dashboard
- editer le fichier **/etc/openstack-dashboard/local_settings.py** et effectuer les actions suivantes
  - Configurer le tableau de bord pour utiliser les services OpenStack sur le nœud contrôleur

```
OPENSTACK_HOST = "controller"
```

  - Configurer le **memcached**service de stockage de session

```
SESSION_ENGINE = 'django.contrib.sessions.backends.cache'


CACHES = {
    'default': {
         'BACKEND': 'django.core.cache.backends.memcached.MemcachedCache',
         'LOCATION': 'controller:11211',
    },
}
```

  - Activer l'API d'identité version 3

```
OPENSTACK_KEYSTONE_URL = "http://%s:5000/v3" % OPENSTACK_HOST
```

  - Activer le support pour les domaines

```
OPENSTACK_KEYSTONE_MULTIDOMAIN_SUPPORT = True
```

- Configurer les versions de l'API

```
OPENSTACK_API_VERSIONS = {
    "identity": 3,
    "image": 2,
    "volume": 2,
}
```

- Configurer **Default** comme domaine par défaut pour les utilisateurs qu'on crée via le tableau de bord

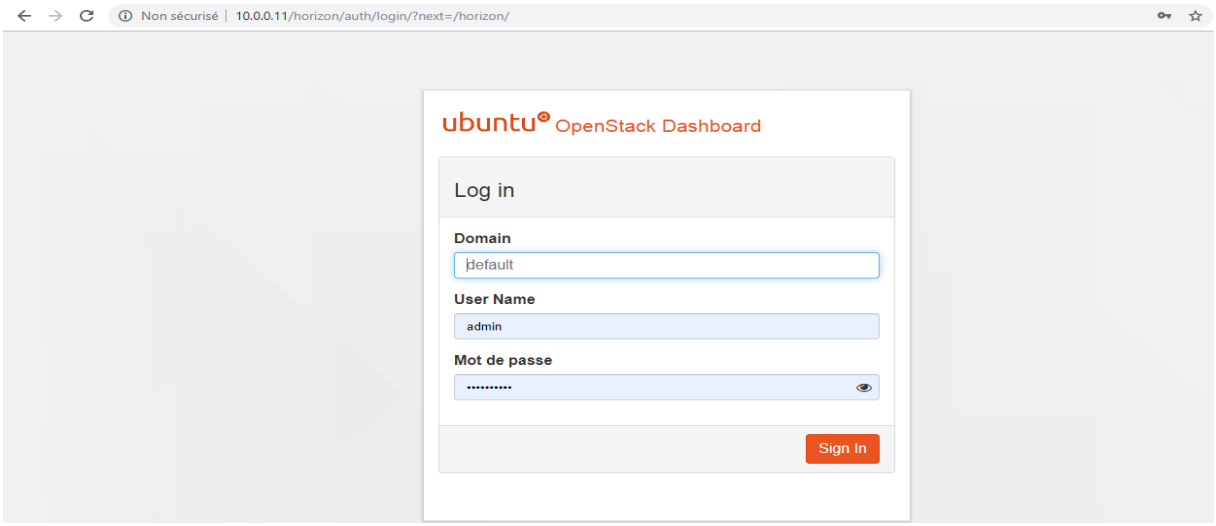
```
OPENSTACK_KEYSTONE_DEFAULT_DOMAIN = 'Default'
```

- Recharger la configuration du serveur Web

```
root@controller:~#  service apache2 reload
```

Vérification du fonctionnement

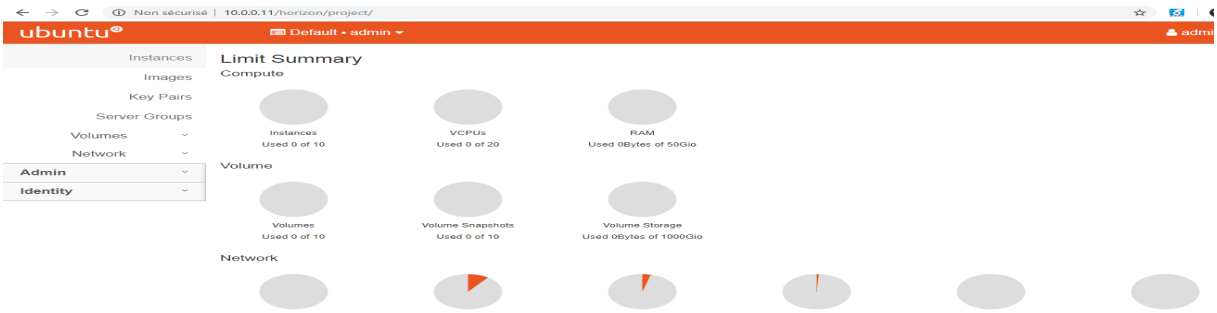
A partir d'un navigateur web on tape l'adresse ip du contrôleur/horizon , la page suivante doit s'afficher



- Après l'installation de tous les service, on peut accéder à la plateforme cloud en entrant **admin** comme utilisateur et le mot de passe c'est celui qu'on a donnée a export OS_PASSWORD dans le fichier admin-openrc (ce fichier a été créé à l'étape où on a installé keystone).

La page d'accueil d'OpenStack est la suivante :

Annexe A:Installation OpenStack



Annexe B : Création d'une instance OpenStack

La création d'une instance OpenStack peut se fait soit via le Dashboard ou le Shell.

A. En utilisant le Shell

Avant de créer une instance OpenStack il faut des prérequis comme le réseau aux quelle on va la liée, le système d'exploitation qu'elle va utiliser donc l'image iso de ce système ….

1- Création d'un réseau virtuelle : comme on l'a expliqué précédemment il existe deux option de réseau dans OpenStack soit un réseau fournisseur ou un réseau libre-service. Dans cette annexe on crée un réseau fournisseur le plus simple. Les étapes suivit sont dans le lien [48].

2- Créer une saveur (gabarit de la machine) on a utilisé la plus petite saveur qui consomme 512 Mo de mémoire par instance cause de manque de ressource.

```
root@controller:~#  openstack flavor create --id 1 --vcpus 1 --ram 64 --disk 1 saveur.nano
+----------------------------+-------------+
| Field                      | Value       |
+----------------------------+-------------+
| OS-FLV-DISABLED:disabled   | False       |
| OS-FLV-EXT-DATA:ephemeral  | 0           |
| disk                       | 1           |
| id                         | 1           |
| name                       | saveur.nano |
| os-flavor-access:is_public | True        |
| properties                 |             |
| ram                        | 64          |
| rxtx_factor                | 1.0         |
| swap                       |             |
| vcpus                      | 1           |
+----------------------------+-------------+
```

3- Télécharger une image iso du système qu'on veut installer (à l'installation du service glance on a téléchargé l'image source 'Cirros' à l'aide du format de disque QCOW2) on peut le confirmer par

```
root@controller:~# openstack image list
+--------------------------------------+--------+--------+
| ID                                   | Name   | Status |
+--------------------------------------+--------+--------+
| 4a6cf4da-8c01-430a-86ab-c0c05f8471a2 | cirros | active |
| d2488c3c-56a5-409a-ba4d-090d35673e6f | ubuntu | active |
+--------------------------------------+--------+--------+
```

On a 2 images mais par manque de ressources on a utilisé l'image Cirros

4- Lancer l'instance

```
root@controller:~# openstack server create --flavor m1.nano --image cirros  --nic net-id=0fe1e27f-ff
dd-43c5-b1dd-0cd1314db45a --security-group default   --key-name mykey INSTANCE-Annexe
+-----------------------------+------------------------------------------------+
| Field                       | Value                                          |
+-----------------------------+------------------------------------------------+
| OS-DCF:diskConfig           | MANUAL                                         |
| OS-EXT-AZ:availability_zone |                                                |
| OS-EXT-STS:power_state      | NOSTATE                                        |
| OS-EXT-STS:task_state       | scheduling                                     |
| OS-EXT-STS:vm_state         | building                                       |
| OS-SRV-USG:launched_at      | None                                           |
| OS-SRV-USG:terminated_at    | None                                           |
| accessIPv4                  |                                                |
| accessIPv6                  |                                                |
| addresses                   |                                                |
| adminPass                   | kKQ3P6SBztEq                                   |
| config_drive                |                                                |
| created                     | 2019-08-31T20:50:42Z                           |
| flavor                      | m1.nano (0)                                    |
| hostId                      |                                                |
| id                          | 3c7c003b-3bdb-4c67-b1ce-5790789b1216           |
| image                       | cirros (4a6cf4da-8c01-430a-86ab-c0c05f8471a2)  |
| key_name                    | mykey                                          |
| name                        | INSTANCE-Annexe                                |
| progress                    | 0                                              |
| project_id                  | bec9bd23182447399f3b6a013d613031               |
| properties                  |                                                |
| security_groups             | name='eb8e7b71-9ec1-49ae-a2f9-1e2e249a1457'    |
| status                      | BUILD                                          |
| updated                     | 2019-08-31T20:50:43Z                           |
| user_id                     | 08c43dcba2024b6aa5cd021e908def71               |
| volumes_attached            |                                                |
+-----------------------------+------------------------------------------------+
```

L'état de la machine est 'BUILD' qui veut dire en construction.

5- Confirmer que la création de l'instance

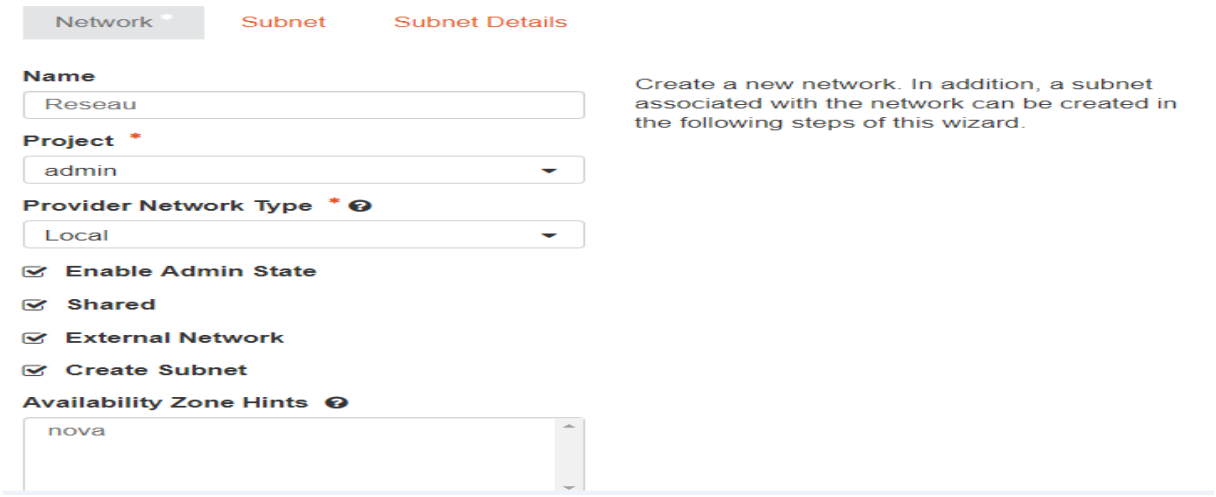
```
root@controller:~# openstack server list
+--------------------------------------+-----------------+---------------+----------------------+----
----+---------+
| ID                                   | Name            | Status        | Networks             | Ima
ge  | Flavor  |
+--------------------------------------+-----------------+---------------+----------------------+----
----+---------+
| 3c7c003b-3bdb-4c67-b1ce-5790789b1216 | INSTANCE-Annexe | ACTIVE        | provider=10.0.0.111 | cir
ros | m1.nano |
```

L'état est 'ACTIVE ' donc l'instance est créer avec succès.

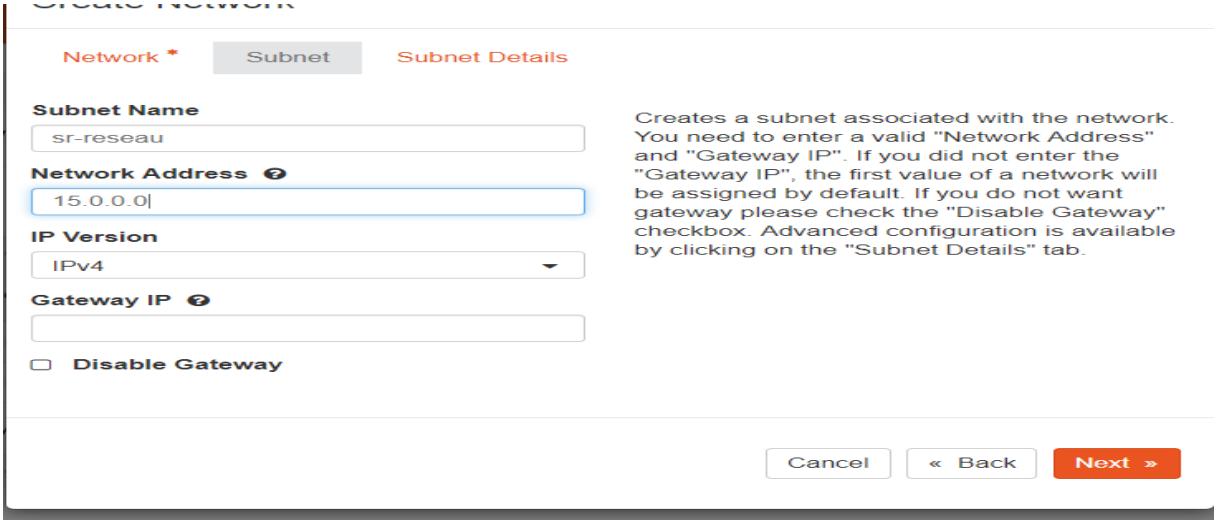
B. En utilisant le Dashboard

1. Créer un réseau : dans Project dans la rubrique network -> network ->create network

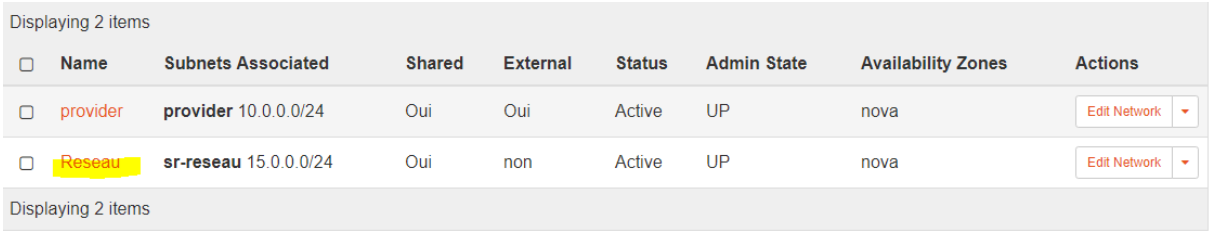
Annexe B: Création d'une instance OpenStack



Ensuite dans Subnet , on crée le sous-réseau



On clique Next ->create



2. Créer un gabarit

Admin->compute->falvors->createflavor

Annexe B: Création d'une instance OpenStack

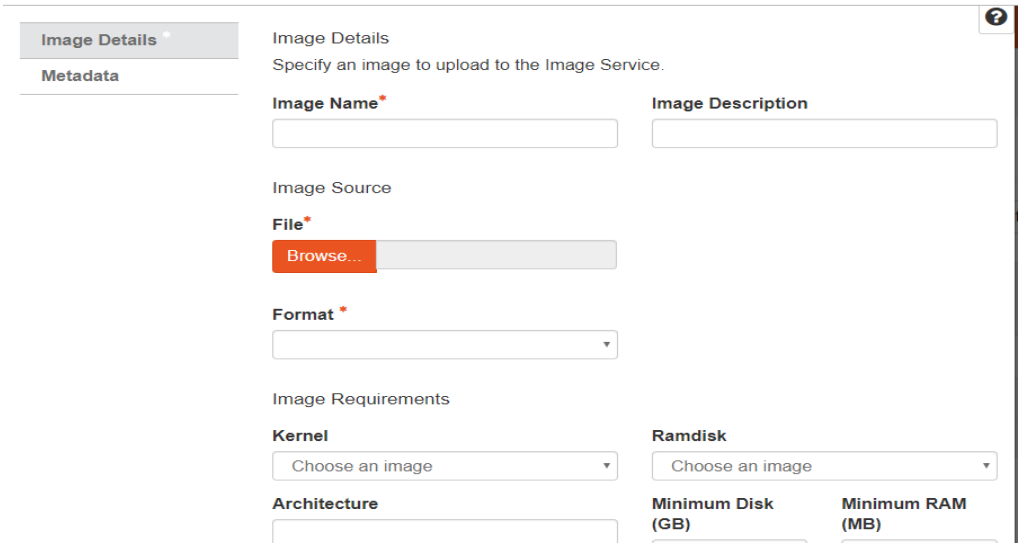
Confirmer la création (l'ID) est diffèrent es autre parce que on l'a laissé auto contrairement aux autre avec la ligne de commande on a précisé l'ID.

Displaying 3 items

Flavor Name	VCPUs	RAM	Root Disk	Ephemeral Disk	Swap Disk	RX/TX factor	ID	Public	Metadata	Actions
m1.nano	1	64Mio	1GB	0GB	0MB	1,0	0	Oui	non	Update Metadata
saveur.nano	1	64Mio	1GB	0GB	0MB	1,0	1	Oui	non	Update Metadata
saveur2	1	64Mio	0GB	0GB	0MB	1,0	d6215f3a-7b0c-4012-a76c-94f1a1605103	Oui	non	Update Metadata

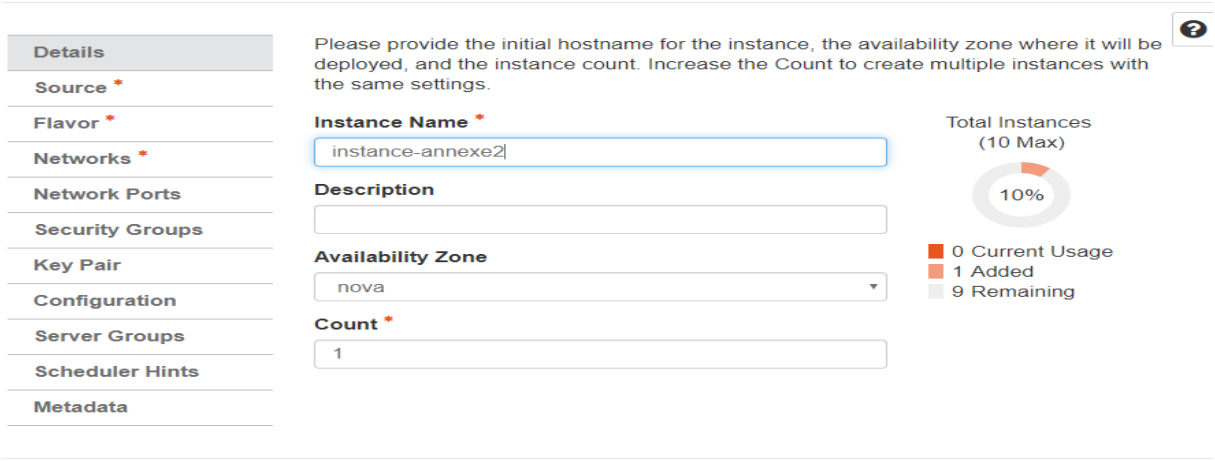
Displaying 3 items

3. Télécharger l'image : dans Admin->image ->create image



4. Création de l'instance : Project->compute->instance->launch instance

Annexe B: Création d'une instance OpenStack



On décent pour continuer de saisir les autres informations puis on clique sur launch instance.

Liste de référence :

[1] KELLA Abedelaziz, vers une gestion de migrations des machines virtuelles pour améliorer la qualité de service dans le Cloud Computing, thèse de doctorat, université d'Oran,2015.

[2] https://www.cisco.com/c/fr_dz/solutions/cloud-systems-management/intelligent-automation-cloud/index.html

[3] https://www.ibm.com/fr-fr/cloud/learn/benefits-of-cloud-computing

[4]http://pfmh.uvt.rnu.tn/896/1/cloud.pdf?fbclid=IwAR1OO6fMJjWjJ_ol8hiuRSGbFtqfHb9ug8P6QmymmgROCnZSBt_bhbDZHQU

[5] https://docplayer.fr/562607-Etat-de-l-art-cloud-computing-sogeti-www-sogeti-com-6-8-rue-duret-75016-paris-t-el-33-0-1-58-44-55-66-fax-33-0-1-58-44-55.html#

[6] http://www.cloud-entreprise.info/historique-cloud-computing/#prettyPhoto/0/

[7] Mcgrawhill-cloud computing

[8] JENDOUBI Maher, Mise en place d'une plateforme de virtualisation et déploiement d'une solution Cloud privé open source, mémoire de master, université virtuelle de Tunis,2015.

[9] SADDI Samy, Techniques de Checkpointing pour la tolérance aux fautes dans le Cloud Computing, thèse de doctorat, université d'Oran, 2017.

[10] MESOUR Leila, Tolérance aux pannes dans les infrastructures Cloud Computing, mémoire de master, université de Tlemcen, 2016.

[11] les rencontres du CIMI, sureté de fonctionnement de l'installation industrielle, 2011

[12] https://methodes.pressbooks.com/chapter/surete-de-fonctionnement/

[13] HAOUATI M.S. Architectures innovantes de système commandes de vol, thèse de doctorat, Université deToulouse,2010.

[14] https://moodle.utc.fr/file.php/498/SupportWeb/co/Module_RCSF_53.html

[15]M.G Venkata, P. G. Bridges et P. M. Widener: Using application communication characteristics to drive dynamic MPI reconfiguration. In IPDPS Proceedings of the IEEE International Symposium on Parallel Distributed, 2009.

[16] Review on fault tolerance techniques in Cloud Computing, International journal of computer applications, April 2015.

[17]LoadBalancing auteurs :MASSAOUDI MOHAMED, CHAHINEZ HACHAICHI, AMENI DHAWEFI, ERIJ MAIJED, EMNA BOUGHANMI.

[18]LIMANI Saïd, gestion de tolérance aux fautes dans le Cloud Computing, thèse de magistère, université d'Oran, 2012.

[19] http://cookieconnecte.fr/2017/12/06/comprendre-cloud-computing/

[20]https://www.itiforums.com/fichiers/2011_06_17_13_28_37_LIVREBLANC_CLOUD_BR.pdf

[21] https://www.lecoindesentrepreneurs.fr/le-mode-saas/

[22] https://support.cloudwatt.com/kb/faq/lecloud/quels-sont-les-avantages-du-cloud-computing.html

[23] http://www.renaudvenet.com/cloud-computing-avantages-et-inconvenients-2011-01-26.html

[24]https://www.uniprint.net/fr/cloud-security-challenges-ensure-security/

[25] FRISSOU Yasmine, réplication des données dans le Cloud Computing, mémoire de master, université de Bejaïa, 2017.

[26]DEGBOUCH Hicham, stratégie de réplication de donnée pour gérer la tolérance aux pannes et l'équilibrage en charge dans le Cloud Computing, université de Saida, 2018.

[27]R. Buyya S. Venugopal and R. Kotagiri. A taxonomy of data grids for distributed data sharing, management and processing. *ACM computing survey*,2007.

[28]XavierBesseron, tolérance aux fautes et reconfiguration dynamique pour les applications distribuées à grande échelle, thèse de doctorat, université de grenoble, 2010.
[29] S.Rajesh, K.R.Devi., "ImprovingFaultTolerance in Virtual Machine Based Cloud Infrastructure", International Conference on Innovations in Engineering and Technology, 2014

[30]W. Zhao, P. M. M.Smith, and L. E. Moser, "FaultTolerance Middleware for Cloud Computing", Article in Proceedings of the IEEE 3rd International Conference on Cloud Computing, ser. CLOUD "10. Washington, DC, USA: IEEE Computer Society, 2010.

[31] A.D.Meshram, A.S.Sambare, S.D.Zade, "FaultTolerance Model for Reliable Cloud Computing", International Journal onRecent and Innovation Trends in Computing and Communication, 2011

[32] https://www.akamai.com/fr/fr/resources/cdn-and-cloud-services-glossary.jsp

[33] http://www.agentsolo.com/ca/fr/blog/fbsc/les-avantages-de-la-virtualisation

[34] http://www.qsp-systems.com/la-virtualisation-mode-demploi/

[35]https://wapiti.telecomlille.fr/commun/ens/peda/options/st/rio/pub/exposes/exposesrio2009/COLIN-DESBUREAUX/La-Virtualisation-Diff%C3%A9rents-types-de-virtualisation.html

[36] https://docs.openstack.org/ocata/install-guide-ubuntu/overview.html

[37]https://www.redhat.com/fr/topics/openstack

[38]https://blog.storagecraft.com/openstack-introduction/

[39]https://docs.openstack.org/fr/install-guide/overview.html

[40]http://ilearnstack.com/2013/04/26/request-flow-for-provisioning-instance-in-openstack/

[41]https://www.networkcomputing.com/data-centers/openstack-neutron-components-and-concepts

[42] https://www.slideshare.net/stmcginnis/introduction-to-openstack-cinder

[43] https://www.sparkmycloud.com/blog/openstack-glance/

[44] DAD Djouhra, optimisation des Datacenter des Cloud sous contrainte d'énergie consommée, thèse de doctorat, université d'Oran, 2016.

[45]https://docs.openstack.org/nova/rocky/admin/configuringmigrations.html#section-configuring-compute-migrations

[46]https://www.lemagit.fr/definition/Machine-Virtuelle-VM

[47]https://www.cnblogs.com/sammyliu/p/4572287.html?fbclid=IwAR2riTfti8kLzsxg1_Lmkk_BuftxUGmMZUiSTqEbXSI3B10avPTJyG2_Mbg

[48] https://docs.openstack.org/install-guide/launch-instance-networks-provider.html

[49] Mohammed Aleslek, Live migration in private Cloud Computing environment, mémoire de master**, Polytechnique de Porto - École d'ingénierie** (**ISEP**) Portugal , 2016.

[50] Bureau d'expertise Technologique Wygwan Le Cloud Computing : Réelle révolution ou simple évolution ?

[51] https://wwhttps://maestropandy.wordpress.com/2016/05/26/openstack-instance-creation-request-work-flow/w.silicon.fr/special-cloud-3-un-monde-tout-en-aas-nouveau-87306.html

[52] Nicolas GREVET, Le Cloud Computing : évolution ou révolution ?, M2IRT, spécialité SIIC, 2009.

[53] http://www.cloud-entreprise.info/historique-cloud-computing/

[54]https://moodle.utc.fr/file.php/498/SupportWeb/co/Module_RCSF_52.html

[55]https://maestropandy.wordpress.com/2016/05/26/openstack-instance-creation-request-work-flow/

[56]https://docs.openstack.org/neutron/rocky/contributor/internals/live_migration.html