الجمهورية الجزائرية الديمقراطية الشعبية
République Algérienne démocratique et populaire

وزارة التعليم العالي والبحث العلمي
Ministère de l'enseignement supérieur et de la recherche scientifique

جامعة سعد دحلب البليدة
Université SAAD DAHLAB de BLIDA

كليةالتكنولوجيا
Faculté de Technologie

قسم الإلكترونيك
Département d'Électronique

Mémoire de Master

Filière Télécommunications

Spécialité Réseaux & Télécommunications

Présenté par :

BENMEHREZ Hocine Amine

&

MOUALHI Zine El Abidine

Mise en place d'une architecture NFV/SDN Cloud Native (5G)

Promoteur : M. BENSEBTI, et H. MAIZI, Co-promotrice

Année Universitaire 2020-2021

REMERCIEMENTS

Tout d’abord, nous tenons à remercier nos parents pour leur présence constante à nos côtés, leurs encouragements dans les moments difficiles et leur soutien inconditionnel tout au long des années.

Nous tenons également à présenter nos remerciements les plus sincères à Madame MAIZI qui nous a encadrés dans ce projet de fin d’étude et nous a fait profiter de précieuses connaissances en nous consacrant une partie de son temps.

Un grand merci, bien qu’à notre sens il ne saurait être assez grand, également à notre promoteur Monsieur BENSEBTI pour ses remarques objectives et ses conseils avisés lors de l’élaboration de ce projet de fin d’étude.

Nous remercions vivement les membres de jury d’avoir accepté de juger notre modeste travail.

Finalement, nous remercions tous nos amis pour leur soutien et leur aide ainsi que toute personne ayant contribué à l’achèvement de ce projet, directement ou indirectement.

DEDICACES

Nous dédions ce travail

A nos parents,

pour leur amour inconditionnel,

leur soutien infaillible,

leurs sacrifices,

pour nous avoir appris le sens de la vie et nous avoir patiemment expliqué les valeurs de l'être humain,

pour leur bonté,

leur constante disponibilité, particulièrement dans les moments difficiles.

A nos frères et sœurs qui nous ont apportés leur soutien lorsqu'on en avait besoin.

A nos amis qui nous ont aidés de près ou de loin tout au long de l'année.

A toute la famille.

ملخص

لضمان حماية أفضل ، تحتاج الأنظمة التي تستخدم بروتوكولات الاتصال إلى شبكة قوية يمكن إصلاحها بسرعة كبيرة ، والتعامل مع تدفق عالي للمستخدمي الشبكة وتعزيز أمن شبكة الاتصال. محولات إيثرنت التقليدية تجعل هذه المهمة صعبة. في عملنا ، سوف نعرض البرنامج السحابي OSM الذي يقدم حلاً يتغلب على العديد من قيود شبكات Ethernet التقليدية . على وجه الخصوص ،هو يوفر إدارة محسّنة للشبكة بالإضافة إلى ميزات أخرى ضرورية لتلبية متطلبات بروتوكولات الاتصال. سنرى الخطوات اللازمة لتحقيق هدفنا المتمثل في تنفيذ بنية NFV / SDN سحابية أصلية للشبكة الخلوية من الجيل الخامس.

RESUME

Pour garantir une meilleure protection, les systèmes qui utilisent des protocoles de communication ont besoin d'un réseau robuste, qu'on peut réparer très rapidement, de gérer des volumes de trafic élevés et d'améliorer la cybersécurité. Les commutateurs Ethernet conventionnels rendent cette tâche difficile. Au cours de nos travaux, nous montrerons que le logiciel cloud natif OSM offre une solution qui évite de nombreuses limitations des réseaux précédents. Ils offrent notamment une gestion améliorée du réseau ainsi que d'autres caractéristiques nécessaires pour répondre aux exigences des protocoles de communication. Nous allons voir les étapes nécessaires qui nous ont permis d'atteindre notre objectif qui est de mettre en place une architecture NFV/SDN cloud native pour le réseau cellulaire 5ème génération.

ABSTRACT

To ensure better protection, systems that use communication protocols need a robust network that can be repaired very quickly, handle high traffic volumes and enhanced cybersecurity. Conventional Ethernet switches make this task difficult. In our work, we will show that the cloud native software OSM offers a solution that overcomes many limitations of traditional Ethernet networks. In particular, they offer improved network management as well as other features necessary to meet the requirements of communication protocols. We will see the steps necessary to achieve our goal of implementing a cloud native NFV / SDN architecture for the 5th generation cellular network.

TABLE DES MATIERES

LISTE DES FIGURES

Chapitre 1

Chapitre 2

Chapitre 3

LISTE DES ABREVIATIONS

API: Application Programming Interface

CAAS: Containers-as a Service

Cloud RAN: Cloud Radio Access Network

CNI: Container Networking Interface

IAAS: Infrastructure as a Service

IP: Internet Protocol

IoT: Internet of things

IT: Information Technology

NAT: Network Address Translation

NS: Network service

NFV: Network function virtualization

OS: Operating System

OSM: Open Source MANO

PAAS: Platform as a Service

SDN: Software Defined Networking

SAAS: Software as a Service

TIC : Technologies de l'information et de la communication

VIM: Virtualized Infrastructure Manager

VM: Virtual Machine

SDN: Software Defined Network

INTRODUCTION GENERALE

Le « Cloud Native Computing » [1] permet aux organisations de créer et d'exécuter des applications évolutives dans des environnements modernes et dynamiques tels que les Clouds publics, privés et hybrides.

Ce modèle donne un accès réseau omniprésent, pratique et à la demande à un pool partagé de ressources informatiques configurables (par exemple, réseaux, serveurs, stockage, applications et services) qui peuvent être rapidement provisionnés et publiés avec un effort de gestion minimal ou interaction avec le fournisseur de services.

Les réseaux d'opérateur mobile n'ont pas été conçus au départ pour fonctionner dans le Cloud [2]. Loin s'en faut. Ces réseaux sont dominés par des équipements dédiés très puissants mais peu flexibles et du matériel spécifique tels que des commutateurs et des passerelles.

En investissant dans le Cloud [3], les opérateurs peuvent moderniser leur propre infrastructure pour réduire les coûts, rationaliser les opérations et accroître l'agilité des services tout en exploitant de nouveaux services. Cette idéologie a déjà été mise en œuvre dans le noyau 4G, IMS/VoLTE et de plus en plus pour le Cloud RAN (Radio Access network).

L'application pratique du Cloud Computing repose sur l'utilisation de technologies logicielles très flexibles pour créer des systèmes faiblement couplés qui sont très faciles à observer, à gérer et à modifier, ce qui est justement en adéquation avec notre objectif d'atteindre un véritable Cloud Computing natif ou ce qu'on appelle maintenant Telcocloud.

Le Telcocloud, que nous définissons comme le déploiement d'infrastructures de télécommunications virtualisées et programmables (NFV, SDN, automatisation, informatique distribuée, etc.) commence à être adopté par les opérateurs de télécommunications du monde entier, principalement en raison de la transition vers la virtualisation du cœur mobile réseaux en réponse à la croissance du trafic de données et en vue du déploiement des réseaux 5G.

Grâce à cette virtualisation des fonctions réseaux (NFV), Cloud et 5G [4] [5] [6] [7] [8] fusionnent pour donner naissance à un « Cloud 5G » qui génèrera des quantités colossales de données. Il n'y a aucun moyen que les opérations héritées puissent suivre, elles ont besoin de beaucoup plus d'automatisation.

La 5G exige de nouvelles performances, de sorte que le Cloud doit se déplacer vers la périphérie, dans un souci de faible latence, de fiabilité localisée et de pilotage du trafic. Pour répondre à ces exigences, nous avons besoin de l'efficacité du Cloud-native.

En effet, l'adoption du Cloud Computing a provoqué le besoin d'avoir des réseaux dynamiques qui offrent la possibilité de s'adapter rapidement aux changements et aux mises à jour des infrastructures. C'est là qu'interviennent les réseaux définis par logiciels SDN (Software Defined Networking) [9] [10]qui permet une adaptation dynamique des applications.

Ce projet est l'une des premières publications dans le domaine à l'échelle nationale et il représente donc un défi qui doit se hisser aux normes de notre institution et de notre hôte de

stage Mobilis. La solution mise en place devra être modulable et personnalisable pour correspondre aux besoins de l'opérateur Mobilis.

L'objectif de notre projet de fin d'étude est la mise en place d'une infrastructure NFV/SDN Cloud Native en utilisant la plateforme open source kubernetes, les conteneurs docker, l'orchestrateur OSM et la plateforme OpenStack.

Notre mémoire est organisé en trois chapitres :

Dans le premier chapitre, nous présentons le concept de Cloud Computing et les différents types de Cloud ainsi que les concepts de NFV/SDN qui sont la base de notre projet.

Dans le deuxième chapitre nous présenterons notre solution de virtualisation d'un réseau d'opérateur et ses composantes de base.

Enfin dans le dernier chapitre, nous expliquerons les différentes étapes de réalisation de notre solution ainsi que les résultats obtenus.

Nous terminons notre mémoire par une conclusion générale, suivie de quelques références bibliographiques qui apporteront plus de détails sur certains points d'importance.

CHAPITRE 1 :

VIRTUALISATION DES RESEAUX DE TELECOMMUNICATION

Introduction

L'opérateur de téléphonie mobile ATM Mobilis est le leader de la téléphonie mobile en Algérie. C'est un opérateur convaincu de l'adoption des technologies du Cloud et de virtualisation des réseaux (NFV et SDN) et qui vise à bénéficier de leurs larges avantages afin de renforcer la qualité et l'efficience de son réseau. Avec les réseaux SDN (Software-Defined Networking) [11] et les plateformes Cloud, il aspire à devenir le principal moteur technologique de la transformation des TIC à l'ère de l'économie numérique.

La dernière tendance dans l'industrie des télécommunications est de virtualiser les réseaux afin de réduire les coûts en faisant converger les organisations et l'infrastructure informatique et de télécommunication, et de réduire le temps nécessaire à l'introduction de nouveaux services tels que la 5G et l'Internet des objets (IoT).

I.1. Le Cloud Computing

L'informatique dans les nuages, connue sous le nom du « Cloud Computing » [12] [13] [14] [15] est un concept qui consiste à :

- Externaliser le stockage et le traitement informatique traditionnellement localisé sur des serveurs locaux ou sur les postes de travail vers des serveurs distants.
- Proposer des services informatiques sous la forme de services à la demande accessibles de n'importe où, n'importe quand et par n'importe qui grâce à un système d'identification via un périphérique terminal et une connexion à internet.

I.2. Eléments constitutifs du Cloud Computing

Les éléments qui peuvent constituer un système Cloud sont les suivants :

- **Data center**

Un centre de traitement de données « data center » est un site physique dans lequel se trouvent regroupés les équipements constituant le système d'information de l'entreprise (mainframes, serveurs, baies de stockage, équipements réseaux et de télécommunications, etc.) qui peut être internes et/ou externes à l'entrepris. Un data center dispose généralement d'un contrôle sur l'environnement (système de prévention contre les incendies, climatisation, etc.), une alimentation d'urgence et redondante, ainsi qu'une sécurité physique élevée.

Chaque entreprise peut avoir sa propre infrastructure, utilisée par elle seule ou à des fins commerciales, pour stocker les données des entreprises ou des particuliers suivant des modalités bien définies.

- **Plateforme collaborative**

Une plateforme de travail collaboratif est un espace de travail virtuel qui centralise tous les outils liés à la conduite d'un projet et les met à disposition des acteurs. L'objectif de

cette collaboration est de faciliter et d'optimiser la communication entre les individus dans le travail ou dans un procédé. [16]

Les plates-formes collaboratives intègrent en général les éléments suivants :

- Des outils informatiques.
- Une communication améliorée à l'aide de guides ou méthodes de travail en groupe.
- Un service de messagerie.
- Un système permettant de partager des fichiers et des ressources.

- **Virtualisation**

La virtualisation des serveurs se définit comme l'ensemble des techniques matérielles et/ou logiciels qui permettent de faire tourner plusieurs systèmes d'exploitation sur un même serveur physique. Il existe de multiples intérêts à la virtualisation ; on peut citer :

- Réduction du nombre de serveurs, ce qui permet de réduire le coût, le câblage et l'espace, tout en diminuant les coûts d'administration
- Réduction de la consommation d'énergie par le matériel et baisse des coûts de son entretien.
- Sauvegarde du bon fonctionnement du système hôte en évitant de l'endommager tout en pratiquant de multiples tests, développements et installation sur ce dernier.
- Amélioration de la flexibilité et de la rapidité des services.
- Amélioration de la qualité de services.

Cependant, comme expliqué précédemment, la virtualisation consiste à faire fonctionner sur un seul ordinateur physique plusieurs Machines Virtuelles (VM) avec différents systèmes d'exploitation, comme s'ils fonctionnaient sur des ordinateurs distincts. Cependant, cette couche du système d'exploitation consomme énormément de ressources.

Ainsi le fait que les **hyperviseurs** (processus qui crée et exécute les machines virtuelles) des machines virtuelles reposent sur une émulation du hardware, nécessite beaucoup de puissance de calcul.

Pour remédier à ces problèmes, de nombreuses firmes se tournent vers la technologie de **Conteneurisation** qui a été introduite pour pallier aux inconvénients de la virtualisation en optimisant l'utilisation des ressources.

1.3. La conteneurisation

La conteneurisation [17] [18] [19] est une méthode de virtualisation au niveau du système d'exploitation. Elle repose sur l'utilisation des conteneurs qui permettent aux applications d'être déployées de manière fiable et de migrer rapidement entre divers environnements informatiques en regroupant le code, les paramètres de configuration et les dépendances nécessaires à son exécution dans un seul objet.

1.3.1. Concept de fonctionnement

Cette méthode [20] permet d'exécuter des applications avec ses dépendances dans une seule zone nommé « Conteneur » qui se présente comme une enveloppe virtuelle. Ces applications accèdent au même noyau de système d'exploitation tout en regroupant le code d'application avec les fichiers de configuration, bibliothèques et variables et en les stockant dans le conteneur.

Ce package unique de logiciel ou « conteneur » est éloigné du système d'exploitation hôte et, par conséquent, il est autonome et devient portable, capable de fonctionner sur n'importe quelle plate-forme ou Cloud, sans problèmes de compatibilité. Ces deux schémas présentés dans la figure 1.1, d'un environnement de virtualisation à gauche et d'un environnement de conteneurisation à droite permettent de bien identifier les différences entre ces deux technologies.

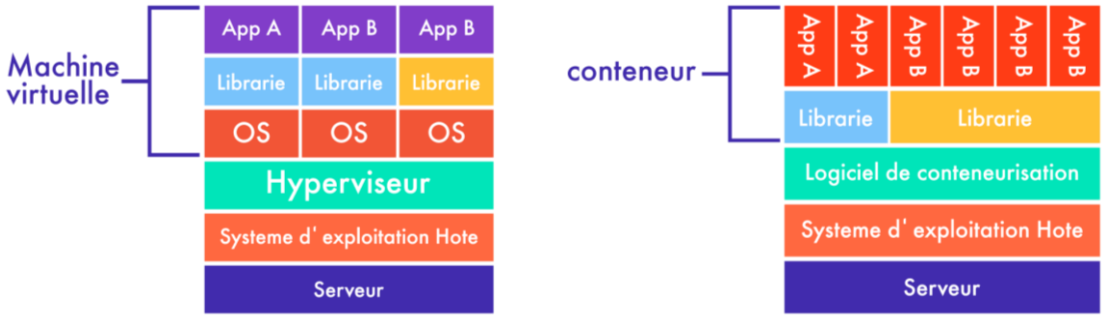

Figure 1. 1 Virtualisation vs. Conteneurisation.

(Original 2021).

1.3.2. Les logiciels de conteneurisation

Le logiciel de conteneurisation, connu sous le nom de Container Runtime, est un programme qui permet d'embarquer une application avec ses dépendances dans un conteneur virtuel qui pourra s'exécuter sur n'importe quel environnement (Linux, Windows…). Nous citerons à titre d'exemple Docker [21] [22] , LXD (LinuX Container) [23] et CRI-O (Container Runtime Interface for Open Container Initiative).

Les principales fonctions d'un Container Runtime sont : la création des conteneurs et la gestion de leurs cycles de vies.

En général, l'architecture du logiciel de conteneurisation (Voir figure 1.5) se décompose en deux niveaux qui sont :

- Logiciel de conteneurisation de bas niveau (Low-Level Container Runtime),
- Logiciel de conteneurisation de haut niveau (High-Level Container Runtime).

- **Les logiciels de conteneurisation de bas niveau**

Avant de présenter le principe de fonctionnement des logiciels de conteneurisation de bas niveau il faut expliquer deux notions de bases dans l'architecture des systèmes, à savoir les espaces de noms et les Cgroups.

-**Les espaces de noms** : ils sont responsables de la virtualisation des ressources système, comme le système de fichiers ou la mise en réseau, pour chaque conteneur. Autrement dit, les espaces de noms fournissent aux processus leur propre vision du système et donc une couche d'isolation. Ainsi chaque conteneur s'exécute dans un espace de noms distinct et son accès est limité à cet espace de noms.

-**Les Cgroups** : ils constituent une fonctionnalité du noyau Linux et permettent de limiter, compter et isoler l'utilisation des ressources (processeur, mémoire, utilisation disques…).

Les containers runtimes de bas niveau sont responsables de la configuration des espaces de noms et cgroups pour les conteneurs, puis de l'exécution de commandes à l'intérieur de ces espaces de noms et cgroups que nous définirons par la suite. Parmi ces containers Runtimes de bas niveau on trouve : Kata-Runtime, Runc.

- **Les logiciels de conteneurisation de haut niveau**

Les containers runtimes de haut niveau sont responsables du transport et de la gestion des images de conteneur, du déballage de l'image et du passage à l'environnement d'exécution de bas niveau pour exécuter le conteneur. En règle générale, les runtimes de haut niveau fournissent une application démon et une API que les applications distantes peuvent utiliser pour exécuter logiquement des conteneurs et surveiller leur cycle de vie. Il existe plusieurs logiciels de conteneurisation de haut niveau tels que : Containerd, CRI-O, Docker.

1.3.3. Les avantages de la conteneurisation

Les principaux avantages de la conteneurisation sont :

• **Légèreté** : Les conteneurs occupent moins d'espace sur le serveur que les machines virtuelles (VM) et ne prennent généralement que quelques secondes à démarrer.

• **Élasticité** : Les conteneurs sont très élastiques et il n'est pas nécessaire d'allouer une quantité donnée de ressources. Cela signifie que les conteneurs sont en mesure de faire une utilisation dynamique plus efficace des ressources du serveur. Lorsque la demande sur un conteneur diminue, les ressources excédentes sont libérées pour être utilisées par d'autres conteneurs.

• **Densité** : la conteneurisation permet des environnements avec un nombre plus important de conteneurs que de machines virtuelles. En effet, les conteneurs n'ont pas besoin d'héberger leur propre système d'exploitation.

• **Performance** : lorsque la demande sur les ressources est importante, les performances des applications sont nettement meilleures avec les conteneurs qu'avec les hyperviseurs. En

effet, avec la virtualisation traditionnelle, le système d'exploitation invité doit satisfaire ses propres besoins en matière de mémoire, au détriment de l'hôte.

• **Efficacité de la maintenance** : avec un seul noyau de système d'exploitation, les mises à jour et autres correctifs du système d'exploitation ne sont effectués qu'une seule fois pour que les modifications prennent effet dans tous les conteneurs. L'exploitation et la maintenance des serveurs sont alors plus efficaces.

I.4. Les formes de déploiement du Cloud Computing

Il existe 4 formes de Cloud Computing : Le Cloud publique (qui est le premier apparu), le Cloud privé, le Cloud communautaire et le Cloud hybride (la combinaison du Cloud public et privé).

- **Le Cloud public**

Un service dans le cloud public est l'équivalent d'un service sur Internet. L'infrastructure et les applications du client sont hébergées chez un prestataire de services, à l'endroit où celui-ci est installé. Le client n'a aucune visibilité ni aucun contrôle sur l'endroit où les services sont hébergés dans le cloud. L'avantage est que l'infrastructure principale est partagée par plusieurs entreprises, mais les données des applications sont séparées logiquement, afin que seuls les utilisateurs autorisés y aient accès. Le service dans le cloud public réduit la complexité et les délais de mise en œuvre, imputables aux tests et au déploiement de nouvelles applications.

- **Le Cloud privé**

Ce type de Cloud est déployé au sein d'une entreprise qui doit ainsi gérer toute seule son infrastructure. Dans ce cas, mettre en œuvre un Cloud privé revient à transformer l'infrastructure interne en utilisant des technologies telles que la virtualisation pour délivrer plus simplement et plus rapidement des services à la demande. Cet environnement présente un avantage certain par rapport au Cloud publique dans les aspects de sécurité et de protection des données.

L'ensemble du matériel est conservé au sein de l'entreprise qui détient et contrôle toutes les ressources en interne. OpenStack, OpenNebula et Eucalyptus sont des solutions pour la mise en place de Cloud privé.

- **Le Cloud communautaire**

L'infrastructure d'un Cloud communautaire est partagée par plusieurs organisations indépendantes pour répondre à des besoins communs. Par exemple, dans le projet Open Cirrus, le Cloud communautaire est partagé par plusieurs universités dans le cadre d'un projet de recherche conjoint. La gestion de son infrastructure peut être prise en charge soit par les organismes utilisateurs, soit par un tiers.

- **Le Cloud hybride**

Le Cloud hybride [24] [25] [26] [27] est la cohabitation d'un Cloud privé et d'un Cloud public dans une organisation. Les données et les applications sont partagées indistinctement entre les deux Cloud (par exemple, un Cloud exclusivement pour les données et un autre pour les applications).

Ce modèle, malgré une gestion de deux Clouds plus contraignante, permet :

- Les avantages du Cloud public et du Cloud privé.
- Un contrôle accru des données privées.

I.5. Les Services du Cloud Computing

Le Cloud Computing était divisé jusqu'à présent en 3 couches : [28] [29] [30]

- Infrastructure (IaaS, Infrastructure as a Service).
- Platform (PaaS, Platform as a Service)
- Application (SaaS, Software as a Service)

I.5.1 Infrastructure as a service « IaaS »

L'infrastructure est un service principal dans le Cloud. Il fournit l'entreprise en composants informatiques (espaces de stockages, équipements réseaux, serveurs, etc). Les utilisateurs peuvent accéder à ces services à la demande via l'internet sans restriction. Les services stockés dans des serveurs virtuels situés dans des Centres des données sont présentés par l'IaaS.

L'infrastructure met à disponibilité des capacités de calcul et de stockage ainsi qu'une connectivité réseau. Les serveurs, les systèmes de stockage, les commutateurs, les routeurs et autres équipements, sont mis à disposition pour gérer une charge de travail demandée par les utilisateurs.

I.5.2. Platform as a service « PaaS »

Plateforme en tant que service est un modèle composé de tous les éléments et services nécessaires pour faciliter les développements des applications. PaaS prépare des environnements spécialisés pour aider les utilisateurs dans la construction, la livraison, l'extension de leurs projets.

I.5.3 Software as a Service « SaaS »

L'application en tant que service est le modèle le plus utilisé dans le monde après le service d'email. C'est un modèle de distribution de logiciels et d'applications hébergées dans des centres de données qui donne la possibilité aux clients d'utiliser ces applications à la demande, via internet, avec une facturation à l'usage réel. Les applications sont prêtes à l'emploi, l'utilisateur étant débarrassé des opérations de maintenance, d'installation de logiciel et de mise à jour, toutes ces opérations étant effectuées par le fournisseur d'application. Dans SaaS l'utilisation des applications reste transparente pour les utilisateurs qui ne se soucient ni de la plateforme, ni du matériel.

Ces dernières années, de nouveaux services de Cloud sont apparus et ne cessent d'apparaitre, comme le SaaS (Security as a Service), NaaS (Network as a Service) et finalement CaaS (Container as a Service) qui représente la solution que nous avons adoptée.

I.5.4 Container as a service (CaaS)

Containers as a service (CaaS) [31] est un service Cloud qui permet aux développeurs de logiciels et aux services informatiques de gérer et déployer des conteneurs, des applications Linux et des clusters en utilisant le concept de conteneurs.

Pour la mise en place de notre solution CaaS, nous utilisons une plateforme logicielle qui permet de créer, déployer et gérer des conteneurs d'applications connue sous le nom de « Docker ». Toutefois, au regard du nombre croissant de conteneurs avec le développement des applications, nous avons dû utiliser une technique d'orchestration des conteneurs connue sous le nom de « kubernetes ». Cette dernière a pour but d'accompagner le logiciel Docker dans la gestion et la maintenance de ces conteneurs. En d'autres termes, Kubernetes permet de gérer le cycle de vie d'un ensemble des conteneurs appelé « pod ».

1.6. Virtualisation du réseau de Mobilis

Les implémentations dans le Cloud du réseau Mobilis doivent passer par trois phases (Figure 1.2):

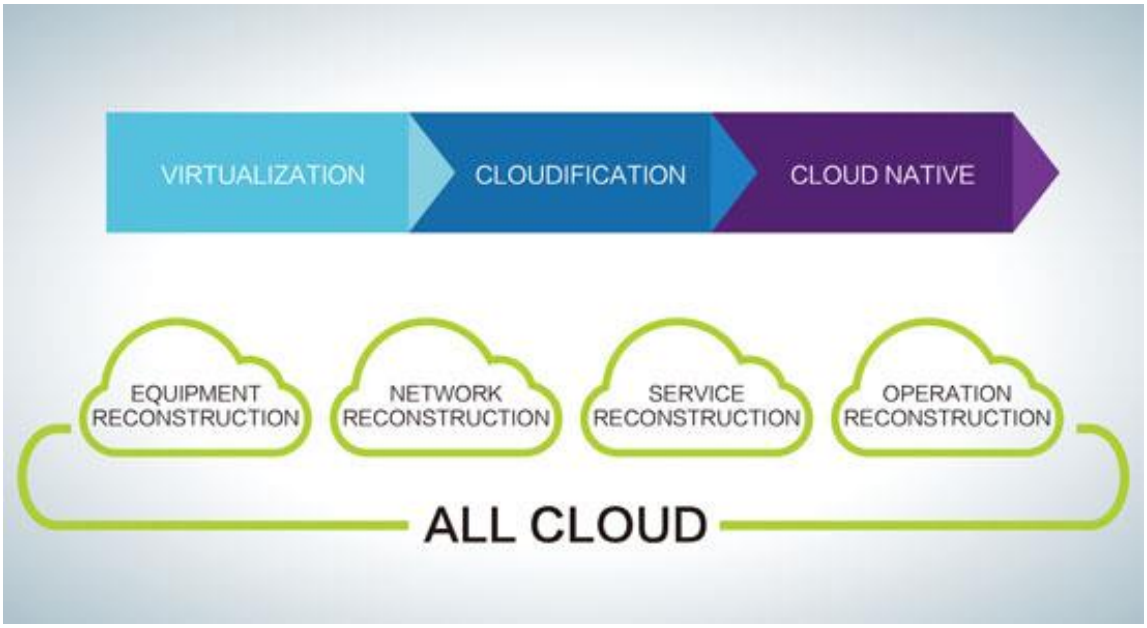

Figure 1. 2 Implémentation dans le Cloud pour Mobilis.

(Mobilis 2021).

- **Virtualisation** [32] [33]**:** c'est la phase de développement NFV (Network Function Virtualization). Elle améliore l'utilisation des ressources en découplant le matériel et les logiciels, permettant ainsi le déploiement de plusieurs fonctions de réseau virtuel (VNF) sur un matériel unifié.

- **Cloudification :** l'architecture VNF est optimisée et reconstruite à l'aide de concepts prenant en charge une architecture logicielle à trois couches. La nouvelle architecture consiste en un équilibrage de charge distribué, une base de données distribuée et des unités de traitement de service sans état. Elle offre la possibilité d'évoluer de manière élastique en quelques secondes sans interruption des sessions en direct et introduit une orchestration automatique des services et des ressources pour améliorer la flexibilité de l'ensemble du système.
- **Cloud natif :** Lorsque les réseaux Mobilis entrent dans la phase cloud natif, le découpage de réseau, l'infrastructure agile [34], les micro-services, les conteneurs, la plate-forme PaaS et d'autres innovations peuvent être intégrés à la construction de nouveaux modèles commerciaux TIC. L'environnement en cloud natif est organisé pour assembler les VNF avec la flexibilité nécessaire pour permettre la configuration et la diffusion des services réseau à tout moment, y compris l'automatisation complète des opérations réseau et des procédures de service.

Le SDN (Software Defined Network) [35] [36] associé à la Network Function Virtualization (NFV) représente une évolution majeure pour les réseaux. Il permet de virtualiser les réseaux de bout-en-bout, de simplifier leur exploitation et leur configuration et d'automatiser leur gestion et leur évolution.

La NFV et le SDN assurent une flexibilité et une réactivité accrues des opérations IT et constituent une réponse adaptée à la demande croissante générée par le développement du Cloud et de la mobilité, tout en garantissant une simplification des architectures, ce qui va engendrer une baisse des coûts de maintenance.

1.6.1. Virtualisation des Fonctions Réseaux (NFV)

La virtualisation des fonctions réseau [37] [38] [39] [40]est une initiative définie par l'ETSI «European Telecommunications Standards Institute» pour virtualiser les fonctions réseau précédemment gérées par du matériel dédié. Il s'agit donc d'une nouvelle façon de concevoir, déployer et gérer les services réseau. NFV sépare les fonctions réseau des systèmes matériels propriétaires et les exécute au niveau logiciel.

La NFV nécessite l'utilisation d'un hyperviseur et d'une couche de virtualisation qui permet l'abstraction du matériel. Le but est de virtualiser le logiciel qui implémente une fonction réseau pour le faire fonctionner sur une machine virtuelle ou un conteneur dans le cloud.

La NFV permet aux opérateurs une réduction des coûts et de la consommation d'énergie induisant ainsi un meilleur retour sur investissement. Ce n'est pas tout, puisqu'elle autorise une plus grande flexibilité pour augmenter, réduire ou faire évoluer les services et qu'elle offre une opportunité d'essayer et de déployer de nouveaux services innovants à moindre risque et avec une plus grande liberté, car la couche de virtualisation permet de fonctionner sur plusieurs types de hardware. De plus, cette solution permet une meilleure réaction en fonction de l'importance du trafic. Si le trafic augmente subitement, on peut monter en charge pour répondre aux besoins.

Architecture standard de NFV « ETSI » :

L'architecture NFV (Figure 1.3) proposée par l'ETSI définit des normes pour la virtualisation des fonctions réseau. Chaque composant de l'architecture est basé sur ces normes afin de fournir un niveau plus élevé de stabilité et d'interopérabilité.

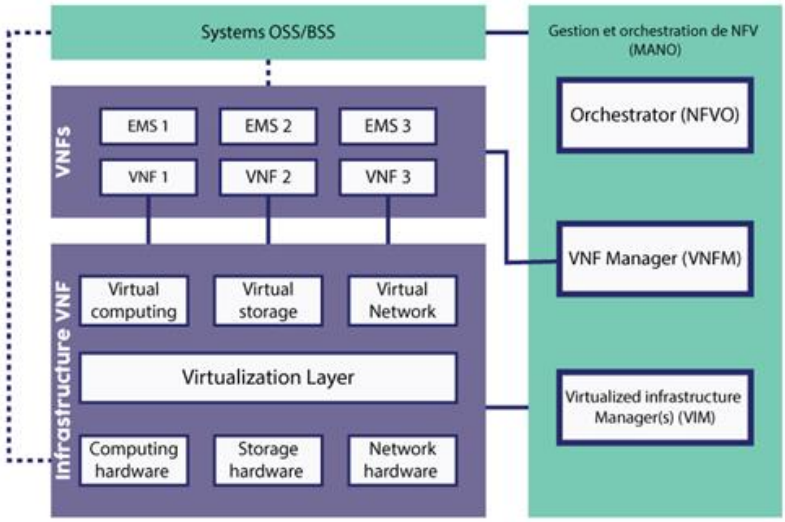

Figure 1. 3 Eléments constituant l'architecture NFV.

(Original 2021).

Une architecture NFV comprend les éléments suivants :

- **Les fonctions réseau virtualisées (VNF) :** ce sont des applications logicielles qui fournissent des fonctions réseau. Nous pouvons citer les services d'annuaire et la configuration IP ainsi que le partage de fichiers.
- **L'infrastructure de virtualisation des fonctions réseau (NFVi):** elle consiste en un ensemble de composants d'infrastructure (calcul, stockage, réseau) sur une plateforme qui prend en charge des logiciels, tels qu'un hyperviseur ou une plateforme de gestion de conteneurs, nécessaire pour exécuter des applications réseau. [41]
- **NFV MANO** (gestion et orchestration de la virtualisation des fonctions réseau) : c'est un cadre architectural pour la gestion et l'orchestration de fonctions réseau virtualisées (VNF) et d'autres composants logiciels visant à faciliter le déploiement et la connexion des services, car ils sont détachés des périphériques physiques dédiés et déplacés vers des machines virtuelles (VM).

NFV MANO [42] [43] [44] comprend trois blocs fonctionnels principaux :

- Les orchestrateurs NFV [45]se composent de deux couches, à savoir l'orchestration des services et l'orchestration des ressources qui contrôlent l'intégration de nouveaux

services réseau et VNF dans un cadre virtuel. Les orchestrateurs NFV valident et autorisent également les demandes de ressources d'infrastructure NFV (NFVI).

- Les responsables VNF supervisent le cycle de vie des instances VNF.
- Les VIM contrôlent et gèrent l'infrastructure NFV qui englobe le calcul, le stockage et les ressources réseau.

L'une des prochaines étapes nécessaires à l'évolution de NFV MANO est d'inclure les contrôleurs SDN dans l'architecture MANO.

1.6.2. Le Contrôleur SDN (Software Defined Network):

Dans une solution Cloud, [46] il est nécessaire de pouvoir consommer chaque ressource en tant que service quel que soit le type de service que nous souhaitons offrir, IaaS ou PaaS. L'objectif du SDN est de fournir des interfaces ouvertes qui permettent le développement de logiciels capables de contrôler la connectivité fournie par un ensemble de ressources réseau et le flux du trafic réseau à travers elles, ainsi que l'inspection et la modification éventuelles du trafic fait sur le net. Ces fonctions primitives peuvent être extraites de services réseau arbitraires dont certains peuvent ne pas être apparents aujourd'hui.

Le SDN est la séparation des fonctions de contrôle d'un réseau de ses fonctions de transmission. Lorsque le SDN est utilisé en conjonction avec la virtualisation des fonctions réseau (NFV), les deux créent des superpositions de réseau virtuel qui existent au-dessus de l'infrastructure de réseau physique.

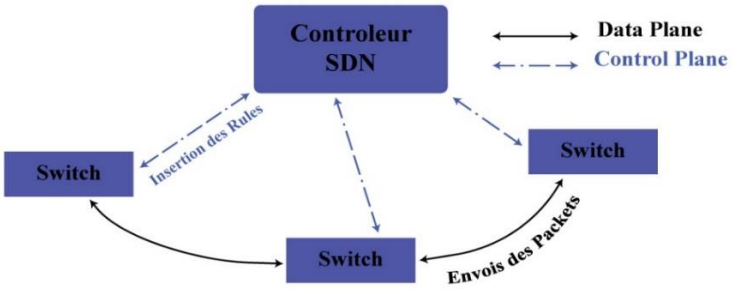

Figure 1. 4 Contrôleur SDN

(Original 2021).

Le SDN (Figure 1.4) obéit à plusieurs principes de base :

- **Découplage du contrôleur et des plans de données :** ce principe nécessite des contrôleurs et des plans de données séparables. Cependant, le contrôle doit nécessairement être exercé dans les systèmes du plan de données. L'interface du plan du contrôleur de données entre le contrôleur SDN et l'élément de réseau est définie de sorte que le contrôleur SDN puisse déléguer des

fonctionnalités importantes à l'élément de réseau, tout en restant conscient de l'état de l'élément de réseau.

- **Contrôle logiquement centralisé :** En comparaison avec le contrôle local, un contrôleur centralisé a une perspective plus large des ressources sous son contrôle et peut prendre de meilleures décisions sur la façon de les déployer. L'évolutivité est améliorée à la fois par le découplage et la centralisation du contrôle, ce qui permet des vues plus globales mais moins détaillées des ressources du réseau. Les contrôleurs SDN peuvent être empilés de manière récursive pour des raisons de mise à l'échelle ou de limite de confiance.

- **Exposé des ressources réseau abstraites et de l'état aux applications externes :** des applications peuvent exister à n'importe quel niveau d'abstraction ou de granularité. Du fait d'une interface qui expose les ressources, l'état peut être considéré comme une interface de contrôleur et la distinction entre application et contrôle n'est pas précise. La même interface fonctionnelle peut être vue sous différents angles par différentes parties intéressées. Tout comme les contrôleurs, les applications peuvent être liées à d'autres applications en tant que pairs ou en tant que clients et serveurs.

Les avantages du SDN :

- Le contrôle du réseau peut être directement programmable car il est séparé des fonctions de transmission.
- L'architecture SDN est agile car l'abstraction du contrôle du transfert permet aux administrateurs d'ajuster dynamiquement le flux de trafic à l'échelle du réseau pour répondre à des besoins changeants.
- Le SDN est géré de manière centralisée ; l'intelligence réseau est (logiquement) centralisée dans des contrôleurs SDN logiciels qui conservent une vue globale du réseau.
- Le SDN permet aux administrateurs réseau de configurer, gérer, protéger et optimiser très rapidement les ressources réseau grâce à des programmes SDN dynamiques et automatisés qu'ils peuvent écrire eux-mêmes (les programmes ne dépendent pas de logiciels propriétaires).
- Le SDN est basé sur des normes ouvertes et indépendantes du fournisseur ; les instructions sont fournies par des contrôleurs SDN au lieu de plusieurs périphériques et protocoles spécifiques au fournisseur. De ce fait, le SDN simplifie la conception et le fonctionnement du réseau.

L'Architecture SDN :

L'architecture SDN permet à un contrôleur SDN de gérer une large gamme de ressources de plan de données. Comme il existe un certain nombre de plans de données et infrastructures différents, le SDN offre la possibilité d'unifier et de simplifier la configuration de cet ensemble diversifié de ressources.

L'architecture reconnaît également la réalité selon laquelle, pour que le SDN réussisse, il doit être déployable dans le contexte d'environnements multi-joueurs largement préexistants, comprenant de nombreuses organisations ou confiances. Dans des environnements moins complexes, tels que des réseaux d'entreprise à échelle limitée, des sous-ensembles fonctionnels appropriés peuvent être profilés à partir de l'architecture.

L'architecture SDN (Figure 1.5) recommande que des modèles et des mécanismes communs soient utilisés chaque fois que possible pour réduire les efforts de normalisation, d'intégration et de validation. Cela implique également l'utilisation des normes existantes ou des meilleures pratiques acceptées lorsque cela est possible. Cette architecture repose sur des contrôleurs pour permettre aux administrateurs réseau de gérer le réseau, tels que :

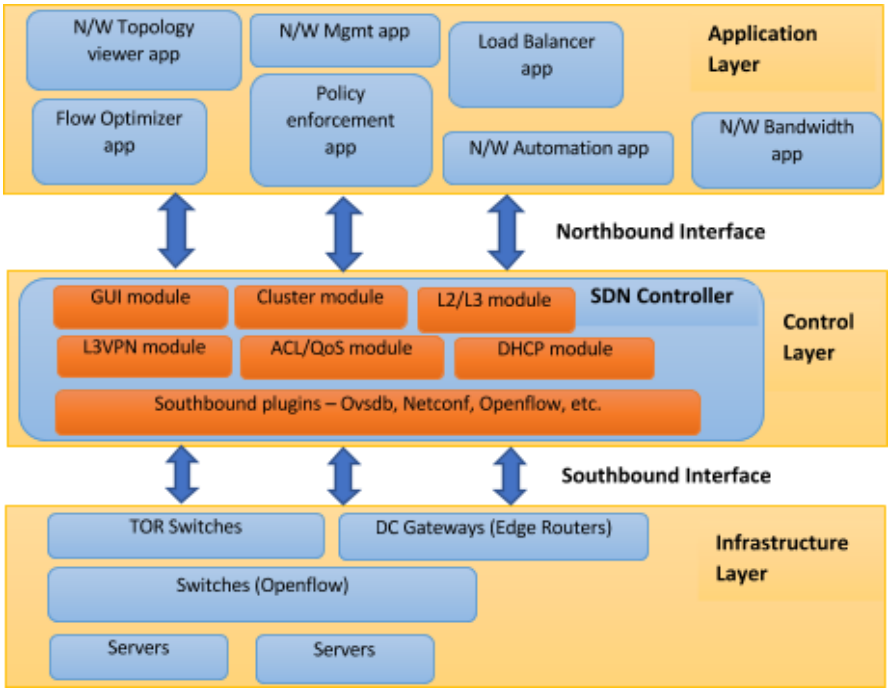

Figure 1. 5 Architecture SDN typique

(Howtoforge 2021).

- **La couche infrastructure :** elle est composée de divers équipements réseau qui forment le réseau sous-jacent pour transmettre le trafic réseau. Cette couche serait la couche physique sur laquelle la virtualisation du réseau serait établie via la couche de contrôle (où les contrôleurs SDN résideraient et géreraient le réseau physique sous-jacent). Elle intègre les ressources qui traitent directement le trafic client ainsi que les ressources de support nécessaires pour garantir une connectivité, une sécurité, une disponibilité et une qualité de virtualisation appropriées.
- **La couche de contrôle :** c'est le cœur du plan de contrôle où la logique intelligente des contrôleurs SDN résiderait pour contrôler l'infrastructure réseau. Cependant, le contrôle est exercé à des degrés divers dans d'autres

plans, le plan contrôleur SDN étant modélisé comme le siège d'un ou plusieurs contrôleurs SDN. C'est le domaine dans lequel chaque fournisseur de réseau travaille pour proposer ses propres produits.

- **La Couche d'application :** c'est une zone qui permet le développement d'applications en mettant à profit toutes les informations relatives au réseau (topologie, état, statistiques, etc.). Plusieurs types d'applications peuvent être développées comme celles liées à l'automatisation du réseau, la configuration et la gestion de réseau, la surveillance du réseau, le dépannage du réseau, les politiques et sécurité du réseau... Ces applications SDN peuvent fournir diverses solutions de bout en bout pour les réseaux d'entreprise et de centre de données du monde réel.
- **L'Interface Northbound :** elle est utilisée pour la communication avec la couche d'application supérieure et est en général réalisée via les API REST des contrôleurs SDN.
- **L'Interface Southbound :** elle est utilisée pour la communication avec la couche d'infrastructure inférieure des éléments de réseau et est en général réalisée via des protocoles Southbound (Openflow, Netconf, Ovsdb, etc.).

Conclusion :

La virtualisation a révolutionné le monde des télécommunications avec l'abstraction du matériel. Etant donné les avancements technologiques et les recherches actuelles menées sur le SDN et le NFV qui permettent un contrôle centralisé des ressources réseau ainsi qu'une une meilleure programmabilité et une orchestration efficace de ces ressources, il est clair que de grands changements vont se produire dans l'industrie des télécommunications.

CHAPITRE 2

APPROCHE CONCEPTUELLE DE VIRTUALISATION DU RESEAU DE MOBILIS : INFRASTRUCTURE LOGICIELLE DE BASE

Introduction

La 4ème génération de téléphonie cellulaire a eu un impact majeur dans le monde des télécommunications. Certes, elle offre des performances plus élevées que celles des réseaux précédents, mais cela n'est plus suffisant. Les systèmes qui utilisent des protocoles de communication ont besoin d'un réseau robuste, capable de se réparer en moins d'une milliseconde, de gérer des volumes élevés de trafic et d'atténuer les vulnérabilités de cyber sécurité. Les commutateurs Ethernet conventionnels rendent cette tâche difficile à réaliser.

Dans ce chapitre, nous allons montrer notre solution qui évite un bon nombre d'obstacles afin d'être adaptée au réseau de 5ème génération.

2.1. La plateforme Open Source Mano (OSM) :

Dans un environnement Cloud natif, le contrôle nécessite un ensemble d'outils, appelé NFV MANO, qui est à l'origine de toute la gestion et de l'orchestration des ressources dans un centre de données virtualisé, y compris les ressources de calcul, de mise en réseau, de stockage et de machine virtuelle (VM). L'objectif principal de NFV MANO est de permettre une intégration flexible en évitant le chaos qui peut être associé à la mise en place rapide des composants du réseau.

NFV MANO (Figure 2.1) peut être divisé en trois blocs fonctionnels :

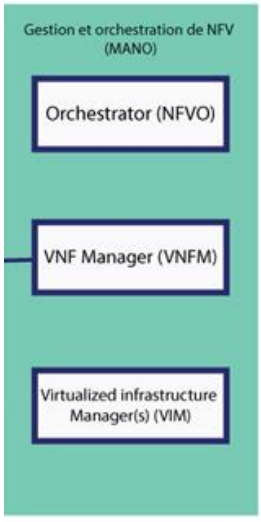

Figure 2. 1 Les différents éléments du NFVMANO

(Original 2021).

- **NFV Orchestrateur :** ce bloc est responsable de l'intégration des nouveaux packages de services réseau (NS) et de fonctions de réseau virtuel (VNF) (gestion du cycle de vie de la SN, gestion globale des ressources,

validation et autorisation des demandes de ressources d'infrastructure de virtualisation des fonctions réseau - NFVI).

- **VNF Manager :** ce gestionnaire supervise le cycle de vie des instances VNF ; il remplit le rôle de coordination et d'adaptation pour la configuration et les rapports d'événements entre l'infrastructure NFV (NFVI) et les systèmes de gestion d'éléments/réseaux. Cependant, dans un environnement cloud natif, nous ne nous intéressons qu'à KVF, qui est l'alternative Kuberentes pour contrôler les fonctions réseau conteneurisées CNF.
- **Gestionnaire d'infrastructure virtualisé (VIM) :** il contrôle et gère les ressources de calcul, de stockage et de réseau NFVI.

L'Open source Mano est une initiative hébergée par l'ETSI pour développer une pile logicielle open source de gestion et d'orchestration NFV (MANO), cohérente avec ETSI NFV, en utilisant des outils et des procédures de travail open source bien établis. Étant hébergé par l'ETSI, le projet OSM (Open Source Mano) collabore étroitement avec le groupe de travail ETSI NFV et suit les spécifications, points de référence et interfaces appropriés.

La plate-forme d'orchestration OSM prend en charge le déploiement de services NFV sur plusieurs VIM. Elle peut facilement installer et intégrer un VIM approprié pour évaluer les capacités de base de NFV. Cependant, la mise en réseau sous-jacente entre les VIM doit être préconfigurée.

Architecture d'OSM :

Le projet Open Source MANO (OSM) se compose de 3 composants de base (Figure 2.2):

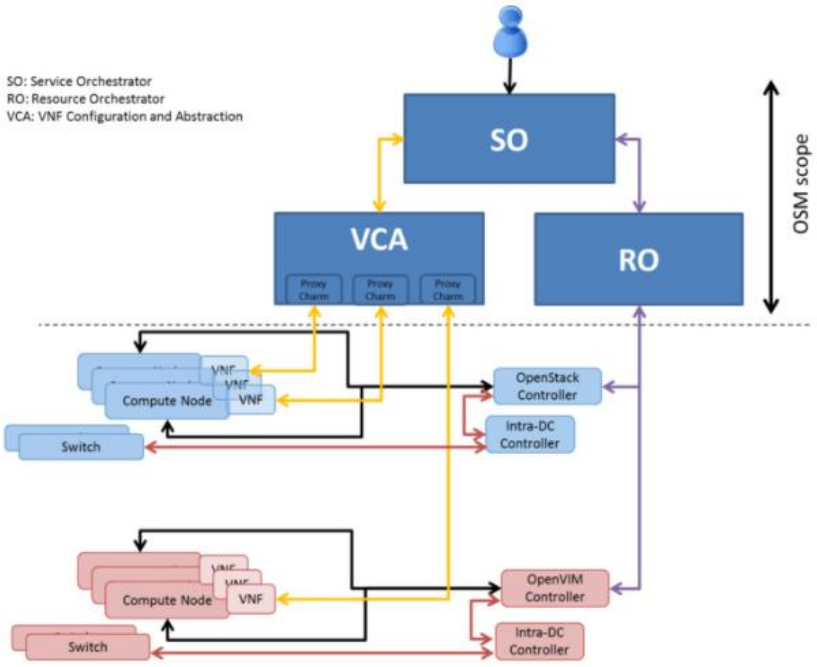

Figure 2. 2 Architecture du projet OSM. (OSM 2020)

- **Resources Orchestrator (RO) :** il est utilisé pour fournir des services chez un fournisseur IaaS particulier à un emplacement donné. Le composant RO est capable de déployer des services réseau sur OpenStack, VMware et OpenVIM. Les composants SO et RO peuvent être mappés conjointement à l'entité NFVO dans l'architecture ETSI MANO.
- **Module VNF Configuration and Abstraction (VCA) :** il exécute la configuration VNF initiale à l'aide de Juju Charms. Pour cela, le module VCA peut être considéré comme un VNFM générique avec un ensemble de fonctionnalités limité.
- **Service Orchestrator (SO) :** il s'occupe de l'orchestration et de l'approvisionnement des services de bout en bout. Le responsable de la sécurité stocke les définitions VNF et les catalogues NS (services réseaux), gère le flux de travail du déploiement de service et peut interroger l'état des services déjà déployés.

2.2. Les solutions NFV

De nombreuses technologies différentes peuvent être utilisées pour réaliser une solution NFV. On trouve parmi celles-ci :

- **OpenStack :**

  OpenStack [47] [48] [49] [50] est une plateforme de virtualisation open source. Il permet aux fournisseurs de services de déployer des fonctions de réseau virtuel (VNF) à l'aide de matériel serveur commercial standard (COTS). Ces applications sont hébergées dans un centre de données mais sont accessibles via le Cloud, le modèle derrière NFV.

OpenStack est souvent utilisé en conjonction avec la technologie de virtualisation des fonctions réseau (NFV) dans les centres de données pour déployer des services Cloud, en particulier les services de communication proposés par les grands fournisseurs de services et les fournisseurs de Cloud.

De nombreuses opérations commerciales adoptent OpenStack afin de pouvoir apporter des améliorations de support, de développement et de service à la plate-forme. De grandes entreprises technologiques telles que Cisco, HP, Oracle et Red Hat ont également construit et commercialisé des outils, des plates-formes et une infrastructure de support autour de la technologie NFV OpenStack.

Open-O :

Le principal avantage visible de la plate-forme OPEN-O est l'architecture flexible et extensible basée sur des microservices. L'approche OPEN-O considère la fourniture de services de bout en bout couvrant plusieurs régions SDN et NFV dès le début. De plus, le projet OPEN-O collabore activement avec la communauté OPNFV pour une intégration

étroite de l'orchestrateur avec la plate-forme OPNFV. Malheureusement la plate-forme OPEN-O nécessite un développement supplémentaire pour être capable de fournir un approvisionnement arbitraire de services NFV, ainsi que plus d'efforts dans l'élaboration de la documentation dont le manque rend difficile la compréhension de la logique du microservice et du flux de travail d'interaction.

CORD/XOS :

Le projet CORD est la bonne option lorsque OpenFlow et les whiteboxes sont la principale option pour l'infrastructure informatique et réseau. La plate-forme prend en compte plusieurs cas d'utilisation, et une grande communauté participe au développement de cette plate-forme. La plate-forme CORD est une solution spécifique autour d'OpenFlow et des technologies associées mises sur le marché.

Gigaspaces Cloudify :

Gigaspaces Cloudify est une plate-forme qui apparaît comme la solution d'orchestration la plus mature parmi les plates-formes examinées. L'architecture enfichable et extensible de Cloudify et le moteur de flux de travail intégré permettent la fourniture de services NFV arbitraires. Cependant, si on considere Cloudify comme un moteur d'orchestration, il faut assurer le risque que le processus de prise de décision concernant la stratégie globale de la plate-forme soit contrôlé uniquement par Gigaspaces.

Docker engine :

Les conteneurs Docker ont fait un grand pas en avant dans la révolutionnarisation de NFV. Comme ils sont légers par rapport aux machines virtuelles, ils nécessitent moins de tracas et de ressources et ils peuvent fournir une isolation satisfaisante des applications s'exécutant dans le même système d'exploitation dans lequel eux-mêmes s'exécutent.

Le conteneur Docker est plus populaire que les conteneurs concurrents, en raison de sa capacité à se charger en tant qu'« image » sur le système d'exploitation hôte de manière simple et rapide. Les dockers sont stockés dans le Cloud sous forme d'images et sont appelés à être exécutés par les utilisateurs en cas de besoin de manière simple. Comme il la rend plus facile à utiliser, l'utilisation du conteneur Docker dans l'architecture NFV se résume en certains points :

- La couche de virtualisation avec les ressources virtuelles est appelée « domaine hyperviseur » (Figure 2.3). La couche de virtualisation est en fait l'hyperviseur qui est responsable de l'abstraction des ressources matérielles d'un domaine de calcul. L'hyperviseur peut partitionner un seul serveur physique en plusieurs mémoires virtuelles et calculs virtuels de manière à ce que chaque entité soit indépendante. La machine virtuelle fournit une virtualisation au niveau du matériel, ce qui signifie que les machines virtuelles classiques prennent un hôte et le partitionnent via un logiciel

hyperviseur. Cela signifie essentiellement que les machines virtuelles sont isolées du système d'exploitation de la machine hôte.

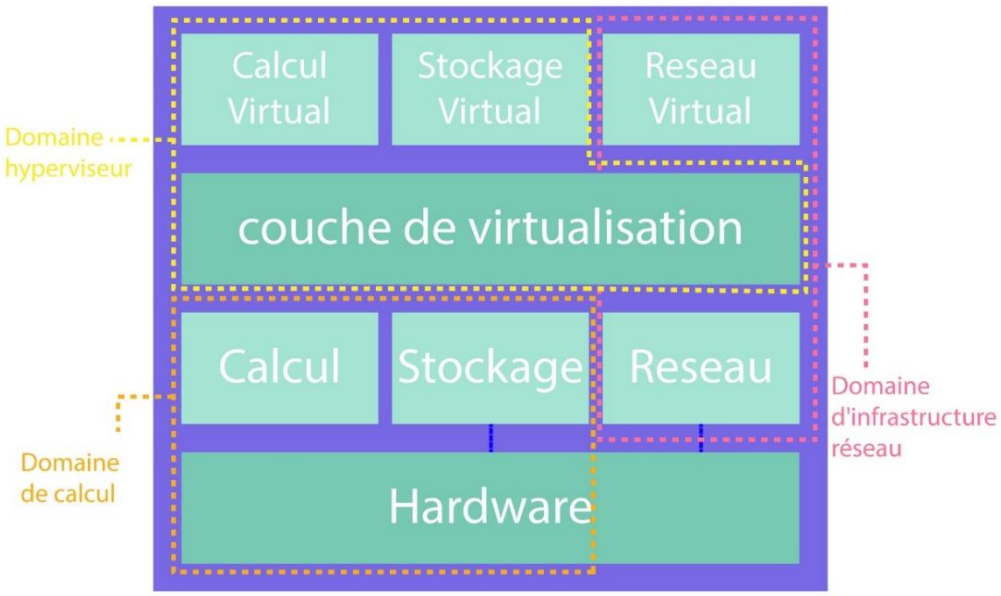

Figure 2. 3 Exploration de l'hyperviseur dans l'infrastructure NFV

(Original 2021).

- Une machine virtuelle fournit un environnement dans lequel un VNF (Virtual Network Function) s'exécute. Chaque machine virtuelle est liée à un VNF. L'architecture est simplifiée en supprimant l'hyperviseur car désormais les conteneurs peuvent conserver une isolation suffisante au niveau du système d'exploitation dans le même système d'exploitation hôte.

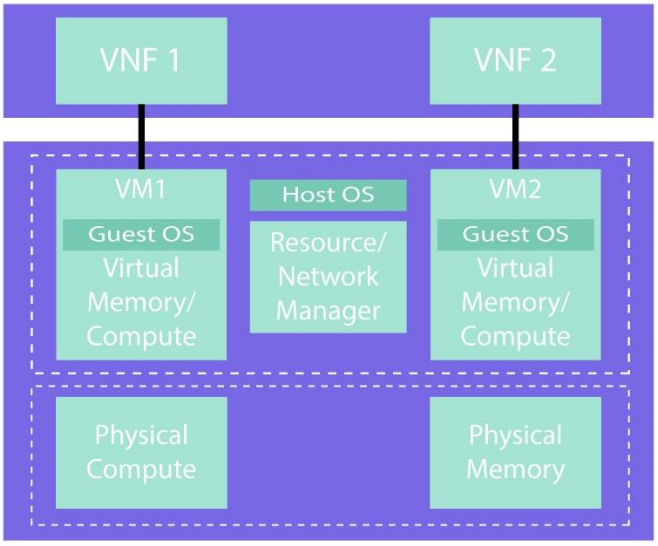

Figure 2. 4 Simplification de la couche de virtualisation

(Original 2021).

- Dans la figure 2.4 ci-dessus VNF1 s'exécute dans le conteneur 1 et VNF2 s'exécute dans le conteneur 2 en offrant les mêmes fonctionnalités que les machines virtuelles et en obtenant les mêmes fonctionnalités et de meilleures performances tout en coupant sur certains blocs comme le Guest OS. Ils sont alors plus légers car il n'y a pas besoin de système d'exploitation invité dans l'environnement de conteneurs (Figure 2.5) et les conteneurs fournissent une virtualisation au niveau du système d'exploitation.

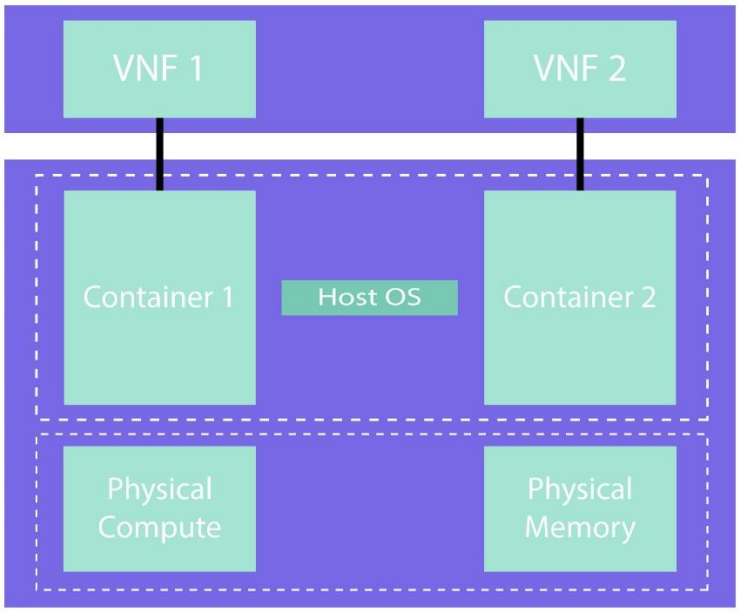

Figure 2. 5 Architecture NFV conteneurisée

(Original 2021).

2.3. La plateforme Kubernetes :

Kubernetes (K8s) [51] [52] [53] [54] [55] est une plateforme open-source extensible et portable permettant d'automatiser le déploiement, la montée en charge et la gestion de conteneurs d'application sur des clusters de serveurs; elle permet de tirer parti des infrastructures qu'elles soient sur site, en Cloud public ou en Cloud hybride.

Cette plateforme aide à gérer facilement et efficacement des clusters au sein desquels sont rassemblés des groupes d'hôtes exécutant des conteneurs Linux. Ces clusters peuvent couvrir des hôtes situés dans des clouds publics, privés ou hybrides. C'est pour cela que Kubernetes représente la plateforme idéale pour héberger les applications cloud-native qui requièrent une montée en charge, comme la diffusion de données en continu et en temps réel via Apache Kafka.

Kubernetes peut être considéré comme une plateforme de conteneurs, une plateforme de microservices ou une plateforme cloud portable et même beaucoup plus, mais elle n'est pas une solution PaaS (Platform as a Service).

Kubernetes opère au niveau des conteneurs plutôt qu'au niveau du matériel. Cette plateforme fournit une partie des fonctionnalités des offres PaaS, telle que le déploiement, la montée en charge, l'équilibrage de charge, le logging et la surveillance. Cependant, ces implémentations par défaut sont optionnelles et interchangeables. Kubernetes fournit les bases permettant de construire des plateformes orientées développeurs, en laissant la possibilité à l'utilisateur de faire ses choix.

Conçue selon les mêmes principes qui permettent à Google de gérer des milliards de conteneurs par semaine, Kubernetes peut évoluer sans accroître l'équipe des opérations et est suffisamment flexible pour fournir les applications de manière cohérente et simple, quelle que soit la complexité des besoins.

2.3.1. Les Pods :

Les pods [56] sont des petites unités de calcul déployables qu'on peut créer et gérer dans Kubernetes. Un pod est un groupe d'un ou plusieurs conteneurs, avec des ressources de stockage et de réseau partagées et une spécification de la manière d'exécuter les conteneurs.

Kubernetes fournit les moyens de prendre en charge le déploiement basé sur des conteneurs dans les Clouds Platform-as-a-Service (PaaS), en se concentrant spécifiquement sur les systèmes basés sur des clusters. Elle autorise le déploiement de plusieurs « pods » sur des machines physiques, permettant la montée en charge d'une application avec la modification de la charge de travail. Chaque pod peut prendre en charge plusieurs conteneurs Docker qui peuvent utiliser des services (par exemple, fichier système et E / S) associé à un pod. Avec un intérêt significatif pour la prise en charge des applications Cloud native (CNA), Kubernetes fournit une approche utile pour y parvenir.

Cela s'appelle des pods à cause de la façon dont elles sont groupées ressemble à des cosses de pois (Pea pods).

2.3.2. Les fonctionnalités Kubernetes :

Kubernetes dispose de beaucoup de fonctionnalités qui la rendent la mieux adaptée à notre projet :

- **Orchestration du stockage :** montage automatique du système de stockage sélectionné, que ce soit à partir du stockage local, d'un fournisseur de cloud public tel que GCP ou AWS ou d'un système de stockage réseau tel que NFS, Cinder ou Flocker.
- **Endpoint Slices :** un suivi évolutif des réseaux Endpoints dans un cluster Kubernetes.
- **Déploiements et restaurations automatisés :** Kubernetes déploie progressivement les modifications apportées à l'application ou à sa configuration, tout en surveillant l'intégrité de l'application pour s'assurer qu'elle ne tue pas toutes les instances en même temps. En cas de problème, Kubernetes annule le changement.

Kubernetes évolue rapidement, mais il manque encore un certain nombre de fonctionnalités importantes pour la gestion et les performances des fonctions de réseau virtuel

(VNF) basées sur des conteneurs dans un environnement NFV. Par défaut, Docker fournit sa propre passerelle Linux (docker0). Ce réseau de pont est automatiquement créé lors de la première installation du Docker. Le Docker démon connecte les conteneurs à ce réseau de pont. Dans notre configuration, nous n'utilisons pas le réseau par défaut de Docker. À la place, nous utilisons l'interface CNI (Container Networking Interface) de Kubernetes.

2.3.3. L'interface CNI (Container Networking Interface) de Kubernetes:

L'interface CNI est une spécification et un ensemble de bibliothèques pour une solution de mise en réseau des conteneurs basée sur des plugins [57] (Figure 2.6). Cette solution réseau a été adoptée par divers moteurs d'orchestration open source, y compris Kubernetes. L'interface CNI se préoccupe uniquement de la connectivité réseau des conteneurs et de la suppression des ressources allouées lorsque les conteneurs sont supprimés.

L'un des plugins CNI les plus couramment utilisés est le plugin flannel qui fournit un réseau de superposition de base (IP unique et routable) pour les conteneurs dans un cluster Kubernetes. Il affecte un sous-réseau à chaque hôte à utiliser avec les environnements d'exécution de conteneur.

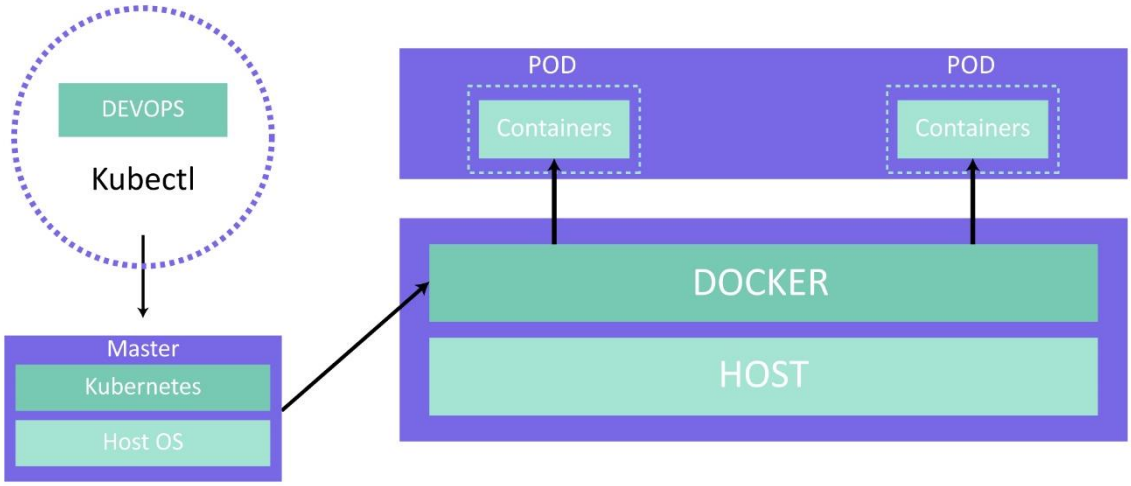

Figure 2. 6 Aperçu de Kubernetes dans une infrastructure conteneurisée. (Original 2021).

Rappelons que les plugins sont des outils qui permettent d'ajouter des fonctions supplémentaires à un logiciel principal. Un certain nombre de ces plugins sont fournis par ansible (les plugins d'action, de cache, de rappel...etc.), mais nous pouvons également créer nos propres plugins.

2.4. La plateforme d'automatisation :

Les technologies natives du cloud utilisent des architectures automatisées pour gérer la complexité et augmenter la vitesse de traitement des tâches.

L'automatisation permet la livraison continue, grâce à la méthodologie de développement de produits Agile qui consiste à déplacer constamment les incréments de modifications logicielles vers la production (les phases de développement et de test sont concurrentes).

La livraison continue rend la publication de logiciels fiable, de sorte que les organisations peuvent fournir des logiciels plus fréquemment avec moins de risques, ce qui se traduit par une gestion plus rapide des pannes.

Ansible :

Ansible [58] est un moteur d'automatisation informatique populaire qui peut automatiser des tâches lourdes, répétitives ou complexes telles que la gestion de la configuration, la configuration du Cloud, le déploiement de logiciels et l'orchestration en service.

Il est utilisé pour les déploiements à plusieurs niveaux et modélise l'ensemble de l'infrastructure informatique en tant que déploiement unique au lieu de gérer chaque déploiement individuellement. Il n'y a pas de proxy dans l'architecture Ansible et il n'est pas nécessaire d'utiliser une architecture de sécurité personnalisée. Le langage utilisé dans ansible est un langage simple appelé YAML, qui signifie "YAML Ain't Markup Language".

En raison du faible facteur d'application géré par OSM dans ce travail, l'automatisation avec ansible n'est pas requise.

Kafka :

Kafka est un logiciel open source qui fournit une structure (framework) pour le stockage, la lecture et l'analyse des données de streaming. Open source implique qu'il est gratuit et qu'il dispose d'un réseau d'utilisateurs et de développeurs qui fournissent des mises à jour, de nouvelles fonctionnalités ainsi qu'une assistance pour les nouveaux utilisateurs.

LXD :

LXD [59] est un gestionnaire de conteneurs système. Il offre une expérience utilisateur similaire aux machines virtuelles utilisant des conteneurs Linux. Il est basé sur des images. Des images prédéfinies sont aussi disponibles pour un grand nombre de distributions Linux. LXD est construit autour d'une API REST puissante, mais simple. C'est donc un outil léger mais parfaitement adapté à la gestion de petits déploiements.

Conclusion :

Dans ce chapitre, nous avons présenté les différents éléments qui composent notre solution et nous nous sommes familiarisés avec ces notions pour passer à la partie pratique dans le prochain chapitre.

CHAPITRE 3

REALISATION D'UNE ARCHITECTURE RESEAU VIRTUELLE CLOUD NATIVE

Introduction

Après l'étude théorique des différents éléments qui constituent notre infrastructure réseau, nous allons, dans ce chapitre, mettre en pratique les notions et concepts présentés précédemment.

Ce chapitre sera en effet dédié à la mise en place d'une solution NFV/SDN dans notre architecture en présentant les différentes étapes d'installation et de configuration des composants réseau et d'OSM.

3.1. Vue globale de l'architecture NFV/SDN Cloud Native:

Pour réaliser notre solution, il faut d'abord commencer par la conception du schéma réseau global ; la figure 3.1 suivante illustre l'interaction d'OSM avec Openstack (i.e. les VIM-Virtualized Infrastructure Manager-) et les VNF :

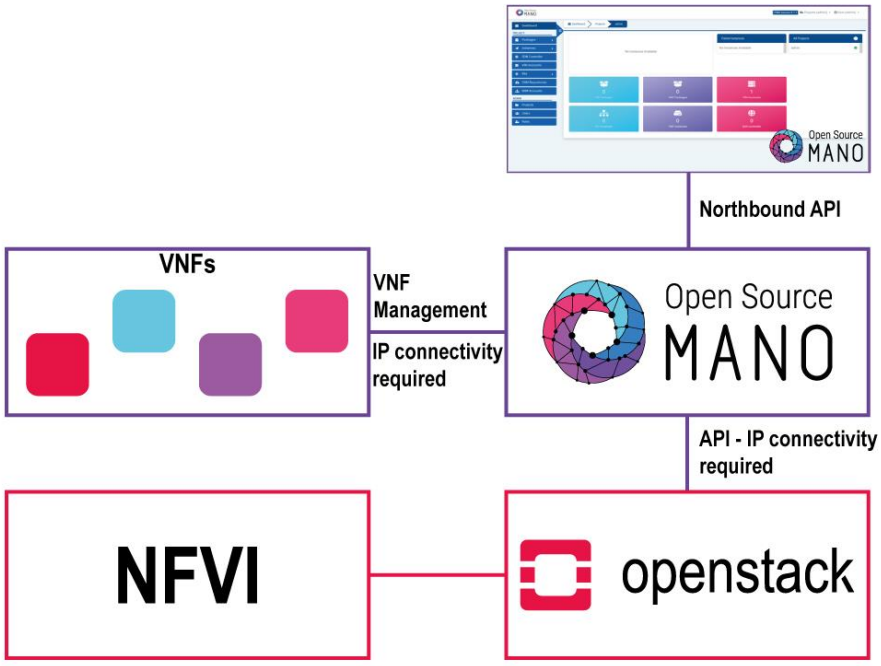

Figure 3. 1 Interaction d'OSM avec les différents composants réseau NFV/SDN (Original 2021).

- **Environnement d'installation**

La plupart des projets dans le domaine qui nous intéresse sont open source et financés par la communauté, ce qui les rend disponibles à tous pour les tests sur une plateforme libre open source.

OSM est disponible sur Linux, nous avons choisi Ubuntu 20 LTS qui est la dernière version stable pour le moment.

Il existe différentes méthodes d'installation, nous citerons :

- Installation via Vagrant.
- Installation via Charmed.
- Installation via script Bash.

Après avoir testé les trois méthodes, nous avons choisi la troisième option qui est la plus simple et la plus rapide à suivre en comparaison aux deux autres qui nécessitent des étapes supplémentaires ne concernant pas notre cas d'utilisation.

Afin d'installer OSM sur notre machine virtuelle, il faut installer quelques dépendances d'Ubuntu. La méthode d'installation du script Bash est un processus automatisé qui échoue lorsqu'il rencontre des erreurs et nécessite une réinstallation complète de l'environnement. Après plusieurs essais, les erreurs causées par une mauvaise utilisation des packages Snap et du référentiel indisponible ont été isolées.

- **Caractéristiques de notre machine virtuelle :**

Nom : OSM

Type et version de système d'exploitation : Ubuntu 20.04 LTS

Mémoire (RAM) : 12 GB

Stockage : 90 GB

Nombre de processeurs : 2

Carte réseau : Bridged.

3.2. Etapes de mise en œuvre de l'architecture NFV/SDN Cloud native et résultats :

3.2.1. Installation de Docker

Docker est un framework de conteneur et l'une des principales dépendances de l'architecture NFV conteneurisée. Malheureusement pour l'installer nous avons du faire un petit détour.

Pour installer Docker, on a besoin d'ajouter le répertoire approprié et pour cela nous avons besoin de la commande « curl ».

Nous utilisons la commande curl pour obtenir les clés publiques du répertoire docker, et la commande echo pour confirmer la validité de ce dernier (Figure 3.2).

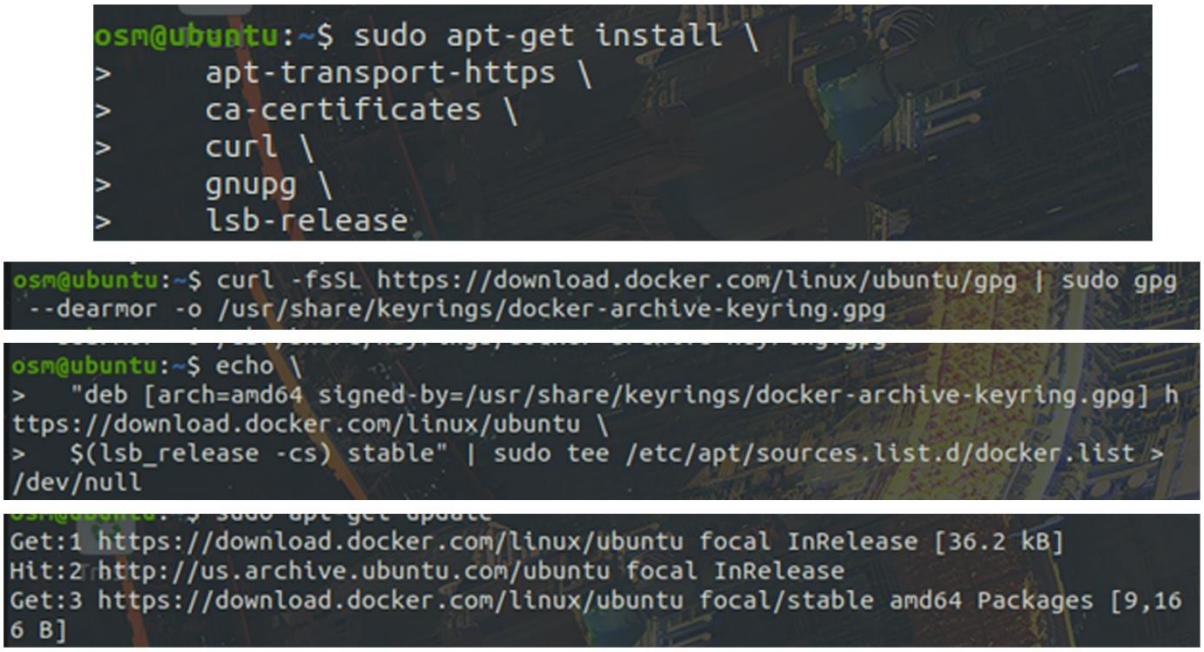

Figure 3. 2 Installation des répertoires docker (Linux terminal 2021).

Une fois l'installation des répertoires docker terminée nous passons à l'installation de docker (Figure 3.3):

Figure 3. 3 Installation de docker et de ses plugins (Linux terminal 2021).

3.2.2. Installation de Kubernetes :

Kubernetes est un système permettant d'automatiser le déploiement, la mise à l'échelle et la gestion d'applications conteneurisées. Son installation (Figure 3.4) nécessite de suivre certaines étapes.

```
osm@osm: $ sudo curl -fsSLo /usr/share/keyrings/kubernetes-archive-keyring.gpg https://packages.cloud.google.com/apt/doc/apt-key.gpg

osm@osm: $ echo "deb [signed-by=/usr/share/keyrings/kubernetes-archive-keyring.gpg] https://apt.kubernetes.io/ kubernetes-xenial main" | sudo tee /etc/apt/sources.list.d/kubernetes.list

osm@osm: $ sudo apt-get update
osm@osm: $ sudo apt-get install -y kubelet kubeadm kubectl
osm@osm: $ sudo apt-mark hold kubelet kubeadm kubectl
```

Figure 3. 4 Installation de Kubernetes plugins (Original 2021).

3.2.3. **Installation d'OSM :**

On utilise la méthode du script pour installer OSM (Figure 3.5) et toutes ses dépendances afin de pouvoir prendre en charge le déploiement de services NFV sur plusieurs VIM. Et pour cela on utilise la commande :

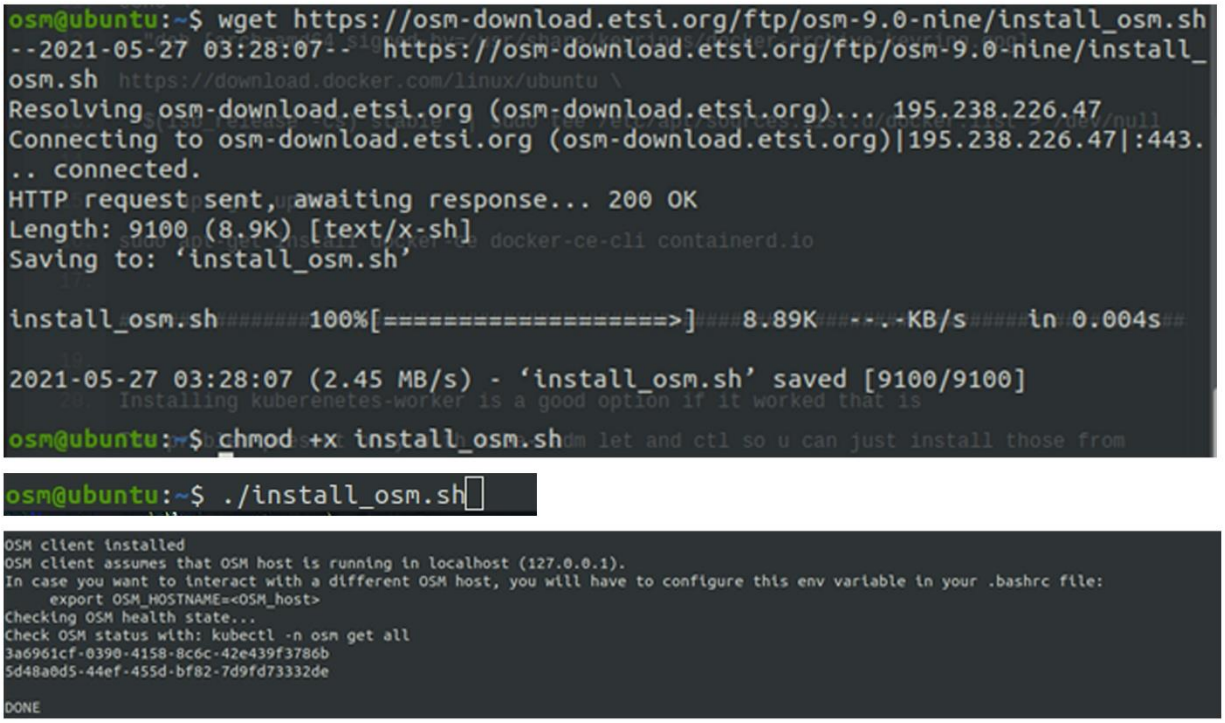

Figure 3. 5 Installation d'OSM (Linux terminal 2021).

Le script d'installation démarre le contrôleur Juju sur Kubernetes pour déployer OSM, configure un cluster LXD sur le système hôte et ajoute celui-ci au contrôleur.

Pour déployer OSM, Juju met en paquets les informations nécessaires sur le contrôleur ; un modèle nommé osm est alors créé par le script. Une fois terminé, il est possible d'accéder à certains services OSM : NBI, NG-UI, Grafana et Prometheus.

Une fois les paquets déployés, un composant logiciel enfichable pour le client OSM est installé.

Le tableau de bord (Dashboard) d'OSM :

Le Dashboard est une interface Web permettant d'agir sur les différents services d'Open Source MANO. C'est un tableau de bord dédié à l'administration du Cloud via une interface web qui permet de gérer facilement la création d'instances, de réseaux virtuels, d'images, etc...

Apres l'installation de docker et d'OSM nous pouvons accéder à l'interface graphique d'OSM (ou Dashboard.). On utilise la commande suivante (Figure 3.6) afin de connaitre l'adresse IP host :

```
osm@ubuntu:~$ hostname -i
127.0.1.1
```

Figure 3. 6 Récupération de l'adresse IP host (Linux terminal 2021).

Après avoir obtenu une nouvelle installation OSM avec ses dernières images de docker préconstruites (les images sont mises à jour quotidiennement), nous pouvons accéder à l'interface utilisateur (Figures 3.7 et 3.8) en utilisant l'adresse IP de l'hôte comme URL avec les identifiants par défaut (utilisateur : admin, mot de passe : admin) :

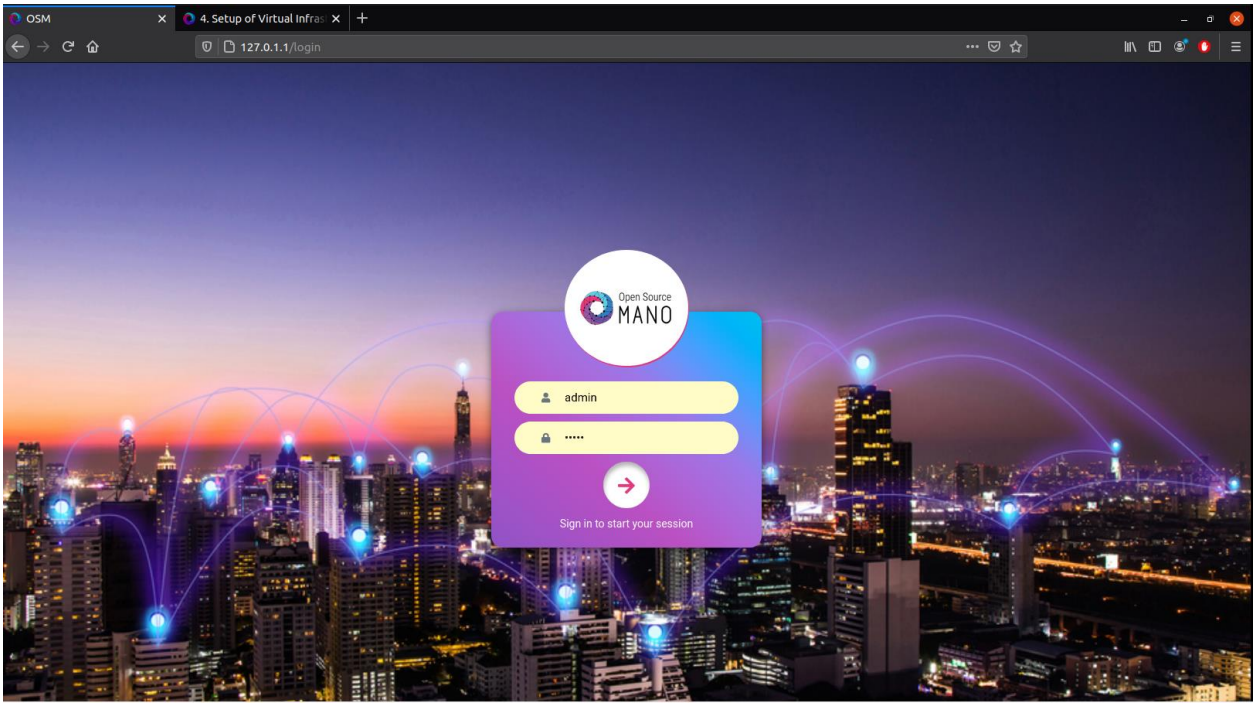

Figure 3. 7 Ecran d'accueil d'OSM (Firefox 2021)

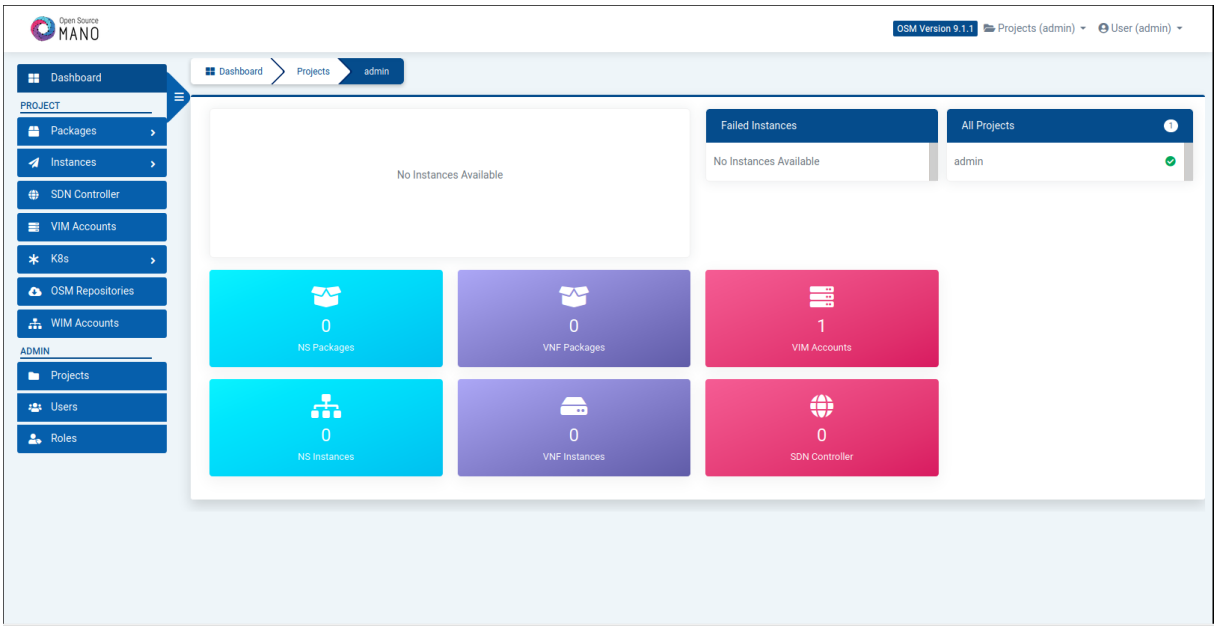

Figure 3. 8 Tableau de bord (dashboard) d'OSM (Firefox 2021)

3.2.3. Intégration des VNF (Virtual Network Function) et des NS (Network Services) :

L'intégration d'un VNF ou un NS dans OSM implique la préparation et l'ajout du package VNF correspondant au système. Ce processus suppose également que les images VM correspondantes sont disponibles dans le(s) VIM où elles seront instanciées.

OSM fournit quelques packages VNF et NS pour les tests d'environnement et ils sont hébergés dans un répertoire git (Figure 3.9).

```
osm@osm:~$ git clone --recursive https://osm.etsi.org/gitlab/vnf-onboarding/osm-packages
Cloning into 'osm-packages'...
warning: redirecting to https://osm.etsi.org/gitlab/vnf-onboarding/osm-packages.git/
remote: Enumerating objects: 2821, done.
remote: Counting objects: 100% (2821/2821), done.
remote: Compressing objects: 100% (2117/2117), done.
remote: Total 5846 (delta 601), reused 2779 (delta 564), pack-reused 3025
Receiving objects: 100% (5846/5846), 48.65 MiB | 748.00 KiB/s, done.
osm@osm:~$ ls
Desktop  Documents  Downloads  install_osm.sh  Music  osm-packages  Pictures  Public  snap  Templates  Videos
```

Figure 3. 9 Téléchargement du contenu du répertoire git (Linux terminal 2021)

Nous utilisons la commande ci-dessous pour aller vers le dossier téléchargé :

osm@osm:~$ Cd osm-packages

Une fois toutes ces étapes effectuées avec succès, nous pouvons alors ajouter n'importe quel VNF ou NS à notre OSM (Figure 3.10) :

```
osm@osm:~/osm-packages$ osm nfpkg-create hackfest_basic_vnf
Validating package hackfest_basic_vnf
Base directory: hackfest_basic_vnf
1 Descriptors found to validate
Validation OK
List of charms in the descriptor: []
Adding File: hackfest_basic_vnf
Package created: ./hackfest_basic_vnf.tar.gz
Uploading package ./hackfest_basic_vnf.tar.gz
89b99fa2-54d1-4ff6-897a-99eb60c3dac0
osm@osm:~/osm-packages$ osm nspkg-create hackfest_basic_ns
Validating package hackfest_basic_ns
Base directory: hackfest_basic_ns
1 Descriptors found to validate
Validation OK
List of charms in the descriptor: []
Adding File: hackfest_basic_ns
Package created: ./hackfest_basic_ns.tar.gz
81e81a35-5dc5-4113-a9fc-c24000cd5b02
```

Figure 3. 10 Importation des packages à partir du dossier (Linux terminal 2021)

Il ne reste plus qu'à valider et vérifier que les VNF et NS sont importés avec succès (Figure 3.11):

```
osm@osm:~/osm-packages$ osm nspkg-list
+-------------------------+--------------------------------------+
| nsd name                | id                                   |
+-------------------------+--------------------------------------+
| hackfest-simple-k8s-ns  | 6b921a0f-574e-408a-b438-7ad5429c3bbd |
| hackfest_basic-ns       | 81e81a35-5dc5-4113-a9fc-c24000cd5b02 |
| squid_cnf_ns            | dac4d414-323b-4ce7-a87f-b88572565f88 |
+-------------------------+--------------------------------------+
osm@osm:~/osm-packages$ osm nfpkg-list
+-------------------------+--------------------------------------+-----------+
| nfpkg name              | id                                   | desc type |
+-------------------------+--------------------------------------+-----------+
| hackfest-simple-k8s-vnf | b3cdb67f-4acc-445b-ab8c-7b2425ea2902 | sol006    |
| hackfest_basic-vnf      | 89b99fa2-54d1-4ff6-897a-99eb60c3dac0 | sol006    |
| squid_cnf               | 6a1e670c-4d1b-4b2e-b9c5-ed49a1013074 | sol006    |
+-------------------------+--------------------------------------+-----------+
```

Figure 3. 11 Vérification des VNF et NS (Linux terminal 2021)

Il est possible de vérifier leur présence ainsi que leur topologie sur le tableau de bord (dashboard) :

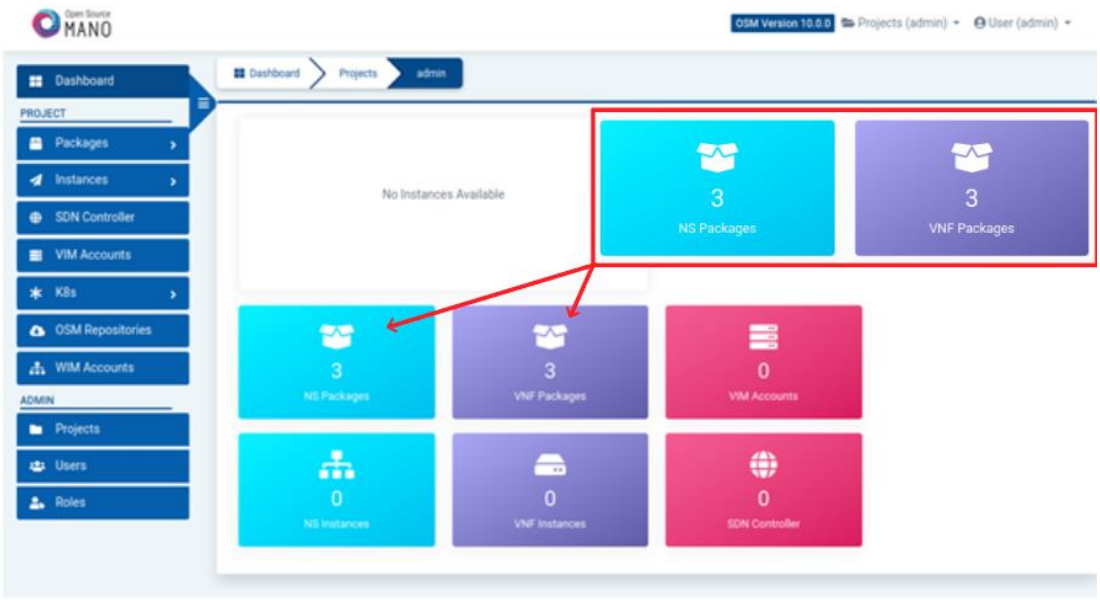

Figure 3. 12 Tableau de bord d'OSM après ajout des VNF et NS (Firefox 2021)

Nous pouvons ainsi détailler la topologie des VNF et NS dans le tableau de bord (Figures 3.12 et 3.13):

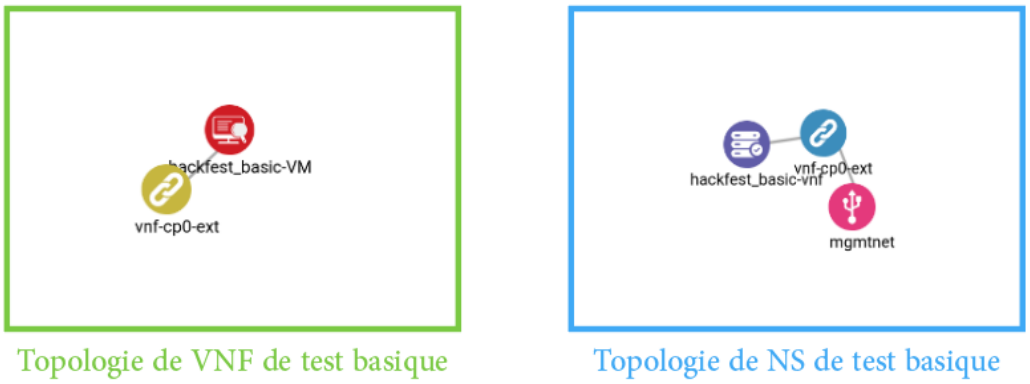

Figure 3. 13 Aperçu des topologies des VNF et NS dans le tableau de bord (Firefox 2021)

3.2.4. Installation du VIM (Virtualized Infrastructure Manager) :

MicroStack permet d'installer rapidement OpenStack, de le déployer à un ou plusieurs nœuds et de l'exécuter directement sur le poste de travail.

MicroStack est préconfiguré avec la mise en réseau, une image, des variantes, des groupes de sécurité ouverts (port TCP 22 et ICMP) et une paire de clés SSH.

Nous avons utilisé une 2ème machine virtuelle avec les caractéristiques suivantes pour installer notre VIM :

Nom : OSM

Type et version de système d'exploitation : Ubuntu 20.04 LTS

Mémoire (RAM) : 12 GB

Stockage : 90 GB

Nombre de processeurs : 2

Carte réseau : Bridged.

Pour l'installation de notre VIM, nous avons utilisé la commande suivante (Figure 3.14) :

```
osm@osm:~$ sudo snap install microstack --beta --devmode
[sudo] password for osm:
microstack (beta) ussuri from Canonical✓ installed
```

Figure 3. 14 Installation de MicroStack (Linux terminal 2021)

Une fois l'installation terminée, MicroStack doit être initialisé pour pouvoir configurer les réseaux et les bases de données. Pour cela, nous utilisons la commande suivante (Figure 3.15):

```
osm@osm:~$ sudo microstack init --auto --control
2021-07-02 00:48:18,018 - microstack_init - INFO - Configuring clustering ...
2021-07-02 00:48:18,111 - microstack_init - INFO - Setting up as a control node.
2021-07-02 00:48:21,049 - microstack_init - INFO - Configuring networking ...
2021-07-02 00:48:26,394 - microstack_init - INFO - Opening horizon dashboard up to *
2021-07-02 00:48:27,305 - microstack_init - INFO - Waiting for RabbitMQ to start ...
Waiting for 192.168.241.132:5672
2021-07-02 00:48:32,655 - microstack_init - INFO - RabbitMQ started!
2021-07-02 00:48:32,656 - microstack_init - INFO - Configuring RabbitMQ ...
2021-07-02 00:48:34,029 - microstack_init - INFO - RabbitMQ Configured!
2021-07-02 00:48:34,047 - microstack_init - INFO - Waiting for MySQL server to start ...
Waiting for 192.168.241.132:3306
2021-07-02 00:48:42,397 - microstack_init - INFO - Mysql server started! Creating databases ...
2021-07-02 00:48:44,095 - microstack_init - INFO - Configuring Keystone Fernet Keys ...
2021-07-02 00:48:57,383 - microstack_init - INFO - Bootstrapping Keystone ...
2021-07-02 00:49:09,961 - microstack_init - INFO - Creating service project ...
2021-07-02 00:49:15,632 - microstack_init - INFO - Keystone configured!
2021-07-02 00:49:15,651 - microstack_init - INFO - Configuring the Placement service...
2021-07-02 00:49:35,952 - microstack_init - INFO - Running Placement DB migrations...
2021-07-02 00:49:39,278 - microstack_init - INFO - Configuring nova control plane services ...
2021-07-02 00:49:52,646 - microstack_init - INFO - Running Nova API DB migrations (this may tak
2021-07-02 00:53:54,674 - microstack_init - INFO - Creating security group rules ...
2021-07-02 00:54:10,460 - microstack_init - INFO - Configuring the Cinder services...
2021-07-02 00:55:12,082 - microstack_init - INFO - Running Cinder DB migrations...
2021-07-02 00:55:23,099 - microstack_init - INFO - restarting libvirt and virtlogd ...
2021-07-02 00:55:27,531 - microstack_init - INFO - Complete. Marked microstack as initialized!
```

Figure 3. 15 Initialisation de MicroStack (Linux terminal 2021)

Pour lancer notre première instance OpenStack (VM) appelée « test » basée sur l'image CirrOS, on exécute la commande suivante (Figure 3.16) :

```
osm@osm:~$ microstack launch cirros --name test
Creating local "microstack" ssh key at /home/osm/snap/microstack/common/.ssh/id_microstack
Launching server ...
Allocating floating ip ...
Server test launched! (status is BUILD)

Access it with `ssh -i /home/osm/snap/microstack/common/.ssh/id_microstack cirros@10.20.20.36`
You can also visit the OpenStack dashboard at http://10.20.20.1:80
```

Figure 3. 16 Lancement de l'instance OpenStack (Linux terminal 2021)

Le message a la fin montre que tout s'est bien déroulé et donne les informations dont nous avons besoin pour SSH à l'instance. Notre machine virtuelle est maintenant opérationnelle.

Dashboard d'OpenStack :

D'après le site officiel d'OpenStack : « Le dashboard est le tableau de bord OpenStack qui fournit aux utilisateurs un portail en libre-service pour provisionner leurs propres ressources dans les limites fixées par les administrateurs. Ceux-ci incluent le provisionnement des utilisateurs, la définition des types d'instance, le téléchargement d'images de machine virtuelle (VM), la gestion des réseaux, la configuration de groupes de sécurité, le démarrage d'instances et l'accès aux instances via une console ».

Après avoir effectué les étapes précédentes, il nous est possible d'accéder au dashboard d'OSM (Figure 3.17) via un navigateur grâce à l'adresse IP 10.20.20.1.

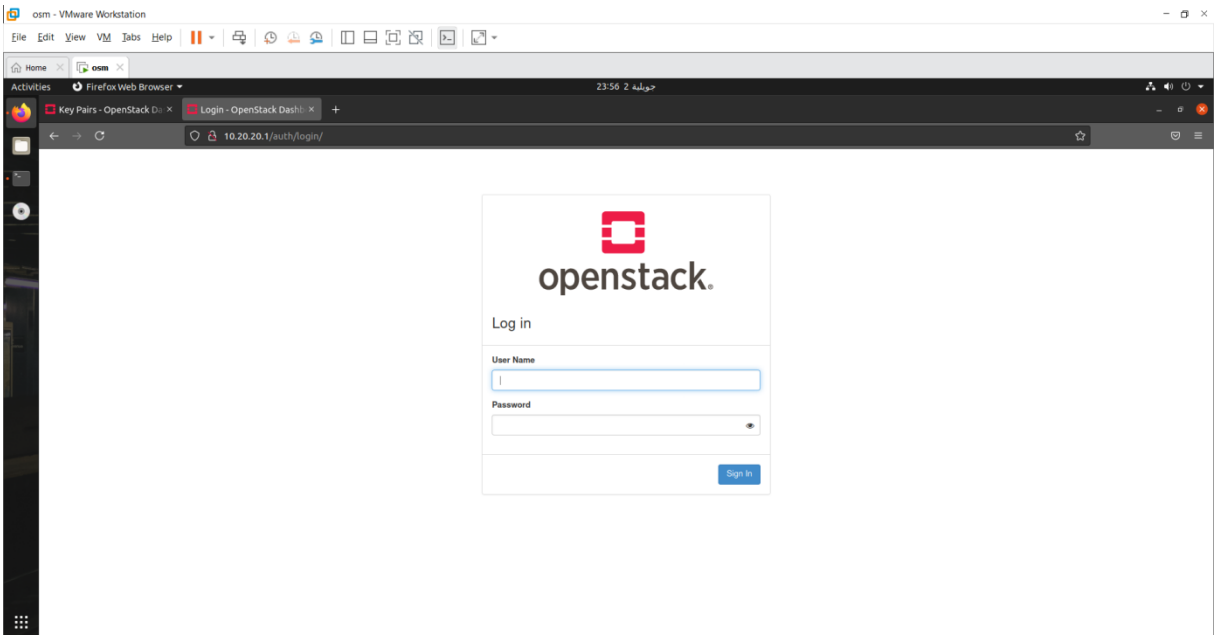

Figure 3. 17 Ecran de connexion d'OpenStack (Firefox 2021)

Avant de se connecter à notre compte OpenStack et accéder au dashboard, nous devons effectuer une commande pour récupérer le mot de passe (Figure 3.18) :

```
osm@osm:~$ sudo snap get microstack config.credentials.keystone-password
[sudo] password for osm:
6ug4TlW909LewHRM0ptwhkRTJVrOEwaa
```

Figure 3. 18 Récupération du mot de passe du compte OpenStack (Linux terminal 2021)

Maintenant, nous pouvons accéder au dashboard d'OpenStack et profiter de ses diverses fonctionnalités (Figure 3.19).

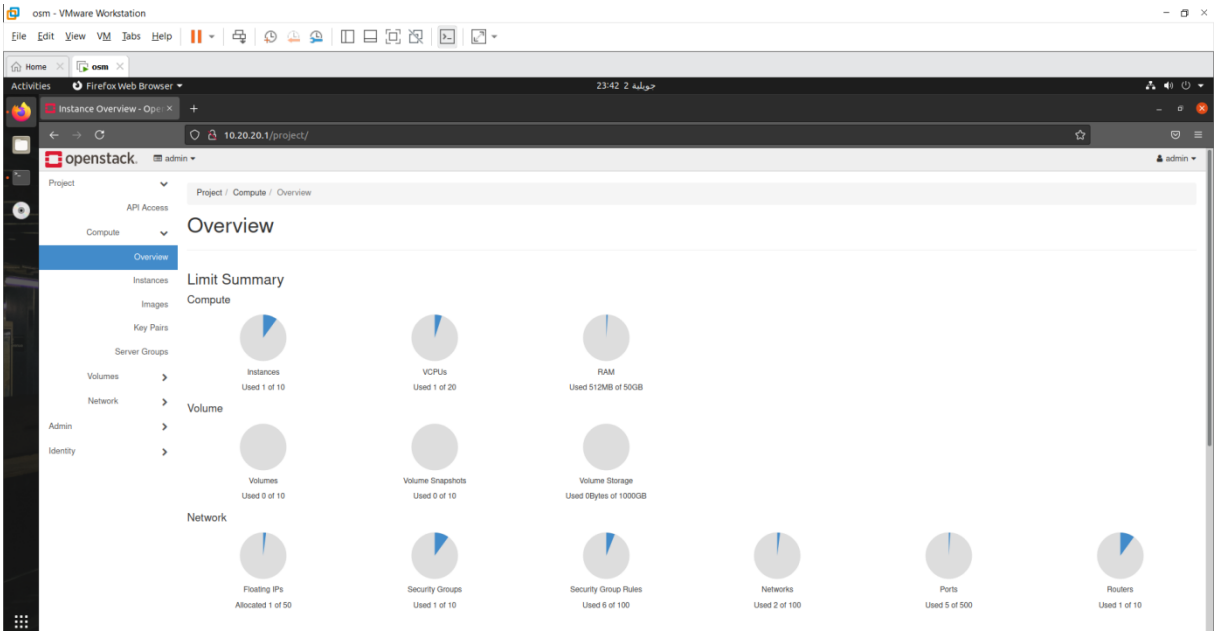

Figure 3. 19 Tableau de bord d'OpenStack (Firefox 2021)

Le service « Compute » d'OpenStack s'exécute dans de nombreux emplacements dans le cloud et interagit avec certains services internes. Il offre des options de configuration pour le déploiement.

Comme nous pouvons le voir, la fonction « Overview » nous donne une vue globale des performances. Mais ce n'est pas la seule, nous disposons de multiples fonctionnalités dans notre interface graphique. On compte parmi elles, la possibilité de créer ou de supprimer une image comme ci-dessous (Figure 3.20) :

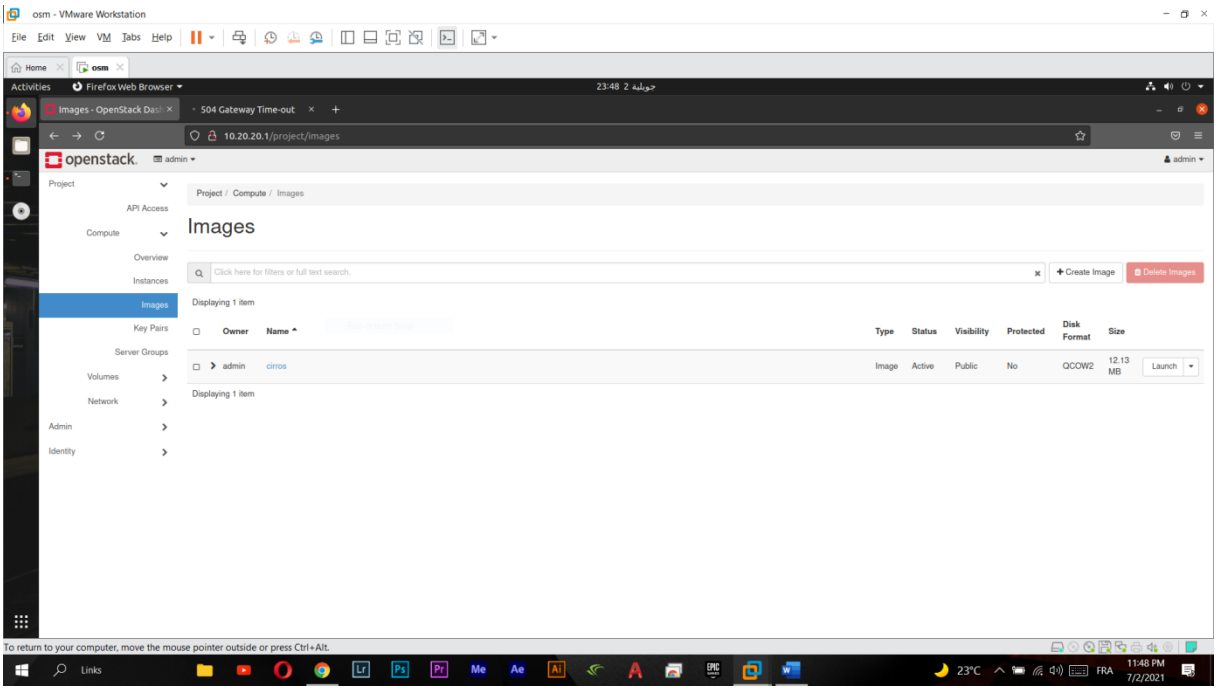

Figure 3. 20 Service "images" dans le tableau de bord d'OpenStack (Firefox 2021)

Ce service permet aux utilisateurs de télécharger et découvrir des actifs de données destinés à être utilisés avec d'autres services. Nous remarquons que notre image CirrOS précédemment évoquée se trouve ici.

Nous avons aussi une autre fonctionnalité qui est le « Key Pairs » (Figure 3.21) qui nous permet de créer, supprimer ou importer une clé publique de type SSH car MicroStack est préconfiguré avec une paire de clés SSH.

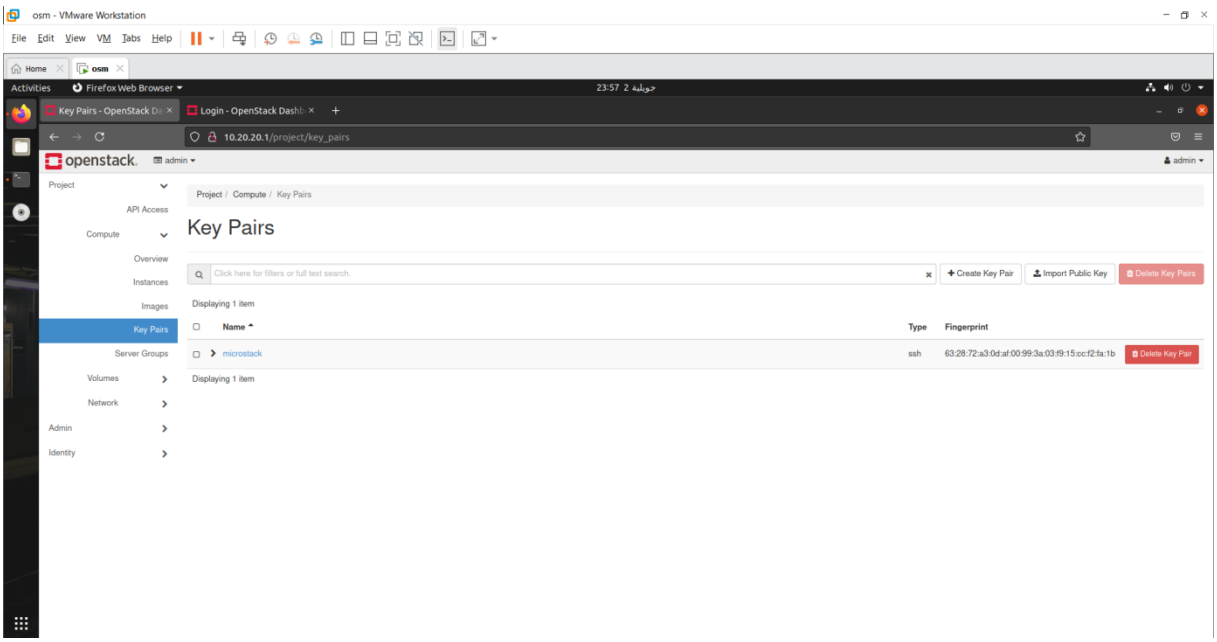

Figure 3. 21 Service "Key Pairs" dans le tableau de bord d'OpenStack (Firefox 2021)

Apres avoir passé en revue quelques-uns des services du dashboard d'OpenStack, on peut maintenant configurer ces services à notre guise, de la manière qui nous arrange, en fonction de nos besoins.

3.2.5. Configuration des gestionnaires d'infrastructure virtuelle (VIM) :

- **Préparation de MicroStack pour être utilisé par OSM :**

Une fois l'installation de notre VIM terminée, nous pouvons désormais nous connecter au dashboard d'OpenStack sur la machine virtuelle où OSM est installée via le réseau local. Mais cela n'est pas suffisant pour qu'ils soient connectés : faire fonctionner MicroStack avec OSM nécessite une série de configurations supplémentaires.

- **Création d'un réseau de gestion, avec DHCP activé, accessible depuis OSM :**

Le déploiement de services réseaux nécessite un réseau de gestion avec DHCP activé pour garantir que ce réseau de gestion soit accessible depuis OSM. Le réseau est utilisé par le VCA (Juju) pour configurer les VNF, une fois qu'ils sont en cours d'exécution.

Il est recommandé de créer un réseau fournisseur isolé de MicroStack. On crée par exemple un réseau de fournisseur en utilisant l'interface physique em1 et VLAN et avec CIDR 10.208.0.0/24 (Figure 3.22).

```
osm@osm:$ microstack.openstack network create  mgmt --share --external \
            --provider-physical-network physnet_em1 \
            --provider-network-type flat provider


osm@osm:$ microstack.openstack subnet create subnet-mgmt --network provider \
            --allocation-pool start=210.208.0.2,end=10.208.0.254 \
            --dns-nameserver 1.1.1.1 \
            --subnet-range 24/10.208.0.0 provider
```

Figure 3. 22 les différents réseaux configurés dans MicroStack (Firefox 2021)

- **Téléchargement d'images nécessaires**

Il est nécessaire de télécharger les images des VNF sur notre VIM, afin qu'elles soient disponibles avant une instanciation (Figure 3.23).

```
osm@osm:~$ microstack.openstack image create \
>                          --public \
>                          --disk-format qcow2 \
>                          --container-format bare \
>                          --file bionic-server-cloudimg-amd64.img \
>                          ubuntu1804
```

Field	Value
checksum	77e181086c6ef6d77164c83e9f1e69c1
container_format	bare
created_at	2021-07-07T19:48:29Z
disk_format	qcow2
file	/v2/images/ea576694-aefc-446a-940b-f979b37d24da/file
id	ea576694-aefc-446a-940b-f979b37d24da
min_disk	0
min_ram	0
name	ubuntu1804
owner	f1353922a35b4719856eb3f6f0dac51e
properties	os_hash_algo='sha512', os_hash_value='f754d42f32f7dd2df28918137cb3c4b3d2b247cefc8ec3393186d744b57295fce26775acc0e4ec815c08b46b4a4fa8ac5609dbda1c8d35e0139890e04b9f5130', os_hidden='False', owner_specified.openstack.md5='', owner_specified.openstack.object='images/ubuntu1804', owner_specified.openstack.sha256='a8406e85777bc19e3cceec978f909ce31ff7c7bcf53277867484e4964f8106106', self='/v2/images/ea576694-aefc-446a-940b-f979b37d24da'
protected	False
schema	/v2/schemas/image
size	370802688
status	active
tags	
updated_at	2021-07-07T19:50:08Z
visibility	public

Figure 3. 23 Création d'image Ubuntu. (Linux terminal 2021)

- **Ajout de MicroStack en tant que cible VIM à OSM :**

Voici un exemple d'utilisation du client OSM pour ajouter un VIM MicroStack (Figure 3.24). Nous pouvons le constater dans le dashboard d'OSM :

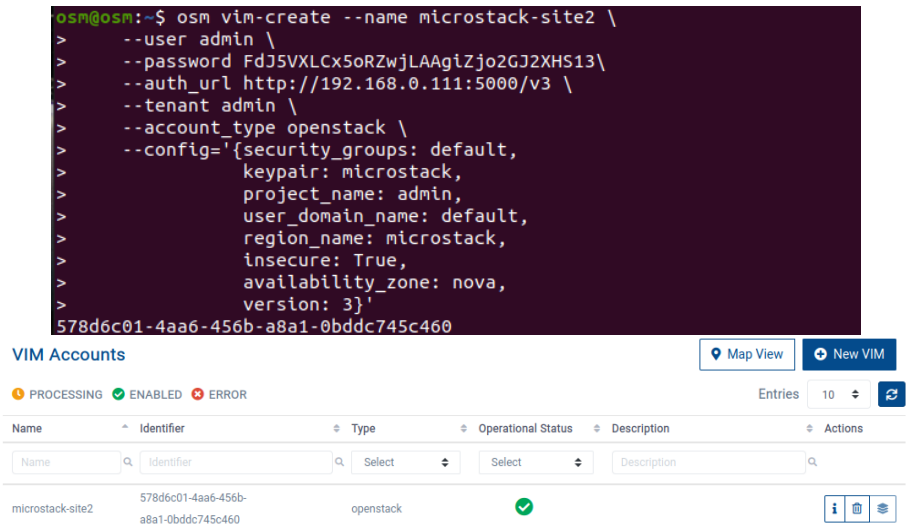

Figure 3. 24 Ajout de MicroStack en tant que cible VIM. (Firefox et Linux terminal 2021)

3.2.6. Déploiement des services réseau :

OSM a la possibilité de modifier les paramètres de NS ou NSI lors de l'instanciation (Day-0 et Day-1), afin que nous puissions facilement décider des paramètres clés du service sans avoir besoin de modifier l'ensemble d'origine des packages validés.

Le cycle de vie d'un VNF passe par différentes étapes de configuration. Qui sont : la configuration de la gestion pendant l'instanciation (Day-0), l'initialisation du service juste après l'instanciation (Day-1) et la reconfiguration pendant l'exécution (Day-2).

Pour notre cas d'utilisation, nous n'avons pas eu besoin de le reconfigurer, nous avons donc fait uniquement (Day-0) et (Day-1).

Lors de la création d'une instance NS, il est possible de passer des paramètres d'instanciation à OSM à l'aide de l'option (--config) du client ou du paramètre (config) du Dashboard qui est l'une des fonctionnalités les plus importantes d'OSM.

Par exemple, nous pouvons modifier les paramètres, comme le réseau auquel se connecter.

```
osm@osm:$~ osm ns-create --ns_name hf-basic --nsd_name hackfest_basic-ns --vim_account openstack1
           --config '{vld: [ {name: mgmtnet, vim-network-name: mgmt} ] }'
```

Par exemple, dans un réseau de grande entreprise, les administrateurs peuvent choisir de créer différents réseaux séparés par logiciel pour héberger différentes fonctions, comme un

réseau de gestion pour les produits et services finalisés, mais aussi un réseau de test pour tester localement de nouvelles fonctionnalités.

```
osm@osm:$ osm ns-create --ns_name <ns-instance-name> --nsd_name hackfest_basic-ns
          --vim_account microstack-site2
```

Nous pouvons visualiser le statut de NS via une commande ou via le dashboard (Figure 3.25) :

```
osm@osm:~/osm-packages/charm-packages$ osm ns-list
+-----------------+--------------------------------------+-------------------+---------+-----------------+--------------+
| ns instance name | id                                   | date              | ns state | current operation | error details |
+-----------------+--------------------------------------+-------------------+---------+-----------------+--------------+
| 5555555         | 9bceb688-7850-4dbf-94da-2e539d7af9d1 | 2021-07-07T22:49:31 | READY    | IDLE (None)       | N/A          |
+-----------------+--------------------------------------+-------------------+---------+-----------------+--------------+
```

Figure 3. 25 Liste de NS déployé. (Linux terminal 2021)

Dans la figure ci-dessous, nous constatons les avantages de l'isolation des services qui offre un confort de la gestion des pannes en toute simplicité.

Figure 3. 26 Graphe des services déployés (Firefox 2021)

Malheureusement, en raison du manque de ressources informatique utilisé par le programme de conteneurisation pour déployer les fonction réseau, nous n'avons pas pu essayer des tests plus complexes comme certaines des offres connues des générations précédentes de réseaux cellulaires (IMS, RAN, etc.).

Conclusion :

Dans ce dernier chapitre, nous avons détaillé les étapes à suivre pour installer les différents éléments de notre solution ainsi que la configuration du VIM afin de le connecter à OSM. Nous avons maintenant un aperçu du mode de fonctionnement de ces derniers et du processus de déploiement des VNF et NS.

CONCLUSION GENERALE

Dans le cadre de notre projet, nous avons mis en place une architecture NFV/SDN cloud native basé sur OpenStack et OSM qui pourra être utilisée dans l'architecture du réseau 5ème génération de Mobilis.

L'utilisation du Cloud Natif ne cesse de croître dans les réseaux de télécommunication et cela dû à ses diverses fonctionnalités et son ensemble d'outils personnalisable. Aujourd'hui, aucun opérateur ne remet en cause la logique de la NFV. Au contraire, ils cherchent des moyens de faire progresser ces gains d'efficacité dans leurs réseaux, elle est même utilisée par des entreprises comme Huawei pour fournir une solution TelcoCloud robuste.

Tous ces avantages et la libération des contraintes d'utilisation d'équipements propriétaires séduisent les opérateurs de réseaux de télécommunication à l'instar de Mobilis.

Cette expérience n'est qu'une brève introduction à ce domaine, les implémentations sont infinies car OSM est désormais déployé dans les réseaux de production, ce qui prouve la valeur des services réseaux multi-fournisseurs orchestrés par des solutions ouvertes basées sur des standards interopérables.

Notre propre expérience de formation dans le domaine en question a été extrêmement enrichissante avec l'assimilation de technologies liées au Cloud Computing en premier lieu et la maîtrise de nouveaux concepts (virtualisation de fonctions, infrastructure de virtualisation des fonctions réseaux (hyperviseur, gestion de conteneurs…). Cette maîtrise est nécessaire lorsqu'on met en œuvre une architecture NFV/SDN.

Nous sommes arrivés à mettre en place l'architecture recherchée. Mais dû au manque de ressources matériels et l'apparition de problèmes causé par le verrouillage de la machine hôte qui nous a compliquée la tâche car on devait recommencer l'installation d'OSM et d'OpenStack dès le début (cela montre à quel point cette plate-forme a besoin de plus de contribution pour l'amélioration de ses fonctionnalités, car elle a encore besoin de se développer davantage), nous n'avons pas pu exécuter des tests plus complexes. Néanmoins, nous avons pu déployer des NS et des VNF ce qui prouve que notre solution OSM avec notre VIM MicroStack peuvent être utilisés dans les réseaux.

BIBLIOGRAPHIE ET WEBOGRAPHIE

[1] N. K. Garrison J., Cloud Native Infrastructure, O'Reilly Media Inc., 2017.

[2] SDxCentral, «Learn: what is the cloud,» [En ligne]. Available: www.SDxCentral.com/cloud. [Consulter en Décembre 2020].

[3] Orange Businesss Services, «transformation digitales des infrastructures réseau,» 19 Mars 1970. [En ligne]. Available: https://www.orange-business.com/fr/presse/transformation-digitale-des-infrastructures-reseaux-orange-business-services-lance-une.. [Consulter en Décembre 2020].

[4] SDxcentral, «SDxcentral,» [En ligne]. Available: www.SDxCentral.com/5g. [Consulter en Décembre 2020].

[5] Elgorma M, R. Beney, «Généralités sur les réseaux de 3eme Génération (UMTS).,» 7 Octobre 2019. [En ligne]. Available: https://summarynetworks.com/info-tlc/generalites-sur-les-reseaux-3eme-generation-umts/. . [Consulter en Décembre 2020].

[6] Ketfi C., «Réseau 5G : déploiement, fonctionnement, usages et smartphones compatibles,» 28 Mars 2021. [En ligne]. Available: https://www.frandroid.com/telecom/488716_reseau-5g-tout-ce-qui-va-changer-quels-usages-et-pourquoi-la-technologie-est-importante.. [Consulter en Avril 2021].

[7] ITU, «itunews,» Février 2017. [En ligne]. Available: https://www.itu.int/en/itunews/Documents/2017/2017-02/2017_ITUNews02-fr.pdf. [Consulter en Décembre 2020].

[8] Capgemini France, Sarrazin T., «Cloud et 5G : les deux agents indissociables de la révolution de la donnée. Capgemini France,» 24 Juin 2020. [En ligne]. Available: https://www.capgemini.com/fr-fr/2020/06/cloud-et-5g-les-deux-agents-indissociables-de-la-revolution-de-la-donnee/.. [Consulter en Janvier 2021].

[9] Nangare S., «What is Software Defined Networking (SDN)?,» 29 Janvier 2018. [En ligne]. Available: https://sagarnangare.com/what-is-software-defined-networking-sdn/.. [Consulter en Février 2021].

[10] Wikipédia, «Mise en réseau définie par logiciel - Software-defined networking.,» [En ligne]. Available: https://fr.wikinew.wiki/wiki/Software-defined_networking..

[Consulter en Novembre 2020].

[11] Researchgate, «Software defined networking- SDN architecture,» [En ligne]. Available: https://www.researchgate.net/figure/Software-defined-networking-SDN-architecture_fig1_333873385. [Consulter en Février 2021].

[12] LeBigData L., B., «Cloud Computing – Définition, Avantages et Exemples d'utilisation. LeBigData,» 5 Janvier 2021. [En ligne]. Available: https://www.lebigdata.fr/definition-cloud-computing. [Consulter en Janvier 2021].

[13] Saaoui T., F. Touahria, «Etude et simulation d'une solution Cloud Computing au sein d'une entreprise privée. Cas d'étude : Cevital.,» 2020. [En ligne]. Available: http://www.univ-bejaia.dz/xmlui/bitstream/handle/123456789/14454/memoir%20corriger.pdf?sequence=1&is. [Consulter en Février 2021].

[14] Mimoune M., «Etude sur la sécurité du cloud computing. Memoire Online.,» 2014. [En ligne]. Available: https://www.memoireonline.com/07/15/9194/Etude-sur-la-securite-du-cloud-computing.html.. [Consulter en Novembre 2020].

[15] Javatpoint, «Cloud Computing Architecture,» [En ligne]. Available: https://www.javatpoint.com/cloud-computing-architecture.. [Consulter en Décembre 2020].

[16] Wikimedia Foundation, «Plate-forme collaborative,» [En ligne]. Available: https://fr.wikipedia.org/wiki/Plate-forme_collaborative#:~:text=Une%20plateforme%20de%20travail%20collaboratif,met%20%C3%A0%20disposition%20des%20acteurs. [Consulter en Décembre 2020].

[17] IBM, «Containerization,» [En ligne]. Available: https://www.ibm.com/cloud/learn/containerization. [Consulter en Janvier 2021].

[18] Redhat, «La conteneurisation, qu'est ce que c'est ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/cloud-native-apps/what-is-containerization. [Consulter en Février 2021].

[19] Hebergeurcloud, «Conteneurs : avantages et inconvénients de la conteneurisation.,» 27 Janvier 2021. [En ligne]. Available: https://www.hebergeurcloud.com/conteneurs-avantages-inconvenients/. [Consulter en Février 2021].

[20] Bigelow S.J., «Quelle différence entre conteneurisation et virtualisation,» 12 Octobre 2015. [En ligne]. Available: https://www.lemagit.fr/conseil/Quelle-est-la-difference-entre-la-conteneurisation-et-la-virtualisation#:~:text=Ce%20proc%C3%A9d%C3%A9%20est%20souven. [Consulter

en Février 2021].

[21] Khan F., «A Beginner's Guide to Docker Container in NFV,» 13 Mars 2021. [En ligne]. Available: https://telcocloudbridge.com/blog/beginners-guide-docker-container-nfv/.. [Consulter en Avril 2021].

[22] Redhat, «Docker : définition, fonctionnement et avantages,» [En ligne]. Available: https://www.redhat.com/fr/topics/containers/what-is-docker.. [Consulter en Mars 2021].

[23] Linuxcontainers Org., «What's LXD? Linux Containers - LXD - Introduction.,» [En ligne]. Available: https://linuxcontainers.org/lxd/introduction/.. [Consulter en Mars 2021].

[24] Comarch, «Quelles sont les différences entre cloud privé, public et hybride ?,» [En ligne]. Available: https://blog.comarch.fr/differences-cloud-prive-public-hybride/.. [Consulter en Décembre 2020].

[25] Microsoft, «Cloud public comparé à cloud privé ou hybride,» [En ligne]. Available: https://azure.microsoft.com/fr-fr/overview/what-are-private-public-hybrid-clouds/.. [Consulter en Novembre 2020].

[26] Redhat, «Quelle est la différence entre un cloud public, privé et hybride ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/cloud-computing/public-cloud-vs-private-cloud-and-hybrid-cloud.. [Consulter en Novembre 2020].

[27] Citrix France, «Définition du cloud hybride,» [En ligne]. Available: https://www.citrix.com/fr-fr/solutions/app-delivery-and-security/what-is-hybrid-cloud.html.. [Consulter en Novembre 2020].

[28] Appvizer, «IaaS, PaaS, SaaS : différences, définition des services de cloud computing,» 23 Avril 2018. [En ligne]. Available: https://www.appvizer.fr/magazine/services-informatiques/stockage/iaas-paas-saas-quelles-differences.. [Consulter en Janvier 2021].

[29] Nageotte A., «Cloud computing - IAAS, PAAS, SAAS : quelles différences ?,» 29 Janvier 2021. [En ligne]. Available: https://www.oodrive.com/fr/blog/innovation/cloud-computing-iaas-paas-saas-quelles-differences/.. [Consulter en Février 2021].

[30] Redhat, «IaaS, PaaS, SaaS : quelle est la différence ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/cloud-computing/iaas-vs-paas-vs-saas.. [Consulter en Février 2021].

[31] Redhat, «Le CaaS, qu'est-ce que c'est ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/cloud-computing/what-is-

caas#:~:text=Le%20CaaS%20(Containers%2Das%2D,dans%20des%20datacenters%20sur%20site.. [Consulter en Février 2021].

[32] Arcep, «Virtualisation des réseaux d'opérateurs,» 19 Juillet 2019. [En ligne]. Available: https://www.arcep.fr/uploads/tx_gspublication/reseaux-du-futur-virtualisation-des-reseaux-juillet2019.pdf. [Consulter en Février 2021].

[33] IONOS, «Aperçu des concepts de virtualisation.,» [En ligne]. Available: https://www.ionos.fr/digitalguide/serveur/configuration/la-virtualisation/?fbclid=IwAR3N_Ud-M3L6_esJudB108o68pagx6mT7jXZRkeXMou_WPc5sKTjA5JOsj0. [Consulter en Février 2021].

[34] Redhat, «Agile methodology,» [En ligne]. Available: redhat.com/fr/develops/what-is-agile-methodology/. [Consulter en Mars 2021].

[35] Sdxcentral, «SDN controller,» [En ligne]. Available: https://www.sdxcentral.com/networking/sdn/definitions/what-is-sdn-controller/. [Consulter en Mars 2021].

[36] Arora H., «Software Defined Networking (SDN) - Architecture and role of OpenFlow,» [En ligne]. Available: https://www.howtoforge.com/tutorial/software-defined-networking-sdn-architecture-and-role-of-openflow/.. [Consulter en Mars 2021].

[37] SDxCentral, «What Is Network Virtualization?,» [En ligne]. Available: www.SDxCentral.com/networking. [Consulter en Février 2021].

[38] Imtech, «NFV-Network Function Virtualization,» 28 Septembre 2020. [En ligne]. Available: https://imtech.wp.imt.fr/2020/04/20/quesaco-le-nfv-network-function-virtualization/?fbclid=IwAR29BXkND1--1xAeFQE9Q2kCtD7UIa88F0dS8giZlKjmkzegzpEiVHZxYdA.. [Consulter en Décembre 2020].

[39] ETSI Org., «Network Functions Virtualization,» [En ligne]. Available: https://www.etsi.org/images/files/ETSITechnologyLeaflets/NetworkFunctionsVirtualization.pdf?fbclid=IwAR0LSQiFqqmgF0A_2nG1p9acKASNKotuUr_ofg06Pf30SmInpKqzLVlG3Vc. [Consulter en Janvier 2021].

[40] Redhat, «Qu'est-ce que la virtualisation des fonctions réseau ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/virtualization/what-is-nfv?fbclid=IwAR2sqJDrhhEwBBGgUC69AD3Jout3KPoi0FqOeRyBv5m0eDqJU6DFq8yY0Sc. [Consulter en Janvier 2021].

[41] Memoireonline, «Monitoring d'une infrastructure informatique sur base d'outils libres.,» [En ligne]. Available: https://www.memoireonline.com/04/12/5604/m_Monitoring-dune-infrastructure-informatique-sur-base-doutils libres10.html?fbclid=IwAR1ZA2NmPxLzMUfIvrZCSVk3fXeRPLgB_. [Consulter en Janvier 2021].

[42] IEEE Org, «OpenSource MANO. IEEE Software Defined Networks.,» [En ligne]. Available: https://sdn.ieee.org/newsletter/july-2016/opensource-mano. [Consulter en Décembre 2020].

[43] English J., «What is NFV MANO (network functions virtualization management and orchestration)?,» 3 Octobre 2016. [En ligne]. Available: https://searchnetworking.techtarget.com/definition/NFV-MANO-network-functions-virtualization. [Consulter en Mars 2021].

[44] Cloudify, «ONAP vs OSM - The Battle for NFV MANO Supremacy.,» 14 Mai 2021. [En ligne]. Available: https://cloudify.co/blog/onap-vs-osm/?fbclid=IwAR0EgL527De9efNRi_RBYGPS94mzwpjUNcKCoIS0VES_PpRBTcb0rGhNKj8.. [Consulter en Mai 2021].

[45] Andrushko, D., , R. H., D. A., T., & A., «What is the best NFV Orchestration platform? A review of OSM, Open-O, CORD, and Cloudify.,» 30 Octobre 2020. [En ligne]. Available: https://www.mirantis.com/blog/which-nfv-orchestration-platform-best-review-osm-open-o-cord-.. [Consulter en Mars 2021].

[46] Petit B., «SDN : principes et fonctionnement.,» 24 Mars 2020. [En ligne]. Available: https://blog.wescale.fr/2018/03/01/sdn-principes-et-fonctionnement/.. [Consulter en Janvier 2021].

[47] Openstack Org., «Openstack,» [En ligne]. Available: https://docs.openstack.org/security-guide/compute.html.. [Consulter en Mars 2021].

[48] Openstack Org., «Openstack - Dashboard,» [En ligne]. Available: https://docs.openstack.org/security-guide/dashboard.html.. [Consulter en mars 2021].

[49] Ubuntu, «Get started with MicroStack.,» [En ligne]. Available: https://ubuntu.com/tutorials/microstack-get-started#5-launch-and-access-a-vm.. [Consulter en Mars 2021].

[50] Microstack, «MicroStack - OpenStack in a snap.,» [En ligne]. Available: https://microstack.run/docs. [Consulter en Mars 2021].

[51] Redhat, «Qu'est-ce que Kubernetes ?,» [En ligne]. Available: https://www.redhat.com/fr/topics/containers/what-is-kubernetes.. [Consulter en Février

2021].

[52] Kubernetes, «Qu'est-ce-que Kubernetes ?,» 3 Septembre 2020. [En ligne]. Available: https://kubernetes.io/fr/docs/concepts/overview/what-is-kubernetes/.. [Consulter en Février 2021].

[53] Kubernetes, «Solution professionnelle d'orchestration de conteneurs.,» [En ligne]. Available: https://kubernetes.io/fr/.. [Consulter en Mars 2021].

[54] Syentis SAS, «Kubernetes: Syentys.,» [En ligne]. Available: https://www.syentys.com/kubernetes.. [Consulter en Février 2021].

[55] Kubernetes, «Solution professionnelle d'orchestration de conteneurs.,» [En ligne]. Available: https://v1-17.docs.kubernetes.io/fr/.. [Consulter en Février 2021].

[56] Kubernetes, «Pods. Kubernetes.,» [En ligne]. Available: https://kubernetes.io/docs/concepts/workloads/pods/. [Consulter en Février 2021].

[57] Journal du Net. La Rédaction, «Plugin : définition simple et pratique.,» [En ligne]. Available: https://www.journaldunet.fr/web-tech/dictionnaire-du-webmastering/1445312-plugin-definition-simple-et-pratique/.. [Consulter en Décembre 2020].

[58] Arora S., «What is Ansible and how to use Ansible in Docker?,» 26 Mai 2021. [En ligne]. Available: www.simplilearn.com/tutorials/ansible-tutorials/what-is-ansible. [Consulter en Juin 2021].

[59] Linuxcontainers Org., «What's LXD? Linux Containers - LXD -,» [En ligne]. Available: https://linuxcontainers.org/lxd/introduction/. . [Consulter en Janvier 2021].

[60] Garrison J., Nova K., «oreilly.com,» [En ligne]. Available: https://www.oreilly.com/library/view/cloud-native-infrastructure/9781491984291/ch01.html?fbclid=IwAR0NR6VNRq_2AD-um2rBGQe548KqIdbBzUhUXV05SBDeO2J2q1y--uHZ1RI.. [Consulter en Décembre 2020].

[61] English J., «What is NFV MANO (network functions virtualization management and orchestration)?king.techtarget.com/definition/NFV-MANO-network-functions-virtualization,» 3 Octobre 2016. [En ligne]. Available: https://searchnetworking.techtarget.com/definition/NFV-MANO-network-functions-virtualization. [Consulter en Mars 2021].

[62] Staragile, «What Is Ansible? Uses, Benefits, Architecture.,» [En ligne]. Available:

https://staragile.com/blog/what-is-ansible-in-devops.. [Consulter en Mars 2021].