Vérification d'anomalies dans les services réseau NFV externalisés vers le cloud

Moubarak Zoure

université de BORDEAUX

THÈSE PRÉSENTÉE

POUR OBTENIR LE GRADE DE

DOCTEUR DE

L'UNIVERSITÉ DE BORDEAUX

ÉCOLE DOCTORALE DE MATHÉMATIQUES ET INFORMATIQUE (EDMI)

SPÉCIALITÉ INFORMATIQUE

Par Moubarak ZOURE

Vérification d'anomalies dans les services réseau NFV externalisés vers le cloud

Sous la direction de : Toufik AHMED
Co-directeur : Laurent RÉVEILLÈRE

Soutenue le 18/03/2022

Membres du jury :

M. DRIRA, Khalil	Directeur de recherche CNRS Toulouse	Président/Examinateur
M. BEYLOT, André-Luc	Professeur INP-ENSEEIHT	Rapporteur
M. BROMBERG, David	Professeur Univesité de Rennes 1	Rapporteur
M. AHMED, Toufik	Professeur Université de Bordeaux	Directeur de thèse
M. RÉVEILLÈRE, Laurent	Professeur Université de Bordeaux	Co-directeur de thèse

Titre : Vérification d'anomalies dans les services réseau NFV externalisés vers le cloud

Résumé

La Virtualisation des Fonctions Réseau (NFV) transforme la façon dont les entreprises déploient, maintiennent et font évoluer leurs services réseau. Avec la NFV, les entreprises peuvent déployer diverses fonctions réseau telles que les routeurs, les pare-feu, les équilibreurs de charges sous forme d'instances logicielles s'exécutant dans des serveurs standards bon marché. En adoptant cette nouvelle approche, les entreprises externalisent de plus en plus leurs services réseau vers le cloud, pour récolter finalement les fruits du cloud computing qui incluent la haute disponibilité, la réduction des couts et l'accès à des ressources théoriquement infinies.

Cependant, de nombreuses entreprises hésitent à emboiter cette nouvelle approche, à cause du manque de confiance envers le cloud. Une telle réticence s'explique par l'opacité du cloud, le risque de comportement malhonnête du fournisseur cloud, et le risque d'attaques de l'intérieur ou de l'extérieur. Ainsi, les entreprises qui envisagent d'externaliser leurs services réseau vers le cloud manquent cruellement de garanties quant à la conformité de leurs services réseau par rapport à leurs spécifications. Les violations de spécifications de service réseau nuisent grandement à la réussite commerciale et à l'image des entreprises, car ces violations compromettent leurs objectifs de sécurité, de qualité de service et de résilience. Ainsi, l'externalisation des services réseau vers le cloud nécessite avant tout d'établir la confiance entre les entreprises et le cloud.

Dans cette thèse, nous soutenons que la vérification systématique de service réseau reste la clé pour combler le manque de confiance des entreprises envers le cloud. La vérification consiste à confronter l'état du service réseau pour détecter des anomalies de service réseau ou pour évaluer la conformité de l'état du service réseau par rapport à sa spécification. Ainsi, avec des mécanismes de vérification, les entreprises peuvent anticiper et détecter les violations de spécifications, *alias anomalies de services réseau*, y remédier et réclamer des compensations auprès des fournisseurs de cloud-NFV.

Cette thèse propose plusieurs contributions dans le contexte de la vérification de services réseau externalisés vers le cloud. D'abord, nous présentons une taxonomie des anomalies de service réseau qui peuvent survenir dans les environnements NFV. Parallèlement à cette taxonomie, nous analysons les impacts négatifs des anomalies de service réseau sur des attributs de service critiques tels que la confidentialité, l'intégrité, et la performance. Comme seconde contribution, nous introduisons VeriNeS, un système de vérification qui observe le comportement de l'Orchestrateur NFV, afin de détecter des anomalies de service réseau. En interceptant les commandes configurations émises par l'Orchestrateur NFV, VeriNeS construit un état global de tous les services réseau au lieu d'un état par service réseau. Avec cette approche, VeriNeS surmonte des limites cardinales de l'architecture NFV qui incluent le manque de détails pour corréler l'état des services réseau avec leurs propriétaires respectifs. VeriNeS

s’intègre à l’architecture NFV sans besoin de modifier les composants existants et répond aux requêtes de vérification en un temps acceptable pour des scénarios réels de déploiement.

Mots clés : NFV, vérification, anomalies, service réseau, externalisation, orchestration, cloud computing**.**

Title: Verification of anomalies in NFV-based network services outsourced to the cloud

Abstract

Network Functions Virtualization (NFV) is subverting the way enterprises deploy, maintain, and evolve their network services. With NFV, enterprises can deploy various network functions such as routers, firewalls, and load balancers as software instances running on commercial off-the-shelf servers. By adopting this new approach, enterprises are increasingly outsourcing their network services to the cloud, ultimately reaping the rewards of cloud computing, which include high availability, lower costs, and theoretically infinite resources.

However, many enterprises are reluctant to embrace this new approach, due to a lack of trust in the cloud. Such reluctance stems from the opacity of the cloud, the risk of dishonest behavior by the cloud provider, and the risk of attacks from inside or outside. As a result, enterprises considering outsourcing their network services to the cloud are sorely lacking in assurances that their network services will meet their specifications. Violations of network service specifications are highly detrimental to the business success and image of enterprises, as these violations compromise their security, quality of service, and resiliency objectives. Thus, outsourcing network services to the cloud requires first and foremost establishing trust between enterprises and the cloud.

In this thesis, we argue that a systematic verification of network services remains the key to addressing the lack of enterprises' trust in the cloud. Verification involves confronting the network service state to detect network service anomalies or to assess the compliance of the network service state with its specification. Thus, with verification mechanisms, enterprises can anticipate and detect specification violations, a.k.a. network service anomalies, remediate them, and claim compensation from cloud-NFV providers.

This thesis proposes several contributions in the context of verification of outsourced network services to the cloud. First, we present a taxonomy of network service anomalies that can occur in NFV environments. Along with this taxonomy, we analyze the negative impacts of network service anomalies on critical service attributes such as confidentiality, integrity, and performance. As a second contribution, we introduce VeriNeS, a verification system that observes the behavior of the NFV Orchestrator to detect network service anomalies. By intercepting configuration commands issued by the NFV Orchestrator, VeriNeS builds a global state of all network services instead of a single state per network service. With this approach, VeriNeS overcomes cardinal limitations of the NFV architecture that include the lack of details to correlate the state of network services with their respective owners. VeriNeS integrates with the NFV architecture without the need to modify existing components and responds to verification requests in an acceptable time frame for real-world deployment scenarios.

Keywords: NFV, verification, anomalies, network service, outsourcing, orchestration, cloud computing.

Unité de recherche

LaBRI (Laboratoire Bordelais de Recherche en Informatique) UMR 5800 - 351 Cours de la Libération, 33405 Talence

Remerciements

Je dédie cette thèse à Micheline, l'amour de ma vie.

Je remercie toutes les personnes qui de près ou de loin ont contribué à l'accomplissement de cette thèse.

Merci à mes deux co-directeurs de thèse, Pr. Laurent RÉVEILLÈRE et Pr. Toufik AHMED d'avoir encadré ma thèse et de m'avoir donné une formation scientifique solide.

Merci à Micheline YAGO, ma fiancée, qui m'a témoigné de beaucoup de patience, de compréhension, d'attention et d'amour afin que je puisse atteindre mes objectifs dans cette thèse.

Merci à ma famille de m'avoir soutenu moralement pendant toutes ses années de thèses.

Merci à Monsieur Kabongo MAMBA qui a été un soutient morale permanent tout au long de ma thèse.

Merci à Tidiane SYLLA pour l'intérêt qu'il a porté sur mes travaux de recherche et de m'avoir aidé à préparer ma soutenance.

Merci à tous.

Table des matières

Tables des figures

Listes des tableaux

Acronymes

Acronyme	Signification	Traduction en français
ACL	Acces Control List	liste de contrôle d'accès
AMD SEV	AMD Secure Encrypted Virtualization	virtualisation sécurisée et cryptée d'AMD
API	Application Programming Interface	interface de programmation d'application
APP	Australian Privacy Principles	principes australiens de protection de la vie privée
ASIC	Application-Specific Integrated Circuit	circuit intégré propre à une application
AWS	Amazon Web Service	
BSS	Business Support Systems	systèmes de support commercial
CCPA	California Consumer Privacy Act	loi californienne sur la protection de la vie privée des consommateurs
CORE RANK	Computing Research and Education Association of Australasia's Rank	système de classement de l'association australienne pour la recherche et l'éducation en informatique
CVE	Common Vulnerabilities and Exposures	vulnérabilités et expositions courantes
CVSS	Common Vulnerability Scoring System	système de notation des vulnérabilités courantes
DPDK	Data Plane Development Kit	kit de développement du plan de données
ERP	Enterprise Resource Planning	planification des ressources de l'entreprise
ETSI	European Telecommunications Standards Institute	Institut européen des normes de télécommunications
ETSI GS	ETSI Group Specification	groupe de spécification de l'ETSI
FPGA	Field-programmable Gate Array	réseau de portes programmables sur site
GCP	Google Cloud Platform	
GPU	Graphics Processing Unit	processeur graphique
HIPAA	Health Insurance Portability and Accountability Act	loi sur la portabilité et la responsabilité en matière d'assurance maladie
IAAS	Infrastructure as a Service	infrastructure en tant que service
IDS	Intrusion Detection System	système de détection d'intrusion
IMA	Linux Integrity Measurement Architecture	architecture de mesures d'intégrité de Linux
INTEL SGX	Intel Software Guard Extensions	extensions de la protection logicielle d'Intel
IPS	Intrusion Prevention System	système de prévention d'intrusion
ISO	International Organization for Standardization	organisation internationale de normalisation
KVM	Kernel-based Virtual Machine	machine virtuelle basée sur le noyau
NAAS	Network as a Service	réseau en tant que service
NAT	Network Address Translation	traduction d'adresse réseau
NFV	Network Function Virtualization	Virtualisation des Fonctions Réseau
NFV MANO	NFV Management and Orchestration	gestion et orchestration de NFV
NFVI	NFV Infrastructure	Infrastructure NFV
NFVO	NFV Orchestrator	Orchestrateur NFV
NIST	National Institute of Standards and Technology	institut national des normes et de la technologie
NPU	Network Processing Unit	unité de calcul réseau
OASIS	Organization for the Advancement of Structured Information Standards	organisation pour l'avancement des normes d'information structurée
ODP	OpenDataPlane	

OS	Operating System	système d'exploitation
OSM	Open Source MANO	
OSS	Operational Support Systems	systèmes de support opérationnel
PAAS	Platform as a Service	plateforme en tant que service
PCI DSS	Payment Card Industry Data Security Standard	norme de sécurité de l'industrie des cartes de paiement
PCR	Platform Configuration Register	registre de configuration de plateforme
QOS	Quality of Service	qualité de service
REST	Representational State Transfer	transfert d'état représentationnel
RGPD	Règlement Général sur la Protection des Données	
SAAS	Software as a Service	logiciel en tant que service
SAS 70	Statement on Auditing Standards no.70	déclaration sur les normes d'audit no.70
SDN	Software Defined Networking	réseau défini par logiciel
SLA	Service Level Agreement	contrat de niveau service
SMT	Satisfiability Modulo Theories	satisfiabilité modulo des théories
TCAM	Ternary Content Adressable Memory	mémoire adressable à contenu ternaire
TOSCA	Topology and Orchestration Specification for Cloud Applications	spécification de la topologie et de la spécification pour les applications cloud
TPM	Trusted Platform Module	module de plateforme de confiance
VIM	Virtualized Infrastructure Manager	Gestionnaire d' Infrastructure Virtualisée
VM	Virtual Machine	machine virtuelle
VNF	Virtualized Network Function	Fonction Réseau virtualisée
VNFM	VNF Manager	Gestionnaire de VNF
XSS	Cross-Site Scripting	scripting intersites

Liste des publications

Publications internationales

- **Moubarak Zouré**, Toufik Ahmed and Laurent Réveillère. (**2021**, March). *VeriNeS : runtime verification of outsourced network services orchestration*. **In Proceedings of the 36th Annual ACM Symposium on Applied Computing** (pp. 1138-1146). https://doi.org/10.1145/3412841.3441988

- **Moubarak Zouré**, Toufik Ahmed and Laurent Réveillère. (**2022, January**). *Network Services Anomalies in NFV : Survey, Taxonomy, and Verification Methods*. **IEEE Transactions on Network and Service Management**. https://doi.org/10.1109/TNSM.2022.3144582

Publications nationales

- **Moubarak Zouré**, Toufik Ahmed and Laurent Réveillère. (**2019**). *Attestation de l'intégrité des chaînes de fonctions de services dans les environnements cloud*. **Abstract paper, Compass 2019**.

- **Moubarak Zouré**, Toufik Ahmed and Laurent Réveillère. (**2019**). *Network service integrity attestation in cloud environments*. **Poster, Compass 2019**

Chapitre 1

1 Introduction

1.1 Vers une externalisation des fonctions réseau vers le cloud

Une adoption massive du cloud

La *migration* vers le cloud s'accélère [1]. Elle s'impose comme « *la nouvelle normalité* », car aujourd'hui, 94 % des entreprises exploitent au moins un service cloud [2]. L'exemple emblématique en la matière est celui de Netflix qui a misé sur la stratégie du « *tout cloud* » depuis 2008. En janvier 2016, la multinationale annonce la fermeture de son dernier centre de données et exécute désormais tous ses services applicatifs dans l'infrastructure d'AWS [3],[4]. Le cloud [5] offre une panoplie de ressources informatiques (calcul, stockage, réseau) qui sont devenues des *commodités* [6] comme l'eau, l'électricité et le gaz. En d'autres termes, les entreprises s'approvisionnent des ressources *à la demande* auprès de fournisseurs cloud, tout en se *déchargeant* de la gestion de ces ressources et en *payant uniquement à l'usage*. Ainsi, en exploitant le cloud, les entreprises substituent leurs dépenses d'investissement en dépenses opérationnelles, variables et réduites. De plus, le cloud délivre des services hautement disponibles, stimule et accélère l'innovation, et permet aux entreprises de rendre leurs services applicatifs accessibles à l'échelle mondiale en quelques minutes [7]. Gartner prédit que d'ici 2025, 85 % des grandes entreprises externaliseront leurs applications vers des fournisseurs cloud [8]. D'ici 2026, 45 % du budget informatique des entreprises sera alloué aux investissements dans le cloud public, contre moins de 17 % en 2021 [9].

Vers des services réseau externalisés vers le cloud

Dès 2012, des acteurs majeurs des télécommunications tels que AT&T, Orange, et Verizon révolutionnent [10] l'architecture des réseaux informatiques en poussant à l'extrême les concepts de cloud computing et de virtualisation. Ces opérateurs veulent s'affranchir des limites des réseaux traditionnels qui sont rigides et difficiles à configurer, nécessitent d'importants investissements, engendrent une dépendance envers les fournisseurs, et requièrent une expertise qui évolue rapidement. Avec l'ETSI (European Telecommunications Standards Institute), ils introduisent la NFV (Network Function Virtualization) [10], un paradigme consistant à implémenter les fonctions réseau [11] physiques telles que les routeurs, les équilibreurs de charge et les pare-feu sous forme de logiciels. Les fonctions réseau virtualisées (VNFs) s'exécutent dans des machines virtuelles (VMs) déployées dans une infrastructure de cloud privé ou public p. ex., AWS, AZURE, GCP. Avec ce paradigme, les entreprises sont en train d'*externaliser massivement* [12], [13] leurs VNFs vers le cloud, afin d'amasser les fruits du cloud. Pour les fournisseurs cloud, la NFV ouvre la voie à des modèles économiques

fortement innovants et lucratifs [14] tels que le Network as a Service (NaaS) [15]. De grands fournisseurs cloud tels qu'AWS proposent déjà un large spectre de VNFs prêtes-à l'emploi, par ex., AWS ELB (équilibreur de charge), AWS Transit Gateway (routeur), AWS Network Firewall, AWS Route 53 (DNS), et AWS Shield (mitigeur d'attaque de déni de service), AWS Cloud Front (diffusion de contenu). Les entreprises créent ainsi rapidement des services réseau performants en chainant diverses VNFs. Le marché global de la NFV devrait croitre de 34,9 % chaque année pour atteindre 122 milliards de dollars d'ici 2027, selon une étude de MeticulousResearch [16].

1.2 Des freins à l'adoption de la NFV

Les entreprises reconnaissent unanimement les bénéfices de l'externalisation des fonctions réseau vers le cloud à travers la NFV. Cependant, avant d'adopter cette externalisation, elles doivent accepter une perte de visibilité et de contrôle direct sur leurs services réseau. Par exemple, les procédures internes du cloud, les fichiers de configuration et de journalisation restent opaques aux entreprises. Par conséquent, ces dernières manquent de garanties quant à la conformité des services réseau déployés dans le cloud par rapport à leurs spécifications. L'opacité du cloud permet à un fournisseur cloud malhonnête de violer une ou plusieurs clauses de spécifications tout en clamant le contraire. Supposons qu'une entreprise de commerce en ligne exige que le fournisseur cloud héberge ses VNFs dans l'espace européen, afin de se conformer au RGPD (Règlement Général sur la Protection des Données). Pour réduire ses couts de fonctionnement, le fournisseur cloud peut relocaliser discrètement les VNFs vers un centre de donnée située hors de l'espace européen. Une telle violation peut compromettre la confidentialité des données des utilisateurs lorsque les VNFs sont relocalisées dans des territoires qui possèdent des législations laxistes et liberticides en matière de protection des données utilisateurs. De plus, lorsque les juridictions européennes constatent cette violation, elles peuvent imposer à l'entreprise de lourdes amendes et des procès interminables, qui résultent sur une détérioration de son image. Avec des risques de telles violations, plusieurs entreprises montrent une réticence solide pour externaliser des VNFs critiques vers le cloud. Les facteurs suivants exacerbent le risque de violation et sont typiques aux environnements NFV basés sur le cloud :

- *Large surface d'attaque* : NFV élargit significativement la surface d'attaque. Les plateformes NFV reposent sur une large pile de logiciels allant des hyperviseurs (par ex., Xen, KVM, Docker) aux outils d'automatisation et de gestion (par ex., OpenStack, Cloudify, Chef, Ansible), en passant par les logiciels d'isolation réseau et d'accélération du plan de données (par ex. OpenVswitch, DPDK, FD.io). Chacun de ses logiciels expose diverses vulnérabilités (par ex., CVE-2018-15402 et CVE-2020-3236), décuplant ainsi les possibilités pour un adversaire de violer les spécifications de services réseau des entreprises.

De plus, l'hétérogénéité de ces logiciels due à la variété de leurs éditeurs et langages de codage complexifie la gestion des vulnérabilités dans les plateformes NFV.

- *Co-résidence des VNFs* : dans les plateformes NFV, les VNFs de différents locataires (entreprises) partagent le même serveur physique ou virtuel. Un colocataire malicieux peut conduire des attaques par canal latéral [17],[18],[19], [20] contre des VNFs d'autres colocataires, pour voler ou corrompre des données sensibles, ou perturber le fonctionnement des VNFs. Par exemple, Youngjoo Shin et *al.* [17] ont prouvé que l'accès partagé au cache du CPU permet à un colocataire d'inférer les règles de filtrage du pare-feu (VNFs) d'un autre colocataire.

- *Problèmes de configuration* : plusieurs administrateurs gèrent l'ensemble de la pile NFV et restent susceptibles à des erreurs de configuration. Une étude de Facebook [21] révèle que 13 % des incidents de réseau à l'intérieur de ses centres de données relèvent d'erreurs de configuration. La même étude conclut que le taux d'erreur humaine double celui du matériel.

- *Problèmes d'incohérence intercouches* : la nature multicouche de NFV introduit des problèmes d'incohérence intercouches[22], [23],[24]. Une couche inférieure peut ne pas appliquer les directives de la couche supérieure à cause des problèmes de synchronisation, de bogues, de pannes ou d'injection de logiciels malveillants.

- *Comportements malhonnêtes du cloud* **:** les fournisseurs cloud détiennent une vue complète des services réseau et des accès privilégiés aux composantes logicielles de la plateforme NFV. Un fournisseur cloud malhonnête peut fournir des services réseau non conformes pour optimiser ses couts. De plus, des entreprises concurrentes peuvent comploter avec le fournisseur cloud pour saboter des services réseau ou soustraire des données confidentielles.

1.3 La vérification systématique

Le *manque de confiance* des entreprises envers les environnements NFV inhibe l'externalisation des fonctions réseau vers le cloud. Ce fossé critique doit être franchi par le champ de recherche de NFV avant de considérer une adoption à grande échelle. *Dans cette thèse, nous soutenons que la vérification systématique est la clé pour construire chez les entreprises réticentes une confiance nécessaire pour externaliser des fonctions réseau critiques dans des environnements NFV dont elles se méfient.* La vérification assure à une entreprise la conformité de ses services réseau par rapport à leurs spécifications respectives. Avec des mécanismes de vérification, l'entreprise anticipe et détecte les violations de

spécifications, *alias anomalies de services réseau*, y remédie et réclame des compensations auprès du fournisseur de cloud-NFV.

Cependant, nous avons observé un manque significatif de travaux de recherche qui traitent la vérification dans NFV. La plupart des travaux de recherche se focalisent sur des thématiques telles que l'orchestration des services réseau, l'optimisation de performance des VNFs et le placement des VNFs. Or, la sécurité, la confiance et la conformité restent les préoccupations majeures des entreprises qui envisagent l'adoption de NFV.

Considérer la vérification dans NFV soulève deux questions de recherche essentielles :

Q1 : *quels types d'anomalies (violations) doit anticiper une entreprise qui externalise ses services réseau dans un cloud NFV* ?

Q2 : *comment vérifier un service réseau pour détecter des violations de spécifications* ?

1.4 Contributions

Cette thèse présente deux contributions majeures.

La première contribution (cf. Section 3) présente une taxonomie des anomalies de service réseau dans les environnements NFV. Nous analysons l'impact de chaque anomalie sur les objectifs de sécurité, de performance et de résilience de l'entreprise. Nous scrutons la littérature pour identifier les mécanismes existants pour vérifier les anomalies de service réseau et identifions des domaines de recherche critiques non encore explorés.

La deuxième contribution (cf. Section 4) présente VeriNeS, un système détectant plusieurs types d'anomalies de service réseau dans des environnements NFV. Les anomalies de services réseau proviennent souvent de la compromission de l'orchestrateur NFV ou de problèmes de configuration au niveau de l'orchestrateur NFV. Notre approche soutient que l'observation du comportement de l'orchestrateur NFV permet de connaitre l'état exact des services réseau, et donc de détecter effacement les violations. Ainsi, VeriNeS intercepte les commandes de configuration envoyées par l'orchestrateur, construit un super-graphe en analysant ces commandes, et répond aux requêtes de vérification en cherchant un monomorphisme entre le graphe du service réseau à vérifier et le super-graphe. Nous avons montré que VeriNeS détecte plusieurs anomalies de services réseau en un temps acceptable pour un déploiement réel de NFV.

Le Tableau 1.1 lie chaque contribution aux questions de recherche qu'elles abordent.

Tableau 1.1 : Questions de recherche abordées dans chaque contribution.

	Q1	Q2
Contribution 1	☑	
Contribution 2		☑

1.5 Organisation de la thèse

Nous structurons le reste de cette thèse comme suit :

- Le Chapitre 2 fournit au lecteur les principaux concepts autour de la NFV, qui permettent de comprendre le reste de cette thèse. Nous présentons le contexte général de cette thèse, qui est la tendance galopante à l'externalisation des services réseau vers le cloud. Aussi, nous discutons des problématiques de confiance que soulève cette nouvelle tendance.

- Le Chapitre 3 présente notre contribution sur la taxonomie des anomalies de service réseau dans les environnements NFV. En outre, nous établissons l'état de l'art sur les méthodes de vérification existantes pour ces anomalies. Enfin, nous identifions des pistes de recherche pertinentes dans le domaine de la vérification d'anomalies de service réseau dans NFV.

- Le Chapitre 4 décrit notre deuxième contribution. Nous introduisons VeriNeS, un système de vérification d'anomalies de service réseau, qui se base sur l'observation du comportement de l'Orchestrateur NFV. Nous décrivons dans ce chapitre la conception, l'implémentation et l'évaluation de VeriNeS.

- Enfin, dans le Chapitre 5, nous dressons un résumé global des contributions que nous avons apportées dans cette thèse. Nous présentons aussi des perspectives pour nos travaux de recherche.

Chapitre 2

2 Vers une externalisation des services réseau vers le cloud

2.1 Les réseaux traditionnels et leurs limites

2.1.1 Caractéristiques des réseaux traditionnels

Du simple commutateur à la boite intermédiaire. Au début d'Internet, les réseaux se composaient de commutateurs et de routeurs, se chargeant du transfert des paquets. Avec les exigences complexes et croissantes de nouvelles applications, de nouveaux types de fonctions réseau appelées *boites intermédiaires* foisonnent et prolifèrent (cf. Figure 2.1) dans le réseau. Ces fonctions réseau « r*emplissent des fonctions autres que les fonctions normales et standard d'un routeur IP sur le chemin des datagrammes entre un hôte source et un hôte de destination* » [11]. Les boites intermédiaires [11] transforment, filtrent, manipulent et inspectent les paquets sur leur chemin, afin de réaliser des objectifs allant de la performance (équilibrage de charge, mise en cache, compression) à la sécurité (chiffrement, filtrage de trafic, détection d'intrusion). Une enquête [25] sur l'utilisation des fonctions réseau en entreprise rapporte que certaines entreprises emploient jusqu'à 1946 boites intermédiaires dans leur réseau (cf. Figure 2.1).

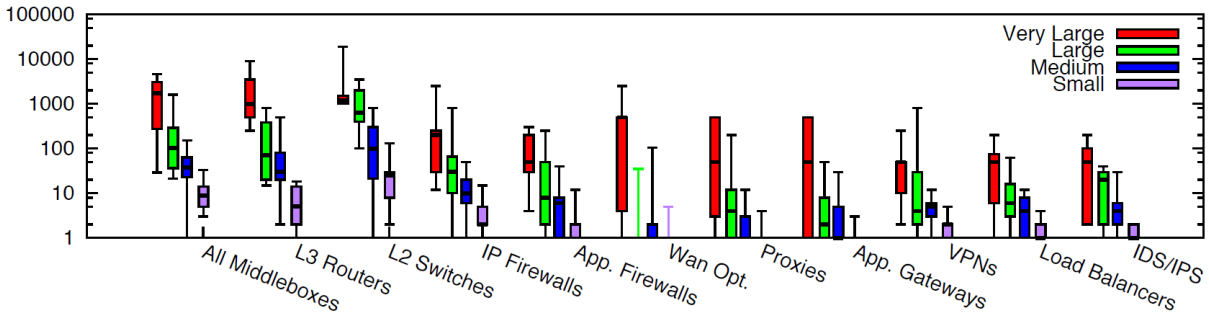


Figure 2.1 : Distribution des types de fonctions réseau utilisés dans les entreprises [25].

Les données ont été récoltées auprès de 57 entreprises, ayant des réseaux de taille variable : petite (inférieur à 1K machines), moyenne (1K-10K), grande (10K-100K), très grande (supérieur à 100K).

Des fonctions réseau fortement couplées aux matériels. Les équipementiers réseau tels que Cisco, HP ou Juniper délivrent traditionnellement les fonctions réseau sous forme d'appliances matérielles. Ces dernières correspondent à des « *boites noires* » (cf. Figure *2.2*) s'appuyant sur du matériel spécialisé dans des types de traitements, p. ex., routage, filtrage, inspection approfondie. Les constructeurs codent leurs fonctions réseau pour qu'elles exploitent les fonctionnalités de modules matériels spécialisés et

propriétaires, p. ex., ASIC, FPGA, NPU, TCAM. La couche logicielle des fonctions réseau reste quasi impossible à modifier car opaque et protégée par des licences propriétaires. De plus, les équipements réseau présentent une faible interopérabilité, souvent induite délibérément par les équipementiers eux-mêmes, pour des raisons commerciales.

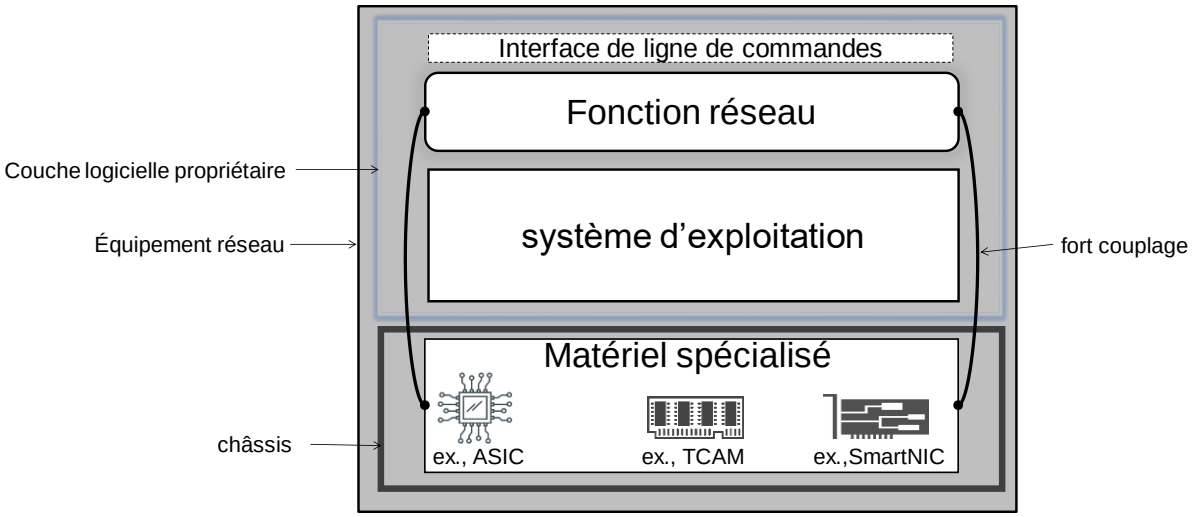


Figure 2.2 : Architecture générique des équipements réseau.

2.1.2 Limites des réseaux traditionnels

Afin de comprendre les limites des réseaux traditionnels, considérons une entreprise imaginaire dénommée *Innover*, dont l'objectif central est de rester compétitif tout en augmentant ses revenus. *Innover* adopte alors une stratégie qui consiste à proposer continuellement des services réseau innovants (p. ex., diffusion vidéo, visioconférence, bureau à distance) à ses clients et à ses employés. Dans la Figure 2.3, nous soulignons les obstacles qu'*Innover* affronte dans chaque étape du cycle d'innovation.

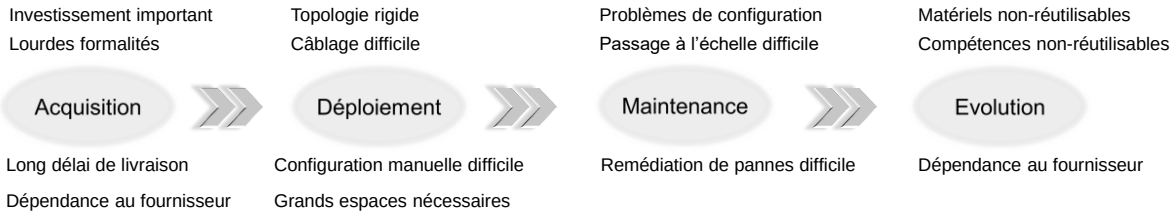


Figure 2.3 : Inconvénients l'approche traditionnelle de la mise en réseau tout au long du cycle de vie du service réseau.

Acquisition du matériel. Pour déployer un nouveau service réseau, les administrateurs réseau d'*Innover* câblent un ensemble de fonctions réseau physiques qu'ils acquièrent auprès de divers équipementiers réseau, p. ex., Cisco, HP, Juniper, Huawei. Chaque fonction réseau nécessite un *budget important,* car son cout d'achat inclut le prix du matériel spécialisé et celui des licences logicielles. Justine Sherry et *al.* [25] rapporte que dans les réseaux d'entreprise de très grandes tailles (plus de cent mille nœuds), les investissements dans les fonctions réseau atteignent souvent un à cinquante millions de dollars sur une

période de cinq ans. De plus, le *délai d'acquisition* des équipements réseau reste important, car *Innover* doit d'abord, leur affecter une ligne budgétaire, négocier leur prix et celui des services de support, négocier un contrat de niveau service, avant d'attendre la livraison du matériel, qui intervient après plusieurs semaines. Ce processus d'acquisition du matériel lent et complexe inhibe l'innovation chez les administrateurs réseau d'*Innover*.

Déploiement. Une fois le matériel livré, *Innover* entrepose les équipements réseau dans des salles serveur. Ceux-ci occupent des espaces volumineux, nécessitent des systèmes d'alimentation, de climatisation et de sécurité sophistiqués et onéreux. Les administrateurs réseau câblent les fonctions selon un ordre logique prédéfini. Une telle approche ossifie la topologie du réseau, qui s'adapte difficilement aux évolutions de la spécification du service réseau.
Lorsque les administrateurs réseau envisagent de changer de topologie, ils doivent entièrement recâbler le réseau, ce qui s'annonce comme une tâche complexe, le nombre de câbles dans une salle serveur atteignant souvent des milliers [26]. Après le câblage, les administrateurs réseau appliquent à chaque fonction réseau diverses configurations (p. ex., politique de routage, règle de filtrage, liste de contrôle d'accès, règles de priorité de trafic). La configuration manuelle nécessite une expertise pointue, car les interfaces de configuration d'un équipement réseau varient d'une version à l'autre et d'un équipementier à l'autre. *Innover* limite ainsi ses choix de fonctions réseau par rapport au capital de compétences de ses administrateurs réseau. Si toutefois elle décide d'acquérir des équipements réseau qu'elle ne maitrise pas, mais qui offrent des fonctionnalités innovantes, elle devra former son personnel [27]. Or, la formation implique des moyens financiers importants et réduit le temps de travail du personnel.

Maintenance et évolution du service réseau. Innover se heurte souvent à des pannes causées par des problèmes de configuration [25]. Ces pannes restent complexes à identifier et à résoudre à cause de l'opacité des équipements réseau. Dans certains cas, la résolution de panne nécessite l'intervention sur site des ingénieurs de l'équipe support de l'équipementier. *Innover* se retrouve donc avec des services réseau indisponibles pendant plusieurs heures et subit des pertes économiques considérables et une baisse de productivité des employés. Aussi, les administrateurs réseau d'*Innover* observent des pics de trafic pendant des périodes de l'année. *Innover* doit alors investir de nouveau pour l'acquisition d'équipements réseau, car ses administrateurs réseau ne peuvent ni augmenter les ressources physiques des fonctions réseau existantes (passage à l'échelle vertical) ni dupliquer ses fonctions réseau (passage à l'échelle vertical). Pour les pics de trafic survenant de façon imprévisible, la marge de manœuvre d'*Innover* reste limitée puisque l'acquisition du matériel est un long processus. Pour anticiper ces situations, *Innover* surdimensionne les ressources des services réseau, augmentant ainsi ses dépenses d'investissements. Lorsqu'*Innover* requiert des fonctionnalités supplémentaires pour une fonction réseau, elle doit ranger le matériel existant et acheter un nouvel équipement. Le plus souvent, *Innover* se fournit auprès du même constructeur (Cisco par exemple) à cause des problèmes d'interopérabilité

entre équipements de constructeurs différents et parce qu'elle veut capitaliser sur son héritage de compétences sur les produits du constructeur. Enfin, lorsqu'*Innover* lance un nouveau service réseau et veut l'abandonner aussitôt à cause d'un revirement de stratégie, les investissements consentis pour l'acquisition du matériel deviennent des pertes économiques. Ainsi, *Innover* hésite souvent à lancer un nouveau service, même si celui-ci regorgeait de potentialités en termes d'innovation et de gains.

En résumé, les réseaux traditionnels se caractérisent par le fort couplage entre les fonctions réseau et le matériel. Pour une entreprise comme *Innover*, les réseaux traditionnels présentent plusieurs limites qui incluent :

- Couts d'investissement et couts opérationnels élevés : ici la notion de cout dépasse l'aspect financier et inclut les ressources humaines, les moyens immobiliers et le temps et les efforts du personnel.
- Réduction de l'agilité commerciale
- L'inhibition de l'innovation

En insistant sur l'approche traditionnelle de la mise en réseau, les entreprises sont condamnées à l'échec dans une arène économique où la compétitivité, l'innovation et l'agilité commerciale deviennent les clés de la réussite. En conséquence, un nouveau paradigme réseau s'impose.

2.2 La virtualisation des fonctions réseau (NFV) : un nouveau paradigme réseau

Le nouveau paradigme NFV. Plusieurs chercheurs ont proposé (depuis 2009) [28] [29], [30] de surmonter les limites des réseaux traditionnels avec un principe architectural simple : 1) *casser la dépendance « fonction réseau/matériel »*, afin de 2) délivrer les fonctions réseau *sous forme d'appliances logicielles* pouvant s'exécuter sur des serveurs physiques *standard bon marché*. Ces serveurs peuvent être localisés dans des nœuds réseau, des salles de serveurs ou dans des centres de données privés ou publics. Ce nouveau paradigme appelé NFV (Network Function Virtualisation) a reçu une attention particulière depuis 2012, quand, sept des principaux opérateurs des réseaux de télécommunication (p. ex., AT&T, Orange, Telefonica) l'ont promu [10] comme l'architecture réseau du futur. La NFV permet de consolider plusieurs fonctions réseau virtualisées (VNFs) dans un même serveur. Les VNFs s'exécutent dans des unités d'isolation (machine virtuelle, conteneurs, unikernels), qui sont fournies par des systèmes de virtualisations appelés hyperviseurs tels que Xen [31], Docker [32], ClickOS [33] et MirageOS [34]. La NFV s'appuie sur des plateformes d'automatisation et d'orchestration [35] pour déployer, gérer, et faire passer à l'échelle des services réseau, correspondant à des chaines de VNFs.

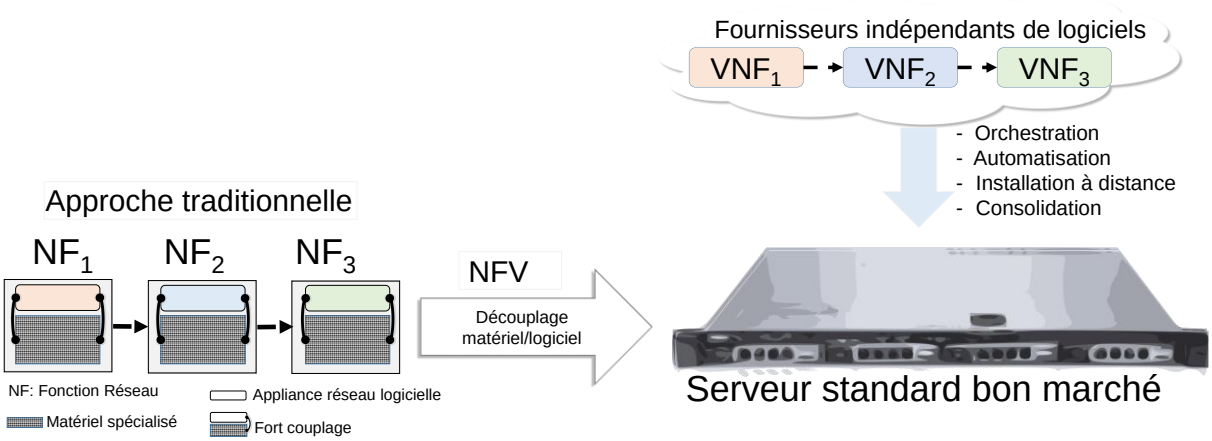


Figure 2.4 : Comparaison entre l'approche traditionnelle de la mise en réseau et l'approche NFV.

Avantages. Pour cerner quelques avantages qu'apporte NFV à la mise en réseau, considérons à nouveau le scénario précédent (cf. Section 2.1.2) illustrant les limites des réseaux traditionnelles. Une entreprise comme *Innover*, qui adopte la NFV s'affranchit des dépenses d'investissement nécessaires pour l'acquisition du matériel, en payant uniquement la licence d'exploitation du logiciel. *Innover* ajoute des nouvelles fonctionnalités au réseau en téléchargeant les VNFs correspondantes sur le site en ligne du fournisseur logiciel, augmentant ainsi la vélocité du temps de mise sur le marché. L'approche logicielle du réseau réduit significativement l'espace et la consommation énergétique des salles de serveurs d'*Innover*. Aussi, la topologie du réseau devient flexible, car les VNFs sont interconnectés par des liens virtuels qu'*Innover* crée, modifie et supprime à la demande. En exploitant des outils d'orchestration, de surveillance et de gestion de pannes des VNFs disponibles sur le marché, *Innover* réduit la complexité des tâches de maintenance et la taille de son équipe d'administrateurs réseau. De plus, *Innover* gère facilement les pics de trafic soit en augmentant les ressources virtuelles des unités d'isolation hébergeant les VNFs ou en créant de nouvelles instances de VNFs. Enfin, *Innover* peut adapter les VNFs à ses besoins en développant de nouveaux modules logiciels.

En résumé, la NFV repose sur trois grands principes : 1) le découplage des fonctions réseau du matériel, 2) l'exécution et la consolidation des VNFs dans des serveurs standard, et 3) l'orchestration et l'automatisation du cycle de vie des services réseau. Les bénéfices offerts par NFV vont de la réduction des dépenses à la flexibilité en passant par la facilité d'innovation.

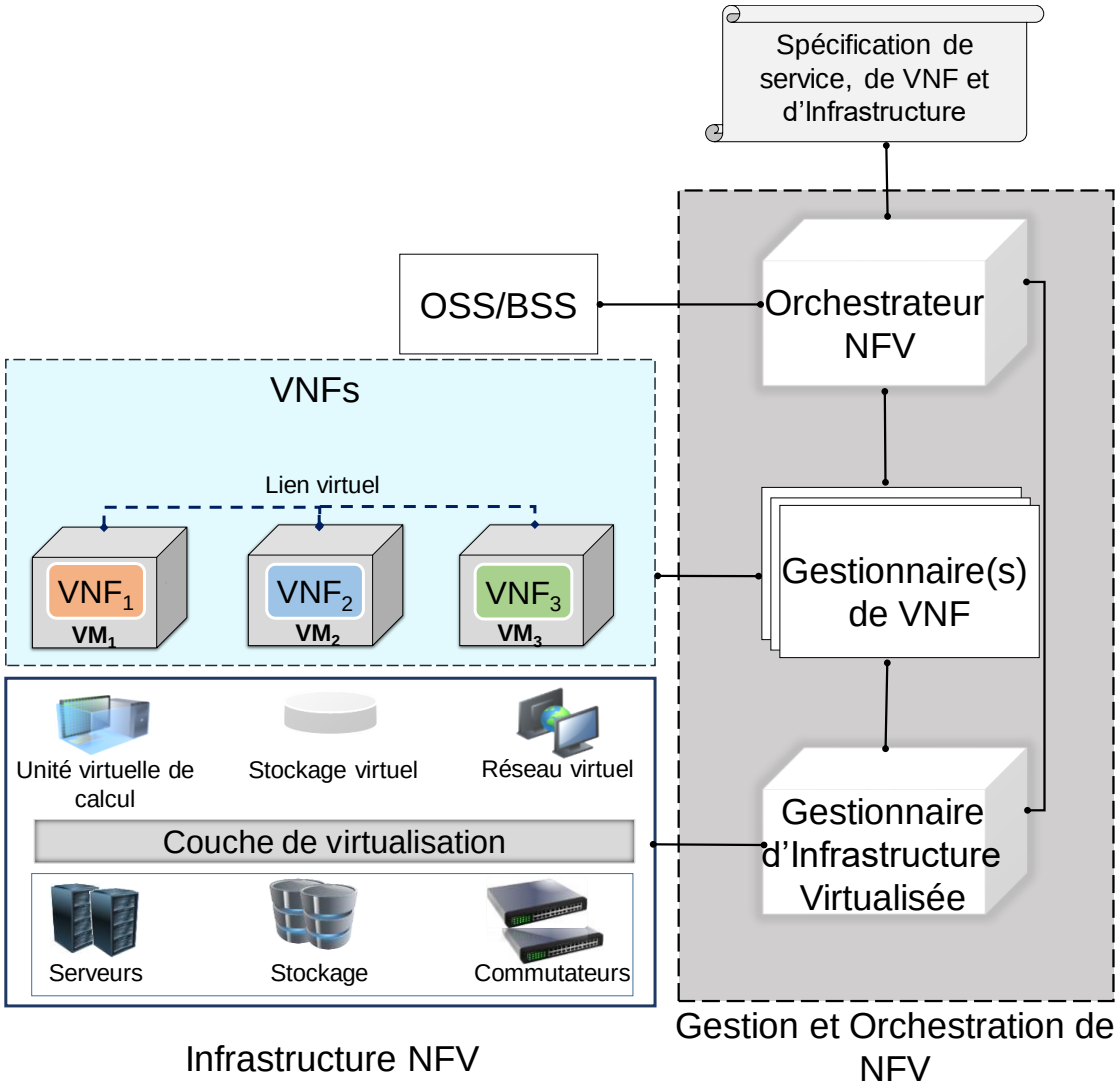


Figure 2.5 : Cadre architectural de référence de NFV [36].

2.3 Le cadre architectural de NFV

La Figure 2.5 dépeint l'architecture de référence [36] de la plateforme NFV définie par l'ETSI[1]. Le cadre architectural est structuré en trois couches : 1) la couche d'Infrastructure NFV (NFVI), 2) la couche des VNFs, et 3) la couche de Gestion et d'Orchestration NFV (NFV MANO).

L'Infrastructure NFV comprend un ensemble de serveurs, de périphériques réseau et de dispositifs de stockage, qui supportent l'exécution des VNFs. Des hyperviseurs (p. ex., Xen, VMware, Docker) partitionnent ces ressources physiques en ressources logiques pour créer des unités d'isolation (VMs, conteneurs, unikernels) et des liens virtuels connectant les unités d'isolation entre elles. Le choix du type d'unités d'isolation reste un compromis entre la performance et l'isolation [37].

[1] Depuis 2012, l'ETSI a dédié un groupe de spécification dénommé « ETSI ISG NFV » qui a publié plus d'une centaine de rapports et documents de spécification sur NFV.

Les VNFs correspondent aux versions logicielles de fonctions réseau physiques telles que les pare-feu, les systèmes de détection d'intrusion, les équilibreurs de charges, les proxys et les transcodeurs vidéo. Ces VNFs sont exécutées dans les unités d'isolation fournies par l'Infrastructure NFV. Ainsi, la quantité des ressources virtuelles et le nombre d'instances de VNFs sont augmentables ou réductibles à la demande. La combinaison du comportement individuel de plusieurs VNFs apporte des fonctionnalités réseau enrichies appelées services réseau.

Le NFV MANO gère et orchestre le cycle de vie des services réseau, des VNFs, et les ressources de l'Infrastructure NFV. Il comprend trois composantes fonctionnelles : 1) le Gestionnaire d'Infrastructure Virtualisée (VIM), 2) le gestionnaire de VNF (VNFM), et 3) l'Orchestrateur NFV (NFVO). Le VIM gère et contrôle l'Infrastructure NFV. Il assure l'allocation et la désallocation des ressources virtuelles aux VNFs. Le VNFM quant à lui gère le cycle de vie et la configuration des instances de VNFs. L'Orchestrateur NFV [38] gère le cycle de vie des services réseau. Il assure le déploiement, le passage à l'échelle, la surveillance et la réparation des services réseau. Les spécifications de services réseau dictent les décisions de l'Orchestrateur NFV, à savoir, la quantité et les caractéristiques de ressources à allouer aux VNFs, les tables de routage à appliquer aux réseaux virtuels, les événements à considérer pour le passage à l'échelle des VNFs, et les exigences de qualité de services à appliquer aux VNFs.

Le Système de Support Opérationnel et Commercial (OSS/BSS) inclut un ensemble d'applications supportant la gestion des réseaux du fournisseur NFV et l'automatisation des fonctions commerciales telles que la gestion des requêtes de service et la facturation. Cependant, OSS/BSS reste une composante optionnelle à l'architecture NFV.

2.4 NFV a besoin de boosters de performance

Exécuter des fonctions réseau sur du matériel standard (serveurs x86) *tout en gardant les mêmes performances* représente un défi majeur pour NFV. Certaines fonctions réseau telles que les applications audio et vidéo exigent des performances élevées et variées en termes de latence, de gigue, de bande passante et d'entrées/sorties. Par exemple, la latence de bout en bout maximale pour la téléphonie sur IP est bornée à 150 *ms* [39]. Certaines applications de sécurité doivent pouvoir traiter chaque seconde 190 Gbits de paquets cryptés. De plus, les cartes réseau physiques de nouvelles générations sont capables de traiter jusqu'à 100 Gbit de paquets par seconde [40], [41], réduisant ainsi le temps maximal de traitement des paquets à 67 *ns* [42]. Ainsi, le monde académique et industriel exploite des technologies d'accélération assisté par le matériel et logiciel pour supporter ces exigences [43], [44]. L'accélération assistée par le matériel consiste à décharger des modules de fonctions réseau dans du matériel spécialisé (GPU, FPGA, NPU, cartes réseau intelligentes) pour ces types de traitements, afin d'atteindre des performances maximales. Les techniques d'accélération assistée par le logiciel ajoutent des couches logicielles (DPDK, Netmap, ODP) aux VNFs, aux hyperviseurs ou aux unités d'isolation

pour éliminer les frais généreux et les traitements qui réduisent les performances réseau. Par exemple la plateforme DPKD (Kit de Développement du Plan de Données) élimine les frais généreux des interruptions système en contournant [45] la pile réseau du noyau Linux, permettant ainsi d'avoir des performances de traitements jusqu'à dix fois plus élevées.

Bien que l'accélération NFV solutionne certains défis de performance, nous notons que cette approche réintroduit une dépendance entre les fonctions réseau et des modules matériels et logiciels. Or, le but ultime du paradigme NFV est de pouvoir exécuter des VNFs sur n'importe quelle Infrastructure NFV, sans se soucier des dépendances matérielles et logicielles. Pour répondre à cette problématique, l'ETSI GS NFV propose des couches d'abstraction pour l'accélération NFV [46]. Ces couches d'abstractions exposent aux VNFs des interfaces pour exploiter les fonctionnalités d'accélération.

2.5 Spécification et orchestration des services réseau NFV

Les utilisateurs NFV définissent les exigences des services réseau en matière de déploiement et de comportement opérationnel à travers des *spécifications de services réseau*, aussi appelées *descripteurs de services réseau* [47]. La structure générique et les exigences fonctionnelles des spécifications de service réseau sont standardisées par un groupe de spécification dénommée *ETSI GS NFV* [47].

L'Orchestrateur NFV maintient un catalogue de fichiers de spécification, qui, chacun, possède un identifiant utilisé dans le cadre des requêtes de création de services. Les spécifications de service réseau contiennent une liste de références qui pointent vers les descripteurs des VNFs composant le service réseau. Les descripteurs de VNF définissent un ensemble d'unités virtuelles de déploiement correspondant à des unités d'isolation (p.ex., VMs, conteneurs) hébergeant les modules logiciels d'une VNF. Pour chaque unité virtuelle de déploiement, l'utilisateur NFV définit son image, ses contraintes de placement, et ainsi que ses demandes de capacité en termes de nombre de processeurs, de taille mémoire vive et d'espace de stockage.

Dans la spécification du service réseau, l'utilisateur NFV décrit la topologie du service réseau en listant un ensemble de liens virtuels, correspondant à des commutateurs virtuels connectant les VNFs entre elles. Les VNFs sont connectées aux liens virtuels à travers des cartes réseau virtuelles appelées points de connexion.

Les spécifications de services réseau incluent des références de descripteurs de graphe d'acheminement qui expriment des politiques de routage appliquées aux profils de trafic entrant dans le service réseau. Les descripteurs de graphe d'acheminement comprennent des classificateurs de flux et des chemins de transfert. Les classificateurs de flux utilisent des règles de classification (p. ex., adresse IP, numéro de port, numéro de protocole) pour décrire les profils de trafic. Un chemin de transfert ou (chaine de fonctions de service) spécifie une liste ordonnée de VNFs qu'un profil de trafic doit suivre.

Les utilisateurs NFV définissent aussi dans la spécification, des règles de mise à l'échelle automatique des instances de VNF et de service réseau. Chaque règle de mise à l'échelle comporte un évènement qui doit déclencher un ensemble d'actions de l'Orchestrateur NFV. Les évènements correspondent à des assertions sur les valeurs des métriques de performances et d'indicateurs de VNFs. La spécification de service réseau inclut des scripts de gestion de cycle de vie (création, réparation) du service réseau écrit dans un langage dédié (DSL). Aussi, les utilisateurs NFV référencent dans la spécification un ensemble de point d'accès de service du service réseau.

Enfin, les spécifications de service réseau possèdent des profils de déploiement qui définissent des exigences de différents cas d'utilisation du service réseau. Un profil de déploiement de service réseau comprend des contraintes de priorité et de disponibilité de service, des règles d'affinité/anti-affinité, des profils de déploiement de VNFs et de liens virtuels. Par exemple, le profil de déploiement d'une VNF spécifie un nombre d'instances minimal et maximal, des règles d'affinité/anti-affinité entre VNFs, et un niveau de disponibilité de service.

L'ETSI GS NFV reconnait la norme TOSCA [48] (Topology and Orchestration Specification for Cloud Applications) comme langage d'implémentation des spécifications de service réseau. Standardisée par l'OASIS[2], TOSCA définit un langage de modélisation décrivant les composantes, les relations, les dépendances, les exigences, et les capacités des applications hébergées dans le cloud. TOSCA déclare aussi les procédures de gestion des applications cloud et permet de les orchestrer et migrer à travers différentes infrastructures cloud. L'OASIS fournit un profil de TOSCA appelé TOSCA NFV[49], qui est adapté aux modèles de données des spécifications de services réseau NFV. Dans l'industrie, plusieurs projets d'implémentation d'Orchestrateur NFV (p. ex., Cloudify, OpenStack Tacker, OpenBaton) s'appuient sur le profil TOSCA NFV.

2.6 Quand NFV rencontre le cloud computing

Le concept de NFV s'inspire de celui du cloud computing qui l'a précédé. En d'autres termes, NFV est une application des techniques de cloud computing aux fonctions réseau. Dans cette section, nous montrons que le cloud computing s'impose comme un catalyseur de NFV et que la combinaison entre NFV et le cloud computing crée une nouvelle tendance qui consiste à externaliser les fonctions réseau vers le cloud. Nous présentons aussi des obstacles à cette externalisation des fonctions réseau vers le cloud.

[2] OASIS : Organization for the Advancement of Structured Information Standards

2.6.1 Un bref aperçu sur cloud computing

Selon le NIST, le cloud computing est le provisionnement *à la demande* via *internet* d'un *ensemble* de ressources informatiques (par exemple, réseaux, serveurs, stockage, applications*) hautement* configurables, *élastiques, évolutives, co-hébergées avec d'autres locataires de cloud,* et avec une *facturation à l'usage* [50]. Le cloud permet ainsi d'exécuter des applications diverses (serveur Web, gestionnaire de base de données, intergiciel) dans des unités d'exécution virtuelles, en se déchargeant de la gestion et de la maintenance de l'infrastructure physique sous-jacente, et de l'orchestration des ressources virtuelles.

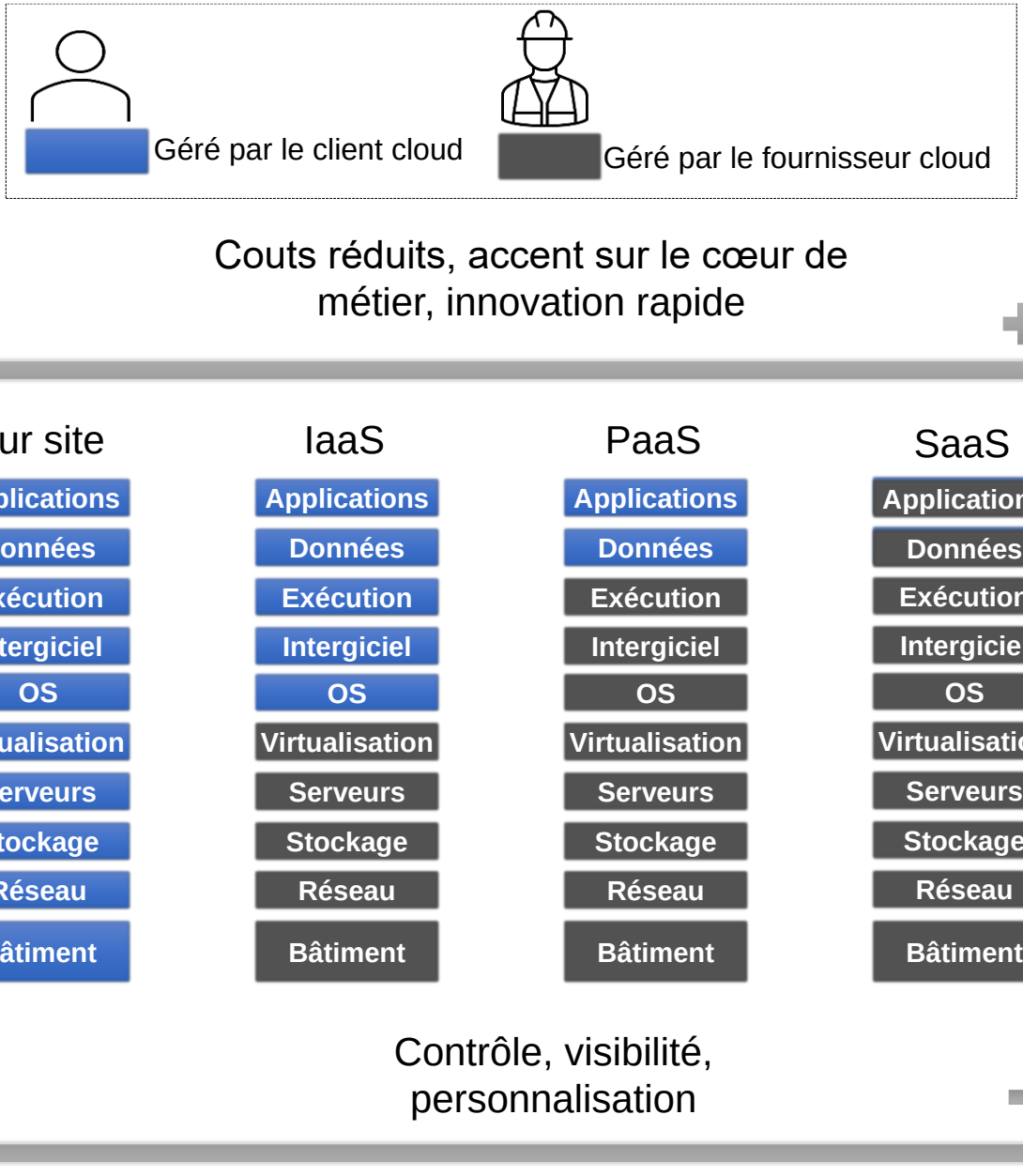


Figure 2.6 : Niveaux de responsabilité de l'utilisateur sur la pile du cloud, en fonction du modèle de service.

Les fournisseurs cloud proposent trois modèles de services qui varient selon le niveau de visibilité, de contrôle et de responsabilité des utilisateurs sur la pile du cloud (cf. Figure 2.6) :

- *Le logiciel en tant que service (SaaS)* : le SaaS correspond à des offres de services tels que Gmail, Slack, Dropbox, ou Office 365, dans lesquels les utilisateurs exploitent via internet des applications prêtes à l'emploi déployées et gérées par des fournisseurs cloud. Ce modèle de service correspond au plus bas niveau de contrôle et de responsabilité de l'utilisateur. Autrement dit, le fournisseur cloud est responsable du provisionnement, de la gestion et de la sécurité de l'application SaaS ainsi que la pile logicielle et matérielle supportant son exécution. Le client SaaS possède néanmoins un accès à des paramètres de configuration lui permettant d'adapter le comportement de l'application à ses besoins.

- *La plateforme en tant que service (PaaS)* : ce modèle de service fournit aux clients un environnement logiciel complet (p. ex., OS, bibliothèques, intergiciels, outils) et des ressources physiques (p. ex., serveurs, stockage) pour développer et déployer leurs propres applications. Les clients PaaS détiennent un contrôle total sur les applications. Les PaaS correspondent par exemple à des gestionnaires de base de données (AWS DynamoDB, AWS RDS), des intergiciels (AWS MQ, AWS IoT Core) ou des environnements de développement logiciels (AWS Correto, AWS CodeStar).

- *L'infrastructure en tant que service (IaaS)* : l'IaaS offre le plus haut niveau de flexibilité et de contrôle pour l'utilisateur cloud et le plus bas niveau de responsabilité pour le fournisseur cloud. Ce dernier gère uniquement l'infrastructure physique (bâtiments, alimentation, serveurs physiques, systèmes de câblage). Le fournisseur cloud fournit aux clients IaaS des pools de ressources physiques gérés par un hyperviseur. Les ressources physiques (calcul, mémoire, stockage, réseau) se présentent souvent aux clients IaaS sous forme de ressources virtuelles (VMs, conteneurs, réseaux virtuels) dont le cycle de vie est indépendant de celui de l'infrastructure physique. Les exemples d'IaaS sur le marché du cloud incluent AWS EC2 et RedHat OpenShift.

Le NIST identifie quatre modèles de déploiement du cloud en fonction des consommateurs finaux du cloud :

- *Le cloud privé* : un cloud privé demeure sous la responsabilité administrative d'une seule organisation et reste uniquement accessible à ses utilisateurs. Le déploiement et la gestion de l'infrastructure cloud (sur site ou hors site) incombe à l'organisation ou à une partie tierce. Par exemple, OVH propose une solution appelée *Hosted Private Cloud* [51] qui délivre aux organisations des infrastructures de cloud entièrement gérées par OVH.

- *Le cloud communautaire :* le cloud communautaire offre des services cloud aux utilisateurs de plusieurs organisations (gouvernements, universités, banques centrales) ayant des intérêts communs.

- *Le cloud public* : un fournisseur de cloud public (entreprise, gouvernement, université) offre des ressources et services cloud (gratuits ou facturés) ouverts au grand public. Le fournisseur de cloud public gère et héberge l'infrastructure cloud. AWS, Microsoft Azure et Google Cloud Platform sont actuellement les leaders mondiaux du cloud public.

- *Le cloud hybride* : le cloud hybride consiste à combiner différents modèles de déploiement (privé, hybride, communautaire) du cloud. Par exemple, AWS fournit une solution appelée *VMWARE Cloud on AWS* permettant d'étendre un cloud privé basé sur VMWARE vers l'infrastructure AWS.

2.6.2 Le cloud computing en tant que catalyseur de NFV

Le cloud, le père, NFV, le fils. NFV et le cloud computing sont deux paradigmes intrinsèquement similaires, mais avec des objectifs différents. En rappel, le cloud computing exploite massivement la virtualisation pour consolider tout type d'applications dans des machines virtuelles, qui sont déployées de manière flexible et dynamique à travers des outils d'orchestration et d'automatisation, tout en assurant une indépendance du cycle de vie de ces applications de l'infrastructure physique sous-jacente. NFV reprend les mêmes concepts au profit des fonctions réseau, qui exigent des performances et une fiabilité largement plus élevées que les applications standards. En termes simples, NFV est un cas d'utilisation du cloud. D'ailleurs, beaucoup de plateformes cloud telles qu'OpenStack et Kubernetes sont intégrées massivement dans l'architecture NFV.

Le cloud comme catalyseur de NFV. Le cloud représente une opportunité pour faciliter et accélérer le déploiement de NFV. Les entreprises peuvent exécuter leurs VNFs sur leurs propres serveurs. Cependant, elles se privent de la flexibilité et de l'élasticité qu'offre le cloud. De plus, l'approche sans le cloud accroit le risque de résurgence de la dépendance aux fournisseurs. Certains fournisseurs tels que CISCO offre des services NFV délivrés avec du matériel et du logiciel propriétaire. Ainsi, l'approche cloud reste la meilleure option pour réaliser la vision de NFV et optimiser ses avantages. Certains blocs et composantes de l'architecture NFV (p. ex. NFVI, NFV MANO, VNFs) peuvent être proposées *en tant que service* par des fournisseurs cloud. Par exemple, certains fournisseurs de logiciels VNFs tels Nokia délivrent des VNFs en tant que services qui s'exécutent dans une Infrastructure NFV (NFVI) entièrement déployée et gérée par un fournisseur d'IaaS tel qu'AWS [52]. Aussi, Cloudify offre

une solution unifiée d'Orchestrateur NFV (NFVO) et de Gestionnaire de VNFs (VNFM) en tant que service [53].

2.6.3 Le cloud : nouveau terrain de jeu des fonctions réseau

La cloudification du réseau. NFV déloge les fonctions réseau des locaux d'entreprises, et les pousse inexorablement dans le cloud. En *externalisant leurs fonctions réseau vers le cloud* [12], [13], [54], [55], [56], [57], les entreprises récoltent les bénéfices cumulés du cloud et de la NFV [58]. Avec cette nouvelle approche, les entreprises se concentrent davantage sur leur cœur de métier tout en ayant la possibilité de déployer des fonctions réseau innovantes, rapidement, de manière flexible et à la demande. De plus, les services réseau opérant dans le cloud héritent de la haute disponibilité et de la résilience du cloud. Par exemple, comme le montre la Figure 2.7, des fournisseurs cloud tels qu'AWS dispose de plus de 80 zones de disponibilités[3] à travers le monde assurant un niveau élevé de disponibilité et de tolérance aux pannes avec de faibles latences. La *« cloudification du réseau »* s'impose comme *« la nouvelle normalité »*.

Cloud privé ou public ? Bien que le cloud privé reste une option pour opérer les fonctions réseau [59], la majorité des entreprises privilégient le cloud public, car il regorge d'avantages concurrentiels, p. ex., économie de l'échelle, ressources théoriquement infinies, passage à l'échelle mondiale en quelques minutes. Même les grandes multinationales telles que Netflix abandonnent leurs centres de données privés pour miser sur le cloud public, car la maintenance, l'évolutivité et la résilience du cloud privé leur posent d'énormes défis [3], [4]. De plus, les entreprises s'appuient de plus en plus sur des clouds privés externalisés [60], [51], c'est-à-dire que la gestion du cloud est déchargée vers une partie tierce. Au moins, les fonctions réseau devront s'exécuter dans des environnements gérés par une entité tierce.

[3] Une zone de disponibilté : centre de données indépendant localisé dans un espace géographique et représentant un point de défaillance unique

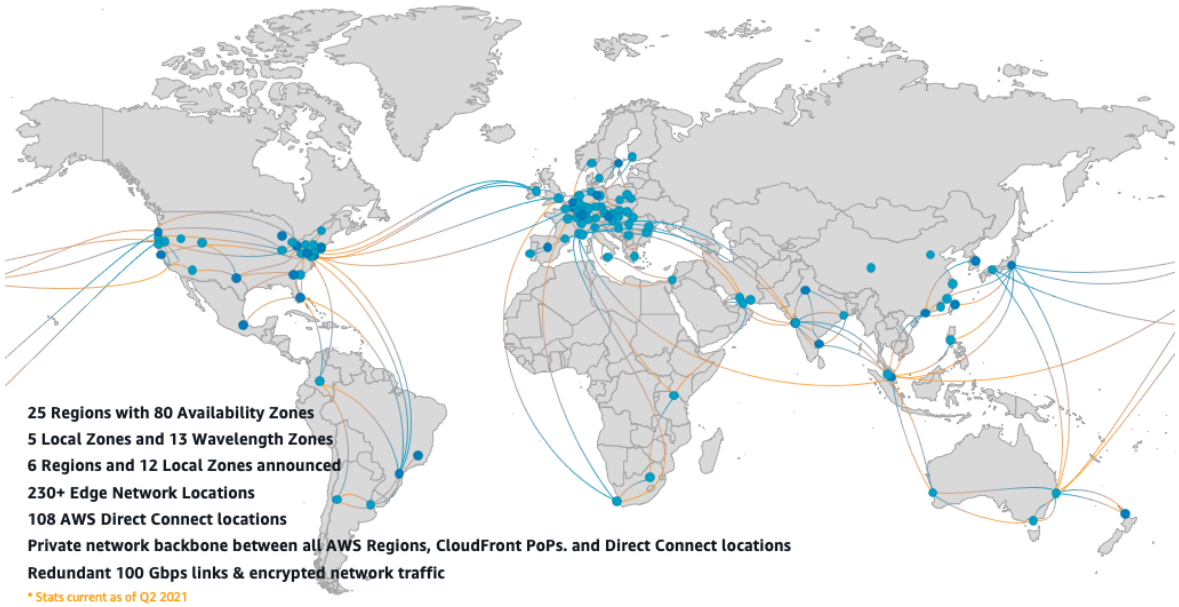


Figure 2.7 : Infrastructure mondiale du cloud d'AWS [61].

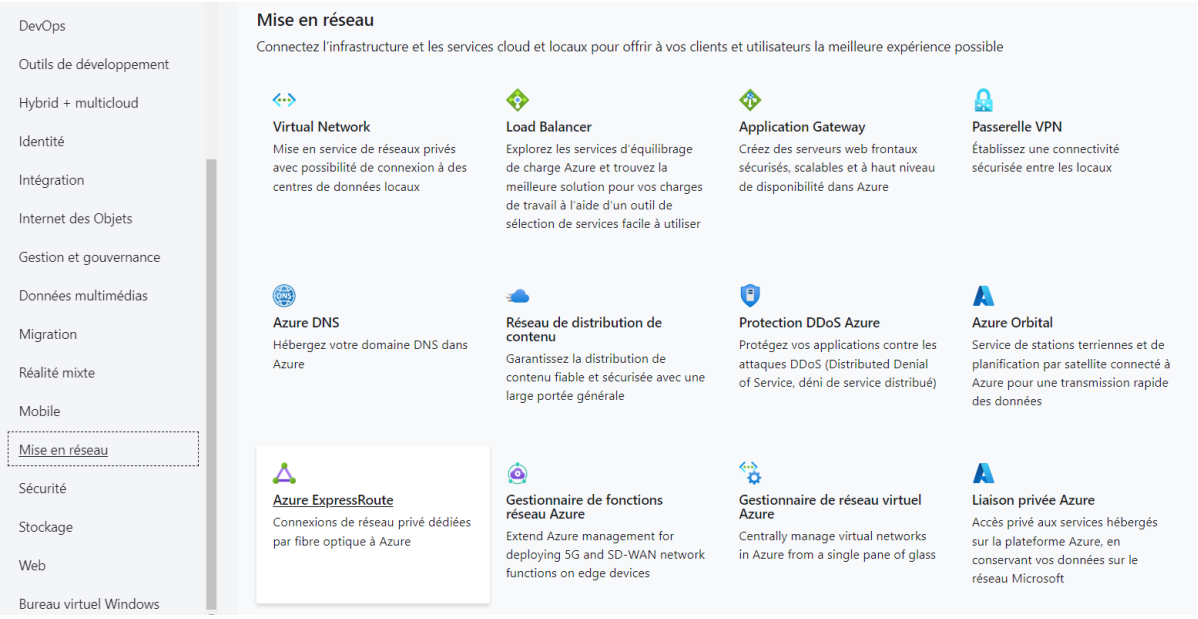


Figure 2.8 : Offre de services cloud de mise en réseau de Microsoft AZURE [62].

Une tendance en plein essor. Plusieurs fournisseurs cloud tels qu'AWS, Microsoft AZURE ou Verizon accompagnent la tendance de l'externalisation des fonctions réseau vers le cloud en offrant diverses fonctions réseau (routeurs, pare-feu, équilibreurs de charge) en tant que service. La Figure 2.8 montre la large gamme de fonctions réseau en tant que service présente sur le portail de services de Microsoft AZURE. Le marché global de la NFV présente un potentiel énorme. Il devrait croitre de 34,9 % chaque année pour atteindre 122 milliards de dollars d'ici 2027, selon une étude de MeticulousResearch [16].

2.7 Des freins à l'externalisation des fonctions réseau vers le cloud

Bien que l'externalisation vers le cloud regorge d'énormes bénéfices, plusieurs entreprises hésitent à migrer des applications critiques vers le cloud. Cette réticence émane du *manque de confiance envers le cloud* [63], [64], qui s'explique par l'opacité du cloud, le risque de comportement malhonnête de la part du fournisseur cloud, et le risque d'attaques de l'intérieur ou de l'extérieur. Ainsi, les entreprises manquent cruellement de garanties quant à la conformité des services réseau s'exécutant dans le cloud par rapport à leurs spécifications. Le strict respect des exigences de service exprimées par les entreprises les assure de maintenir le même niveau de qualité de service et de sécurité et de respect de normes nationales (HIPAA, CCPA, The Privacy Act 1988, NIST 800-53), régionales (RGPD) ou internationales (ISO/CEI 27001, SAS 70, PCI DSS). Cependant, le risque de violations de spécification reste important dans des environnements non fiables, dynamiques et partagés comme le cloud. Et pire encore, les entreprises disposent de peu de moyens pour détecter les violations et les remédier, car l'état des services réseau leur sont souvent opaque. En plus, certains fournisseurs cloud malhonnêtes peuvent manipuler l'état des services réseau pour masquer des violations de spécifications. Au regard de ces risques importants de violations de spécifications, les entreprises doivent trouver des mécanismes pour vérifier leurs services réseau opérant dans le cloud.

2.8 Conclusion

Dans ce chapitre, nous avons détaillé le paradigme NFV et avons décrit l'architecture standard NFV, ainsi que la spécification et l'orchestration des services réseau. Nous avons montré que le cloud computing représente un catalyseur pour l'adoption de la NFV. Ainsi, nous assistons à une externalisation massive des services réseau vers le cloud. Cependant, nous avons souligné que malgré les énormes bénéfices de cette dernière approche, les entreprises expriment une réticence solide à externaliser leurs services réseau vers le cloud, à cause du manque de confiance envers le cloud. Les entreprises redoutent ainsi des anomalies de service réseau, qui peuvent affecter négativement leurs objectifs de sécurité et de qualité de service. Ainsi, les mécanismes de vérification d'anomalies de service réseau deviennent décisifs pour permettre aux entreprises d'externaliser sereinement leurs services réseau vers le cloud.

Dans le chapitre suivant (cf. Section 3), nous présentons notre première contribution. Nous présentons une taxonomie des anomalies de service réseau dans les environnements NFV. Nous scrutons l'état de l'art sur les techniques de vérification existantes pour ces anomalies, et dégageons des pistes de recherche critiques.

Chapitre 3

3 Les anomalies de services réseau dans NFV

Le chapitre précédent (cf. Section 2.7) a montré que les entreprises reconnaissent unanimement les avantages de l'externalisation des fonctions réseau vers le cloud, mais redoutent des violations de spécifications de service réseau que nous appelons aussi anomalies de service réseau. Dans ce chapitre, nous présentons une taxonomie des anomalies de service réseau (cf. Section 3.3), et dressons un état de l'art sur leurs méthodes de vérification (cf. Section 3.4).

3.1 Introduction

Dans les environnements NFV basés sur le cloud, les anomalies de service réseau sont fréquentes, diverses et difficilement identifiables. Pour cause, ces environnements exposent une large surface d'attaque [65], [66] et restent hautement dynamiques [67], [68], et opaques de l'extérieur [63], [69], [70], [71]. Ainsi, l'adoption massive de NFV nécessite chez les entreprises des garanties sur la conformité de leurs services réseau par rapport à leurs spécifications.

Nous défendons que le recours systématique à la vérification fournisse aux entreprises davantage de confiance, leur permettant d'externaliser avec sérénité leurs services réseau vers le cloud. La vérification de service réseau consiste à élaborer des techniques pour confronter l'état des services réseau à leur spécification, afin de détecter des anomalies ou d'établir une conformité.

Avant d'envisager la vérification de service réseau, les entreprises doivent dresser un tableau complet des clauses critiques de spécification de service réseau, et des anomalies de service réseau potentielles. Cependant, la littérature manque d'une analyse approfondie des anomalies de service réseau et manque d'approches de vérification pour ces anomalies. Les études précédentes [65], [66], [72], [73], [74] ont exclusivement abordé les menaces et les vulnérabilités dans les environnements NFV sans identifier les anomalies qu'elles peuvent introduire dans les services réseau déployés.

Cette lacune dans la littérature nous a motivés à mener une analyse profonde de l'état de l'art et à identifier les directions de recherche importantes. Notre objectif est de fournir un aperçu complet des anomalies de service réseau et des techniques existantes permettant de les vérifier.

Contributions. Cette étude comprend les contributions suivantes :

- Tout d'abord, nous analysons les causes profondes des anomalies de service réseau. Nous identifions les menaces et les vulnérabilités inhérentes au cadre NFV qui conduisent à des

anomalies de service réseau. Nous proposons ensuite une définition générique d'une anomalie de service réseau.

- Ensuite, nous scrutons les anomalies possibles de service réseau dans NFV, tout en précisant leurs impacts négatifs sur des attributs de service bien connus tels que la sécurité, la performance et la résilience. Cette étude fournit au lecteur une taxonomie complète des anomalies de service réseau dans les environnements NFV.

- Troisièmement, nous explorons et comparons les techniques existantes pour la vérification de service réseau.

- Quatrièmement, nous identifions les futures directions de recherche pour les approches de vérification afin de détecter les anomalies de service réseau.

Grandes lignes. Le reste du chapitre est structuré comme suit. La Section 3.2 analyse les causes potentielles des anomalies de service réseau. Cette section propose également une définition générique d'une anomalie de service réseau. Nous présentons dans la Section 3.3 une taxonomie des anomalies de service réseau et analysons leur impact sur les attributs de service. Dans la Section 3.4, nous étudions les techniques et les approches existantes pour vérifier les anomalies de service réseau. Nous discutons enfin des futures directions de recherche pour la vérification des anomalies dans la Section 3.5, et concluons ce chapitre dans la Section 3.6.

3.2 Les causes potentielles des anomalies de service réseau

Pour comprendre les causes des anomalies de services réseau, 1) nous identifions les auteurs potentiels d'anomalies et 2) leurs moyens, c'est-à-dire la surface d'attaque de NFV.

3.2.1 Les auteurs d'anomalies de service réseau dans les environnements NFV

Dans les environnements NFV basés sur le cloud, plusieurs acteurs humains ou logiciels provoquent des anomalies de service réseau, intentionnellement ou non. Nous désignons les auteurs potentielles d'anomalies de service réseau par le terme « adversaire ». Nous distinguons cinq types d'adversaires, selon leur appartenance ou non au périmètre du cloud, leur caractère intentionnel ou non et leur niveau d'accès aux points de contrôle du cloud :

- *Les adversaires internes* : les adversaires internes [75] (p. ex., employé du cloud, sous-traitant, ex-employé) appartiennent au périmètre du cloud, et détiennent un niveau d'accès et de contrôle élevé à la couche logicielle et matérielle de la pile NFV. D'une part, les actions des adversaires

internes restent souvent intentionnelles. Par exemple, pour des raisons personnelles ou par corruption, certains administrateurs cloud peuvent exploiter leurs accès privilégiés pour voler des données sensibles [76] ou reconfigurer malicieusement des services réseau. Les adversaires internes concernent aussi des logiciels de la pile NFV (p. ex., Orchestrateur NFV) qui peuvent être infectés par des logiciels malveillants modifiant leur comportement. D'autre part, certains adversaires internes provoquent involontairement des anomalies de service réseau à travers des erreurs de configuration ou d'implémentation logicielle.

- *Le fournisseur cloud malhonnête* : certains fournisseurs cloud profitent de l'opacité de leurs systèmes et procédures internes pour fournir des services réseau non conformes. Les motivations peuvent être d'ordre économique ou politique. Par exemple, un fournisseur cloud malhonnête peut fournir une quantité de ressources virtuelles en dessous de celle demandée par le locataire, afin de servir davantage de locataires. De plus, des entreprises concurrentes peuvent comploter avec le fournisseur cloud pour saboter des services réseau ou soustraire des données confidentielles. Nous distinguons les fournisseurs de cloud malhonnêtes des adversaires internes. Le comportement malhonnête d'un fournisseur cloud résulte d'une politique interne assumée alors que les adversaires internes agissent souvent de manière isolée et opposée aux intérêts du fournisseur cloud. Par ailleurs, les fournisseurs de cloud malhonnêtes peuvent masquer les anomalies de service réseau en présentant aux locataires des états falsifiés de leurs services réseau.

- *Les colocataires cloud* : Parmi les caractéristiques essentielles de NFV, figure la *multilocation*, qui consiste à exécuter des machines virtuelles de locataires différents sur le même hôte (serveur) physique. Cependant, la multilocation soulève des problématiques de sécurité, car un locataire malveillant peut attaquer les machines virtuelles des autres colocataires. Plusieurs travaux [77], [78] ont démontré que dans la plupart des clouds publics (AWS, Azure, GCP), des adversaires peuvent cibler un locataire, et exploiter facilement des vulnérabilités des stratégies de placement de VMs du cloud, afin d'obtenir une colocation avec les machines virtuelles de la cible. Ensuite, l'adversaire devenu colocataire avec la cible peut exploiter des failles de l'hyperviseur ou des techniques d'attaques sophistiquées [17], [19], [20] pour voler des données sensibles [17] ou perturber [79] les services réseau d'autres colocataires. En outre, lorsque les mécanismes d'isolation des hyperviseurs comportent des défaillances, les performances des machines virtuelles des locataires peuvent être impactées par les opérations des machines virtuelles d'autres colocataires [79], [80].

- *Les adversaires externes* : les adversaires externes ne disposent pas d'accès autorisés à la pile logicielle de NFV, mais exploitent des vulnérabilités de cette pile pour compromettre les

services réseau. Ces adversaires sont souvent des concurrents du fournisseur ou du locataire cloud, des activistes ou des groupes de pirates motivés le profit de gain.

- *Le demandeur du service réseau lui-même* : le locataire peut transmettre au fournisseur cloud une spécification erronée due à des erreurs de transcriptions de leurs besoins, des problèmes de versions ou des problèmes d'incohérence [81], [82]. En outre, le demandeur de service peut être à l'origine de problèmes de configuration provoquant des anomalies de service réseau. Si le locataire ne s'aperçoit pas de ses erreurs, une vive controverse peut éclater entre le locataire et le fournisseur cloud. Le locataire se plaindra (à tort) de violations de sa spécification, alors que le fournisseur cloud revendiquera (à raison) l'implémentation correcte de la spécification.

3.2.2 La surface d'attaque de NFV

En plus des menaces communes des environnements cloud [83], le cadre NFV présente des menaces spécifiques, qui émanent de la nouvelle pile logicielle (cf. Section 2.3) introduite par NFV. Par conséquent, le cadre NFV expose une surface d'attaque plus étendue que les réseaux traditionnels. Dans ce contexte, Pattaranantakul et al. [65] ont examiné les multiples menaces liées à l'architecture NFV. Ils ont proposé une taxonomie qui comprend des menaces pour les couches de gestion et d'orchestration NFV (NFV MANO), de l'Infrastructure NFV et de VNFs. Nous résumons cette taxonomie comme suit :

- *Menaces de la couche NFV MANO* : NFV MANO expose des vulnérabilités qu'un adversaire pourrait exploiter pour compromettre les services réseau. Une personne de confiance ayant un accès privilégié au logiciel de NFV MANO peut reconfigurer de manière malveillante les services réseau déployés. Les adversaires peuvent également exploiter des vulnérabilités (p. ex., CVE-2018-15402, CVE-2019-1946, CVE-2020-3478, CVE-2019-1971) dans les interfaces externes du logiciel NFV MANO pour contrôler les services déployés. En outre, des problèmes de synchronisation ou une configuration incohérente dans les logiciels de la couche NFV MANO pourraient entraîner des incohérences entre les services réseau et leur spécification.

- *Menaces de la couche d'Infrastructure NFV* : les menaces de l'Infrastructure NFV vont des réseaux traditionnels aux menaces de la virtualisation. Ces menaces comprennent les problèmes de sécurité des machines virtuelles, des hyperviseurs, des interfaces de gestion, des ressources virtuelles et des attaques matérielles telles que les attaques par canal latéral. D'autres menaces spécifiques aux environnements NFV comprennent l'application inadéquate des politiques de sécurité, les menaces liées à la multilocation et les personnes de confiance malveillantes.

- *Menaces de la couche VNFs* : la couche VNFs présente des menaces liées à la gestion de la sécurité. Des adversaires pourraient profiter des vulnérabilités du logiciel des VNFs, par exemple CVE-2012-2663, CVE-2006-5276, pour contrôler les VNFs ou violer les politiques de sécurité définies par les locataires. En outre, la menace provient souvent de la configuration par défaut appliquée aux VNFs ou de l'implémentation non sécurisée des protocoles de communication, par exemple SSL et TLS. Une personne de confiance disposant de permissions suffisantes peut avoir accès aux VNFs et exfiltrer des données sensibles. Un adversaire peut également exploiter les vulnérabilités des interfaces de gestion pour compromettre les VNFs ou violer la confidentialité des données des utilisateurs.

3.2.3 Une définition générique des anomalies de services réseau

Un bref rappel sur la spécification de service réseau. Un service réseau augmente la valeur ajoutée du réseau avec des fonctionnalités qui résultent de la combinaison du comportement distinct de plusieurs VNFs. Les locataires utilisent une spécification pour définir le comportement attendu du service réseau tout au long de son cycle de vie (instanciation, mise à jour, mise à l'échelle, fin). La spécification se compose de clauses qui contribuent individuellement à différents aspects du comportement global du service réseau. Parmi les exemples de clauses de spécification figurent la topologie de service réseau, le graphe d'acheminement, les contraintes d'allocation des ressources, la politique d'isolation et la politique de mise à l'échelle.

Une définition générique. Nous appelons *« anomalie de service réseau »* la *violation* d'une ou plusieurs clauses de spécification. Nous utilisons les termes « *anomalie », et « violation »* de manière interchangeable. Les anomalies de services réseau sont provoquées intentionnellement ou non par les adversaires mentionnés dans (cf. Section 3.2.1), qui exploitent la surface d'attaque de NFV décrite dans (cf. Section 3.2.2). Par exemple, un adversaire interne disposant de permissions suffisantes pourrait migrer des VNFs vers des territoires qui appliquent des réglementations laxistes en matière de protection de données des utilisateurs.

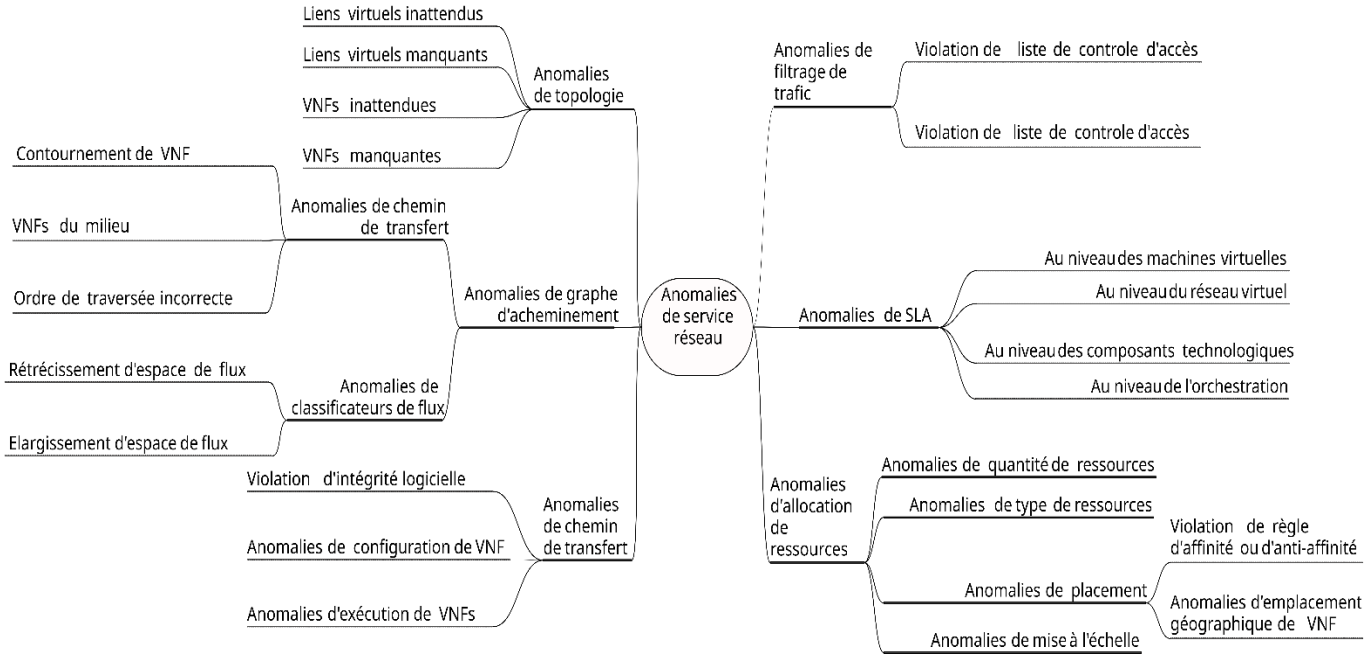


Figure 3.1 : Taxonomie des anomalies de service réseau dans NFV.

Un nœud avec une ligne pleine représente une classe ou une sous-classe d'anomalies. Un nœud avec une ligne pointillée représente un exemple d'anomalie dans une classe ou une sous-classe d'anomalies.

3.3 Taxonomie des anomalies de services réseau dans NFV

La Figure 3.1 illustre la taxonomie que nous proposons pour les anomalies des services réseau dans NFV. Ces anomalies engendrent des problèmes de sécurité et de qualité de service (QoS). Nous décrivons six catégories d'anomalies liées à six types de violation de clauses : (1) Topologie, (2) Graphe d'acheminement, (3) VNFs, (4) Filtrage du trafic, (5) SLA[4], et (6) Allocation de ressources. Simultanément, nous identifions et analysons les éventuels impacts négatifs des anomalies sur les attributs critiques de service : confidentialité, intégrité, disponibilité, performance et résilience.

3.3.1 Les anomalies de topologie

En rappel, une topologie de service réseau capte la structure du service réseau en décrivant les VNFs qui le composent et les liens virtuels qui relient ces VNFs. Deux VNFs connectées à deux liens virtuels différents ne peuvent pas échanger directement des paquets. Ainsi, une spécification de topologie peut implémenter une politique d'isolation du trafic. Comme le montre la Figure 3.2, nous identifions quatre anomalies de topologies [84], [85] qui affectent la sécurité et la qualité de service :

- *Liens virtuels inattendus* : cette anomalie se produit lorsqu'un adversaire crée un lien virtuel entre deux VNFs dont la communication est interdite par une politique d'isolation du trafic.

[4] SLA : contrat de niveau de service

L'adversaire peut exploiter le lien virtuel créé pour envoyer du trafic malveillant à une autre VNF. Par exemple, le lien virtuel inattendu peut servir à inonder une autre VNF de paquets de déni de service.

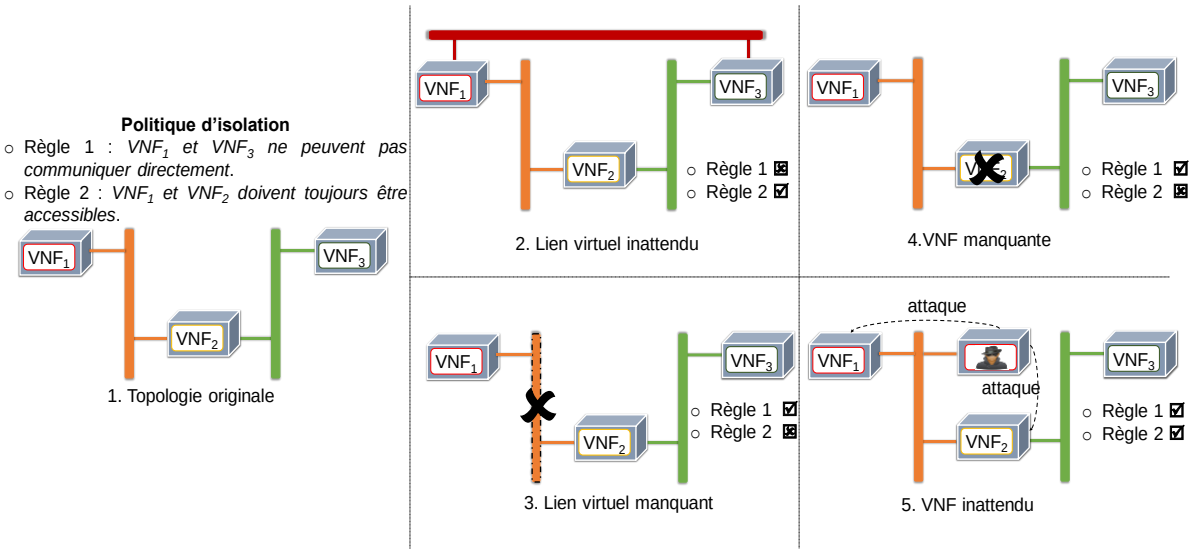


Figure 3.2 : Scénarios illustrant quatre anomalies de topologie.

Les barres larges représentent des liens virtuels. Les barres minces relient les liens virtuels aux VNFs.

- *Liens virtuels manquants* : cette anomalie correspond à une instance de service réseau à laquelle il manque un ou plusieurs liens virtuels spécifiés dans la topologie. Nous identifions deux conséquences possibles de cette anomalie. Premièrement, la connexion cesse entre les VNFs qui communiquent par les liens virtuels manquants, ce qui entraîne des interruptions de service. Deuxièmement, une attaque sophistiquée peut consister à introduire cette anomalie pour forcer certaines VNFs à utiliser un autre lien virtuel qui a été préalablement compromis. Ensuite, l'adversaire peut espionner ou altérer le trafic traversant le lien virtuel compromis.

- *VNFs manquantes* : comme l'anomalie précédente, cette anomalie se produit lorsqu'une ou plusieurs VNFs spécifiées dans la topologie sont manquantes dans l'instance de service réseau. Par conséquent, une partie du trafic échappe au traitement par les VNFs manquantes, ce qui entraîne une violation des objectifs de service, selon les fonctions des VNFs manquantes.

- *VNFs inattendues* : un adversaire peut insérer une VNF non spécifiée dans la topologie. La VNF insérée peut servir de renifleur qui extrait des données sensibles du trafic. L'adversaire peut également modifier le trafic arrivant aux VNFs insérées.

3.3.2 Les anomalies de graphe d'acheminement

Un graphe d'acheminement comprend deux parties logiques qui, ensemble, spécifient une politique d'aiguillage du trafic entre VNFs : (1) les *chemins de transfert* qui sont des listes ordonnées de VNFs qui traitent une classe de trafic spécifique, et (2) les *classificateurs de flux* qui définissent les classes de trafic avec des règles de classification, par exemple, l'adresse IP source, l'adresse IP de destination, et les numéros de port.

Nous classons ainsi les anomalies du graphe d'acheminement en deux sous-classes : (1) les *anomalies de chemins de transfert* et (2) les *anomalies des classificateurs de flux*. Contrairement aux anomalies de graphe d'acheminement, qui ont été intensivement explorées dans les contextes non NFV [86] et NFV [87], les anomalies des classificateurs de flux ont été omises dans la littérature. Cette dernière sous-classe d'anomalies découle de notre analyse plus approfondie sur les anomalies de graphe d'acheminement. Nous avons observé qu'en plus des chemins de transfert, les règles de classification peuvent également être violées, une possibilité que la communauté scientifique avaient négligée. Par exemple, un adversaire qui prend contrôle de l'Orchestrateur NFV peut modifier les règles de classification spécifiées par un locataire. De même, un administrateur cloud peut mal configurer les règles de classification.

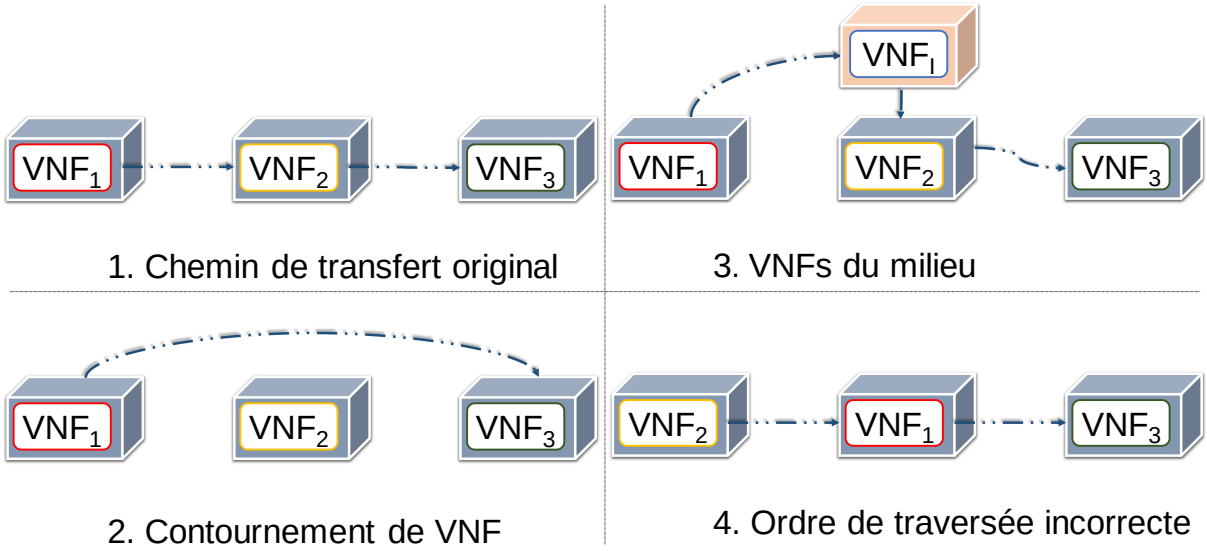


Figure 3.3 : Les anomalies de chemins de transfert

3.3.2.1 Les anomalies de chemins de transfert

Un chemin de transfert définit une liste de VNFs qu'une classe de trafic spécifique doit traverser dans un ordre particulier. Également appelés chaînes de fonctions de service, les chemins de transfert permettent un traitement différencié des classes de trafic en fonction de leurs exigences de sécurité et de qualité de service. Une anomalie de chemin de transfert se produit lorsqu'un adversaire viole l'une des

propriétés d'un chemin de transfert, à savoir l'ordre, le nombre ou le type de VNFs. La Figure 3.3 illustre trois types d'anomalies de chemins de transfert que nous avons identifiés :

- *Contournement de VNF* : nous illustrons cette anomalie par le scénario suivant. Un adversaire planifie une attaque par déni de service contre un service Web. Cependant, avant d'atteindre le service Web, chaque paquet doit traverser deux VNFs : un équilibreur de charge et un système de prévention d'intrusion (IPS). L'IPS détecte et atténue les schémas des attaques de déni de service. Pour échapper aux mécanismes de sécurité de l'IPS, l'adversaire prend le contrôle [88] du commutateur qui relie les VNFs. Ensuite, l'adversaire reconfigure le commutateur compromis pour transmettre les paquets sortants de l'équilibreur de charge au service Web au lieu de l'IPS. Une telle situation caractérise l'anomalie de contournement de VNF. Cette anomalie peut violer plusieurs attributs de service, en fonction des fonctions des VNFs contournées, par exemple, le chiffrement, l'authentification, l'autorisation et l'optimisation du trafic.

- *VNF du milieu* : un adversaire peut insérer une ou plusieurs VNFs en tête, au milieu ou en queue d'un chemin de transfert. En conséquence, l'adversaire peut écouter le trafic des utilisateurs finaux. Les VNFs insérées peuvent également modifier ou extraire des secrets à partir des flux de trafic redirigés.

- *Ordre de traversée incorrecte* : cette anomalie correspond à une permutation aléatoire de l'ordre des VNFs sans insérer ou contourner certaines VNFs. Cette permutation déforme le comportement attendu du service réseau en un comportement aléatoire. Par exemple, avant d'être chiffrés par un proxy de cryptage, les paquets doivent d'abord atteindre un système de prévention d'intrusion (IPS). Une inversion de cet ordre de traitement des paquets empêche l'IPS d'analyser la charge utile des paquets, ce qui empêche la détection de certaines intrusions.

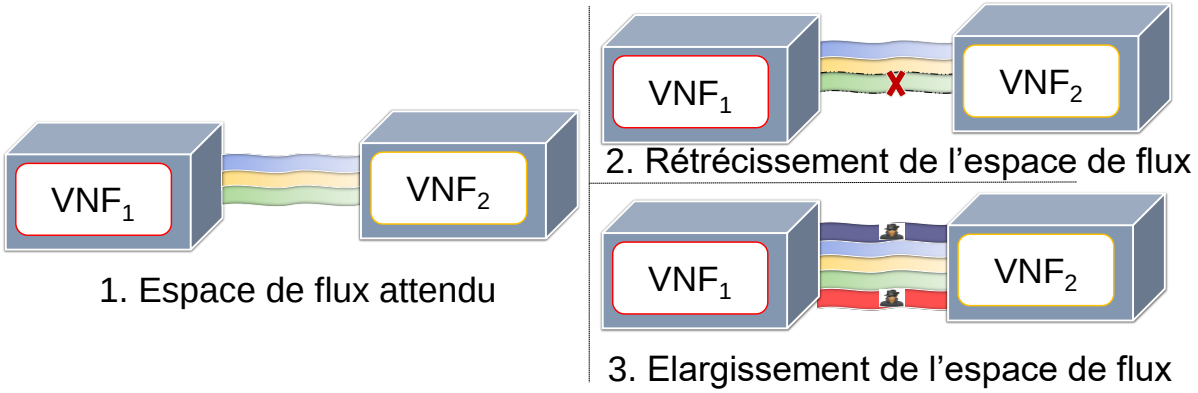


Figure 3.4 : Les anomalies de classificateur de flux

3.3.2.2 Les anomalies de classificateurs de flux

Les règles de classification (par exemple, l'adresse IP, numéro de port TCP/UDP) spécifient les classes de trafic entrant dans un chemin de transfert. L'ensemble des en-têtes de paquets correspondant à ces règles définit un espace géométrique [89], [90], comme l'ont suggéré certains chercheurs. Un rétrécissement ou un élargissement de cet espace géométrique modifie la classe de trafic autorisée dans un chemin de transfert spécifique. Ainsi, comme le montre la Figure 3.4, nous identifions deux anomalies dans de classificateur de flux :

- *Rétrécissement de l'espace de flux* : cette anomalie résulte d'un adversaire qui modifie les règles de classification pour empêcher le trafic légitime d'entrer dans un chemin de transfert. En conséquence, les VNFs jettent illégitimement certains flux de trafic, rendant le service final inaccessible aux utilisateurs.

- *Élargissement de l'espace de flux* : les règles de classification sont modifiées pour accepter des flux de trafic non autorisés dans un chemin de transfert. Par exemple, un adversaire peut introduire cette anomalie pour autoriser des flux de trafic malveillants à entrer dans un chemin de transfert.

3.3.3 Les anomalies de VNFs

La conformité des services réseau de bout en bout dépend des propriétés d'exactitude des VNFs individuelles en termes d'intégrité logicielle [91], [92], [93] de configuration et d'exécution [94]. Nous décrivons trois types d'anomalies de VNF liées à ces propriétés comme suit :

- *Violation d'intégrité logicielle* : Les VNFs sont préemballés sous forme d'images de machine virtuelle qui consistent en une pile logicielle, y compris le système d'exploitation invité, les applications préinstallées et les bibliothèques. Une violation de l'intégrité logicielle se produit lorsqu'un adversaire altère au moins un de ces éléments logiciels. Une telle violation se produit dans plusieurs scénarios. Premièrement, l'adversaire peut remplacer une version sécurisée d'image de machine virtuelle par une version plus ancienne présentant des vulnérabilités inattendues. Par exemple, la Figure 3.5 présente une analyse statique de vulnérabilités que nous avons réalisée pour montrer la différence de surface d'attaque des images Docker de Nginx[5], en fonction la version de l'image. Deuxièmement, l'adversaire peut injecter un logiciel malveillant dans une image de machine virtuelle pour en prendre le contrôle. Troisièmement, l'adversaire peut exploiter les vulnérabilités de la plate-forme NFV pour altérer une machine virtuelle

[5] Nginx reste l'un des logiciels les plus utilisé pour l'équilibrage de charge.

pendant sa migration [95] ou au repos dans la base de données des images de machine virtuelle. Les violations d'intégrité logicielle introduisent des failles de sécurité dans les VNFs, modifient leur comportement attendu ou dégradent leurs performances.

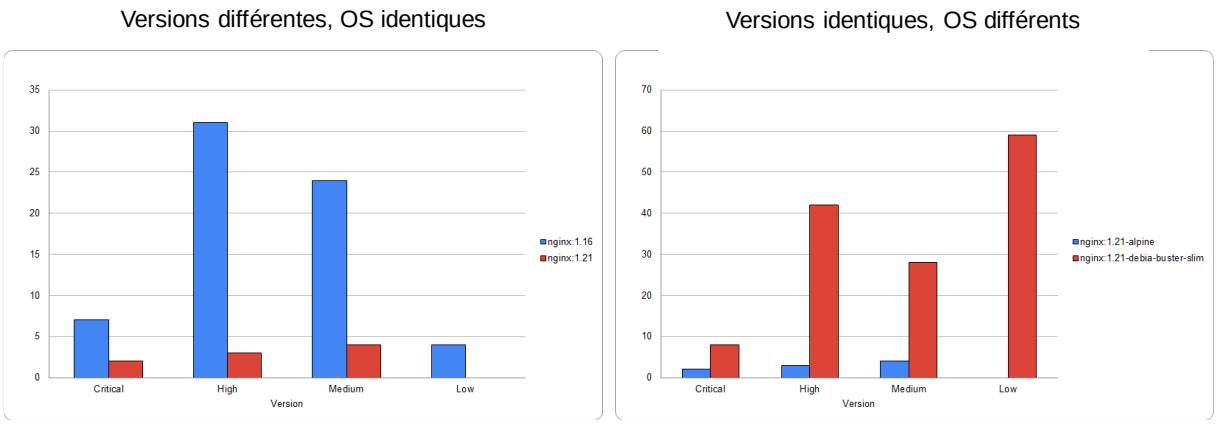


Figure 3.5 : Différence de la surface d'attaque de Nginx selon la version de l'image Docker.

À gauche, les images docker intègrent des versions différentes de Nginx (1.16 et 1.26), mais s'appuient sur le même OS. À droite, les images Docker intègrent la même version de Nginx (1.26), mais s'appuient sur des OS différents (alpine, debian-buster-slim). L'axe Y compte le nombre de vulnérabilités. L'axe X correspond aux classes de vulnérabilités CVSS (Critial, High, Medium, Low)

- *Anomalies de configuration de VNF* : les paramètres de configuration contrôlent le comportement attendu des VNFs. Par exemple, le jeu de règles *iptables* défini par un locataire détermine le trafic entrant et sortant qui traverse un pare-feu. En outre, pour configurer un équilibreur de charge avec un algorithme d'équilibrage de charge basé sur la pondération, un locataire attribue un poids aux serveurs dorsaux en fonction de leur capacité de traitement. Considérons C comme un ensemble de configurations produisant un comportement spécifique Ω d'une VNF. Un ensemble de configuration C' produira un comportement différent Ω'. L'anomalie de configuration de VNF se produit lorsqu'un locataire s'attend à ce qu'une VNF soit à la configuration C alors que la VNF est à la configuration C'. Cette anomalie peut avoir un impact sur la sécurité et les performances du service final du locataire. Les exemples précédents montrent qu'un pare-feu mal configuré peut permettre à des utilisateurs non autorisés d'accéder à des données sensibles ou de les altérer. La mauvaise configuration d'un équilibreur de charge peut entraîner une surcharge des serveurs dorsaux, augmentant ainsi la latence des requêtes. Nous soulignons qu'un tel type de violation est resté non identifié dans la littérature. Cela est probablement dû au fait que les violations de configuration peuvent facilement être confondues avec les violations d'intégrité logicielle. Cependant, ces deux violations doivent être différenciées, car, même si l'intégrité logicielle d'une VNF reste intacte, un changement de sa configuration affectera les performances et la sécurité du service réseau.

- *Anomalies d'exécution de VNF* : Chaque VNF met en œuvre une fonction réseau $F(p, C)$ qui doit renvoyer une sortie attendue *S* pour chaque paquet *P*, sous une configuration *C*. La sortie *S* représente une décision de transfert (abandon, transfert) ou la génération d'un nouveau paquet. Une anomalie d'exécution [94] de VNF correspond aux situations où la fonction *F* reçoit un paquet *p* et renvoie une sortie S'au lieu de *S*, alors que *C* reste inchangé. Cette anomalie résulte de bogues, de logiciels malveillants ou d'attaques par flux de contrôle. Selon la VNF présentant l'anomalie, la confidentialité, l'intégrité, la disponibilité ou les performances du service de haut niveau peut être impacté. Par exemple, un système de détection d'intrusion défectueux peut ne pas détecter des schémas d'attaque tels que les injections SQL ou les XSS (*cross-site scripting*).

3.3.4 Les anomalies de filtrage de trafic

Les locataires définissent la politique de filtrage du trafic à l'aide de groupes de sécurité et de listes de contrôle d'accès (ACLs) de réseau, chacun offrant une ligne de défense particulière. Alors que les groupes de sécurité s'appliquent à des VNFs individuelles, les ACLs de réseau s'appliquent à toutes les VNFs appartenant aux mêmes sous-réseaux. Les anomalies de filtrage du trafic [96] se produisent lorsqu'un adversaire autorise une VNF à envoyer ou à recevoir un trafic spécifique, alors qu'une politique de filtrage de trafic interdit cette communication. Une telle anomalie brise la ligne de défense offerte par le filtrage du trafic, permettant ainsi à des motifs de trafic malveillants (par exemple, déni de service) d'atteindre les VNFs.

3.3.5 Les anomalies de SLA

Selon la norme ISO/IEC 20000-10:2018, un accord de niveau de service (SLA) est un contrat qui oblige le fournisseur cloud à fournir des services réseau aux locataires en respectant un niveau de performance convenu. Le fournisseur cloud négocie avec chaque locataire certains indicateurs clés de qualité [97]. Chaque indicateur clé de qualité capture un aspect critique du service réseau du locataire, p. ex., le temps d'indisponibilité des VNFs, la fiabilité des VNFs, le taux de défaillance des VMs, le temps de blocage des VMs et les actions de gestion du cycle de vie de la qualité du service [98]. Les locataires surveillent chaque indicateur clé de qualité en collectant et en agrégeant un ensemble de mesures d'indicateurs clés de performance, notamment la bande passante minimale et la perte de paquets, le temps moyen entre les pannes. En outre, les fournisseurs et les locataires circonscrivent les limites que les indicateurs clés de qualité/performance doivent respecter. Les limites de chaque indicateur clé de qualité/performance définissent des seuils d'alerte inférieurs et supérieurs et des seuils d'erreur inférieurs et supérieurs. Par exemple, un locataire peut exiger une bande passante minimale de 1 Gbit/s pour tous les flux transitant entre ses VNFs. En outre, le locataire peut exiger que seule la latence d'un flux sur mille dépasse 300 ms [99]. Une violation d'accord de niveau de service [99], [100],

[101] se produit lorsqu'un indicateur clé de qualité/performance dépasse ses limites spécifiées. Conformément à la taxonomie des métriques de qualité de service proposée par l'ETSI [102], nous distinguons quatre types de violations de SLA, en fonction du niveau auquel la violation se produit :

- *Les violations de SLA au niveau de la machine virtuelle (VM)* : elles correspondent à des violations d'indicateurs clés de qualité au niveau de la VM. Un SLA peut exiger certains indicateurs clés de qualité pour évaluer la qualité de service des VMs, surtout après leur instanciation. De tels indicateurs incluent le temps de blocage de la VM, le taux de libération prématurée de la VM, la latence d'ordonnancement de la VM ou l'erreur d'horloge de la VM [102].

- *Violations des SLA au niveau du réseau virtuel* : ces violations concernent les indicateurs clés de qualité assurant la qualité de service des réseaux virtuels qui connectent les VNFs et d'autres éléments de service réseau tels que les bases de données, les systèmes de stockage et les appliances matérielles de mise en réseau. Les exemples d'indicateurs clés de qualité au niveau du réseau virtuel comprennent le taux perte de paquets, la latence et la gigue.

- *Violations des accords de niveau de service au niveau des composants technologiques* : les services réseau reposent souvent sur des composants technologiques externes (par exemple, des bases de données en tant que service, des systèmes de stockage en tant que service) proposés par des fournisseurs cloud dans le modèle de plateforme en tant que service. Des indicateurs clés de qualité tels que la fiabilité du service, la latence et les pannes sont nécessaires pour évaluer la qualité de service de ces composants technologiques. Les violations de SLA au niveau des composants technologiques se produisent lorsqu'un ou plusieurs de ces indicateurs clés de qualité dépassent les limites spécifiées.

- *Violations au niveau de l'orchestration* : elles concernent les violations d'indicateurs clés de qualité qui garantissent la qualité de l'orchestration des machines virtuelles et des réseaux virtuels. Ces indicateurs couvrent des mesures telles que la latence et la fiabilité du provisionnement des machines virtuelles et la conformité de la diversité des réseaux virtuels.

Les violations de SLA nuisent à la fois aux opérations commerciales des fournisseurs cloud et à celles des locataires. Lorsqu'un fournisseur cloud viole une clause de SLA, sa réputation s'érode vis-à-vis du locataire concerné, qui peut réclamer des compensations. Ces compensations prennent souvent la forme de crédits de service ou de remises sur les futures facturations. Toutefois, ces compensations restent insignifiants en comparaison des répercussions néfastes des violations de SLA pour le locataire, notamment la perte de clients et de revenus, une image ternie et une productivité réduite.

3.3.6 Les anomalies d'allocation de ressource

3.3.6.1 Les anomalies de quantité de ressources

Les exigences relatives à la quantité de ressources virtuelles telles que le nombre de processeurs virtuels, la taille de mémoire vive, et la bande passante du réseau varient d'une VNF à l'autre, en fonction de ses caractéristiques fonctionnelles et de son niveau de qualité de service attendu. Les défauts dans la quantité de ressource [98] se produisent lorsque la quantité de ressources allouée à une VNF ne correspond pas à sa spécification. Ce défaut peut résulter d'erreurs lors du traitement des spécifications de service réseau. Il peut également être dû à un mensonge du fournisseur cloud sur la quantité de ressources fournie à une VNF. Un mensonge proche des limites reste difficile à distinguer, mais peut néanmoins affecter les performances du service.

3.3.6.2 Les anomalies de type de ressources

L'exécution optimale de certaines VNFs nécessite des ressources présentant des caractéristiques spécifiques, telles que l'architecture du processeur, le type de stockage (par exemple, disque SSD) et un environnement d'exécution fiable. Par exemple, les VNFs qui préservent la confidentialité des données utilisateur [103] necessitent souvent de processeurs prenant en charge Intel SGX [104] pour s'exécuter. Une anomalie de type de ressource [98] se produit lorsqu'une instance de VNF s'exécute avec des ressources présentant des caractéristiques différentes de celles spécifiées.

3.3.6.3 Les anomalies de placement de VNF

Le placement d'une VNF [105], [106] consiste à déterminer un hôte (serveur) physique pour exécuter la machine virtuelle de la VNF, tout en respectant les préférences, les objectifs et les contraintes du locataire, p. ex., les règles d'affinité/d'anti-affinité, les contraintes d'emplacement géographique. En suivant cette définition, les anomalies de placement de VNF incluent les violations de règle d'affinité ou d'anti-affinité et les violations de géolocalisation de VNF :

- *Violation de règle d'affinité ou d'anti-affinité* : les locataires contrôlent le placement des VNFs sur les hôtes (serveurs) physiques à l'aide de règles d'affinité et d'anti-affinité. Une règle d'affinité s'applique à un groupe de VNFs et exige leur placement sur les mêmes hôtes. À l'inverse, une règle d'anti-affinité répartit un groupe de VNFs sur différents hôtes physiques. Les locataires définissent généralement les règles d'anti-affinité pour atteindre leurs objectifs de haute disponibilité et de résilience. La violation d'une règle d'affinité [102] se produit lorsque certaines VNFs appartenant au même groupe d'affinité sont déployées sur différents hôtes. Une telle violation ajoute par exemple de la latence dans le transfert de paquets entre les VNFs du groupe d'affinité. Inversement, une violation de la règle d'anti-affinité [98], [102] se

produit lorsque certaines VNFs du même groupe d'anti-affinité sont exécutées sur le même hôte. Une violation de règle d'anti-affinité compromet les propriétés de haute disponibilité, car la défaillance d'un hôte physique dégrade les performances de l'ensemble du groupe d'anti-affinité ou entraîne l'arrêt total de l'ensemble du groupe.

- *Les anomalies d'emplacement géographique de VNF* : la spécification des services réseau permet aux locataires de définir des contraintes d'emplacement géographique des hôtes (serveurs) physiques sur lesquels leurs VNFs s'exécutent. Les contraintes d'emplacement géographique servent : 1) à répondre aux exigences des services à faible latence en rapprochant les VNFs des utilisateurs, 2) à se conformer aux réglementations (par exemple, RGPD, APPs[6]) en hébergeant les VNFs sur des juridictions adaptées aux utilisateurs, ou 3) à assurer la tolérance aux pannes en dispersant les VNFs dans différentes zones de disponibilité. Les anomalies d'emplacement géographique de VNF [92] se produisent lorsqu'un adversaire place une VNF dans un emplacement géographique différent de celui spécifié. Une telle anomalie entraîne des problèmes de performance et de confidentialité. Par exemple, les utilisateurs finaux subissent des latences importantes lorsqu'ils s'éloignent des VNFs. En outre, lorsque les VNFs sont déplacés vers des territoires où la réglementation en matière de protection de la vie privée est laxiste, les gouvernements en place peuvent s'arroger le droit de compromettre la vie privée des utilisateurs finaux.

3.3.7 Les anomalies de mise à l'échelle

Les locataires reposent sur une politique de mise à l'échelle pour ajuster automatiquement le nombre d'instances d'une VNF (mise à l'échelle horizontale) ou la quantité de ressources allouée à une VNF (mise à l'échelle verticale). Les politiques de mise à l'échelle définissent les événements de mise à l'échelle qui déclenchent les opérations de mise à l'échelle. Un événement de mise à l'échelle correspond à un dépassement de seuil des métriques de mise à l'échelle, notamment l'utilisation moyenne du CPU et le délai de réponse moyen. Une anomalie de mise à l'échelle [98] survient lorsqu'un événement de mise à l'échelle se produit sans que l'opération de mise à l'échelle correspondante soit invoquée.

3.3.8 Résumé des anomalies de service réseau et leur impact

[6] APPs : Australian Privacy Principles ; https://www.oaic.gov.au/privacy/australian-privacy-principles/

À partir de la taxonomie précédente des anomalies de service réseau, nous résumons dans le Tableau 3.1 les impacts négatifs possibles des anomalies de service réseau sur les attributs de service, à savoir la confidentialité, l'intégrité, la disponibilité, la performance et la résilience.

Tableau 3.1 : Impacts négatifs des anomalies des services de réseau sur les objectifs de sécurité et de qualité de service.

Network services anomalies			Confidentialité	Intégrité	Disponibilité	Performance	Résilience
	Anomalies de topologie	Liens virtuels inattendus	✓	✓	✓		
		Liens virtuels manquants	✓	✓	✓		
		VNFs manquantes	✓	✓	✓	✓	
		VNFs inattendues	✓	✓	✓		
Anomalies de graphe d'acheminement	Anomalies de chemin de transfert	Contournement de VNFs	✓	✓	✓	✓	
		VNF du milieu	✓	✓	✓		
		Ordre de traversée incorrecte	✓	✓	✓	✓	
	Anomalies de classificateur de flux	Rétrécissement d'espace de flux			✓		
		Élargissement d'espace de flux	✓	✓	✓		
	Anomalies de VNF	Violation d'intégrité logicielle	✓	✓	✓	✓	
		Anomalies de configuration de VNF	✓	✓	✓	✓	
		Anomalies d'exécution de VNF	✓	✓	✓	✓	
Anomalies de filtrage de trafic		Violation de groupe de sécurité	✓	✓	✓		
		Violation de liste de contrôle d'accès réseau	✓	✓	✓		
	Anomalies de SLA	Au niveau des machines virtuelles				✓	
		Au niveau du réseau virtuel				✓	
		Au niveau des composants technologiques				✓	
		Au niveau de l'orchestration				✓	
Anomalies d'allocation de ressources		Anomalies de quantité de ressources			✓	✓	
		Anomalies de type de ressources	✓	✓		✓	
	Anomalies de placement	Violation de règle d'affinité ou d'anti-affinité	✓	✓		✓	✓
		Anomalies d'emplacement géographique de VNF	✓	✓		✓	✓
		Anomalies de mise à l'échelle			✓	✓	

3.4 Vérification des services réseau dans NFV : état de l'art et perspectives

Le concept de la vérification. Le dictionnaire Larousse définit le verbe « vérifier » comme l'action de « soumettre quelque chose à un examen, à une confrontation avec des faits, des preuves pour en contrôler l'exactitude ». Vérifier un service réseau consiste donc à confronter son état avec sa spécification, pour détecter des anomalies ou évaluer la conformité de l'état avec la spécification (cf. Figure 3.6). La vérification concerne une ou plusieurs clauses de spécification de service réseau, p. ex., topologie,

graphe d'acheminement, contraintes d'emplacement géographique. Dans cette section, nous scrutons les techniques de vérification existantes qui détectent les anomalies de service réseau.

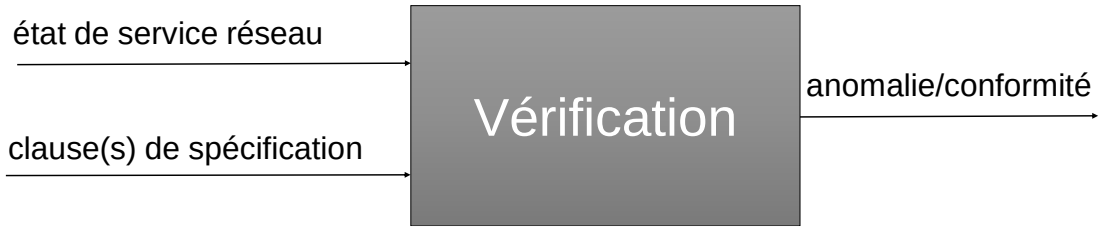


Figure 3.6 : Schéma générique de la vérification

3.4.1 Critères de sélection pour les techniques de vérification

Organisation. Nous organisons les techniques de vérification en suivant la taxonomie d'anomalies de service réseau que nous avons précédemment présentée. En d'autres termes, nous classons les techniques de vérification en fonction des clauses qu'elles vérifient.

Couverture. Lors de la revue de la littérature, nous avons détecté un manque partiel ou total de techniques de vérification pour plusieurs anomalies. Ainsi, pour certaines anomalies, nous considérons également des techniques de vérification non spécifiques à la NFV. Avec une intégration appropriée, ces techniques peuvent être appliquées dans le contexte NFV.

Qualité. Dans un premier temps, nous avons examiné les articles publiés dans des journaux, des conférences, des conférences-ateliers et des organisations de normalisation telles que l'IETF et l'ETSI. Dans un deuxième temps, par souci de concision, nous nous concentrons sur les contributions représentatives de chaque domaine de vérification en fonction de la solidité scientifique, de l'exhaustivité et de la qualité des expérimentations. Une analyse quantitative et qualitative des articles cités (voir Tableau 3.2) révèle qu'une cohorte importante de références provient de journaux et de conférences ayant des facteurs d'impact et des classements *core rank* élevés. À titre d'exemple, citons IEEE INFOCOM, USENIX NSDI et ACM SIGCOMM Computer Communication Review.

Tableau 3.2 : Carastéristiques quantitatives et qualitatives des sources des références.

	Total du nombre de citations pour ce papier	**CORE rank (source : core.edu.au)**	**h5 Index (source : Google Scholar)**	**h5 Median (source : Google Scholar)**
ACM CoNEXT	4	A	32	52
ACM SIGCOMM	6	A*	63	118
CAV	3	A*	37	52
CloudCom	2	C	21	42
ESORICS	2	A	31	51
EuCNC	2	Non référencé	25	52
ICDCS	2	A	45	77

Conférence	ICIN	3	National:France	17	30
	IEEE INFOCOM	7	A*	70	112
	IEEE NetSoft	4	B	27	40
	NSS	2	B	11	24
	USENIX NSDI	14	National:USA	66	105
	USENIX Security	2	A*	81	138
Journal	ACM SIGCOMM CCR	5	Non référencé	N/A	N/A
	IEEE/ACM Transaction on Networking	3	A*	65	94
Rapport	ETSI GS NFV	2	Non référencé	N/A	N/A

L'échantillon sélectionné comprend des sources qui compte au moins deux papiers. L'indice h5 correspond à l'indice h des articles publiés au cours des cinq dernières années révolues. Il correspond à la valeur la plus élevée de h pour que h articles publiés entre 2016 et 2020 soient cités au moins h fois chacun. La médiane h5 d'une publication correspond à la moyenne du nombre de fois où les articles composant son indice h5 sont cités.

Fenêtre de temps. La plupart des articles considérés ont été publiés entre 2014 et 2021. Les références comprennent un nombre insignifiant d'anciens articles marquants que nous considérons comme pertinents pour les champs de vérification non abordés dans NFV.

3.4.2 La vérification de chemin de transfert

Vérification hors contexte NFV. Des chercheurs ont proposé des protocoles de routage [86], [107], pour la vérification de la conformité de chemin de transfert dans les réseaux IP traditionnels. Ces approches s'appuient essentiellement sur des techniques cryptographiques pour valider la traversée correcte des paquets à travers les nœuds du chemin de transfert prédéfini. ICING [86] se distingue comme une contribution fondamentale à la vérification de chemin de transfert (VdC). ICING vérifie que chaque un paquet suit un chemin préapprouvé dans le réseau. L'approche de vérification s'appuie sur la preuve de consentement et la preuve de provenance pour vérifier le respect du chemin. ICING récupère d'abord preuve de consentement de chaque nœud sur le chemin. Les preuves de consentement sont des jetons cryptographiques créés par les serveurs de consentement. Une preuve de consentement certifie que les nœuds consentent à transporter le paquet le long du chemin. Les preuves de provenance embarquent des codes d'authentification de message et vérifient que les nœuds en amont ont traité un paquet. ICING résiste aux comportements malveillants et byzantins des serveurs de consentement, des nœuds et des fournisseurs de services Internet.

Problématiques inhérentes à NFV. Les protocoles traditionnels de vérification de chemin de transfert ne s'appliquent pas au contexte NFV. Leurs hypothèses sous-jacentes deviennent irréalistes dans un contexte NFV [87]. Premièrement, les protocoles traditionnels supposent que les chemins d'acheminement demeurent fixes et sont connus à l'avance. Dans le contexte NFV, les chemins de transfert sont dynamiques et imprévisibles. Deuxièmement, les approches traditionnelles supposaient que le réseau se compose uniquement de routeurs et de paquets, qui restent transparents aux paquets. Cependant, les boîtes intermédiaires telles que les NATs et les Proxies modifient légitimement l'en-tête

et la charge utile des paquets. De telles modifications exacerbent la difficulté de suivre les paquets le long des chemins de transfert. En outre, la distinction entre les modifications légitimes et illégitimes devient difficile [87]. Par exemple, supposons qu'un adversaire compromette un proxy HTTP et injecte un code malveillant dans les champs HTTP Body des paquets. Un protocole de vérification non sophistiqué fera confiance à tort à ces modifications, car les proxys sont connus pour modifier les paquets. Troisièmement, les protocoles traditionnels supposent que les décisions de transfert du réseau dépendent uniquement des tables de flux des nœuds réseau. Or, la NFV fait intervenir des VNFs à état dont les décisions de transfert dépendent de l'état interne des VNFs. En d'autres termes, les paquets appartenant au même flux peuvent être acheminés vers des chemins différents, en fonction de l'état interne des VNFs. Enfin, les protocoles traditionnels ne peuvent pas gérer la complexité introduite par la nature hautement dynamique des plateformes NFV, qui offrent des capacités de migration et de mise à l'échelle automatique. Par exemple, les opérations de mise à l'échelle nécessitent que les répliques de VNFs synchronisent de manière cohérente leurs états sur les flux [108], [109] afin de distribuer de manière transparente le trafic entre ces répliques.

Des travaux récents [87], [110], [111], [112], [113], [114], [115], [116], [117] ont abordé la vérification des chemins de transfert en tenant compte des nouveaux problèmes introduits par la NFV.

Seyed Kaveh Fayazbakhsh *et al*. [112] ont analysé les problèmes introduits par les actions des boîtes intermédiaires dynamiques et à état (par exemple, les modifications de paquets, les comportements opaques). Ils montrent que les actions des boîtes intermédiaires peuvent introduire des chemins de transfert imprévus, conduisant à une mauvaise détection des anomalies de chemin de transfert. Ils illustrent également comment les boîtes intermédiaires violent le concept de *liaison de l'origine* (c'est-à-dire la capacité de relier chaque paquet à son expéditeur), qui reste une exigence pour la vérification de chemin de transfert. Pour résoudre ces problèmes, les auteurs préconisent que les boîtes intermédiaires génèrent et ajoutent des étiquettes aux paquets sortants. Les boîtes intermédiaires en aval utilisent les étiquettes pour lier les paquets à leur origine, et les commutateurs dirigent les paquets à travers les boîtes intermédiaires en fonction des étiquettes. Cependant, le marquage naïf des paquets devient inefficace lorsqu'un ou plusieurs commutateurs sont compromis.

Ainsi, Kai Bu et *al.* [114] ont démontré qu'un attaquant pouvait réaliser une attaque de contournement de boite intermédiaire qui résiste techniques de vérification basées sur l'étiquetage traditionnel et les statistiques. Les auteurs suggèrent un étiquetage sophistiqué des paquets qui consiste à générer des étiquettes de façon aléatoire, ce qui les rend imprédictibles de façon probabiliste par les commutateurs malveillants. Les auteurs s'appuient sur cette dernière technique d'étiquetage pour concevoir FlowCoak, un protocole de vérification en temps réel qui détecte et empêche l'attaque par contournement de la boîte intermédiaire.

La mise en place de mécanismes vérification de chemin de transfert nécessite dans certains cas la modification de la logique interne des VNFs ou de la plateforme cloud [118]. De telles modifications entraînent un développement et un déploiement complexe et coûteux. Pour relever ce défi, Xiaoli Zhang

[111] a présenté vSFC, un schéma de vérification de chemin de transfert en temps réel qui dispense les développeurs de modifier les VNFs existantes. Leur schéma de vérification repose sur une couche de vérification séparée du traitement des VNFs et intégrée dans les machines virtuelles supportant les VNFs. Les paquets entrant ou sortant des VNFs passent par la couche de vérification, qui comprend un module d'entrée et un module de sortie. Les modules de sortie étiquettent les paquets sortant des VNFs. Ensuite, les modules d'entrée des VNFs en aval vérifient les étiquettes des paquets entrants pour détecter diverses anomalies de chemin de transfert. Comme vSFC sépare la couche de vérification de la logique des VNFs, les modules d'entrée et de sortie peuvent être déployés dans des environnements d'exécution sécurisés. Dans ce sens, cSFC [113] propose d'exécuter les couches de traitement et de vérification des VNFs dans des enclaves SGX afin de préserver la confidentialité des données et l'intégrité du logiciel des VNFs contre des adversaires puissants.

Un cadre d'audit des systèmes NFV a été présenté dans AuditBox [87], pour vérifier la conformité des chemins de transfert. AuditBox garantit en temps réel la conformité aux politiques de chemin de transfert. Il prend en compte les chemins de transfert dynamiques, les modifications en cours de chemin des paquets, les comportements basés sur l'état des VNFs, et nécessite des modifications minimales des VNFs. Le système de vérification adopte un protocole d'attestation saut par saut pour prendre en charge la vérification de chemin de transfert dynamique, et réduire la taille de paquet nécessaire. AuditBox fait confiance aux modifications en cours de chemin des paquets en exécutant l'intégralité du code de chaque VNF dans des enclaves SGX, qui offrent une protection de l'intégrité basée sur le matériel. Cette dernière propriété garantit que les modifications des paquets proviennent uniquement de VNFs non corrompues. Cependant, la technique proposée impose des frais généreux significatifs, probablement induite par les goulots d'étranglement matériels d'Intel SGX. L'exécution du code de vérification uniquement dans des enclaves [111] pourrait limiter la surcharge induite par Intel SGX.

Dans un travail publié en cours [119], l'IETF (Internet Engineering Task Force) a proposé un protocole de *preuve de transit* de paquet dans le contexte du chaînage des fonctions de service et de l'ingénierie du trafic. Le protocole met en œuvre un *schéma de partage de secret de Shamir* pour vérifier de manière cryptographique que le paquet a traversé tous les nœuds sur un chemin spécifique. Cependant, le protocole néglige des aspects essentiels de NFV, notamment l'imprévisibilité des chemins. En outre, la version actuelle de l'algorithme de vérification a révélé certaines faiblesses, comme l'incapacité à détecter les nœuds furtifs. Aguado et *al.* [116] ont déployé un système similaire dans le *réseau quantique de Madrid*. Les auteurs suggèrent de nombreuses améliorations au protocole de preuve de transit de l'IETF, par exemple le cryptage des métadonnées du protocole dans les paquets et des frais généreux du protocole. Ils utilisent un système de distribution de clés quantiques pour sécuriser les clés secrètes utilisées pour chiffrer les métadonnées du protocole de preuve de transit.

D'autres travaux [110], [115] préconisent une approche statique pour la vérification de chemin de transfert. Plus précisément, Brendan Tschaen et *al.* [110] proposent SFC-Checker, un cadre basé sur

l'analyse statique, pour vérifier le bon comportement en matière de routage, d'un réseau comprenant des chemins de transfert dynamiques. À partir d'un instantané de l'état du réseau comprenant la topologie, les tables de flux et le modèle de VNF, SFC-Checker reconstruit un graphe d'achemi dynamique à l'aide d'une machine à états finis. SFC-Checker utilise ensuite un algorithme de vérification statique pour analyser le comportement des chemins de transfert.

Fulvio Valenza et *al.* [115] présentent un modèle formel qui permet aux administrateurs de réseau de spécifier de manière flexible et extensible des politiques de transfert et un large éventail d'anomalies. Le modèle formel permet de détecter les anomalies de chemin de transfert avant leur implémentation, évitant ainsi le gaspillage des ressources nécessaires à la traduction et au déploiement de politiques de routage non conformes.

Le Tableau 3.3 présente une comparaison des travaux représentatifs sur la vérification de chemin de transfert.

Tableau 3.3 : Comparaison des travaux représentatifs sur la vérification du chemin de transfert

Références	**Modifications légitimes de paquet**	**Chemin de paquets imprédictible**	**VNFs à état**	**VNFs sans état**	**Chronologie de la vérification**	**Année**
ICING [86]					Temps réel	2011
OPT [107]				✓	Temps réel	2014
FlowTags [112]	✓	✓	✓		Temps réel	2014
vSFC [111]	✓	✓	✓		Temps réel	2017
FlowCoak [114]	✓	✓	✓	✓	Temps réel	2018
AuditBox [87]	✓	✓	✓		Temps réel	2021
SFC-Checker [110]	✓	✓	✓		Statique	2016
Fulvio Valenza *et al.* [115]		✓	N/A	N/A	Statique (phase de la spécification)	2019

3.4.3 Vérification de l'intégrité logicielle de VNF

L'intégrité logicielle des VNFs s'impose comme l'attribut de service le plus critique. Lorsque le logiciel devient corrompu, les fonctions logicielles assurant le respect des autres attributs (par exemple, la confidentialité, la disponibilité et les performances) peuvent être compromises [91]. Ainsi, la *vérification de l'intégrité logicielle de VNF* devient cruciale dans le cadre de l'externalisation des services réseau vers le cloud.

Cette sous-section passe en revue les travaux existants sur la vérification de l'intégrité logicielle. Nous identifions à cet effet deux techniques principales : la signature numérique et l'informatique de confiance. Dans le Tableau 3.4, nous fournissons une comparaison des travaux représentatifs qui exploitent ces deux techniques.

Tableau 3.4 : Comparaison des travaux représentatifs sur la vérification de l'intégrité des logiciels

Références	**Techniques**	**Technologies de virtualisation supportées**	**Racines de confiance matérielles (RdCs)**		**Support du passage à l'échelle**	**Préservation de la confidentialité : isolement des mesures (états) des locataires**	**Vérification de la Chaine de Graphe de Service**	**Intégration dans NFV**
			RdCs testées	**Support de l'hétérogénéité**				
OpenStack [93]	Signature numérique	VM+Conteneur			N/A	N/A		
Shankar Lal *et al.* [92]	Signature numérique	VM+ Conteneur			N/A	N/A		✓
TM [120]	Attestation à distance	VM+ Conteneur	TPM	✓				✓
DIVE [121]	Attestation à distance	Conteneur	TPM					
Container-IMA [122]	Attestation à distance	Conteneur	TPM		✓	✓		
CloudVaults [123]	Attestation à distance	Conteneur	TPM		✓	✓	✓	✓

3.4.3.1 La signature numérique

La signature numérique garantit l'intégrité et l'authenticité d'une donnée. Les techniques de vérification d'intégrité logicielle basée sur la signature numérique [92], [93] consistent à comparer les signatures des VNFs par rapport à des signatures de référence. Les signatures de référence sont générées à partir d'images de VNF connues pour être fiables. Des plateformes populaires de cloud telles qu'OpenStack [93] permettent aux locataires d'exploiter les signatures numériques et les certificats pour valider les images avant leur stockage ou leur téléchargement dans ou depuis la base de données d'images (Glance). Ainsi, cette validation empêche les locataires de stocker ou de télécharger des images compromises à partir de Glance. En outre, Shankar Lal et *al.* [92] reposent sur une autorité de signature et une autorité de vérification pour vérifier l'intégrité des images de VNF. L'autorité de signature génère au préalable des hachages SHA256 d'images de VNF, puis les signe avec sa clé privée. Lorsque l'autorité de vérification reçoit une requête avec en entrée le hachage de l'image d'une VNF, elle utilise le certificat racine de l'autorité de signature pour vérifier la signature de l'image. Cependant, un adversaire qui a accès [124], [125] à la base de signatures de confiance peut manipuler certains enregistrements pour les faire correspondre à des signatures d'images de VNF altérées, contournant ainsi le mécanisme de vérification d'intégrité logicielle. En outre, un adversaire qui altère les fonctions cryptographiques générant les signatures peut contourner le protocole de vérification d'intégrité.

3.4.3.2 L'attestation à distance

Concept. Pour résoudre les problèmes précédents, le Trusted Computing Group [126] préconise d'utiliser une racine de confiance matérielle pour mesurer et stocker les signatures ou les hachages des logiciels à vérifier, rendant ainsi ces mesures immuables. Les racines de confiance matérielles les plus connues incluent le Trusted Platform Module (TPM) [127] ou Intel Software Guard Extension (SGX) [104], ARM's TrustZone [128]. Ces technologies supportent un protocole d'attestation à distance [129] dans lequel un *examiné* (élément logiciel) doit convaincre un *examinateur* distant qu'il (*examiné*) reste dans un état digne de confiance. À la réception d'une demande d'attestation, la racine de confiance calcule une signature numérique de l'*examiné* qui utilise cette signature comme jeton pour s'authentifier auprès de l'*examinateur*. L'*examinateur* s'appuie sur une base de données de signatures de confiance pour évaluer le niveau de confiance du jeton envoyé par l'*examiné*. Plusieurs travaux exploitent les technologies de Trust Computing pour la vérification d'intégrité logicielle [130], [120], [122], [123].
Problèmes d'hétérogénéité des racines de confiances. Les infrastructures cloud comprennent une grande variété d'hôtes (serveurs) physiques présentant des caractéristiques différentes. Ainsi, les techniques de vérification d'intégrité logicielle basée sur l'attestation à distance doivent s'adapter à des hôtes supportant diverses racines de confiance. Pour relever ce défi, Trust Monitor (TM) [120] exploite des pilotes d'attestation sous forme de *plugins* pour mettre en œuvre des flux d'attestation à distance adaptés à diverses racines de confiance, notamment TPM, Intel SGX et AMD SEV. En outre, TM s'intègre dans la couche de gestion et d'orchestration de NFV en tant que module autonome. En d'autres termes, TM sépare les procédures de surveillance et de rapport d'attestation des flux de travail standards de la couche de gestion et d'orchestration de NFV.

Plateformes de virtualisation légère. D'autres travaux [122], [123], [121] portent sur la vérification d'intégrité logicielle dans des plateformes de virtualisation légère telles que Docker. Plus précisément, DIVE [121] exploite une version modifiée de l'architecture de mesures d'intégrité (IMA) de Linux [131] pour fournir des preuves d'intégrité à l'exécution des conteneurs, de l'hôte physique et du moteur de conteneur, p. ex., Docker Engine. Lorsque DIVE détecte la compromission d'un conteneur, il peut mettre fin à ce dernier ou le remplacer par un nouveau. Bien que les auteurs de DIVE mènent leurs travaux en dehors de NFV, les fournisseurs cloud pourraient intégrer DIVE dans la couche de gestion et d'orchestration de NFV pour vérifier l'intégrité des fonctions réseau conteneurisées.

Sécurité des états de Linux IMA. L'architecture de mesures d'intégrité de Linux (IMA) expose les états internes des conteneurs d'un locataire donné à ses colocataires. Le protocole IMA standard stocke les états de tous les conteneurs hébergés sur le même serveur physique dans un seul journal de mesures. Pendant le protocole de vérification, chaque *examinateur* récupère l'intégralité du journal de mesures. Par conséquent, les environnements cloud étant *multilocataires* par nature, un adversaire ayant accès au

journal de mesures peut voler des informations sur les états internes des autres conteneurs co-hébergés. À partir de ces informations, l'adversaire peut déduire et ensuite exploiter les vulnérabilités des conteneurs co-hébergés. C'est pourquoi des travaux ultérieurs [122], [123] ont envisagé la préservation de la confidentialité des états des conteneurs dans la vérification d'intégrité logicielle basée sur Linux IMA. Container-IMA [122] divise les journaux de mesures en plusieurs parties logiques appelées cPCRs (containers' Platform Configuration Registers). En analysant l'espace de noms des conteneurs, Container-IMA les lie aux cPCRs dans une association biunivoque. Pour garantir la protection des cPCRs, Container-IMA les sauvegarde dans une racine de confiance matérielle, tel que le TPM. En outre, contrairement à DIVE, Container-IMA peut mesurer et vérifier l'intégrité des dépendances des conteneurs (par exemple, les bibliothèques, les fichiers) et du démarrage (par exemple, les images et les configurations de démarrage).

Intégrité de bout en bout. DIVE et Container-IMA se concentrent uniquement sur la vérification de l'intégrité des conteneurs individuels. CloudVaults [123] va plus loin en considérant la vérification de bout en bout de l'intégrité de l'ensemble des conteneurs du graphe de chaînes de service, tout en préservant la confidentialité. CloudVaults marque un graphe de chaînes de service comme étant de confiance si et seulement si tous les conteneurs composant le graphe sont correctement attestés.

3.4.4 Vérification de l'application du filtrage de trafic

La vérification de l'application du filtrage de trafic pourrait être reformulée comme un problème d'accessibilité qui détermine quels paquets peuvent être échangés entre deux hôtes [72], et par extension, entre deux VNFs. Ainsi, lorsqu'une politique de filtrage interdit à deux hôtes ou VNFs de communiquer, l'algorithme de vérification du filtrage de trafic vérifie que ces derniers sont inatteignables. Il existe deux approches principales pour analyser l'accessibilité dans un réseau : l'analyse statique et l'analyse dynamique.

3.4.4.1 L'analyse statique

En général, les techniques d'analyse statique [89], [90], [96], [132], [133], [134], [135], [136], [137], [138], [139], [140], [141], [142], [143] fonctionnent sur un instantané de l'état de configuration des dispositifs de mise en réseau, notamment les commutateurs, les routeurs, les filtres de paquets (par exemple, les pare-feu) et les transformateurs de paquets (par exemple, les NATs, les proxys). L'état de configuration collecté chez chaque élément du réseau sert d'entrée pour générer un état global du réseau. L'accessibilité peut ensuite être analysée à l'aide d'une méthode formelle. Les approches d'analyse statique diffèrent selon méthode formelle utilisée pour modéliser et raisonner sur l'état du réseau, par exemple, le solveur SAT [134], le diagramme de décision binaire [139], le SMT [140], l'exécution symbolique [89], [90].

Plus précisément, Anteater [134] modélise l'état du réseau comme un ensemble de fonctions booléennes en utilisant la topologie du réseau et les tables de flux de divers dispositifs du réseau, par exemple, les pare-feu, les routeurs et les commutateurs. Les opérateurs réseau spécifient un large éventail d'invariants réseau (p. ex., isolation, absence de boucle réseau) qu'un solveur SAT peut vérifier.

HSA [89] exploite les bits d'en-tête de paquet pour modéliser les paquets comme des points dans un espace géométrique à L dimensions, où L est la longueur maximale de l'en-tête. HSA modélise le comportement de bout en bout du réseau en composant des fonctions de transfert pour capturer le comportement de divers dispositifs du réseau. Une fonction de transfert transforme des sous-espaces de l'espace à L dimensions en d'autres sous-espaces. Enfin, HSA calcule l'accessibilité et vérifie l'isolation des tranches à l'aide d'algorithmes basés sur des opérations algébriques, telles que l'intersection, l'union et la différence sur les espaces d'en-tête.

Tiros [96] et SecGuru [140], [141] représentent deux études de cas industrielles qui s'appuient sur des démonstrateurs automatiques de théorèmes pour fournir des outils de raisonnement sur l'accessibilité des réseaux aux clients AWS et Microsoft AZURE, respectivement. Tiros construit un modèle statique du réseau AWS pour vérifier les propriétés d'accessibilité. Le modèle comprend une spécification qui formalise les composants AWS (p. ex., les sous-réseaux, les passerelles NAT, les pare-feu, les équilibreurs de charge) et un instantané qui décrit la topologie et les détails du réseau. Tiros encode la spécification, l'instantané et les requêtes d'accessibilité avec le langage de divers moteurs de raisonnement tels que le solveur Datalog Soufflé [144], le solveur SMT MonoSAT [145] et le démonstrateur de théorème du premier ordre Vampire [146]. Enfin, Tiros exploite les solveurs pour répondre aux questions d'accessibilité. SecGuru valide automatiquement l'exactitude et la cohérence des politiques d'accessibilité du réseau dans le cloud Microsoft AZURE. Il encode les politiques et les différences sémantiques avec des formules logiques à vecteur binaire. SecGuru utilise Z3, un solveur SMT, comme moteur d'analyse. Avec SecGuru, les opérateurs d'AZURE peuvent réaliser des suites de tests de régression pour vérifier de manière proactive les politiques avant leur déploiement. La suite de tests de régression évite d'appliquer des politiques susceptibles d'introduire des failles de sécurité ou des problèmes de disponibilité dans le réseau.

D'autres travaux [90], [132], [147], [148] se concentrent sur la surveillance des changements de configuration du réseau pour détecter les événements de changement qui introduisent des défaillances de sécurité.

VeriFlow [90] surveille en permanence la configuration des réseaux SDN pour vérifier les invariants du réseau en temps réel. Il s'appuie sur un proxy qui intercepte et vérifie les règles de transfert envoyées par le contrôleur SDN aux équipements du réseau. VeriFlow optimise le temps de vérification avec des classes d'équivalence.

Cloud Radar [132] surveille les changements de configuration des infrastructures virtualisées pour détecter en temps quasi réel les défaillances de sécurité (y compris l'isolation du réseau) liées à la topologie. Cloud Radar représente les infrastructures virtualisées avec un modèle de graphe appelé

modèle de réalisation. Il met à jour le modèle de réalisation avec des informations récupérées à partir de l'analyse des événements de changement. D'autre part, Cloud Radar exprime les politiques de sécurité telles que l'isolation du réseau, l'isolation du stockage et le placement des machines virtuelles comme des *états d'attaque* en utilisant le modèle de graphe. Pour détecter les violations de sécurité, Cloud Radar cherche des correspondances entre les états d'attaque avec le modèle de réalisation.

3.4.4.2 L'analyse dynamique

Contrairement à l'analyse statique, les techniques d'analyse dynamique [149], [150], [151], [152], [153], [154] génèrent et injectent des paquets de sondage dans le réseau pour détecter les problèmes d'accessibilité. *ping* et *traceroute* figurent parmi les principaux outils que les administrateurs réseau utilisent pour l'analyse dynamique. L'un des principaux défis de l'analyse dynamique consiste à générer un ensemble *minimal* de paquets de sondage qui couvre entièrement les tests de vérification de la politique d'isolement.

Pour la vérification en ligne de l'accessibilité du réseau, ATPG [149] génère automatiquement un ensemble minimal de paquets de test. ATPG emploie l'approche d'analyse d'espace d'en-tête (la même que celle utilisée par HSA [89]) pour modéliser l'état du réseau collecté à partir de diverses sources telles que les tables de flux, les listes de contrôle d'accès et les fichiers de configuration. En utilisant l'analyse de l'espace d'en-tête pour trouver l'accessibilité entre un ensemble de terminaux de test, ATPG génère l'ensemble minimal de paquets requis pour tester l'application correcte de chaque règle de transfert. Périodiquement, des terminaux de test envoient des paquets de tests dans le réseau et utilisent un algorithme de localisation des erreurs pour localiser la cause des erreurs.

Monocle [151] vérifie la correspondance du plan de données avec l'état du réseau maintenu par le contrôleur SDN en présence de pannes et de bogues matériels et logiciels. Il adopte une surveillance active pour détecter les règles et les liens défaillants en quelques secondes. Monocle exploite un proxy qui intercepte toutes les modifications de règles émises vers un commutateur spécifique pour maintenir son état attendu. Monocle utilise ensuite l'état attendu pour générer des paquets de sondage afin de tester l'application correcte de chaque règle attendue sur le commutateur. Les paquets de sondage sont générés en formulant les règles comme un problème de satisfaction booléenne. En injectant les paquets de sondage dans le réseau et en observant comment le commutateur les modifie, Monocle vérifie la correspondance entre la vue du contrôleur SDN et le comportement du commutateur.

Pronton [152] adopte une approche similaire à Monocle, mais se concentre sur l'optimisation du temps de génération des paquets de sondage et du nombre de paquets de sondage générés. Pronto exploite le concept de prédicat atomique pour déterminer l'ensemble des règles testées par une sonde. Ainsi, Pronto peut générer tous les paquets de sondage en quelques secondes. En outre, contrairement aux travaux antérieurs comme ATPG et Monocle, Pronto utilise une sonde pour tester simultanément plusieurs règles, ce qui réduit le nombre de sondes générées.

3.4.5 Vérification de SLA

Vérification de SLA hors NFV. La vérification de SLA consiste à déterminer si les métriques de performance et de disponibilité du SLA sont dans les limites spécifiées [155]. Même avant l'avènement des NFV, la vérification de SLA a reçu une attention significative dans la littérature [155], [156], [157], [158], [158], [159], [160]. Deux approches fondamentales pour vérification de SLA sont le *mesurage actif* [155], [156], [157], [158] et la *modélisation passive* [159]. Le mesurage actif implique l'envoi périodique de paquets de sondage dans le réseau pour collecter des mesures sur l'état du réseau telles que le délai et le débit et la gigue. Les mesures actives génèrent un trafic réseau supplémentaire et détectent les violations de SLA seulement après leur occurrence. À l'inverse, la modélisation passive détecte les violations de SLA avant leur occurrence. La modélisation passive consiste à analyser la configuration du réseau pour générer un modèle quantitatif du réseau. Ainsi, la modélisation passive peut identifier une mauvaise configuration menaçant la réalisation des objectifs de SLA. Cependant, la modélisation passive ne parvient pas à capturer les changements dynamiques du trafic, manquant ainsi la détection certaines violations de SLA.

Vérification de SLA dans NFV. Avec l'émergence de nouveaux cas d'utilisation de la NFV, tels que l'externalisation des services de réseau et le découpage en tranches du réseau, la vérification de SLA a gagné davantage en importance. De rares papiers ont abordé la vérification SLA dans le contexte des NFV. SLA-Verifier [99] est le premier système de vérification à évaluer la conformité des mesures de performance de SLA, telles que la latence, le nombre de sauts et la disponibilité du réseau dans les environnements NFV. Les auteurs introduisent un modèle quantitatif du réseau pour effectuer une vérification statique. Cependant, la vérification statique peut manquer la détection de certaines violations de SLA en raison des changements dynamiques du trafic. Pour résoudre ce problème, SLA-Verifier effectue aussi des mesures en ligne. En exploitant des algorithmes heuristiques, ils sélectionnent le type de mesure optimal (mesure passive, mesure active) pour la vérification, en fonction des résultats de la vérification statique. Xiaoli Zhang et al. [100] ont proposé un schéma de vérification de la conformité des performances pour les boites intermédiaires à état externalisées dans un cloud non fiable, qui peut délibérément manipuler les procédures de vérification. Leur approche exploite les techniques d'échantillonnage retardé et d'engagement pour se défendre contre les comportements malhonnêtes du fournisseur cloud. Jaafar Bendriss et al. [161] ont présenté un cadre cognitif pour l'application de SLA dans les réseaux SDN/NFV. Leur cadre recueille les mesures brutes des VNFs en cours d'exécution et utilise un réseau de neurones artificiel pour prédire les futures violations de SLA. D'autres travaux [101], [162], [163], [164] fournissent des principes architecturaux pour la gestion et la vérification de SLA dans les réseaux 5 G. Plus précisément, Apostolos Papageorgiou et al. [162] ont conçu une architecture

de gestion de SLA et des flux de travail adaptée au découpage en tranches du réseau 5G. Ils ont défini des formules dynamiques pour le calcul de métriques de SLA spécifiques au découpage en tranches des réseaux 5 G.

Le Tableau 3.5 compare les contributions représentatives sur la vérification de SLA, en fonction l'approche de vérification, l'hypothèse d'un environnement non fiable et la prévisibilité. L'approche de vérification est basée soit sur du mesurage actif, soit sur la modélisation passive. L'hypothèse d'un environnement non fiable signifie que le schéma de vérification peut résister aux comportements malhonnêtes du fournisseur cloud.

Tableau 3.5 : Comparaison des travaux représentatifs sur la vérification de SLA

Références	Mesurage actif (en ligne)	Modélisation passive (hors ligne)	Hypothèse sur un environnement non digne de confiance	Prédictibilité
SLA-Verifier [99]	✓	✓		✓
Xiaoli Zhang and al. [100]	✓		✓	
Jaafar Bendriss and al. [161]	✓			✓
5GTANGO [164]	✓			✓

3.4.6 Vérification d'emplacement géographique de VNF

Les mécanismes de vérification d'emplacement géographique de VNF vérifient que les machines virtuelles allouées aux VNFs sont hébergées dans un emplacement géographique attendu. Nous avons observé que la vérification d'emplacement géographique de VNF n'a pas encore reçu une attention suffisante dans la littérature. À l'exception d'un travail [92], presque aucun travail ne considère la vérification d'emplacement géographique dans le contexte de NFV. En revanche, les chercheurs ont largement abordé la question de vérification d'emplacement géographique dans les réseaux IP traditionnels et dans les scénarios de cloud computing non-NFV. Les techniques de vérification proposées peuvent être recyclées pour le contexte NFV. Nous décrivons ces dernières techniques dans ce qui suit.

Les approches pratiques de vérification d'emplacement géographique comprennent : les protocoles de délimitation de distance [165], la géolocalisation basée sur les points de repère [166], [167], la géolocalisation basée sur la cartographie des adresses IP [168], [169], [170], [171] la géolocalisation tenant compte de la topologie réseau [172], [173] et la géolocalisation basée sur le matériel [92], [174], [175], [176].

Les protocoles de délimitation de distance [165] vérifient de manière cryptographique la limite supérieure de la distance d'un *examiné* à un *examinateur*. L'*examinateur* mesure le temps d'aller-retour entre l'envoi des bits de défi et la réception des bits correspondants chez l'*examinateur* pour estimer l'emplacement de l'examiné.

Les techniques de géolocalisation basées sur des points de repère [166], [167] supposent une corrélation ou une correspondance entre différentes mesures du réseau (par exemple, le temps de trajet aller-retour, le nombre de sauts, la largeur de bande, etc. Pour géolocaliser un serveur cible, elles considèrent un ensemble de points de repère dont la localisation géographique est connue. Un protocole de géolocalisation basé sur les points de repère envoie d'abord des paquets de sondage tels que des paquets ICMP et HTTP entre le serveur cible et les points de repère, puis mesure les paramètres réseau correspondants. Les métriques de réseau collectées servent à générer une *fonction distance-délai*. Cette dernière fonction prédit la distance entre le serveur cible et les points de repère. Les distances prédites sont ensuite fournies en entrées à des algorithmes de triangulation qui permettent la géolocalisation du serveur cible. Cependant, la précision de la *fonction distance-délai* peut être affectée [167] par des problèmes liés à la topologie du réseau, tels que la présence de circuit.

Les techniques de géolocalisation tenant compte de la topologie [172], [173] améliorent la précision des algorithmes de géolocalisation basés sur les points de repère en géolocalisant les routeurs intermédiaires entre le serveur cible et les points de repère. « *En partant des points de repère, l'algorithme de géolocalisation estime itérativement l'emplacement de tous les routeurs intermédiaires sur le chemin entre le point de repère et la cible. Cela se fait uniquement sur la base des délais de liaison à un seul saut, qui sont généralement beaucoup moins tortueux que les chemins de bout en bout à sauts multiples, ce qui permet à la géolocalisation tenant compte de la topologie d'être plus résiliente aux chemins de réseau tortueux que la géolocalisation basée sur les délais.* » [167]

Les approches de géolocalisation basées sur le matériel [92], [174], [175], [176] s'appuient sur des modules matériels inviolables attachés aux serveurs cloud pour garantir sa localisation. Les modules matériels servent de racines de confiance qui stockent des informations sur l'emplacement géographique du serveur. Par exemple, GeoProof [174] combine un protocole de preuve de stockage et un protocole de délimitation de distance pour fournir une assurance de l'emplacement géographique de données hébergées dans le cloud. L'architecture de GeoProof emploie un dispositif inviolable, doté d'un GPS et relié au réseau local du fournisseur cloud. Le dispositif attaché est utilisé pour exécuter un protocole de délimitation de distance avec les centres de données. Dans [175], des TPMs sont placés sur les machines physiques comme un identifiant unique. Un *examinateur* tiers maintient une base de données fiable sur l'emplacement des TPMs. Shankar Lal et al. [92] présentent une preuve de concept pour l'attribution de VNFs à des hôtes équipés de TPM qui répondent à des contraintes d'emplacement géographique. Ils suggèrent d'inclure ces contraintes d'emplacement géographique dans les métadonnées des images de VNFs et d'intégrer les informations géographiques des hôtes dans les TPMs qui leur sont rattachés. Les TPMs attestent de l'emplacement géographique des hôtes. Le placement des VNFs implique de les faire correspondre aux hôtes dont les informations d'emplacement géographique des TPMs correspondent aux contraintes d'emplacement géographique des VNFs.

Tableau 3.6 : Analyse des manques dans le domaine de la vérification des anomalies de services de réseau NFV.

		Anomalies de service réseau	**Application à NFV**	**Réseaux traditionnels**
Anomalies de topologie		Liens virtuels inattendus	✓	
		Liens virtuels manquants	✓	
		VNFs manquantes	✓	
		VNFs inattendues	✓	
Anomalies de graphe d'acheminement	Anomalies de chemin de transfert	Contournement de VNFs	✓	✓
		VNFs du milieu	✓	✓
		Ordre de traversée incorrecte	✓	✓
	Anomalies de classificateur de flux	Rétrécissement d'espace de flux		
		Élargissement d'espace de flux		
Anomalies de VNF		Violation d'intégrité logicielle	✓	✓
		Anomalies de configuration de VNF		
		Anomalies d'exécution de VNF	✓×	✓
Anomalies de filtrage de trafic		Violation de groupe de sécurité	✓	✓
		Violation de liste de contrôle d'accès réseau	✓	✓
Anomalies de SLA		Au niveau des machines virtuelles	✓	✓
		Au niveau du réseau virtuel	✓	✓
		Au niveau des composants technologiques	✓	✓
		Au niveau de l'orchestration	✓	✓
Anomalies d'allocation de ressources		Anomalies de quantité de ressources		
		Anomalies de type de ressources		
	Anomalies de placement	Violation de règle d'affinité ou d'anti-affinité		
		Anomalies d'emplacement géographique de VNF	✓×	✓
		Anomalies de mise à l'échelle		

3.5 Analyse des lacunes et futures directions de recherche

En tant que technologie récente, NFV reste encore dans son cycle de maturation. Son adoption à grande échelle exige de relever des défis critiques, notamment la sécurité, la confiance et la conformité [177]. Nous soutenons que la vérification de service de réseau puisse assurer l'établissement de la confiance dans les scénarios de déploiement NFV. Cependant, la littérature NFV n'a pas encore investi dans certaines activités de recherche nécessaires dans le domaine de la vérification. Dans cette section, nous identifions quelques directions de recherche pour la vérification de service de réseau dans le contexte NFV. Le Tableau 3.6 fournit un résumé de l'existence de techniques de vérification pour chaque anomalie de service réseau. Nous distinguons les techniques de vérification traditionnelles des techniques spécifiques au contexte NFV, car des efforts de recherche restent nécessaires pour intégrer les techniques traditionnelles de l'architecture NFV. Même les anomalies abordées dans le contexte NFV nécessitent une investigation plus approfondie. Nous nous concentrons sur les pistes de recherche suivantes :

- *Vérification intercouche*. La principale force de NFV réside dans son architecture multicouche, qui consiste en un découplage et une indépendance des couches de gestion et d'orchestration, d'Infrastructure NFV et de VNFs. Ces propriétés simplifient la gestion des services de réseau et

permettent l'automatisation de leur cycle de vie. Cependant, comme l'ont montré certaines études antérieures [22], [23], [24] la nature multicouche de NFV introduit des problèmes d'incohérence entre les couches. Par exemple, considérons un scénario dans lequel l'Orchestrateur NFV envoie des commandes de configuration à aux autres composants NFV pour déployer un service réseau. Supposons que ces composants n'appliquent ces commandes en raison de problèmes de synchronisation, de bogues, de compromission logicielle ou de défaillances. Dans ce cas, l'état du service réseau sera incohérent entre l'Orchestrateur NFV et ces composants. Par conséquent, la vérification de service réseau sur une seule couche devient biaisée. La vérification intercouches devient donc cruciale pour exploiter les services réseau dans les environnements NFV. Une telle approche de vérification peut nécessiter des détails pour la mise en correspondance des états des services réseau entre les couches [24]. Cependant, ce mappage est en contradiction avec la philosophie NFV qui repose sur le découplage et l'indépendance des couches. Ainsi, une question décisive pour la communauté NFV sera de savoir s'il faut enrichir l'architecture NFV avec des détails de mappage au détriment du découplage et de l'indépendance des couches ou s'il faut envisager d'autres pistes de recherche pour réaliser la vérification intercouches. Une piste d'étude permettant d'éviter les détails de mappage consiste à observer le comportement de chaque couche pour déduire [178] l'état des services réseau à cette couche.

- *Vérification du point de vue du locataire*. Lorsque les locataires se méfient d'un fournisseur cloud, ils ne peuvent pas définir leurs mécanismes de vérification en se fiant naïvement aux états des services de réseau revendiqués par le fournisseur cloud. Un fournisseur cloud malhonnête peut violer délibérément les spécifications du réseau et dissimuler les violations en mentant sur les états des services réseau. Par conséquent, la vérification des services réseau du point de vue des locataires est essentielle pour l'adoption généralisée de la NFV. Une telle perspective de vérification reste difficile, car les locataires ont une visibilité partielle sur les états des services réseau. La visibilité des locataires diffère en fonction de la politique de transparence du fournisseur de cloud et de son modèle de service. Par exemple, le modèle SaS n'expose que des informations sur les instances logicielles. En revanche, le modèle IaaS expose des informations sur les instances de VMs. Une question de recherche essentielle est donc de savoir comment déduire les états des services réseau à partir d'une observabilité externe et partielle.

- *Protocole de vérification sans état du graphe d'acheminement*. NFV offre les avantages du déploiement de services réseau élastiques, résilients et hautement disponibles. Des répliques de VNFs peuvent être lancées en cas de défaillance ou pour satisfaire aux accords de niveau de service et aux exigences de disponibilité. Les VNFs peuvent également être déplacés vers

différents points de présence pour optimiser la latence du réseau ou permettre la maintenance du matériel. Les migrations et réplications constantes de VNFs nécessitent de migrer les états entre les instances de VNFs tout en maintenant la cohérence de ces états [179], [180], [181]. Ainsi, les protocoles de vérification du graphe d'acheminement doivent prendre en compte les aspects de migration et de réplication des VNFs. La conception de protocoles de vérification sans état est une piste possible pour relever ces défis.

- *Sécurité des clés cryptographiques*. Les protocoles de vérification de chemin de transfert reposent généralement sur des clés symétriques partagées entre les VNFs. Pendant le transit dans le réseau ou au niveau du stockage, la gestion de la sécurité de ces clés reste difficile dans le cadre de NFV en raison de ses propriétés dynamiques (VNFs volatiles, migrations et réplications). Une seule clé compromise pourrait briser la sécurité de tout le protocole. Par exemple, un attaquant pourrait échapper au protocole de vérification en manipulant les procédures de vérification à l'aide de la clé compromise. Il convient donc d'envisager des protocoles exploitant le matériel et des procédures cryptographiques pour protéger le transit et le stockage des clés. Par exemple, les travaux récents [116] de 2020 ont exploité une distribution de clés quantiques - le *réseau quantique de Madrid* - pour un provisionnement hautement sécurisé des clés secrètes.

- *Vérification des classificateurs de flux.* A notre connaissance, nous avons observé que la vérification des anomalies des classificateurs de flux n'a pas été étudiée dans la littérature, malgré les problèmes de sécurité qu'ils pourraient introduire.

- *Vérification de l'exécution des VNFs*. L'un des inconvénients majeurs de l'externalisation de services réseau est la perte de contrôle direct et de visibilité sur la bonne exécution des VNFs. Par conséquent, il n'y a aucune garantie que les VNFs traitent les paquets comme prévu. Par exemple, un système de détection d'intrusion compromis pourrait classer les paquets de manière incohérente par rapport à sa base de signatures, ce qui conduirait à des intrusions non détectées. Ainsi, les efforts de recherche doivent se concentrer sur les mécanismes de vérification de l'exécution correcte des VNFs externalisés vers le cloud. Yuan et al. [94] ont posé la première pierre dans ce domaine en proposant un système qui vérifie le bon traitement de motifs de chaînes de caractères dans des environnements cloud non fiables.

- *Adaptation des métriques de SLA aux caractéristiques des VNFs*. Les métriques de SLA doivent définir précisément les attentes des locataires. Cependant, les métriques traditionnelles telles que la bande passante et la latence ne sont pas adaptées à la mesure de l'efficacité de VNFs ayant des objectifs fonctionnels différents, allant de l'optimisation du réseau aux fonctions de

sécurité. La sécurité d'un système de détection d'intrusion (IDS) peut être évaluée en fonction de paramètres tels que le taux de détection et le taux de fausses alarmes [182]. En revanche, l'évaluation d'un pare-feu peut reposer sur sa robustesse aux tests de pénétration [183]. Les locataires peuvent évaluer les performances d'un équilibreur de charge en termes de débit et de tolérance aux pannes. En résumé, les métriques de SLA doivent être adaptées aux caractéristiques de chaque VNF. Ainsi, l'application efficace des SLAs dans les NFV nécessite une taxonomie sur les métriques d'évaluation de l'efficacité des VNFs, en fonction de leurs objectifs fonctionnels.

- *Compensation automatique des SLAs.* Les fournisseurs cloud sont légalement engagés à atteindre les objectifs de SLA dans la mesure du possible. Si un fournisseur ne parvient pas à satisfaire aux exigences de service, il doit compenser financièrement les locataires par des crédits de service. Cependant, la soumission d'une demande de crédit de service et la réception de la compensation reste une procédure complexe et manuelle qui se termine probablement par des litiges entre les fournisseurs et les locataires [184]. Ainsi, il est crucial d'intégrer un mécanisme de compensation automatique de SLA dans le cadre NFV. D'une part, une telle automatisation implique que les fournisseurs et les locataires surveillent les services réseau déployés. D'autre part, chacune des deux parties doit se fier aux mesures de l'autre. Les contrats intelligents s'appuyant sur des grands registres numériques sécurisés tels que Blockchain peuvent aider à établir la confiance requise. Plusieurs travaux [184], [185] ont commencé cet investissement, mais cette piste mérite plus d'attention, notamment dans le contexte des NFV.

3.6 Conclusion

La maturation du NFV a atteint son stade critique. Son adoption massive dans l'industrie nécessite d'instaurer la confiance dans les plateformes et les fournisseurs de services NFV. Nous pensons que la vérification sera un élément crucial pour donner confiance aux entreprises dans le cadre de l'adoption de la NFV, surtout le cas d'utilisation consistant à externaliser les fonctions réseau vers le cloud.

Le présent chapitre a fourni un état des connaissances sur la vérification dans NFV, en introduisant une taxonomie des anomalies de service de réseau et en examinant les mécanismes existants pour détecter ces anomalies. Nous avons motivé l'importance de la vérification des anomalies de service de réseau en analysant leurs impacts négatifs sur les attributs de service les plus critiques, c'est-à-dire la sécurité, la performance et la résilience. En outre, nous avons examiné les lacunes et les défis dans la réalisation de la vérification dans NFV. Nous sommes convaincus que notre étude stimulera la production de travaux abondants et pertinents sur ce sujet. Par exemple, nous avons observé que plusieurs anomalies ne sont toujours pas abordées dans la littérature. En outre, de nombreux systèmes de vérification traditionnels devraient également être adaptés au contexte NFV en raison de ses

caractéristiques uniques, multicouches et dynamiques. Nous avons aussi souligné que l'architecture NFV manque de certains détails techniques pour réaliser certains types de de vérification, p.ex., la vérification intercouches.

Dans le chapitre suivant (cf. Section 4), nous introduisons VeriNeS, notre deuxième contribution. VeriNeS est un système de vérification d'anomalies de service réseau qui repose sur l'observation et l'analyse du comportement de l'Orchestrateur NFV. VeriNeS s'appuie sur un modèle de vérification basé sur les graphes qui permet de surmonter une limite fondamentale de l'architecture NFV, à savoir le manque de détails techniques pour corréler les services réseau à leurs propriétaires (locataires).

Chapitre 4

4 VeriNeS : vérification de l'orchestration des services réseau externalisés dans le cloud

Dans le chapitre précédent (cf. Section 3), nous avons présenté une taxonomie des anomalies de service réseau (cf. Section 3.3) et l'état de l'art sur les méthodes de vérification de ces anomalies (cf. Section 3.4). Les anomalies de service réseau peuvent être provoquées par des attaques contre des logiciels critiques de NFV tels que l'Orchestrateur NFV. De telles attaques émanent souvent d'adversaires internes ou d'adversaires externes.

Le présent chapitre se focalise sur la détection d'anomalies de service réseau causées par des attaques contre l'Orchestrateur NFV, qui rendent son comportement malicieux. Nous présentons VeriNeS, un système de vérification qui détecte des anomalies de service réseau à partir de l'observation du comportement de l'Orchestrateur NFV.

VeriNeS intercepte les commandes de configuration émises par l'Orchestrateur NFV et construit une représentation des services réseau à partir d'un modèle de graphe. Il détecte les anomalies de services réseau en s'appuyant sur le monomorphisme de graphe. Les résultats de nos expérimentations montrent que VeriNeS détecte un large éventail d'anomalies de services réseau avec un taux de détection élevé, dans un délai acceptable pour des scénarios réels de déploiement NFV.

4.1 Introduction

Dis-moi ce que ton Orchestrateur NFV fait et je te dirai l'état de tes services réseau. L'Orchestrateur NFV joue le rôle central dans l'architecture NFV [38], [186]. L'occurrence ou non d'anomalies de service réseau dépend du comportement de l'Orchestrateur NFV vis-à-vis des spécifications des services réseau [187]. Le comportement de l'Orchestrateur NFV se traduit par un ensemble de *commandes de configuration* qu'il envoie aux autres composantes (Gestionnaire de VNFs et Gestionnaire d'Infrastructure Virtualisée) pour créer, modifier, supprimer ou mettre à l'échelle les services réseau. Or, dans les environnements NFV, le risque de déviation du comportement de l'Orchestrateur NFV reste élevée [187] et peut émaner de diverses sources (cf. Section 3.2.1), p. ex., administrateurs cloud malicieux, logiciels malveillants, bogues, pannes, problèmes de configuration, exploitation de vulnérabilités. Il devient alors crucial pour les fournisseurs cloud et les locataires d'obtenir à travers la vérification, des garanties sur la conformité de l'orchestration des services réseau.

Défis et manques dans la littérature. Cependant, comme l'ont observé Thirunavukkarasu et *al.* [24], l'architecture NFV manque encore de détails techniques pour vérifier les services réseau dans NFV. Par

exemple, il reste impossible de corréler les commandes de configuration émises par l'Orchestrateur NFV aux services réseau, car la description standard [36], [186] de NFV ne prévoit pas d'identifiant de service réseau à l'intérieur des commandes de configuration. Dans leur contribution, les auteurs se sont limités à concevoir un modèle de déploiement de NFV à plusieurs niveaux de l'architecture NFV, et à décrire un processus général de vérification, sans proposer et implémenter un modèle de vérification.

La vérification de service réseau requiert avant tout une connaissance de l'état des services réseau. Dans ce contexte, Lin et *al*. [178] proposent d'inférer la structure et l'état internes des services de réseau en utilisant des observations externes provenant à la fois des demandes de flux des utilisateurs et des mesures de performance. Cette solution reste une étape essentielle vers la vérification de service réseau. Cependant, l'état inféré du service réseau peut être incohérent avec son état réel, ce qui conduit à une vérification biaisée.

Contributions. Dans ce chapitre, nous présentons VeriNeS, un système de vérification qui détecte des anomalies de service réseau, en observant le comportement de l'Orchestrateur NFV. VeriNeS intercepte les commandes de configuration émises par l'Orchestrateur NFV, les analyse, et reconstruit l'état des services réseau. VeriNeS utilise un modèle de graphe pour représenter l'état des services réseau. Au lieu de générer un graphe par service réseau et par locataire, VeriNeS génère un graphe global que nous appelons *super-graphe*, qui représente l'agrégation des services réseau de tous les locataires. Le concept de *super-graphe* permet à VeriNeS de surmonter une lacune importante de NFV, qui est le manque de détails techniques pour corréler les commandes de configuration aux services réseau, et les états des services réseau à leurs locataires. Pour détecter les anomalies de service réseau, VeriNeS cherche un monomorphisme entre la représentation sous forme de graphe de la spécification de service réseau et le *super-graphe*.

Nous décrivons aussi une implémentation de VeriNeS et son intégration avec des plateformes de références de NFV tels qu'OpenSource MANO et OpenStack. Une évaluation approfondie de VeriNeS montre qu'il peut détecter un large éventail d'anomalies de services réseau avec un taux de détection élevé, en quelques secondes seulement.

Organisation du chapitre. Nous structurons le reste du chapitre comme suit. Dans la Section 4.2, nous décrivons le processus général de déploiement des services réseau et soulignons le rôle critique de l'Orchestrateur NFV dans ce processus. Dans la Section 4.3, nous présentons un modèle de menace pour l'orchestration des services réseau. Dans la Section 4.4, nous formalisons le problème de déviation du comportement de l'Orchestrateur NFV et présentons le principe général de VeriNeS. Dans la Section 4.5, nous proposons un modèle de vérification à base de graphes qui permet la représentation et la vérification des services réseau. Dans la Section 4.6, nous décrivons la conception et l'implémentation techniques de VeriNeS. Dans la Section 4.7, nous décrivons l'évaluation de VeriNeS. Dans la Section 4.8, nous présentons des travaux connexes à VeriNeS. Enfin, dans la Section 4.9, nous concluons le chapitre.

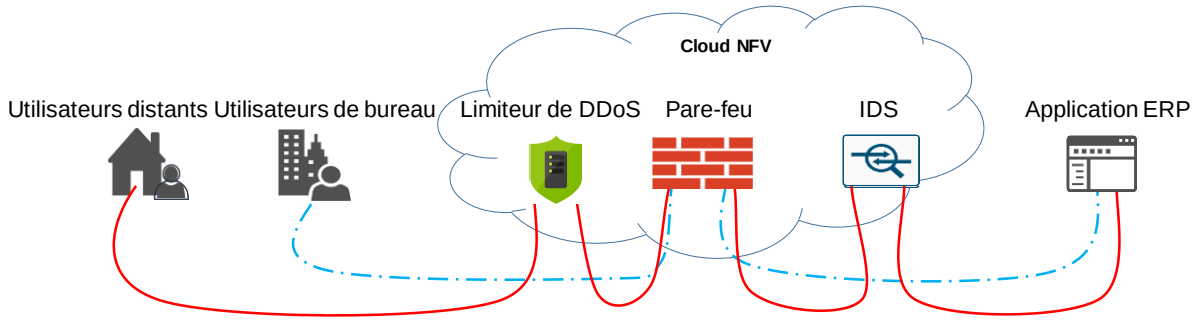


Figure 4.1 : Exemple de scénario de déploiement de service réseau d'entreprise externalisé vers le cloud. Les flux de trafic des utilisateurs du bureau et des utilisateurs distants suivent des chemins de transfert différents.

4.2 Provisionnement des services réseau

Dans cette section, nous détaillons le rôle critique de l'Orchestrateur NFV dans le déploiement des services réseau à travers un scénario concret de déploiement de service réseau dans le cloud.

4.2.1 Scénario

Scénario de motivation. Considérons le scénario de déploiement de service réseau décrit dans la Figure 4.1. L'entreprise *Innover* souhaite que son application ERP (Enterprise Resource Planning) soit accessible à la fois de l'intérieur et de l'extérieur de son intranet. Afin de protéger l'application ERP contre les intrusions (par exemple, les accès non autorisés, les attaques par déni de service), l'entreprise interpose plusieurs VNFs sur le chemin des flux de trafic entre les utilisateurs finaux et l'application ERP. Les VNFs sont déployées dans une infrastructure cloud. *Innover* peut recourir à des passerelles sophistiquées telles que APLOMB [12] pour rediriger le trafic de ses utilisateurs vers les VNFs déployées dans le cloud. Cet exemple met en évidence une politique de sécurité basée sur la localisation qui distingue deux profils de trafic : un profil qui décrit le trafic en provenance des *utilisateurs du bureau* qui accèdent à l'application ERP depuis l'intranet de l'entreprise, et un autre profil correspondant aux *utilisateurs distants* qui se trouvent en dehors de l'intranet de l'entreprise.

Exigences de l'entreprise Innover. Chacun des deux profils de trafic emprunte un chemin de transfert qui lui est spécifique. En rappel, un chemin de transfert définit une liste ordonnée de VNFs qu'un profil de trafic doit traverser. Les administrateurs réseau d'*Innover* accordent une confiance au trafic des *utilisateurs du bureau* qui passe uniquement par un pare-feu avant d'atteindre l'application ERP. En revanche, les *utilisateurs distants* sont moins fiables, et leur trafic nécessite une inspection plus minutieuse. Ainsi, le trafic des *utilisateurs distants* doit être traité par un limiteur d'attaques de déni de service distribué (DDoS), un pare-feu, puis un système de détection d'intrusion (IDS) avant d'atteindre l'application ERP.

Traduction des exigences en spécification. Comme nous l'avons largement abordée dans la Section 2.5, *Innover* doit exprimer de manière déclarative les exigences de déploiement et de comportement opérationnel du service réseau par le biais d'une spécification. La Figure 4.2 représente un extrait du fichier de spécification représentant le service réseau décrit dans le scénario précédent. La spécification comprend des descriptions de topologie et de graphe d'acheminement, qui définissent conjointement la structure du service réseau (cf. Section 2.5).

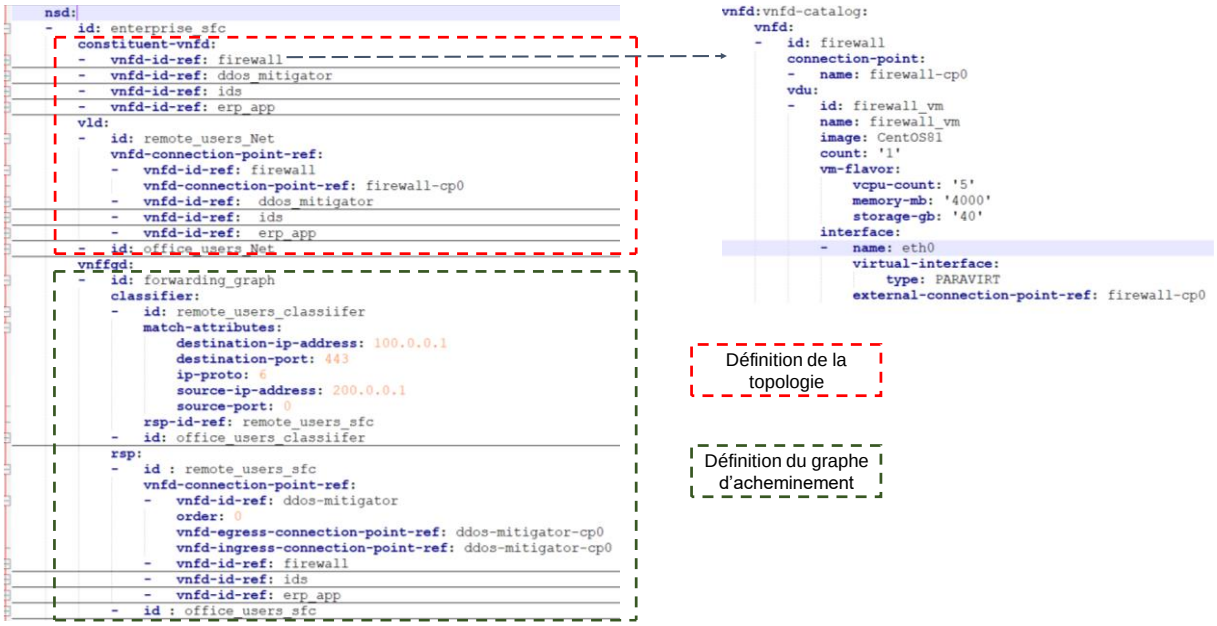


Figure 4.2: Extrait d'une spécification de service réseau suivant la norme définie par l'ETSI et mise en œuvre par OpenSource MANO.

4.2.2 Déploiement du service

Modèle de déploiement. La Figure 4.3 illustre le modèle de déploiement des services réseau, en mettant en évidence l'interaction entre les entités et les acteurs concernés. L'Orchestrateur NFV reçoit en entrée la spécification de service réseau de l'entreprise (locataire du cloud) et déploie une instance de service réseau en générant le *flux de travail* correspondant. Un flux de travail comprend un ensemble séquentiel de *commandes de configuration* envoyées par l'Orchestrateur NFV aux composants sous-jacents (Gestionnaire de VNFs, Gestionnaire d'Infrastructure Virtualisée) pour configurer les services réseau.

Commandes de configuration. Une commande de configuration représente une instruction (p. ex., création/suppression de VM, création/suppression de liens virtuels) que le Gestionnaire de VNFs ou le Gestionnaire d'Infrastructure Virtualisée doivent appliquer pour réaliser un aspect du service réseau. Le Gestionnaire de VNFs et le Gestionnaire d'Infrastructure Virtualisée exposent généralement une interface de programmation d'application (API) de type REST accessible à l'aide d'un protocole standard tel que HTTP. L'Orchestrateur NFV utilise cette API pour leur envoyer des commandes de configuration qui correspondent généralement à des messages HTTP POST ou HTTP PUT. Le

Tableau 4.1 présente un échantillon de l'API REST d'OpenStack, un Gestionnaire d'Infrastructure Virtualisée largement utilisée dans NFV.

Orchestration des services réseau. Tout d'abord, l'Orchestrateur NFV instancie la topologie du service réseau en envoyant un lot de commandes de configuration au Gestionnaire d'Infrastructure Virtualisée pour allouer les ressources virtuelles requises, par exemple, les machines virtuelles, les ports virtuels et les liens virtuels. Ensuite, l'Orchestrateur NFV configure le graphe d'acheminement en envoyant au Gestionnaire d'Infrastructure Virtualisée des commandes de configuration pour la création de classificateurs de flux et de chemins de transfert. Enfin, l'Orchestrateur NFV ordonne au Gestionnaire de VNFs d'instancier les VNFs. Le Gestionnaire de VNFs se connecte alors aux machines virtuelles associées aux VNFs et exécute les scripts de démarrage des VNFs. Les administrateurs du cloud NFV gèrent le cloud NFV avec un accès privilégié à l'Orchestrateur NFV. Ils peuvent modifier ou supprimer les services réseau déployés.

Dans la Figure 4.4, nous présentons une capture réseau des échanges de messages HTTP entre OpenSource MANO (Orchestrateur NFV) et OpenStack (Gestionnaire d'Infrastructure Virtualisée) pour le déploiement d'un service réseau.

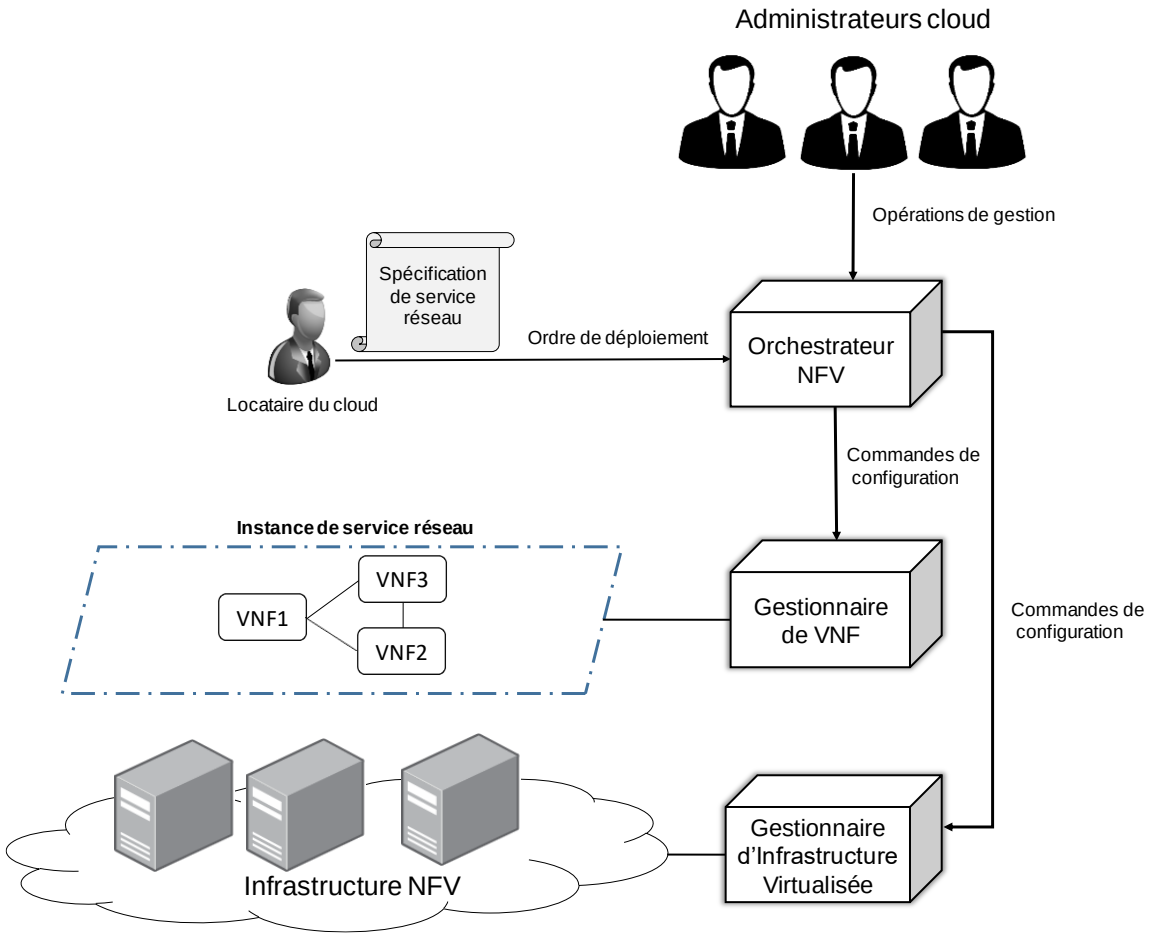


Figure 4.3 : Interaction entre les composants NFV dans le cadre de déploiement des services réseau.

4.3 Modèle de menace

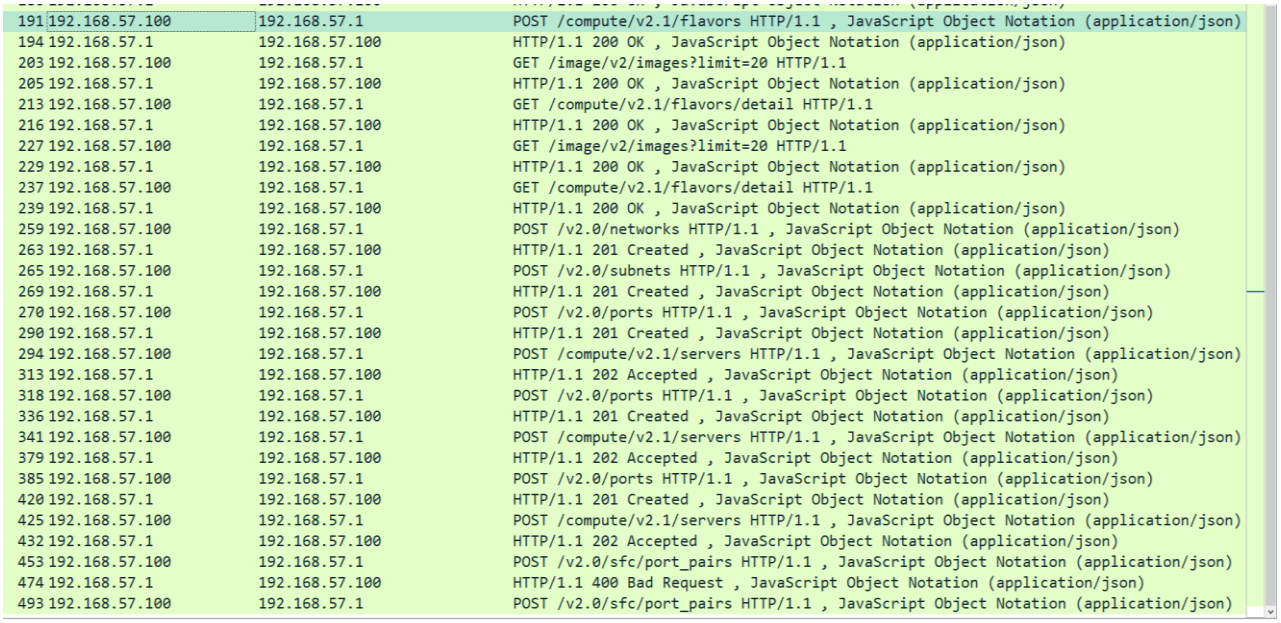
Dans notre modèle de menace, nous considérons que le fournisseur cloud en tant qu'*entité administrative* se comporte de manière honnête vis-à-vis des locataires. Le succès du fournisseur cloud dépend de la satisfaction des locataires, et donc de la conformité des services réseau par rapport à leur spécification.

Cependant, certains employés du cloud qui détiennent des accès privilégiés aux interfaces de programmation de l'Orchestrateur NFV peuvent se comporter de manière malveillante [7] [188], [189], à l'encontre des intérêts du fournisseur cloud et des locataires. Ainsi, nous supposons les administrateurs cloud comme potentiellement malveillants. Les administrateurs cloud peuvent mener des attaques intentionnelles contre les services réseau ou commettre des erreurs de configuration.

Nous considérons également l'Orchestrateur NFV comme potentiellement malveillant [187]. Il peut contenir des bogues ou être infecté par un logiciel malveillant. Aussi, l'Orchestrateur NFV peut subir des pannes logicielles et matérielles, et peut être contrôlé de manière malveillante par un adversaire qui a obtenu un accès non autorisé à son interface de programmation. Comme l'Orchestrateur NFV est potentiellement malveillant, nous supposons en outre que les informations qu'il maintient sur les services réseau (p. ex., fichiers de journalisation, états des instances de service réseau) pourraient être incohérentes avec les états réels des services réseau. Par exemple, supposons qu'un logiciel malveillant et intelligent infecte l'Orchestrateur NFV. Après avoir provoqué les anomalies de service réseau, le logiciel malveillant peut masquer les anomalies en manipulant les informations sur les états des services réseau. Par conséquent, tout mécanisme de vérification basé sur ces informations est biaisé.

Nous considérons le Gestionnaire d'Infrastructure Virtualisée et le Gestionnaire de VNFs comme des entités de confiance : ils mettent correctement en œuvre les commandes de configuration qu'ils reçoivent de l'Orchestrateur NFV. De plus, nous faisons confiance à l'Infrastructure NFV. Ces hypothèses impliquent que l'ensemble des commandes de configuration que l'Orchestrateur NFV emploie pour gérer un service réseau reflète l'état réel de ce dernier. Enfin, nous supposons que l'Orchestrateur NFV communique avec le Gestionnaire d'Infrastructure Virtualisée et le Gestionnaire de VNFs via un canal de confiance, et qu'ils restent donc exempts d'attaques de type *homme du milieu*.

[7] De manière *malveillante* : de manière à provoquer des anomalies de service réseau



191	192.168.57.100	192.168.57.1	POST /compute/v2.1/flavors HTTP/1.1 , JavaScript Object Notation (application/json)
194	192.168.57.1	192.168.57.100	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
203	192.168.57.100	192.168.57.1	GET /image/v2/images?limit=20 HTTP/1.1
205	192.168.57.1	192.168.57.100	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
213	192.168.57.100	192.168.57.1	GET /compute/v2.1/flavors/detail HTTP/1.1
216	192.168.57.1	192.168.57.100	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
227	192.168.57.100	192.168.57.1	GET /image/v2/images?limit=20 HTTP/1.1
229	192.168.57.1	192.168.57.100	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
237	192.168.57.100	192.168.57.1	GET /compute/v2.1/flavors/detail HTTP/1.1
239	192.168.57.1	192.168.57.100	HTTP/1.1 200 OK , JavaScript Object Notation (application/json)
259	192.168.57.100	192.168.57.1	POST /v2.0/networks HTTP/1.1 , JavaScript Object Notation (application/json)
263	192.168.57.1	192.168.57.100	HTTP/1.1 201 Created , JavaScript Object Notation (application/json)
265	192.168.57.100	192.168.57.1	POST /v2.0/subnets HTTP/1.1 , JavaScript Object Notation (application/json)
269	192.168.57.1	192.168.57.100	HTTP/1.1 201 Created , JavaScript Object Notation (application/json)
270	192.168.57.100	192.168.57.1	POST /v2.0/ports HTTP/1.1 , JavaScript Object Notation (application/json)
290	192.168.57.1	192.168.57.100	HTTP/1.1 201 Created , JavaScript Object Notation (application/json)
294	192.168.57.100	192.168.57.1	POST /compute/v2.1/servers HTTP/1.1 , JavaScript Object Notation (application/json)
313	192.168.57.1	192.168.57.100	HTTP/1.1 202 Accepted , JavaScript Object Notation (application/json)
318	192.168.57.100	192.168.57.1	POST /v2.0/ports HTTP/1.1 , JavaScript Object Notation (application/json)
336	192.168.57.1	192.168.57.100	HTTP/1.1 201 Created , JavaScript Object Notation (application/json)
341	192.168.57.100	192.168.57.1	POST /compute/v2.1/servers HTTP/1.1 , JavaScript Object Notation (application/json)
379	192.168.57.1	192.168.57.100	HTTP/1.1 202 Accepted , JavaScript Object Notation (application/json)
385	192.168.57.100	192.168.57.1	POST /v2.0/ports HTTP/1.1 , JavaScript Object Notation (application/json)
420	192.168.57.1	192.168.57.100	HTTP/1.1 201 Created , JavaScript Object Notation (application/json)
425	192.168.57.100	192.168.57.1	POST /compute/v2.1/servers HTTP/1.1 , JavaScript Object Notation (application/json)
432	192.168.57.1	192.168.57.100	HTTP/1.1 202 Accepted , JavaScript Object Notation (application/json)
453	192.168.57.100	192.168.57.1	POST /v2.0/sfc/port_pairs HTTP/1.1 , JavaScript Object Notation (application/json)
474	192.168.57.1	192.168.57.100	HTTP/1.1 400 Bad Request , JavaScript Object Notation (application/json)
493	192.168.57.100	192.168.57.1	POST /v2.0/sfc/port_pairs HTTP/1.1 , JavaScript Object Notation (application/json)

Figure 4.4 : Échantillon des messages HTTP échangés entre l'Orchestrateur NFV (OpenSourceMano) et le Gestionnaire d'Infrastructure Virtualisée (OpenStack) pour le déploiement d'un service réseau.

192.168.57.100 = @IP Orchestrateur NFV. 192.168.57.1 = @IP Gestionnaire d'Infrastructure Virtualisée.

Tableau 4.1 : Échantillon des commandes de configuration exposées par l'API d'OpenStack [190], [191], [192]

Nom de la commande de configuration	**Signification**	**URI de requête**	**Exemples de paramètres**	**Service OpenStack**	**Aspect du service réseau ciblé**
Create flavor	Crée un nouveau *flavor* (profil de déploiement)	`/compute/v2.1/flavors`	ID, taille de disque, taille de RAM, taille de disque swap, nombre de vCPUs, etc.	Nova	Topologie
Create server	Crée une nouvelle instance de machine virtuelle.	`/compute/v2.1/servers`	image de VM, adresse IP, ID d'un *flavor*, ID de carte réseau virtuel, groupe de sécurité, zone de disponibilité, etc.	Nova	Topologie, emplacement géographique, filtrage de trafic, allocation de ressources
Update server	Met à jour une instance de machine virtuelle	`/compute/v2.1/server/{server_id}`	ID de l'instance de machine virtuelle, nouvelle adresse IP, etc.	Nova	Topologie
Delete Server	Supprime une instance de machine virtuelle	`/compute/v2.1/server/{server_id}`	ID de l'instance de machine virtuelle	Nova	Topologie
Migrate server	Migre une instance de machine virtuelle vers un nouvel hôte	`/compute/v2.1/servers/{server_id}/action`	Hôte de destination	Nova	Emplacement géographique
Create network	Créée un nouveau lien (réseau) virtuel	`/v2.0/networks`	MTU du réseau, domaine DNS, ID de politique de QoS, etc.	Neutron	Topologie

Create port	Crée un nouvelle carte réseau virtuelle et l'attache à un lien virtuel	`/v2.0/ports`	ID du lien virtuel attaché, "port-security" activé (vrai, faux), etc.	Neutron	Topologie
Create port pairs	Crée une paire de cartes réseau virtuelles, un pour le trafic entrant et l'autre le trafic sortant	`/sfc/port_pairs`	ID port d'entrée, ID port de sortie, etc.	Neutron	Graphe d'acheminement
Create flow classifier	Crée un classificateur pour un flux donné.	`/sfc/flow_classifiers`	ID du classificateur, règles de classification (protocole IP, port source, port destination), etc.	Neutron	Graphe d'acheminement
Create port chain	Crée un chemin de transfert composé d'un ensemble ordonné de groupe de "paires de ports"	`/sfc/port_chains`	Liste des IDs de classificateurs, chemins de transfert (liste ordonnée de "paires de ports")	Neutron	Graphe d'acheminement

4.4 Modélisation de VeriNeS

4.4.1 Formalisation du problème

Soit M l'ensemble des locataires du cloud et $\omega = \text{card}(M)$ le nombre de locataires du cloud. Chaque locataire $m \in M$ demande un ou plusieurs services réseau, chacun noté $SR_{m,k}$, et possédant une spécification $Spec_{m,k}$.

Chaque spécification $Spec_{m,k}$ comprend un ensemble d'exigences $E = \{E_i\}$, exprimant chacun un aspect du service. Par exemple, une exigence E_i peut exprimer la quantité de mémoire vive requise pour une VNF, ou la définition d'un chemin de transfert.

Pour déployer un service réseau $SR_{m,k}$, l'Orchestrateur NFV génère un *flux de travail* (voir exemple dans la Figure 4.5) composé d'un ensemble séquentiel de commandes de configuration $CFG_{m,k} = \{cfg_{m,k,j}\}$. Chaque commande de configuration $cfg_{m,k,j}$ implémente une ou plusieurs exigences E_i et provoque sur le service réseau $SR_{m,k}$ un changement d'état $Transit_{m,k,j}$. La composition ordonnée de la suite de changement d'état $Transit_{m,k,j}$ représente l'état actuel du service réseau $SR_{m,k}$.

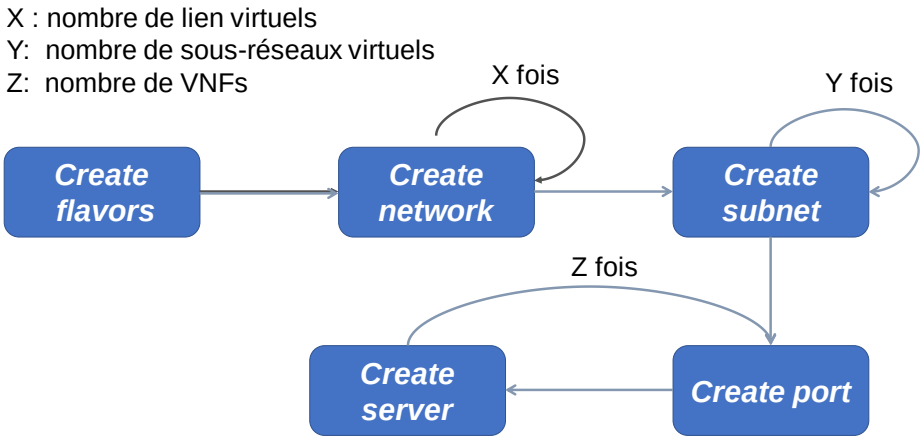


Figure 4.5 : Exemple flux de travail pour déployer une topologie de service réseau.

En raison du modèle de menace décrit dans la Section 4.3, l'Orchestrateur NFV peut générer un ensemble de commandes de configuration $CFG'_{m,k} \neq CFG_{m,k}$ non conforme à la spécification $Spec_{m,k}$. Dans le cas échéant, $CFG'_{m,k}$ contient au moins une commande de configuration *malveillante* $cfg_{m,k,malv}$ qui introduit une violation $v(cfg_{m,k,malv})$ sur une ou plusieurs exigences E_i. Une ou plusieurs violations peuvent provoquer une anomalie α.

Bien que notre modèle s'applique à n'importe quelle anomalie α décrite dans la Section 3.3, dans cette étude, nous considérons α parmi les anomalies listées dans le Tableau 4.2. Ces anomalies concernent des anomalies de topologie et de graphe d'acheminement qui affectent la structure du service réseau. Dans le reste du chapitre, nous référencerons ces anomalies avec la numérotation proposée dans le Tableau 4.2.

Tableau 4.2 : Listes des sept anomalies vérifiées par VeriNeS.

Classe		**Anomalies**	**Numéro d'anomalie**
Anomalie de topologie		Liens virtuels inattendus	❶
		VNFs inattendues	❷
Anomalies de graphe d'acheminement	Anomalies de chemin de transfert	Contournement de VNFs	❸
		VNFs du milieu	❹
		Ordre de traversée incorrecte	❺
	Anomalies de classificateur de flux	Élargissement d'espace de flux	❻
		Rétrécissement d'espace de flux	❼

Nous dirigeons le lecteur vers le chapitre 3 pour plus de détails sur la description de ses anomalies.

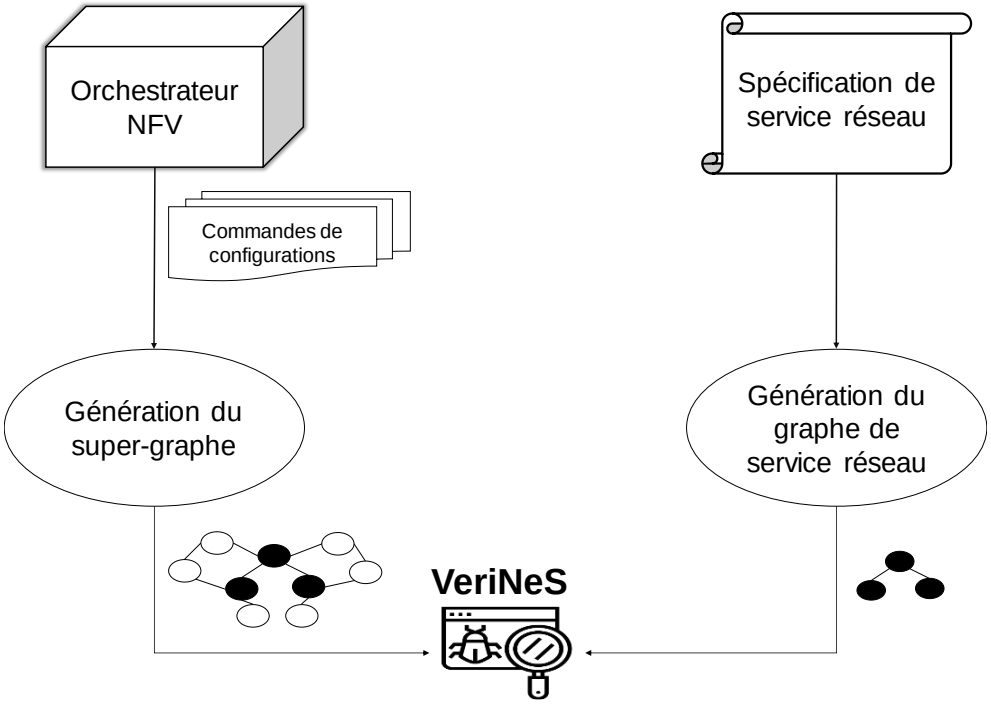


Figure 4.6 : Approche de vérification de VeriNeS.

4.4.2 Approche de VeriNeS

Principe général. Notre approche de vérification repose sur un constat simple : l'ensemble des commandes de configuration émises par l'Orchestrateur NFV reflète l'état des services réseau. En d'autres termes, un observateur (VeriNeS) qui voit une suite de commande de configuration $\mathsf{cfg}_{m,k,j}$ voit en réalité passer une suite de changement d'état $\mathrm{Transit}_{m,k,j}$ sur le service réseau $\mathsf{SR}_{m,k}$. En composant la suite de changement d'état $\mathrm{Transit}_{m,k,j}$, cet observateur peut donc maintenir l'état actuel des services réseau. Ainsi, VeriNeS se dresse comme un *proxy* transparent (passerelle) entre l'Orchestrateur NFV et les autres composants. Il intercepte les commandes de configuration émises par l'Orchestrateur NFV, et les analyse pour construire l'état des services réseau.

Défi technique. En rappel, pour déployer un service réseau $\mathsf{SR}_{m,k}$, l'Orchestrateur NFV émet un ensemble de commandes de configuration $\mathsf{CFG}_{m,k} = \{cfg_{m,k,j}\}$. La situation idéale serait que VeriNeS puisse reconstituer de façon incrémentielle un état T ($\mathsf{SR}_{m,k}$) pour chaque service réseau $\mathsf{SR}_{m,k}$ à partir de chaque commande de configuration $cfg_{m,k,j} \in \mathsf{CFG}_{m,k}$. Ainsi, VeriNeS pourrait confronter cet état T ($\mathsf{SR}_{m,k}$) à la spécification $\mathsf{Spec}_{m,k}$, afin de détecter des anomalies dans le service réseau $\mathsf{SR}_{m,k}$ du locataire m. Cependant, comme l'ont souligné Thirunavukkarasu et *al*. [24], l'architecture de NFV proposée par l'ETSI manque de détails pour corréler chaque $cfg_{m,k,j}$ à $\mathsf{SR}_{m,k}$, et chaque $\mathsf{Spec}_{m,k}$ à $\mathsf{SR}_{m,k}$.

La réponse de VeriNeS. Pour surmonter le défi mentionné ci-dessus, au lieu de considérer des états individuels T ($\mathsf{SR}_{m,k}$), VeriNeS repose sur un état global T_{global} correspondant à l'agrégation des états T

($SR_{m,k}$). Ainsi, quel que soit la commande de configuration $cfg_{m,k,j}$ analysée, VeriNeS met à jour l'état global T_{global}. Nous représentons l'état global T_{global} avec un modèle de graphe (voir les détails dans la Section 4.5) et l'appelons *super-graphe*.

Détection d'anomalies. Afin de détecter les anomalies dans la structure des services réseau, VeriNeS compare la spécification des services réseau avec le *super-graphe* pour vérifier s'il existe une relation d'inclusion entre les deux. Nous résumons notre schéma de vérification dans la Figure 4.6, VeriNeS intercepte chaque commande de configuration qui a un impact sur l'état des services réseau et génère de manière incrémentielle le *super-graphe*. Enfin, VeriNeS modélise la vérification du service réseau comme un problème de monomorphisme entre le *super-graphe* et la représentation sous forme de graphe de la spécification de service réseau.

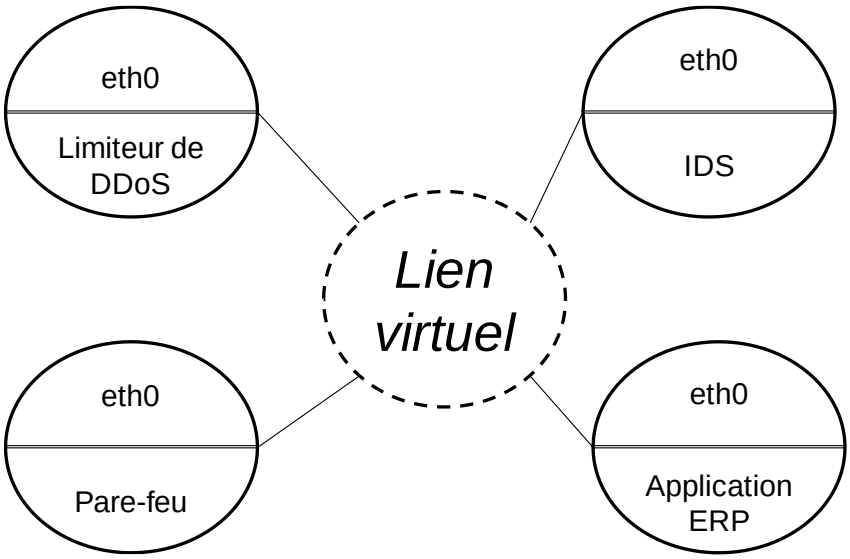


Figure 4.7 : Modèle de représentation de la topologie de service réseau.

4.5 Modèle de vérification

Dans cette section, nous décrivons un modèle à base de graphes pour la représentation et la vérification des services réseau. Nous nous focalisons sur la vérification de la structure du service réseau qui comprend la topologie et le graphe d'acheminement.

Nous modélisons la structure d'un service réseau comme un couple S = $\{TG, FG\}$, où TG et FG sont respectivement les représentations sous forme de graphes de la topologie et du graphe d'acheminement. Dans ce qui suit, nous détaillons le modèle de représentation de la topologie et celui du graphe d'acheminement.

Représentation sous forme de graphe de la topologie (TG). Nous représentons une topologie de service réseau sous la forme d'un graphe non orienté et étiqueté $TG = (P,C)$, où P est un ensemble de nœuds et

C un ensemble d'arêtes (cf. Figure 4.7). Un nœud représente soit un port virtuel (représenté par un cercle), soit un lien virtuel (représenté par un cercle en pointillés). Une arête représente la connexion entre un lien virtuel et un port virtuel appartenant à une VNF donnée. Un nœud représentant un port virtuel est décrit par le *nom du port*, et l*'image de machine virtuelle (VM) de la VNF* auquel le port virtuel est attaché. Nous notons cet ensemble d'attributs comme {*nom_de_port*, *image_de_vnf*}.

Représentation sous forme de graphe du graphe d'acheminement (FG). Un graphe d'acheminement est représenté par un multigraphe orienté et étiqueté $FG = (V, E)$, où V est un ensemble de nœuds, et E est un ensemble d'arêtes étiquetées (cf. Figure 4.8). Un nœud représente un port virtuel de VNF et est associé à un attribut unique appelé *image_de_vnf*. Une arête représente la transmission d'un flux de trafic entre deux VNFs. Une arête E_i est étiquetée avec un identifiant de flux L_i = *haché* (F), où F= *(IP_source, IP_destination, port_source, port_destination, numéro_de_protocol)* est un tuple décrivant le flux de trafic traversant l'arête, et *haché* est une fonction de hachage.
La Figure 4.7 et la Figure 4.8 illustrent respectivement la représentation sous forme de graphe de la topologie et du graphe d'acheminement du service réseau décrit dans notre exemple de scénario (cf. 4.2.1). La topologie comprend quatre ports virtuels connectés par un lien virtuel. Le graphe d'acheminement comprend deux chemins de transfert avec l'identifiant de flux des *utilisateurs du bureau* ($L1$) et des *utilisateurs distants* ($L2$).

Construction du super-graphe. Soit $T_{global} = \{STG, SFG\}$ l'état global de tous les services du réseau. STG et SFG représentent respectivement le *super-graphe* de la topologie et le *super-graphe* du graphe d'acheminement. Du point de vue théorique, STG et SFG correspondent à l'agrégation, respectivement, des TG et FG des services réseau de tous les locataires. Cependant, la construction de STG et SFG par VeriNeS reste un processus incrémental. Nous avons observé que l'agrégation des graphes équivaut à une union incrémentale de leurs nœuds et de leurs arêtes. VeriNeS met progressivement à jour STG ou SFG en fonction de la portée (topologie, graphe d'acheminement) de la commande de configuration $cfg_{m,k,j}$ en cours. De chaque commande de configuration $cfg_{m,k,j}$, VeriNeS extrait soit des nœuds et/ou des arêtes pour mettre STG ou SFG.

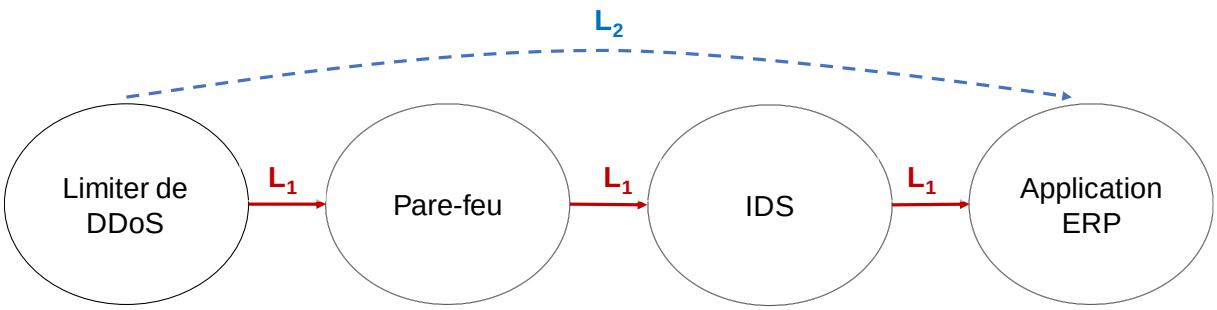


Figure 4.8 : Modèle de représentation sous forme de graphe du graphe d'acheminement

La vérification de service réseau en utilisant la comparaison de graphe. Nous modélisons la vérification de service réseau par un monomorphisme [193] de graphe. *Un motif de graphe $G2$ est un monomorphisme d'un graphe cible $G1$ si 1) les nœuds de $G2$ sont un sous-ensemble des nœuds de $G1$, et 2) les arêtes de $G2$ sont un sous-ensemble des arêtes de $G1$ relatives aux nœuds de $G2$.* Nous définissons formellement la vérification du service réseau comme suit : considérant un *super-graphe* $\mathsf{T}_{global} = \{STG, SFG\}$ et un graphe de spécification T ($\mathsf{Spec}_{m,k}$) = $\{TG, FG\}$, le service réseau contient une anomalie si T ($\mathsf{Spec}_{m,k}$) n'est pas un motif de T_{global}, c'est-à-dire, si TG n'est pas monomorphe à STG ou si FG n'est pas monomorphe à SFG.

4.6 Conception et implémentation de VeriNeS

Cette section présente une conception architecturale de VeriNeS, ainsi que son implémentation et son intégration dans une plateforme NFV réaliste.

4.6.1 Conception du système

Nous avons conçu VeriNeS comme un système distribué qui est transparent pour les composants existants dans l'architecture NFV. Comme le montre la Figure 4.9, VeriNeS représente une couche entre l'Orchestrateur NFV et le Gestionnaire d'Infrastructure Virtualisée. L'ensemble du système est composé de trois modules : un *proxy transparent*, une *base de données d'états* et un *vérificateur*.

Le *proxy* intercepte en permanence les commandes de configuration envoyées par l'Orchestrateur NFV. Le *proxy* extrait des informations ad hoc avant de transmettre chaque commande de configuration, sans aucune modification, au Gestionnaire d'Infrastructure Virtualisée. À partir des informations extraites, il met à jour le *super-graphe* de la topologie STG ou le *super-graphe* du graphe d'acheminement SFG. La *base de données d'états* stocke chaque *super-graphe* sous la forme d'un enregistrement *JSON Object*. Le v*érificateur* expose une interface de programmation d'application (API) de type REST pour permettre aux locataires de lui soumettre des requêtes de vérification.

Une requête de vérification nécessite que le locataire fournisse la spécification du service réseau à vérifier. À la réception d'une requête de vérification, le *vérificateur* génère le graphe de la spécification du service réseau, récupère le *super-graphe* dans la *base de données d'états*, et exécute l'algorithme de détection des anomalies des services réseau décrit dans la Section 4.5.

4.6.2 Implémentation de VeriNeS

Nous avons implémenté un prototype de la conception du système proposé dans un déploiement NFV réaliste, en associant OpenSourceMANO (OSM) [194] et OpenStack [195], deux logiciels NFV source ouverte bien adoptés. Nous utilisons OSM comme Orchestrateur NFV et OpenStack comme Gestionnaire d'Infrastructure Virtualisée. Pour configurer les services réseau, OSM utilise un plug-in

pour se connecter à l'API HTTP REST d'OpenStack. Grâce à un ensemble d'appels API, OSM peut configurer des machines virtuelles, des liens virtuels et des graphes d'acheminement dans OpenStack.

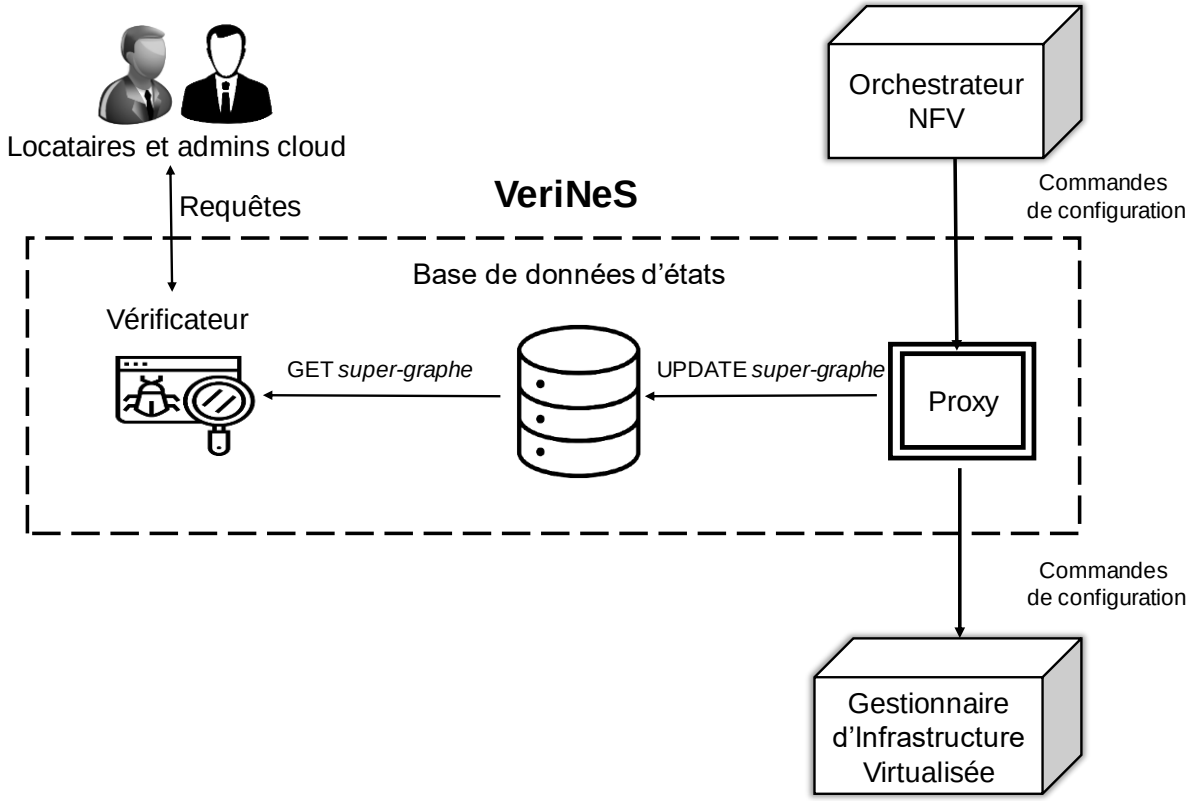


Figure 4.9 : Architecture de VeriNeS et son intégration dans l'architecture NFV

Nous avons implémenté *la base de données d'états* comme un magasin *clé-valeur* en mémoire basé sur Redis [196]. Les modules *proxy* et *vérificateur* sont tous deux implémentés en langage Python, ce qui représente un total de 2200 lignes de code. Le programme du *proxy* est un proxy HTTP *multithread* émulant l'API REST d'OpenStack. Nous avons implémenté le *vérificateur* en l'équipant d'une API de type HTTP REST à travers laquelle il reçoit les requêtes de vérification. Pour prendre en charge les demandes de vérification multiples et parallèles, nous combinons le *multiprocessing* avec une *boucle d'événements* asynchrones pour traiter les requêtes. La détection des anomalies repose sur VF2 [193], [197], un algorithme rapide de comparaison de graphes avec une faible empreinte mémoire. Cet algorithme est adapté au traitement de graphes avec attributs.

4.7 Évaluation

Nous avons basé notre évaluation sur la question de recherche suivante : *VeriNeS peut-il détecter 1) avec précision les anomalies de service réseau 2) dans un délai acceptable* ? Par conséquent, nous effectuons deux expérimentations distinctes. Premièrement, nous évaluons l'efficacité de notre modèle de vérification en mesurant le taux de détection (DR) pour chaque anomalie. Deuxièmement, nous évaluons le temps de détection des anomalies des services réseau en fonction de ω, le nombre total

de locataires du cloud. Toutes les expérimentations sont réalisées sur une machine DELL PowerEdge R910 avec 48 cœurs Intel(R) Xeon(R) CPU E7540 à 2.00 GHz, et avec 132 GB de RAM, exécutant le système d'exploitation Ubuntu 20.04 LTS 64 bits.

4.7.1 Validation du modèle de vérification

Nous évaluons la précision de notre modèle de vérification en mesurant son taux de détection. Nous lançons les demandes de vérification une fois que le *proxy* a stocké dans la *base de données d'états* les *super-graphes* de la topologie et du graphe d'acheminement. Ainsi, nous pouvons nous concentrer sur le processus de vérification plutôt que sur le processus d'orchestration, qui fonctionne indépendamment de la vérification.

Génération des anomalies. Nous générons d'abord un corpus de ω services réseau, un pour chacun des locataires. Un service réseau est généré en créant entre 3 et 8 VNFs à partir d'un catalogue existant de 150 images de VNFs, et en créant entre 1 et 5 liens virtuels, et entre 1 et 5 chemins de transfert. Pour chaque anomalie α (voir Tableau 4.2), nous appliquons une règle de mutation qui introduit l'anomalie α sur une fraction P du corpus. La valeur de P indique le degré de malveillance de l'Orchestrateur NFV. Nous considérons des valeurs moyennes de P allant de 10 % à 30 % de mutants.

Tableau 4.3 : Taux de détection des anomalies de service réseau avec ω =1000

		Pourcentage (%) des mutants insérés dans le super-graphe		
		10 %	20 %	30 %
❶	Liens virtuels inattendus	0 %	0 %	0 %
❷	VNFs inattendues	0 %	0 %	0 %
❸	Contournement de VNFs	100 %	100 %	100 %
❹	VNF du milieu	73 %	73 %	80 %
❺	Ordre de traversée incorrecte	100 %	100 %	100 %
❻	Élargissement d'espace de flux	100 %	100 %	100 %
❼	Rétrécissement d'espace de flux	100 %	100 %	100 %

Taux de détection des anomalies des services du réseau. Pour évaluer le taux de détection de chaque anomalie α en fonction de la valeur de P, nous sélectionnons des échantillons de service réseau, chacun comptant ω/10 mutants. Pour nous concentrer sur l'étape de vérification, nous générons les *super-graphes* de la topologie et du graphe d'acheminement et les téléchargeons dans la *base de données d'états*. Pour chaque échantillon, nous effectuons une vérification et comptons les faux négatifs (FN) et les vrais positifs (TP). Nous utilisons la formule suivante pour calculer le taux de détection :

$$DR = \frac{TP}{TP+FN}$$

Le Tableau 4.3 présente le taux de détection de chaque anomalie α avec différents pourcentages (P) de mutants dans le *super-graphe*. Les résultats montrent que VeriNeS peut détecter toutes les anomalies des graphes d'acheminement de ❸à ❼avec un taux de détection élevé (73 % - 100 %). L'importance sémantique de l'ordre des nœuds et des étiquettes des arêtes dans les graphes d'acheminement justifie cette observation. L'algorithme de monomorphisme détecte tout changement dans ces attributs. Les résultats montrent également que VeriNeS n'est pas adapté à la détection des anomalies de topologie (de ❶à ❷), puisque nous observons un taux de détection de 0 % pour ces anomalies. La raison en est que les anomalies de topologie introduisent des nœuds supplémentaires dans les graphes de topologie. Ainsi, la spécification de la topologie reste toujours monomorphe au *super-graphe*, qui inclut la topologie originale et les nœuds ajoutés. Cela conduit donc à une mauvaise détection de ces anomalies.

4.7.2 Temps de vérification

Du côté des locataires, nous évaluons le temps de vérification pour une mutation donnée α en fonction de ω, le nombre total de locataires. Nous considérons une valeur fixe de $P = 10$, ce qui signifie que 10 % des services réseau représentent des mutants. Nous faisons varier ω de 1000 à 5000 pour imiter un cloud de petite taille. Nous sélectionnons un échantillon de 10 % de ω. Nous effectuons une demande de vérification pour chaque service réseau de l'échantillon, et nous mesurons T, le temps écoulé entre l'envoi de la requête par le locataire et la réception de la réponse de la part de VeriNeS. Nous présentons la distribution de T pour chaque scénario dans la Figure 4.10. Les résultats montrent que la médiane du temps de vérification ne dépasse pas 6 secondes dans tous les scénarios, ce qui est un temps d'exécution raisonnable pour une telle opération. Nous observons également que le type d'anomalie n'a pas d'impact sur le temps de vérification. De plus, nous observons que le temps de vérification croît linéairement avec ω. Lorsque ω croît, le nombre de nœuds et d'arêtes du *super-graphe* augmente, ce qui augmente le temps de vérification des monomorphismes.

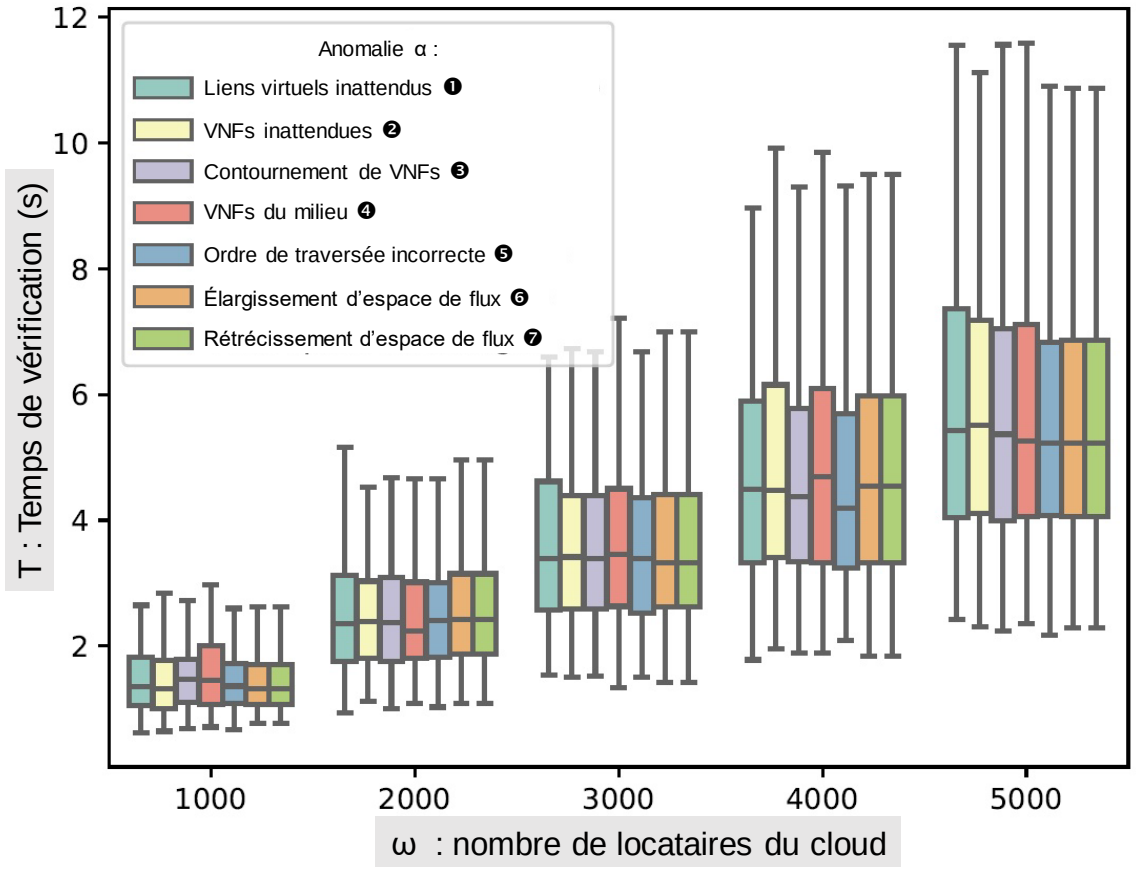


Figure 4.10 : Temps de vérification

4.8 Travaux connexes

Plusieurs travaux de recherche ont étudié le sujet de la vérification des réseaux en dehors du contexte de NFV. Les problématiques de vérification sont abordées à différentes couches, par exemple, le plan de données du réseau, le plan de contrôle du réseau, la couche d'infrastructure virtualisée. Cependant, le développement récent de NFV soulève de nouvelles questions. Nous divisons les travaux principalement en deux catégories : la vérification hors contexte NFV et la vérification dans le contexte NFV.

4.8.1 Vérification hors contexte NFV

Plusieurs chercheurs ont envisagé la vérification d'invariants à l'échelle du réseau (par exemple, isolation du trafic, absence de boucle réseau, la cohérence) ou l'application de politiques de haut niveau. Ces travaux adoptent actuellement deux approches de base : l'analyse du plan de données et l'analyse du plan de contrôle. Les algorithmes d'analyse du plan de données utilisent les informations des tables de transfert des dispositifs du réseau comme entrée. En revanche, les algorithmes d'analyse du plan de contrôle raisonnent directement sur la configuration du réseau.

Analyse du plan de données. Les approches d'analyse du plan de données [89], [90], [134], [198] permettent la détection de dispositifs réseau malveillants ou mal configurés. Header Space Analysis (HSA) [89] modélise l'état du plan de données avec un espace géométrique. HSA effectue une vérification hors ligne des invariants du réseau. En revanche, VeriFlow [90] adopte une vérification en temps réel des invariants du réseau et exploite les classes d'équivalence pour optimiser le temps de vérification. VeriFlow et VeriNeS présentent une approche similaire. VeriFlow est un proxy qui intercepte et vérifie chaque règle de transfert envoyée par le contrôleur réseau[8] au plan de données. Cependant, la conception de VeriFlow reste uniquement adaptée à la vérification des réseaux basés sur SDN. D'autres travaux [199], [112], [200], [201] se sont concentrés sur la vérification de l'application des politiques de haut niveau (par exemple, la vérification et la validation de chemin de transfert, l'authenticité de la source) dans le plan des données. Xu et *al*. [199] ont conçu un schéma de vérification intercouches pour les réseaux OpenStack basés sur SDN. Les politiques de haut niveau définies dans le contrôleur SDN sont comparées à l'état du réseau des hôtes finaux pour détecter les incohérences. Leur schéma de vérification est similaire à celui de VeriNeS, mais n'est pas adapté à la vérification des services réseau basés sur NFV. En particulier, la vérification de la conformité des chemins de transfert a été étudiée en profondeur dans de nombreux travaux [107], [201], [200], [201]. Ils ont exploité des techniques cryptographiques telles que le code d'authentification de message (MAC) pour vérifier la cohérence entre les politiques de transfert de haut niveau et le comportement en termes de transfert réseau des dispositifs réseau. Comme VeriNeS, ils peuvent détecter des anomalies de chemin de transfert telles que la déviation de chemin, ou le contournement de routeur. En complément de ces travaux, VeriNeS vérifie les politiques de transfert réseau attendues au niveau de l'orchestration des services réseau. FlowTAG [112] utilise une architecture de marquage des flux pour garantir l'application de la politique du réseau, même en présence de boîtes intermédiaires qui reclassent les flux de trafic. ECTSENP [200] propose un cadre basé sur une méthode formelle pour tester simultanément un ensemble de politiques de réseau. Ce cadre peut vérifier l'application de politiques complexes dans de grands réseaux en un temps acceptable. MindGap [201] propose une architecture de surveillance qui permet aux opérateurs réseau de vérifier la cohérence entre le plan de contrôle et le plan de données de leur réseau.

Analyse du plan de contrôle. Dans l'analyse du plan de contrôle [198], [143], [202], la configuration du réseau est vérifiée avant son application par le plan de données. En particulier, Minesweeper [198] traduit les fichiers de configuration du réseau en formule logique SMT pour vérifier l'état du plan de contrôle par rapport à des propriétés attendues. Batfish [143] adopte une vérification basée sur la

[8] Le controleur réseau peut être mappé avec l'Orchestrateur NFV pour établir la comparaison entre VeriFlow et VeriNeS.

simulation en générant un état du plan de données à partir des fichiers de configuration du réseau en utilisant le langage LogiQL. Il prend un environnement donné en entrée pour vérifier les invariants du réseau.

Vérification de l'infrastructure cloud. Dans le domaine de la vérification du cloud, plusieurs travaux [132], [203], [147] ont proposé des approches pour vérifier la conformité de la sécurité des infrastructures virtualisées du cloud par rapport aux normes, réglementations et politiques des locataires. Plus précisément, Weatherman [132] vérifie de manière proactive les opérations de gestion du cloud afin de détecter la violation des politiques de sécurité (par exemple, isolement réseau, contraintes de placement des VMs) dans la topologie du cloud. Weatherman s'appuie sur un formalisme de transformation des graphes pour modéliser l'impact des opérations de gestion sur la topologie globale du cloud et vérifie la bonne application des politiques de sécurité en utilisant la comparaison de graphes. Par rapport à Weatherman, VeriNeS permet une vérification plus fine des politiques. VeriNeS vérifie les services réseau individuellement, tandis que Weatherman vérifie la topologie globale du cloud.

4.8.2 Vérification dans le contexte NFV

Très peu d'études [204], [84], [82], [22], [24] considèrent les problèmes de vérification dans les environnements NFV, et en particulier la vérification des services réseau basés sur NFV. Pire encore, une étude récente [24] a observé que la spécification de l'architecture de référence NFV manque de détails (par exemple, le mappage entre les VNFs et les instances de service réseau, le mappage entre les instances de services réseau et l'Infrastructure NFV) pour la vérification de la cohérence des services réseau. Shin et al. [22] ont proposé un cadre de vérification basé sur le modèle d'algèbre de processus pour détecter les incohérences (boucles de transfert, violation de politique) dans les environnements SDN/NFV. Cependant, leur cadre de vérification reste générique, et ils ne l'ont pas implémenté dans un cadre NFV réel. Bondan et al. [84] ont présenté un module d'intégrité des services réseau appelé SIM. Ils ont utilisé l'entropie de l'information de Shannon pour détecter diverses anomalies de topologie. SIM récupère les informations concernant les états des services réseau en utilisant l'API nord de l'Orchestrateur NFV. Cependant, leur approche de vérification devient biaisée dans notre modèle de menace (cf. Section 4.3), où nous avons considéré l'Orchestrateur NFV comme une entité non fiable, et donc pouvant fournir de fausses informations. D'autres travaux tels que [82] ont considéré la vérification et la validation des VNFs et des descripteurs de service réseau avant leur déploiement dans une plateforme NFV donnée.

4.9 Conclusion

Dans ce chapitre, nous avons présenté VeriNeS, un cadre de vérification qui peut détecter les anomalies des services réseau dans un environnement NFV basé sur le cloud. VeriNeS reste une couche

d’abstraction dans le plan d’orchestration NFV qui intercepte les commandes de configuration des services réseau pour construire les états des services réseau. VeriNeS utilise un modèle basé sur les graphes pour représenter à la fois les états et les spécifications des services réseau, et détecte les anomalies de service réseau par une vérification du monomorphisme. Nous avons implémenté VeriNeS et montré qu’il peut détecter plusieurs anomalies en quelques secondes. Nos travaux futurs visent à étendre VeriNeS au contexte de l’orchestration multidomaines dans un environnement NFV.

Chapitre 5

5 Conclusion générale et perspective

Dans cette thèse, nous avons identifié le manque de confiance qu'ont les entreprises envers le cloud comme un obstacle majeur à l'externalisation des fonctions réseau vers le cloud à travers NFV. Nous avons préconisé la vérification de service réseau comme une réponse ultime à ce manque de confiance. Ainsi, nous avons apporté plusieurs contributions dans le cadre de la vérification de service réseau dans NFV. Dans ce chapitre, nous dressons un résumé global de ces contributions et nous présentons des perspectives futures pour nos travaux de recherche.

5.1 Résumé des contributions

Nous avons présenté deux contributions majeures dans le domaine de la vérification de service réseau dans NFV. Nous résumons ces deux contributions comme suit :

- **Taxonomie des anomalies de service réseau et état de l'art sur les méthodes de vérification** (cf. Section 3) : dans cette contribution, nous avons d'abord analysé les causes possibles d'anomalies de service réseau en identifiant les menaces et les vulnérabilités des plateformes NFV. Ensuite, nous avons scruté les anomalies potentielles de service réseau et avons proposé une taxonomie de ces anomalies. Aussi, l'analyse des anomalies de service réseau a relevé les impacts négatifs que peuvent avoir les anomalies sur des attributs de service critiques tels que la sécurité et la performance. La taxonomie des anomalies de service réseau fournit aux entreprises et à la communauté scientifique les aspects critiques du service réseau qui doivent être vérifiés, afin d'établir la confiance dans les plateformes cloud basées NFV. Ainsi, nous avons dressé un état de l'art sur les techniques de vérification existantes pour les anomalies de service réseau. Nous avons identifié des manques dans la littérature ainsi que des pistes de recherche dans le domaine de la vérification dans NFV.

- **VeriNeS** (cf. Section 4) : À travers VeriNeS, nous avons montré qu'en observant le comportement de l'Orchestrateur NFV, nous pouvons détecter des anomalies de service réseau. VeriNeS intercepte les commandes de configuration émises par l'Orchestrateur NFV, les analyse, construit des états de service réseau, et détecte des anomalies de service réseau. De façon incrémentale, VeriNeS construit un état global des services réseau à partir des informations extraites de l'analyse des commandes de configuration. En nous appuyant sur un état global au lieu d'un état unique pour chaque service réseau, nous avons surmonté une limite

fondamentale de l'architecture NFV qui reste le manque détails pour corréler les commandes configuration aux services réseau, et pour corréler les services réseau à leurs propriétaires (locataires) respectifs. Aussi appelé *super-graphe*, l'état global est représenté à partir d'un modèle de graphe. La détection d'anomalies consiste à chercher un monomorphisme de graphe entre la représentation sous forme de graphe des spécifications de service réseau et le *super-graphe*. VeriNeS s'intègre facilement dans l'architecture NFV et peut détecter plusieurs anomalies de services réseau en un temps acceptable.

5.2 Perspectives

Passage à l'échelle de VeriNeS

Le temps de vérification de VeriNeS n'excède pas 12 secondes lorsque le cloud compte entre 1000 et 5000 locataires (cf. Section 4.7.2). Cependant, la taille de certains clouds peut atteindre jusqu'à 100 000 locataires [147]. Le passage à l'échelle de VeriNeS nécessite d'optimiser l'algorithme de vérification employé par VeriNeS. Une piste pertinente pour l'optimisation consisterait à paralléliser l'algorithme de vérification. Ainsi, plusieurs modules de vérification peuvent être déployés sur plusieurs nœuds de calcul et répondre aux requêtes de vérification en parallèle.

Adapter VeriNeS au contexte de l'orchestration multidomaines

L'orchestration multidomaines [38], [205], [206] émerge comme une nouvelle architecture pour déployer des services réseau dans NFV. Dans cette architecture, les services réseau s'étendent sur plusieurs domaines NFV qui sont représentés chacun par un orchestrateur NFV de domaine. Au-dessus de ces orchestrateurs de domaine se dresse un *orchestrateur multidomaines* qui déploie les services réseau à travers les domaines, en sollicitant individuellement les orchestrateurs de domaine. Ainsi, la vérification de service réseau basée sur l'approche de VeriNeS requiert d'abord l'interception des commandes de configuration envoyées par l'*orchestrateur multidomaines* aux orchestrateurs de domaines et l'interception des commandes de configuration envoyées par chaque orchestrateur de domaine à ses composantes sous-jacentes (Gestionnaire d'Infrastructure Virtualisée, Gestionnaire de VNFs). Ensuite, les informations récoltées pour chaque commande de configuration doivent être agrégées pour construire un état global des services réseau. Les problèmes de synchronisation constituent un défi majeur pour la vérification de service réseau dans le contexte de l'orchestration multidomaines.

Garantir l'intégrité du code logiciel de VeriNeS

VeriNeS détecte des anomalies de service réseau à travers trois modules de vérification (cf. Section 4.6.1). Cependant, lorsque le code de ses modules devient compromis, les résultats de la

vérification perdent leur fiabilité. Par exemple, un adversaire intelligent peut modifier le code logiciel de l'algorithme de calcul de monomorphisme de graphe utilisé par VeriNeS, afin de contourner le mécanisme de vérification de VeriNeS. En falsifiant les résultats de la vérification, l'adversaire peut occulter des anomalies de service réseau qu'il aurait provoqué. Ainsi, il devient primordial de garantir l'intégrité du code de VeriNeS. Les environnements d'exécution sécurisée (TEE) [207] fournissent une intégrité et une confidentialité basées sur le matériel. Ainsi, en exécutant le code logiciel de VeriNeS dans un TEE tel que Intel SGX [104], nous pouvons assurer l'immuabilité du code de VeriNeS, et ainsi faire confiance aux résultats de la vérification.

Prévenir l'occurrence des anomalies de service réseau

Le risque d'occurrence des anomalies de service réseau reste élevé dans les environnements NFV à cause de la large surface d'attaque des plateformes NFV. En rappel, l'architecture NFV implique une grande pile de logiciels qui présentent chacun des vulnérabilités par lesquelles un adversaire peut compromettre les services réseau. Ce dernier inconvénient peut être détourné en avantage en appliquant le concept de *défense par cibles mobiles* [208]. La *défense par cibles mobiles* consiste à changer dynamiquement la surface d'attaque de la cible à protéger afin d'augmenter la difficulté de réussite d'une attaque contre la cible. Ainsi, en faisant varier la surface d'attaque des plateformes NFV au cours du temps, il serait difficile pour un attaquant de compromettre les services réseau. Par exemple, en alternant dynamiquement les logiciels (p. ex., Orchestrateur NFV, Gestionnaire de VNFs, hyperviseurs) qui composent l'architecture NFV, un adversaire peut difficilement exploiter les vulnérabilités, car celles-ci varient d'une implémentation ou d'une version du logiciel à l'autre.

6 Bibliographie

[1] “Cloud Computing Trends : 2021 State of the Cloud Report,” *Flexera Blog*, Mar. 15, 2021. https://www.flexera.com/blog/cloud/cloud-computing-trends-2021-state-of-the-cloud-report/ (accessed Nov. 07, 2021).

[2] “Cloud Adoption Statistics - It’s Everywhere & Everyone’s Using It in 2021!,” *HostingTribunal*, Jan. 15, 2021. https://hostingtribunal.com/blog/cloud-adoption-statistics/ (accessed Nov. 06, 2021).

[3] “About Netflix - Completing the Netflix Cloud Migration,” *About Netflix*. https://about.netflix.com/, https://about.netflix.com/en/news/completing-the-netflix-cloud-migration (accessed Nov. 13, 2021).

[4] Amazon Web Services, *Migrating to Cloud - Lessons from Netflix, Brought Up to Date*, (Mar. 23, 2018). Accessed: Nov. 13, 2021. [Online]. Available: https://www.youtube.com/watch?v=XrWII4ewrXA

[5] M. Armbrust *et al.*, “A view of cloud computing,” *Communications of the ACM*, vol. 53, no. 4, pp. 50–58, 2010.

[6] E. Knorr, “Cloud computing : IT as commodity,” *InfoWorld*, Jan. 31, 2011. https://www.infoworld.com/article/2625231/cloud-computing--it-as-commodity.html (accessed Nov. 06, 2021).

[7] Amazon Web Services, “8 Business Drivers That Motivate Cloud Migrations.” Accessed: Nov. 07, 2021. [Online]. Available: https://pages.awscloud.com/rs/112-TZM-766/images/AWS_Migration_8_drivers_FINAL.pdf

[8] “Forecast Analysis: Cloud Consulting and Implementation Services, Worldwide,” *Gartner*. https://www.gartner.com/en/documents/3981831/forecast-analysis-cloud-consulting-and-implementation-se (accessed Nov. 08, 2021).

[9] “Gartner Says Four Trends Are Shaping the Future of Public Cloud,” *Gartner*. https://www.gartner.com/en/newsroom/press-releases/2021-08-02-gartner-says-four-trends-are-shaping-the-future-of-public-cloud (accessed Nov. 08, 2021).

[10] N. F. Virtualisation, “An introduction, benefits, enablers, challenges & call for action,” in *White Paper, SDN and OpenFlow World Congress*, 2012, p. 73. [Online]. Available: https://portal.etsi.org/nfv/nfv_white_paper.pdf

[11] B. Carpenter and S. Brim, “Middleboxes: Taxonomy and issues,” RFC 3234, February, 2002.

[12] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, “Making middleboxes someone else’s problem: Network processing as a cloud service,” *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 13–24, 2012.

[13] J. M. Sherry, “Middleboxes as a Cloud Service,” University of California, Berkeley, United States of America, 2016. [Online]. Available: https://escholarship.org/content/qt4kj7g8dz/qt4kj7g8dz.pdf

[14] “Impact of SDN/NFV on Business Models - IEEE Software Defined Networks.” https://sdn.ieee.org/newsletter/january-2016/impact-of-sdn-nfv-on-business-models (accessed Nov. 12, 2021).

[15] “What Is Network as a Service (NaaS) ?,” *Cisco*. https://www.cisco.com/c/en/us/solutions/enterprise-networks/network-as-service-naas.html (accessed Nov. 12, 2021).

[16] “Network Function Virtualization (NFV) Market | Meticulous Market Research Pvt. Ltd.” https://www.meticulousresearch.com/product/network-function-virtualization-5104 (accessed Nov. 12, 2021).

[17] Y. Shin, D. Koo, and J. Hur, “Inferring Firewall Rules by Cache Side-channel Analysis in Network Function Virtualization,” in *IEEE INFOCOM 2020 - IEEE Conference on Computer Communications*, Jul. 2020, pp. 1798–1807. doi: 10.1109/INFOCOM41043.2020.9155449.

[18] N. Alhebaishi, L. Wang, and S. Jajodia, “Modeling and Mitigating Security Threats in Network Functions Virtualization (NFV),” in *Data and Applications Security and Privacy XXXIV*, Cham, 2020, pp. 3–23. doi: 10.1007/978-3-030-49669-2_1.
[19] A. Alnaim, A. Alwakeel, and E. B. Fernandez, “A misuse pattern for compromising VMs via virtual machine escape in NFV,” in *Proceedings of the 14th International Conference on Availability, Reliability and Security*, 2019, pp. 1–6.
[20] A. O. F. Atya, Z. Qian, S. V. Krishnamurthy, T. La Porta, P. McDaniel, and L. M. Marvel, “Catch me if you can: A closer look at malicious co-residency on the cloud,” *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 560–576, 2019.
[21] J. Meza, T. Xu, K. Veeraraghavan, and O. Mutlu, “A large scale study of data center network reliability,” in *Proceedings of the Internet Measurement Conference 2018*, 2018, pp. 393–407.
[22] M.-K. Shin, Y. Choi, H. H. Kwak, S. Pack, M. Kang, and J.-Y. Choi, “Verification for NFV-enabled network services,” in *2015 International Conference on Information and Communication Technology Convergence (ICTC)*, 2015, pp. 810–815.
[23] A. Oqaily *et al.*, “NFVGuard: Verifying the Security of Multilevel Network Functions Virtualization (NFV) Stack,” in *2020 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2020, pp. 33–40.
[24] S. L. Thirunavukkarasu *et al.*, “Modeling NFV Deployment to Identify the Cross-level Inconsistency Vulnerabilities,” in *2019 IEEE International Conference on Cloud Computing Technology and Science (CloudCom)*, 2019, pp. 167–174.
[25] J. Sherry, S. Hasan, C. Scott, A. Krishnamurthy, S. Ratnasamy, and V. Sekar, “Making middleboxes someone else’s problem: Network processing as a cloud service,” *ACM SIGCOMM Computer Communication Review*, vol. 42, no. 4, pp. 13–24, 2012.
[26] “Everything you should know about server rooms,” *RackSolutions*, Jun. 21, 2019. https://www.racksolutions.com/news/blog/server-room-explained/ (accessed Nov. 26, 2021).
[27] “Moving Network Spend From Products To People,” *Gartner*. https://www.gartner.com/smarterwithgartner/moving-network-spend-from-products-to-people (accessed Nov. 27, 2021).
[28] V. Sekar, S. Ratnasamy, M. K. Reiter, N. Egi, and G. Shi, “The middlebox manifesto: enabling innovation in middlebox deployment,” in *Proceedings of the 10th ACM Workshop on Hot Topics in Networks*, 2011, pp. 1–6.
[29] C. Dixon, A. Krishnamurthy, and T. E. Anderson, “An End to the Middle.,” in *HotOS*, 2009, vol. 9, pp. 2–2.
[30] V. Sekar, N. Egi, S. Ratnasamy, M. K. Reiter, and G. Shi, “Design and implementation of a consolidated middlebox architecture,” in *9th ${$USENIX$}$ Symposium on Networked Systems Design and Implementation (${$NSDI$}$ 12)*, 2012, pp. 323–336.
[31] P. Barham *et al.*, “Xen and the art of virtualization,” *ACM SIGOPS operating systems review*, vol. 37, no. 5, pp. 164–177, 2003.
[32] “Empowering App Development for Developers | Docker.” https://www.docker.com/ (accessed Dec. 02, 2021).
[33] J. Martins *et al.*, “ClickOS and the art of network function virtualization,” in *11th ${$USENIX$}$ symposium on networked systems design and implementation (${$NSDI$}$ 14)*, 2014, pp. 459–473.
[34] “MirageOS.” https://mirage.io/ (accessed Dec. 02, 2021).
[35] S. Palkar *et al.*, “E2: A framework for NFV applications,” in *Proceedings of the 25th Symposium on Operating Systems Principles*, 2015, pp. 121–136.
[36] ETSI GS NFV 002, “Network Function Virtualization (NFV); Architectural Framework.” ETSI, 2013. Accessed: Dec. 03, 2021. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV/001_099/002/01.01.01_60/gs_NFV002v010101p.pdf
[37] R. Morabito, J. Kjällman, and M. Komu, “Hypervisors vs. lightweight virtualization: a performance comparison,” in *2015 IEEE International Conference on Cloud Engineering*, 2015, pp. 386–393.

[38] N. F. Saraiva de Sousa, D. A. Lachos Perez, R. V. Rosa, M. A. S. Santos, and C. Esteve Rothenberg, "Network Service Orchestration : A survey," *Computer Communications*, vol. 142–143, pp. 69–94, Jun. 2019, doi : 10.1016/j.comcom.2019.04.008.
[39] "Quality of Service for Voice over IP," *Cisco*. https://www.cisco.com/c/en/us/td/docs/ios/solutions_docs/qos_solutions/QoSVoIP/QoSVoIP.html (accessed Dec. 06, 2021).
[40] "The Year of 100GbE in Data Center Networks," *Data Center Knowledge*, Aug. 28, 2018. https://www.datacenterknowledge.com/networks/year-100gbe-data-center-networks (accessed Dec. 06, 2021).
[41] "P2100G - 2 x 100GbE PCIe NIC." https://www.broadcom.com/products/ethernet-connectivity/network-adapters/100gb-nic-ocp/p2100g (accessed Dec. 06, 2021).
[42] P. Li, X. Wu, Y. Ran, and Y. Luo, "Designing virtual network functions for 100 gbe network using multicore processors," in *2017 ACM/IEEE Symposium on Architectures for Networking and Communications Systems (ANCS)*, 2017, pp. 49–59.
[43] X. Fei, F. Liu, Q. Zhang, H. Jin, and H. Hu, "Paving the way for NFV acceleration: A taxonomy, survey and future directions," *ACM Computing Surveys (CSUR)*, vol. 53, no. 4, pp. 1–42, 2020.
[44] ETSI GS NFV-IFA 001, "Network Functions Virtualisation (NFV); Acceleration Technologies; Report on Acceleration Technologies & Use Cases." ETSI. Accessed: Dec. 06, 2021. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/001/01.01.01_60/gs_NFV-IFA001v010101p.pdf
[45] S. Peter *et al.*, "Arrakis: The operating system is the control plane," *ACM Transactions on Computer Systems (TOCS)*, vol. 33, no. 4, pp. 1–30, 2015.
[46] ETSI GS NFV-IFA 004, "Network Functions Virtualisation (NFV) Release 2; Acceleration Technologies; Management Aspects Specification." ETSI. Accessed: Dec. 06, 2021. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-IFA/001_099/004/02.04.01_60/gs_NFV-IFA004v020401p.pdf
[47] ETSI GS NFV-IFA 014, "Network Functions Virtualisation (NFV) Release 3; Management and Orchestration; Network Service Templates Specification." ETSI, 2019. Accessed: Dec. 04, 2021. [Online]. Available: https://docbox.etsi.org/isg/nfv/open/Publications_pdf/Specs-Reports/NFV-IFA%20014v3.3.1%20-%20GS%20-%20Network%20Service%20Templates%20Spec.pdf
[48] "Topology and Orchestration Specification for Cloud Applications Version 1.0." http://docs.oasis-open.org/tosca/TOSCA/v1.0/TOSCA-v1.0.html (accessed Dec. 05, 2021).
[49] "TOSCA Simple Profile for Network Functions Virtualization (NFV)—Version 1.0." http://docs.oasis-open.org/tosca/tosca-nfv/v1.0/csd04/tosca-nfv-v1.0-csd04.html#_Toc482896036 (accessed Dec. 04, 2021).
[50] P. Mell and T. Grance, "The NIST Definition of Cloud Computing," National Institute of Standards and Technology, NIST Special Publication (SP) 800-145, Sep. 2011. doi: 10.6028/NIST.SP.800-145.
[51] "Cloud Privé | Hosted Private Cloud Solutions & Services | OVHcloud." https://www.ovhcloud.com/fr/hosted-private-cloud/ (accessed Dec. 07, 2021).
[52] "Nokia makes its network functions available on AWS - Telecoms.com." https://telecoms.com/500959/nokia-makes-its-network-functions-available-on-aws/ (accessed Dec. 08, 2021).
[53] "Cloudify as a Service | Cloudify Documentation Center." https://docs.cloudify.co/latest/trial_getting_started/set_trial_manager/hosted_trial/ (accessed Dec. 08, 2021).
[54] G. Gibb, H. Zeng, and N. McKeown, "Outsourcing network functionality," in *Proceedings of the first workshop on Hot topics in software defined networks*, 2012, pp. 73–78.
[55] A. Singhvi, J. Khalid, A. Akella, and S. Banerjee, "Snf: Serverless network functions," in *Proceedings of the 11 th ACM Symposium on Cloud Computing*, 2020, pp. 296–310.
[56] C. Lan, J. Sherry, R. A. Popa, S. Ratnasamy, and Z. Liu, "Embark: Securely outsourcing middleboxes to the cloud," in *13th ${$USENIX$}$ Symposium on Networked Systems Design and Implementation (${$NSDI$}$ 16)*, 2016, pp. 255–273.

[57] X. Yuan, X. Wang, J. Lin, and C. Wang, “Privacy-preserving deep packet inspection in outsourced middleboxes,” in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr. 2016, pp. 1 –9. doi: 10.1109/INFOCOM.2016.7524526.
[58] T. Benson, A. Akella, A. Shaikh, and S. Sahu, “CloudNaaS: a cloud networking platform for enterprise applications,” in *Proceedings of the 2nd ACM Symposium on Cloud Computing*, 2011, pp. 1–13.
[59] R. Mijumbi, J. Serrat, J. -L. Gorricho, N. Bouten, F. De Turck, and R. Boutaba, “Network Function Virtualization: State-of-the-Art and Research Challenges,” *IEEE Communications Surveys & Tutorials*, vol. 18, no. 1, pp. 236–262, 2016, doi: 10.1109/COMST.2015.2477041.
[60] “Finally, private cloud identical to public cloud,” *Lydia Leong*, Jul. 09, 2020. https://blogs.gartner.com/lydia_leong/2020/07/09/finally-private-cloud-identical-to-public-cloud/ (accessed Dec. 08, 2021).
[61] “Introduction to Network Transformation on AWS – Part 1,” *Amazon Web Services*, Jun. 21, 2021. https://aws.amazon.com/blogs/networking-and-content-delivery/introduction-to-network-transformation-on-aws-part-1/ (accessed Dec. 09, 2021).
[62] “Services de cloud computing | Microsoft Azure.” https://azure.microsoft.com/fr-fr/ (accessed Dec. 09, 2021).
[63] Md. T. Khorshed, A. B. M. S. Ali, and S. A. Wasimi, “A survey on gaps, threat remediation challenges and some thoughts for proactive attack detection in cloud computing,” *Future Generation Computer Systems*, vol. 28, no. 6, pp. 833–851, Jun. 2012, doi: 10.1016/j.future.2012.01.006.
[64] S. M. Habib, S. Hauke, S. Ries, and M. Mühlhäuser, “Trust as a facilitator in cloud computing : a survey,” *Journal of Cloud Computing : Advances, Systems and Applications*, vol. 1, no. 1, p. 19, Aug. 2012, doi: 10.1186/2192-113X-1-19.
[65] M. Pattaranantakul, R. He, Q. Song, Z. Zhang, and A. Meddahi, “NFV Security Survey : From Use Case Driven Threat Analysis to State-of-the-Art Countermeasures,” *IEEE Communications Surveys Tutorials*, vol. 20, no. 4, pp. 3330–3368, Fourthquarter 2018, doi : 10.1109/COMST.2018.2859449.
[66] S. Lal, T. Taleb, and A. Dutta, “NFV : Security Threats and Best Practices,” *IEEE Communications Magazine*, vol. 55, no. 8, pp. 211–217, Aug. 2017, doi : 10.1109/MCOM.2017.1600899.
[67] H. Hawilo, M. Jammal, and A. Shami, “Orchestrating network function virtualization platform: Migration or re-instantiation?,” in *2017 IEEE 6 th International Conference on Cloud Networking (CloudNet)*, 2017, pp. 1–6.
[68] S. Kim, S. Park, Y. Kim, S. Kim, and K. Lee, “VNF-EQ : dynamic placement of virtual network functions for energy efficiency and QoS guarantee in NFV,” *Cluster Comput*, vol. 20, no. 3, pp. 2107–2117, Sep. 2017, doi : 10.1007/s10586-017-1004-3.
[69] W. Pauley, “Cloud Provider Transparency : An Empirical Evaluation,” *IEEE Security Privacy*, vol. 8, no. 6, pp. 32–39, Nov. 2010, doi : 10.1109/MSP.2010.140.
[70] D. Yimam and E. B. Fernandez, “A survey of compliance issues in cloud computing,” *J Internet Serv Appl*, vol. 7, no. 1, p. 5, May 2016, doi: 10.1186/s13174-016-0046-8.
[71] H. Baek, A. Srivastava, and J. Van der Merwe, “CloudSight: A Tenant-Oriented Transparency Framework for Cross-Layer Cloud Troubleshooting,” in *2017 17th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, May 2017, pp. 268 –273. doi: 10.1109/CCGRID.2017.97.
[72] W. Yang and C. Fung, “A survey on security in network functions virtualization,” in *2016 IEEE NetSoft Conference and Workshops (NetSoft)*, Jun. 2016, pp. 15–19. doi: 10.1109/NETSOFT.2016.7502434.
[73] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “Analysis of threats and countermeasures in NFV use cases,” in *2019 IEEE International Systems Conference (SysCon)*, Apr. 2019, pp. 1–6. doi: 10.1109/SYSCON.2019.8836849.
[74] A. M. Alwakeel, A. K. Alnaim, and E. B. Fernandez, “A Survey of Network Function Virtualization Security,” in *SoutheastCon 2018*, Apr. 2018, pp. 1–8. doi: 10.1109/SECON.2018.8479121.

[75] A. J. Duncan, S. Creese, and M. Goldsmith, “Insider attacks in cloud computing,” in *2012 IEEE 11th international conference on trust, security and privacy in computing and communications*, 2012, pp. 857–862.
[76] M.-D. Nguyen, N.-T. Chau, S. Jung, and S. Jung, “A demonstration of malicious insider attacks inside cloud iaas vendor,” *International Journal of Information and Education Technology*, vol. 4, no. 6, p. 483, 2014.
[77] V. Varadarajan, Y. Zhang, T. Ristenpart, and M. Swift, “A placement vulnerability study in multi-tenant public clouds,” in *24th ${$USENIX$}$ Security Symposium (${$USENIX$}$ Security 15)*, 2015, pp. 913–928.
[78] Z. Xu, H. Wang, and Z. Wu, “A measurement study on co-residence threat inside the cloud,” in *24th ${$USENIX$}$ Security Symposium (${$USENIX$}$ Security 15)*, 2015, pp. 929–944.
[79] V. Varadarajan, T. Kooburat, B. Farley, T. Ristenpart, and M. M. Swift, “Resource-freeing attacks: improve your cloud performance (at your neighbor’s expense),” in *Proceedings of the 2012 ACM conference on Computer and communications security*, 2012, pp. 281–292.
[80] C. Delimitrou and C. Kozyrakis, “Bolt: I know what you did last summer... in the cloud,” *ACM SIGARCH Computer Architecture News*, vol. 45, no. 1, pp. 599–613, 2017.
[81] A. N. Sylla *et al.*, “Formal Verification of Orchestration Templates for Reliable Deployment with OpenStack Heat*,” in *2019 15th International Conference on Network and Service Management (CNSM)*, Oct. 2019, pp. 1–5. doi: 10.23919/CNSM46954.2019.9012739.
[82] M. Peuster *et al.*, “Introducing automated verification and validation for virtualized network functions and services,” *IEEE Communications Magazine*, vol. 57, no. 5, pp. 96–102, 2019.
[83] M. A. Khan, “A survey of security issues for cloud computing,” *Journal of Network and Computer Applications*, vol. 71, pp. 11–29, Aug. 2016, doi : 10.1016/j.jnca.2016.05.010.
[84] L. Bondan, T. Wauters, B. Volckaert, F. De Turck, and L. Z. Granville, “Anomaly detection framework for SFC integrity in NFV environments,” in *2017 IEEE Conference on Network Softwarization (NetSoft)*, Jul. 2017, pp. 1–5. doi: 10.1109/NETSOFT.2017.8004204.
[85] L. Bondan, “NFV Environments Security through Anomaly Detection,” Universidade Federal do Rio Grande do Sul, 2019. Accessed: Oct. 06, 2021. [Online]. Available: https://lume.ufrgs.br/handle/10183/197460
[86] J. Naous, M. Walfish, A. Nicolosi, D. Mazières, M. Miller, and A. Seehra, “Verifying and enforcing network paths with icing,” in *Proceedings of the Seventh COnference on emerging Networking EXperiments and Technologies*, New York, NY, USA, Dec. 2011, pp. 1–12. doi: 10.1145/2079296.2079326.
[87] G. Liu, H. Sadok, A. Kohlbrenner, B. Parno, V. Sekar, and J. Sherry, “Don’t Yank My Chain: Auditable {NF} Service Chaining,” presented at the 18th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 21), 2021. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi21/presentation/liu-guyue
[88] “SDN switches aren’t hard to compromise, researcher says,” *CIO*. https://www2.cio.com.au/article/581260/sdn-switches-aren-t-hard-compromise-researcher-says/ (accessed Apr. 27, 2021).
[89] P. Kazemian, G. Varghese, and N. McKeown, “Header Space Analysis: Static Checking for Networks,” in *9th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 12)*, 2012, pp. 113–126. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi12/technical-sessions/presentation/kazemian
[90] A. Khurshid, X. Zou, W. Zhou, M. Caesar, and P. B. Godfrey, “VeriFlow: Verifying Network-Wide Invariants in Real Time,” in *10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, 2013, pp. 15–27. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi13/technical-sessions/presentation/khurshid
[91] “An Overview of The Linux Integrity Subsystem,” White paper. [Online]. Available: https://deac-ams.dl.sourceforge.net/project/linux-ima/linux-ima/Integrity_overview.pdf

[92] S. Lal, S. Ravidas, I. Oliver, and T. Taleb, “Assuring virtual network function image integrity and host sealing in Telco cloue,” in *2017 IEEE International Conference on Communications (ICC)*, May 2017, pp. 1–6. doi: 10.1109/ICC.2017.7997299.
[93] “OpenStack Docs : Image Signature Verification.” https://docs.openstack.org/glance/latest/user/signature.html (accessed Apr. 27, 2021).
[94] X. Yuan, H. Duan, and C. Wang, “Assuring String Pattern Matching in Outsourced Middleboxes,” *IEEE/ACM Transactions on Networking*, vol. 26, no. 3, pp. 1362 –1375, Jun. 2018, doi: 10.1109/TNET.2018.2822837.
[95] M. Aiash, G. Mapp, and O. Gemikonakli, “Secure Live Virtual Machines Migration : Issues and Solutions,” in *2014 28th International Conference on Advanced Information Networking and Applications Workshops*, May 2014, pp. 160–165. doi: 10.1109/WAINA.2014.35.
[96] J. Backes *et al.*, “Reachability Analysis for AWS-Based Networks,” in *Computer Aided Verification*, Cham, 2019, pp. 231 –241. doi: 10.1007/978-3-030-25543-5_14.
[97] “SLA Management Handbook.” The Open Group, 2004. [Online]. Available: https://pubs.opengroup.org/onlinepubs/009295499/toc.pdf
[98] “Network Functions Virtualisation (NFV); Accountability; Report on Quality Accountability Framework,” ETSI ISG, Group Specification DGS/NFV-REL005, 01/20216. Accessed: Sep. 23, 2021. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/NFV-REL/001_099/005/01.01.01_60/gs_nfv-rel005v010101p.pdf
[99] Y. Zhang, W. Wu, S. Banerjee, J.-M. Kang, and M. A. Sanchez, “SLA-verifier : Stateful and quantitative verification for service chaining,” in *IEEE INFOCOM 2017 - IEEE Conference on Computer Communications*, May 2017, pp. 1–9. doi: 10.1109/INFOCOM.2017.8057041.
[100] X. Zhang, H. Duan, C. Wang, Q. Li, and J. Wu, “Towards Verifiable Performance Measurement over In-the-Cloud Middleboxes,” in *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, Apr. 2019, pp. 1162–1170. doi: 10.1109/INFOCOM.2019.8737435.
[101] E. Kapassa, M. Touloupou, and D. Kyriazis, “SLAs in 5G: A Complete Framework Facilitating VNF - and NS- Tailored SLAs Management,” in *2018 32nd International Conference on Advanced Information Networking and Applications Workshops (WAINA)*, May 2018, pp. 469–474. doi: 10.1109/WAINA.2018.00130.
[102] “Network Functions Virtualisation (NFV); Service Quality Metrics,” ETSI ISG, DGS/NFV-INF010, 2014. [Online]. Available: https://www.etsi.org/deliver/etsi_gs/nfv-inf/001_099/010/01.01.01_60/gs_nfv-inf010v010101p.pdf
[103] R. Poddar, C. Lan, R. A. Popa, and S. Ratnasamy, “SafeBricks: Shielding Network Functions in the Cloud,” in *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, 2018, pp. 201–216. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi18/presentation/poddar
[104] “Intel® Software Guard Extensions (Intel® SGX),” *Intel*. https://www.intel.com/content/www/fr/fr/architecture-and-technology/software-guard-extensions.html (accessed Apr. 27, 2021).
[105] A. Alashaikh, E. Alanazi, and A. Al-Fuqaha, “A Survey on the Use of Preferences for Virtual Machine Placement in Cloud Data Centers,” *ACM Comput. Surv.*, vol. 54, no. 5, p. 96:1-96:39, May 2021, doi: 10.1145/3450517.
[106] A. Laghrissi and T. Taleb, “A survey on the placement of virtual resources and virtual network functions,” *IEEE Communications Surveys & Tutorials*, vol. 21, no. 2, pp. 1409–1434, 2018.
[107] T. H.-J. Kim, C. Basescu, L. Jia, S. B. Lee, Y.-C. Hu, and A. Perrig, “Lightweight source authentication and path validation,” in *Proceedings of the 2014 ACM conference on SIGCOMM*, New York, NY, USA, Aug. 2014, pp. 271–282. doi: 10.1145/2619239.2626323.
[108] P. Sun, J. Lan, J. Li, Z. Guo, Y. Hu, and T. Hu, “Efficient flow migration for NFV with Graph-aware deep reinforcement learning,” *Computer Networks*, vol. 183, p. 107575, Dec. 2020, doi: 10.1016/j.comnet.2020.107575.

[109] A. Gember-Jacobson *et al.*, “OpenNF: enabling innovation in network function control,” *SIGCOMM Comput. Commun. Rev.*, vol. 44, no. 4, pp. 163–174, Aug. 2014, doi : 10.1145/2740070.2626313.

[110] B. Tschaen, Y. Zhang, T. Benson, S. Banerjee, J. Lee, and J.-M. Kang, “SFC-Checker: Checking the correct forwarding behavior of Service Function chaining,” in *2016 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, Nov. 2016, pp. 134–140. doi: 10.1109/NFV-SDN.2016.7919488.

[111] X. Zhang, Q. Li, J. Wu, and J. Yang, “Generic and agile service function chain verification on cloud,” in *2017 IEEE/ACM 25th International Symposium on Quality of Service (IWQoS)*, Jun. 2017, pp. 1–10. doi: 10.1109/IWQoS.2017.7969150.

[112] S. K. Fayazbakhsh, L. Chiang, V. Sekar, M. Yu, and J. C. Mogul, “Enforcing Network-Wide Policies in the Presence of Dynamic Middlebox Actions using FlowTags,” in *11th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 14)*, 2014, pp. 543–546. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/fayazbakhsh

[113] S. Yao, M. Xu, Q. Li, J. Cao, and Q. Song, “cSFC: Building Credible Service Function Chain on the Cloud,” in *2019 IEEE Global Communications Conference (GLOBECOM)*, Dec. 2019, pp. 1–6. doi: 10.1109/GLOBECOM38437.2019.9013473.

[114] K. Bu, Y. Yang, Z. Guo, Y. Yang, X. Li, and S. Zhang, “FlowCloak: Defeating Middlebox-Bypass Attacks in Software-Defined Networking,” in *IEEE INFOCOM 2018 - IEEE Conference on Computer Communications*, Apr. 2018, pp. 396–404. doi: 10.1109/INFOCOM.2018.8486230.

[115] F. Valenza, S. Spinoso, and R. Sisto, “Formally specifying and checking policies and anomalies in service function chaining,” *Journal of Network and Computer Applications*, vol. 146, p. 102419, Nov. 2019, doi: 10.1016/j.jnca.2019.102419.

[116] A. Aguado, V. López, J. P. Brito, A. Pastor, D. R. López, and V. Martin, “Enabling Quantum Key Distribution Networks via Software-Defined Networking,” in *2020 International Conference on Optical Network Design and Modeling (ONDM)*, May 2020, pp. 1–5. doi: 10.23919/ONDM48393.2020.9133024.

[117] N. C. Thang and M. Park, “Detecting Compromised Switches And Middlebox-Bypass Attacks In Service Function Chaining,” in *2019 29th International Telecommunication Networks and Applications Conference (ITNAC)*, Nov. 2019, pp. 1–6. doi: 10.1109/ITNAC46935.2019.9077969.

[118] S. K. Fayazbakhsh, M. K. Reiter, and V. Sekar, “Verifiable network function outsourcing: requirements, challenges, and roadmap,” in *Proceedings of the 2013 workshop on Hot topics in middleboxes and network function virtualization*, New York, NY, USA, Dec. 2013, pp. 25–30. doi: 10.1145/2535828.2535831.

[119] Frank Brockners, Shwetha Bhandari, Tal Mizrahi, Sashank Dara, and Stephen Youell, “Proof of Transit.” Internet Engineering Task Force (IETF), Sep. 28, 2021. [Online]. Available: https://datatracker.ietf.org/doc/draft-ietf-sfc-proof-of-transit/

[120] M. De Benedictis and A. Lioy, “A proposal for trust monitoring in a Network Functions Virtualisation Infrastructure,” in *2019 IEEE Conference on Network Softwarization (NetSoft)*, Jun. 2019, pp. 1–9. doi: 10.1109/NETSOFT.2019.8806655.

[121] M. De Benedictis and A. Lioy, “Integrity verification of Docker containers for a lightweight cloud environment,” *Future Generation Computer Systems*, vol. 97, pp. 236–246, Aug. 2019, doi : 10.1016/j.future.2019.02.026.

[122] W. Luo, Q. Shen, Y. Xia, and Z. Wu, “Container-IMA : A privacy-preserving Integrity Measurement Architecture for Containers,” in *22nd International Symposium on Research in Attacks, Intrusions and Defenses ({RAID} 2019)*, 2019, pp. 487–500. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/raid2019/presentation/luo

[123] B. Larsen, H. B. Debes, and T. Giannetsos, “CloudVaults: Integrating Trust Extensions into System Integrity Verification for Cloud-Based Environments,” in *Computer Security*, Cham, 2020, pp. 197–220. doi: 10.1007/978-3-030-66504-3_12.

[124] R. G. M. Gallagher, P. MaassMarch 21 2014, and 12:07 A.m, “Inside the NSA’s Secret Efforts to Hunt and Hack System Administrators,” *The Intercept*. https://theintercept.com/2014/03/20/inside-nsa-secret-efforts-hunt-hack-system-administrators/ (accessed Apr. 27, 2021).
[125] T. L. Foundation, “The State of Kernel Self Protection Project by Kees Cook, Google,” *Linux.com*, Sep. 12, 2016. https://www.linux.com/training-tutorials/state-kernel-self-protection-project-kees-cook-google/ (accessed Apr. 27, 2021).
[126] “Welcome To Trusted Computing Group,” *Trusted Computing Group*. https://trustedcomputinggroup.org/ (accessed Apr. 27, 2021).
[127] “TPM 2.0 Library,” *Trusted Computing Group*. https://trustedcomputinggroup.org/resource/tpm-library-specification/ (accessed Apr. 27, 2021).
[128] A. Ltd, “TrustZone for Cortex-A – Arm,” *Arm | The Architecture for the Digital World*. https://www.arm.com/why-arm/technologies/trustzone-for-cortex-a (accessed Apr. 27, 2021).
[129] A. Francillon, Q. Nguyen, K. B. Rasmussen, and G. Tsudik, “A minimalist approach to Remote Attestation,” in *2014 Design, Automation Test in Europe Conference Exhibition (DATE)*, Mar. 2014, pp. 1–6. doi: 10.7873/DATE.2014.257.
[130] A. Suriano, D. Striccoli, G. Piro, R. Bolla, and G. Boggia, “Attestation of Trusted and Reliable Service Function Chains in the ETSI-NFV Framework,” in *2020 6th IEEE Conference on Network Softwarization (NetSoft)*, Jun. 2020, pp. 479–486. doi: 10.1109/NetSoft48620.2020.9165316.
[131] R. Sailer, X. Zhang, T. Jaeger, and L. Van Doorn, “Design and Implementation of a TCG-based Integrity Measurement Architecture.,” in *USENIX Security symposium*, 2004, vol. 13, no. 2004, pp. 223–238.
[132] S. Bleikertz, C. Vogel, and T. Groß, “Cloud radar : near real-time detection of security failures in dynamic virtualized infrastructures,” in *Proceedings of the 30th Annual Computer Security Applications Conference*, New York, NY, USA, Dec. 2014, pp. 26–35. doi: 10.1145/2664243.2664274.
[133] G. G. Xie *et al.*, “On static reachability analysis of IP networks,” in *Proceedings IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies.*, Mar. 2005, vol. 3, pp. 2170–2183 vol. 3. doi : 10.1109/INFCOM.2005.1498492.
[134] H. Mai, A. Khurshid, R. Agarwal, M. Caesar, P. B. Godfrey, and S. T. King, “Debugging the data plane with anteater,” *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 4, pp. 290–301, Aug. 2011, doi : 10.1145/2043164.2018470.
[135] A. Panda, O. Lahav, K. Argyraki, M. Sagiv, and S. Shenker, “Verifying Reachability in Networks with Mutable Datapaths,” in *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, 2017, pp. 699–718. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/panda-mutable-datapaths
[136] L. Yuan, H. Chen, J. Mai, C.-N. Chuah, Z. Su, and P. Mohapatra, “FIREMAN: a toolkit for firewall modeling and analysis,” in *2006 IEEE Symposium on Security and Privacy (S P’06)*, May 2006, p. 15 pp. – 213. doi: 10.1109/SP.2006.16.
[137] H. Hu, G.-J. Ahn, and K. Kulkarni, “Detecting and Resolving Firewall Policy Anomalies,” *IEEE Transactions on Dependable and Secure Computing*, vol. 9, no. 3, pp. 318–331, May 2012, doi : 10.1109/TDSC.2012.20.
[138] H. Zeng *et al.*, “Libra : Divide and Conquer to Verify Forwarding Tables in Huge Networks,” in *11th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 14)*, 2014, pp. 87–99. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi14/technical-sessions/presentation/zeng
[139] E. Al-Shaer, W. Marrero, A. El-Atawy, and K. ElBadawi, “Network configuration in a box: towards end-to-end verification of network reachability and security,” in *2009 17th IEEE International Conference on Network Protocols*, Oct. 2009, pp. 123–132. doi: 10.1109/ICNP.2009.5339690.

[140] K. Jayaraman, N. Bjørner, G. Outhred, and C. Kaufman, “Automated analysis and debugging of network connectivity policies,” *Microsoft Research*, pp. 1–11, 2014.
[141] N. Bjørner and K. Jayaraman, “Checking Cloud Contracts in Microsoft Azure,” in *Distributed Computing and Internet Technology*, Cham, 2015, pp. 21–32. doi: 10.1007/978-3-319-14977-6_2.
[142] C. Basile, D. Canavese, C. Pitscheider, A. Lioy, and F. Valenza, “Assessing network authorization policies via reachability analysis,” *Computers & Electrical Engineering*, vol. 64, pp. 110–131, 2017.
[143] A. Fogel *et al.*, “A General Approach to Network Configuration Analysis,” in *12th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 15)*, 2015, pp. 469–483. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi15/technical-sessions/presentation/fogel
[144] H. Jordan, B. Scholz, and P. Subotić, “Soufflé : On Synthesis of Program Analyzers,” in *Computer Aided Verification*, Cham, 2016, pp. 422–430. doi : 10.1007/978-3-319-41540-6_23.
[145] S. Bayless, N. Bayless, H. Hoos, and A. Hu, “SAT Modulo Monotonic Theories,” *AAAI*, vol. 29, no. 1, Art. no. 1, Mar. 2015, Accessed: Apr. 27, 2021. [Online]. Available: https://ojs.aaai.org/index.php/AAAI/article/view/9755
[146] L. Kovács and A. Voronkov, “First-Order Theorem Proving and Vampire,” in *Computer Aided Verification*, Berlin, Heidelberg, 2013, pp. 1–35. doi: 10.1007/978-3-642-39799-8_1.
[147] Y. Wang, “TenantGuard: Scalable Runtime Verification of Cloud-Wide VM-Level Network Isolation,” masters, Concordia University, 2017. Accessed: Apr. 27, 2021. [Online]. Available: https://spectrum.library.concordia.ca/982513/
[148] S. Majumdar *et al.*, “Proactive Verification of Security Compliance for Clouds Through Pre-computation : Application to OpenStack,” in *Computer Security – ESORICS 2016*, Cham, 2016, pp. 47–66. doi: 10.1007/978-3-319-45744-4_3.
[149] H. Zeng, P. Kazemian, G. Varghese, and N. McKeown, “Automatic test packet generation,” in *Proceedings of the 8th international conference on Emerging networking experiments and technologies*, New York, NY, USA, Dec. 2012, pp. 241–252. doi: 10.1145/2413176.2413205.
[150] S. K. Fayaz, T. Yu, Y. Tobioka, S. Chaki, and V. Sekar, “${$ BUZZ $}$: Testing Context-Dependent Policies in Stateful Networks,” in *13th ${$USENIX$}$ Symposium on Networked Systems Design and Implementation (${$NSDI$}$ 16)*, 2016, pp. 275–289.
[151] P. Perešíni, M. Kuźniar, and D. Kostić, “Monocle : dynamic, fine-grained data plane monitoring,” in *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, New York, NY, USA, Dec. 2015, pp. 1–13. doi: 10.1145/2716281.2836117.
[152] Y. Zhao, H. Wang, X. Lin, T. Yu, and C. Qian, “Pronto : Efficient Test Packet Generation for Dynamic Network Data Planes,” in *2017 IEEE 37 th International Conference on Distributed Computing Systems (ICDCS)*, Jun. 2017, pp. 13–22. doi: 10.1109/ICDCS.2017.55.
[153] K. Bu, X. Wen, B. Yang, Y. Chen, L. E. Li, and X. Chen, “Is every flow on the right track?: Inspect SDN forwarding with RuleScope,” in *IEEE INFOCOM 2016 - The 35th Annual IEEE International Conference on Computer Communications*, Apr. 2016, pp. 1–9. doi: 10.1109/INFOCOM.2016.7524333.
[154] T. Nelson, D. Yu, Y. Li, R. Fonseca, and S. Krishnamurthi, “Simon : scriptable interactive monitoring for SDNs,” in *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, New York, NY, USA, Jun. 2015, pp. 1–7. doi: 10.1145/2774993.2774994.
[155] J. Sommers, P. Barford, N. Duffield, and A. Ron, “Accurate and efficient SLA compliance monitoring,” in *Proceedings of the 2007 conference on Applications, technologies, architectures, and protocols for computer communications*, New York, NY, USA, Aug. 2007, pp. 109–120. doi: 10.1145/1282380.1282394.
[156] Y. Chen, D. Bindel, H. Song, and R. H. Katz, “An algebraic approach to practical and scalable overlay network monitoring,” in *Proceedings of the 2004 conference on Applications, technologies, architectures, and protocols for computer communications*, New York, NY, USA, Aug. 2004, pp. 55–66. doi: 10.1145/1015467.1015475.

[157] B.-Y. Choi, S. Moon, R. Cruz, Z.-L. Zhang, and C. Diot, “Practical delay monitoring for ISPs,” in *Proceedings of the 2005 ACM conference on Emerging network experiment and technology*, New York, NY, USA, Oct. 2005, pp. 83–92. doi: 10.1145/1095921.1095933.
[158] J. Sommers, P. Barford, N. Duffield, and A. Ron, “A Framework for Multi-Objective SLA Compliance Monitoring,” in *IEEE INFOCOM 2007 - 26 th IEEE International Conference on Computer Communications*, May 2007, pp. 2446–2450. doi: 10.1109/INFCOM.2007.298.
[159] Y.-W. E. Sung, C. Lund, M. Lyn, S. G. Rao, and S. Sen, “Modeling and understanding end-to-end class of service policies in operational networks,” *SIGCOMM Comput. Commun. Rev.*, vol. 39, no. 4, pp. 219–230, Aug. 2009, doi : 10.1145/1594977.1592595.
[160] A. Sahai, V. Machiraju, M. Sayal, A. van Moorsel, and F. Casati, “Automated SLA Monitoring for Web Services,” in *Management Technologies for E-Commerce and E-Business Applications*, Berlin, Heidelberg, 2002, pp. 28–41. doi: 10.1007/3-540-36110-3_6.
[161] J. Bendriss, I. G. Ben Yahia, and D. Zeghlache, “Forecasting and anticipating SLO breaches in programmable networks,” in *2017 20th Conference on Innovations in Clouds, Internet and Networks (ICIN)*, Mar. 2017, pp. 127–134. doi: 10.1109/ICIN.2017.7899402.
[162] A. Papageorgiou, A. Fernández-Fernández, L. Ochoa-Aday, M. S. Peláez, and M. Shuaib Siddiqui, “SLA Management Procedures in 5G Slicing-based Systems,” in *2020 European Conference on Networks and Communications (EuCNC)*, Jun. 2020, pp. 7–11. doi: 10.1109/EuCNC48522.2020.9200904.
[163] M. Touloupou, E. Kapassa, C. Symvoulidis, P. Stavrianos, and D. Kyriazis, “An Integrated SLA Management Framework in a 5G Environment,” in *2019 22nd Conference on Innovation in Clouds, Internet and Networks and Workshops (ICIN)*, Feb. 2019, pp. 233–235. doi: 10.1109/ICIN.2019.8685916.
[164] C. Parada *et al.*, “5Gtango: A Beyond-Mano Service Platform,” in *2018 European Conference on Networks and Communications (EuCNC)*, Jun. 2018, pp. 26–30. doi: 10.1109/EuCNC.2018.8443232.
[165] S. Brands and D. Chaum, “Distance-Bounding Protocols,” in *Advances in Cryptology — EUROCRYPT ’93*, Berlin, Heidelberg, 1994, pp. 344–359. doi: 10.1007/3-540-48285-7_30.
[166] M. Irain, J. Jorda, and Z. Mammeri, “Landmark-based data location verification in the cloud: review of approaches and challenges,” *Journal of Cloud Computing*, vol. 6, no. 1, p. 31, Dec. 2017, doi: 10.1186/s13677-017-0095-y.
[167] P. Gill, Y. Ganjali, B. Wong, and D. Lie, “Dude, where’s that IP? circumventing measurement-based IP geolocation,” in *Proceedings of the 19th USENIX conference on Security*, USA, Aug. 2010, p. 16.
[168] V. N. Padamanabban and L. Subramanian, “Determining the geographic location of Internet hosts,” in *Proceedings of the 2001 ACM SIGMETRICS international conference on Measurement and modeling of computer systems*, New York, NY, USA, Jun. 2001, pp. 324–325. doi: 10.1145/378420.378814.
[169] I. Poese, S. Uhlig, M. A. Kaafar, B. Donnet, and B. Gueye, “IP geolocation databases: unreliable?,” *SIGCOMM Comput. Commun. Rev.*, vol. 41, no. 2, pp. 53–56, Apr. 2011, doi : 10.1145/1971162.1971171.
[170] C. Guo, Y. Liu, W. Shen, H. J. Wang, Q. Yu, and Y. Zhang, “Mining the Web and the Internet for Accurate IP Address Geolocations,” in *IEEE INFOCOM 2009*, Apr. 2009, pp. 2841–2845. doi: 10.1109/INFCOM.2009.5062243.
[171] B. Eriksson, P. Barford, J. Sommers, and R. Nowak, “A Learning-Based Approach for IP Geolocation,” in *Passive and Active Measurement*, Berlin, Heidelberg, 2010, pp. 171–180. doi: 10.1007/978-3-642-12334-4_18.
[172] E. Katz-Bassett, J. P. John, A. Krishnamurthy, D. Wetherall, T. Anderson, and Y. Chawathe, “Towards IP geolocation using delay and topology measurements,” in *Proceedings of the 6th ACM SIGCOMM conference on Internet measurement*, New York, NY, USA, Oct. 2006, pp. 71–84. doi: 10.1145/1177080.1177090.

[173] B. Wong, I. Stoyanov, and E. G. Sirer, “Octant : A Comprehensive Framework for the Geolocalization of Internet Hosts,” in *NSDI*, 2007, vol. 7.
[174] A. Albeshri, C. Boyd, and J. G. Nieto, “GeoProof: Proofs of Geographic Location for Cloud Computing Environment,” in *2012 32nd International Conference on Distributed Computing Systems Workshops*, Jun. 2012, pp. 506–514. doi: 10.1109/ICDCSW.2012.50.
[175] C. Krauß and V. Fusenig, “Using Trusted Platform Modules for Location Assurance in Cloud Networking,” in *Network and System Security*, Berlin, Heidelberg, 2013, pp. 109–121. doi: 10.1007/978-3-642-38631-2_9.
[176] M. Bartock, “Trusted Geolocation in the Cloud: Proof of Concept Implementation,” *Publication NISTIR*, vol. 7904, p. 59, 2015.
[177] A. U. Rehman, R. L. Aguiar, and J. P. Barraca, “Network Functions Virtualization: The Long Road to Commercial Deployments,” *IEEE Access*, vol. 7, pp. 60439–60464, 2019, doi: 10.1109/ACCESS.2019.2915195.
[178] Y. Lin, T. He, S. Wang, K. Chan, and S. Pasteris, “Looking Glass of NFV : Inferring the Structure and State of NFV Network From External Observations,” *IEEE/ACM Transactions on Networking*, vol. 28, no. 4, pp. 1477–1490, Aug. 2020, doi : 10.1109/TNET.2020.2985908.
[179] S. Rajagopalan, D. Williams, H. Jamjoom, and A. Warfield, “Split/Merge : System Support for Elastic Execution in Virtual Middleboxes,” in *10th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 13)*, 2013, pp. 227–240.
[180] M. Kablan, A. Alsudais, E. Keller, and F. Le, “Stateless Network Functions: Breaking the Tight Coupling of State and Processing,” in *14th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 17)*, 2017, pp. 97–112. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi17/technical-sessions/presentation/kablan
[181] S. Woo, J. Sherry, S. Han, S. Moon, S. Ratnasamy, and S. Shenker, “Elastic Scaling of Stateful Network Functions,” in *15th {USENIX} Symposium on Networked Systems Design and Implementation ({NSDI} 18)*, 2018, pp. 299–312. Accessed: Apr. 27, 2021. [Online]. Available: https://www.usenix.org/conference/nsdi18/presentation/woo
[182] G. Gu, P. Fogla, D. Dagon, W. Lee, and B. Skorić, “Measuring intrusion detection capability: an information-theoretic approach,” in *Proceedings of the 2006 ACM Symposium on Information, computer and communications security*, New York, NY, USA, Mar. 2006, pp. 90–101. doi: 10.1145/1128817.1128834.
[183] “Using leak tests to evaluate firewall effectiveness.” https://securelist.com/using-leak-tests-to-evaluate-firewall-effectiveness/36182/ (accessed Apr. 27, 2021).
[184] E. J. Scheid, B. B. Rodrigues, L. Z. Granville, and B. Stiller, “Enabling Dynamic SLA Compensation Using Blockchain-based Smart Contracts,” in *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, Apr. 2019, pp. 53–61.
[185] E. J. Scheid and B. Stiller, “Leveraging Smart Contracts for Automatic SLA Compensation-The Case of NFV Environments.,” in *AIMS*, 2018, pp. 70–74.
[186] E. G. NFV-MAN, “Network Functions Virtualisation (NFV); Management and Orchestration,” ETSI ISG, GS NFV 004 v1.1.1, Oct. 2014.
[187] A. J. Gonzalez, G. Nencioni, A. Kamisiński, B. E. Helvik, and P. E. Heegaard, “Dependability of the NFV orchestrator: State of the art and research challenges,” *IEEE Communications Surveys & Tutorials*, vol. 20, no. 4, pp. 3307–3329, 2018.
[188] “CERT Definition of ‘Insider Threat’ - Updated,” *SEI Blog*. https://insights.sei.cmu.edu/blog/cert-definition-of-insider-threat-updated/ (accessed Dec. 23, 2021).
[189] “Top Threats to Cloud Computing : Egregious Eleven Deep Dive,” Cloud Security Alliance, Survey, 2020.
[190] “Compute API — nova documentation.” https://docs.openstack.org/api-ref/compute/ (accessed Dec. 28, 2021).
[191] “Networking API v2.0 ; Networking API Reference documentation.” https://docs.openstack.org/api-ref/network/v2/index.html (accessed Dec. 28, 2021).

[192] “API Model ; networking-sfc 13.1.0.dev3 documentation.” https://docs.openstack.org/networking-sfc/latest/contributor/api.html (accessed Dec. 28, 2021).
[193] “NetworkX — NetworkX documentation.” https://networkx.org/ (accessed Dec. 24, 2021).
[194] “1. OSM Quickstart — Open Source MANO 6.0 documentation.” https://osm.etsi.org/docs/user-guide/01-quickstart.html (accessed Dec. 25, 2021).
[195] “Open Source Cloud Computing Platform Software,” *OpenStack*. https://www.openstack.org/software/ (accessed Dec. 25, 2021).
[196] “Redis.” https://redis.io/ (accessed Dec. 25, 2021).
[197] L. P. Cordella, P. Foggia, C. Sansone, and M. Vento, “A (sub) graph isomorphism algorithm for matching large graphs,” *IEEE transactions on pattern analysis and machine intelligence*, vol. 26, no. 10, pp. 1367–1372, 2004.
[198] R. Beckett, A. Gupta, R. Mahajan, and D. Walker, “A general approach to network configuration verification,” in *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, 2017, pp. 155–168.
[199] Y. Xu, Y. Liu, R. Singh, and S. Tao, “SDN state inconsistency verification in openstack,” *Computer Networks*, vol. 110, pp. 364–376, 2016.
[200] Y. Yuan, S. Chandrasekaran, L. Jia, and V. Sekar, “Efficient and correct test scheduling for ensembles of network policies,” in *15th ${$USENIX$}$ Symposium on Networked Systems Design and Implementation (${$NSDI$}$ 18)*, 2018, pp. 437–452.
[201] P. Zhang *et al.*, “Mind the gap: Monitoring the control-data plane consistency in software defined networks,” in *Proceedings of the 12th International on Conference on emerging Networking EXperiments and Technologies*, 2016, pp. 19–33.
[202] H. Kim, J. Reich, A. Gupta, M. Shahbaz, N. Feamster, and R. Clark, “Kinetic: Verifiable dynamic network control,” in *12th ${$USENIX$}$ Symposium on Networked Systems Design and Implementation (${$NSDI$}$ 15)*, 2015, pp. 59–72.
[203] T. Madi *et al.*, “ISOTOP: auditing virtual networks isolation across cloud layers in OpenStack,” *ACM Transactions on Privacy and Security (TOPS)*, vol. 22, no. 1, p. 1, 2019.
[204] C. Basile, F. Valenza, A. Lioy, D. R. Lopez, and A. P. Perales, “Adding support for automatic enforcement of security policies in NFV networks,” *IEEE/ACM Transactions on Networking*, vol. 27, no. 2, pp. 707–720, 2019.
[205] K. Katsalis, N. Nikaein, and A. Edmonds, “Multi-domain orchestration for nfv: Challenges and research directions,” in *2016 15th International Conference on Ubiquitous Computing and Communications and 2016 International Symposium on Cyberspace and Security (IUCC-CSS)*, 2016, pp. 189–195.
[206] P. Karamichailidis, K. Choumas, and T. Korakis, “Enabling multi-domain orchestration using Open Source MANO, OpenStack and OpenDaylight,” in *2019 IEEE International Symposium on Local and Metropolitan Area Networks (LANMAN)*, 2019, pp. 1–6.
[207] M. Sabt, M. Achemlal, and A. Bouabdallah, “Trusted execution environment: what it is, and what it is not,” in *2015 IEEE Trustcom/BigDataSE/ISPA*, 2015, vol. 1, pp. 57–64.
[208] J.-H. Cho *et al.*, “Toward proactive, adaptive defense: A survey on moving target defense,” *IEEE Communications Surveys & Tutorials*, vol. 22, no. 1, pp. 709–745, 2020.