UFR de mathématique et d'informatique
Université de Strasbourg

Master de Science, Technologie, Santé
mention Informatique
spécialité Science et Ingénierie des Réseaux,
de l'Internet et des Systèmes

Présenté par
Constantin Divriotis
constantin.divriotis@etu.unistra.fr

Communications sécurisées intra-cloud et inter-cloud

Encadré par
Andréas Guillot
andres.guillot@unistra.fr

Au sein de
Equipe réseaux - UFR Mathématiques et informatique de l'université de Strasbourg

TABLE DES MATIÈRES

CHAPTER 1

INTRODUCTION

L'objectif de ce sujet du Travail d'Étude et de Recherche est de modéliser un système de *workflow* répondant à :

- une authentification sécurisée de plusieurs acteurs d'un même système,
- une communication sécurisée entre plusieurs acteurs,
- une gestion de l'ordonnancement de plusieurs tâches entre elles.

Pour pouvoir répondre convenablement au sujet, nous nous limiterons à un cas d'étude particulier des systèmes de *workflow* : le cloud informatique, appelé aussi *cloud computing*. Nous nous intéresserons plus particulièrement à la communication sécurisée entre plusieurs acteurs au sein du *cloud computing*.
La définition, fournie par l'institut national des standards et de la technologie (NIST) en 2011, permet de comprendre le but et les objectifs du cloud informatique ([14]) : « accès à distance via le réseau, à la demande et en libre- service, à des ressources informatiques partagées configurables ». Le *cloud computing* se compose par plusieurs fonctionnalités importantes telles que : il est accessible à distance n'importe où, et n'importe quand (mondialisation), c'est un service à la demande (Rapidité, facilité, flexibilité, scalabilité) et proposant une haute disponibilité du service (*Service Level Agreement* (*Service Level Agreement*)) et une sécurité des données (stockage sécurisé) et impliquant une démarche éco-responsable. Le NIST présente également différents modèles de cloud. Tout d'abord, il existe le cloud privé proposant une infrastructure exclusivement réservée à une entreprise ou une organisation. Le cloud communautaire, lui, est une infrastructure réservée à une communauté d'organisations partageant des objectifs communs (politique de sécurité par exemple). Puis, le cloud public est une infrastructure appartenant à une entité et proposant des services payants accessible par le grand public. Enfin, il existe le cloud hybride, qui est une infrastructure composée de plusieurs types de cloud. Dans ce document, les principales cibles d'attaques sont les clouds publics c'est pourquoi nous nous focaliserons dessus. Le nombre de données et de personnes plus élevés que sur les autres modèles de cloud ouvrent plus de possibilités aux *Black Hats*.

Les caractéristiques du cloud informatique ont créées une certaine popularité de cette nouvelle forme de stockage dans l'industrie et dans les académies, à tel point que même certaines infrastructures importantes et critiques (banque, NASA) soient tentées de migrer vers le cloud. L'enjeu et obstacle principal du cloud informatique à l'heure actuelle est la sécurité du cloud, qui empêche notamment son utilisation unanime [12].

La sécurité du cloud implique des concepts tels que la sécurité des réseaux, du matériel, mais également la sécurité de nouveaux concepts tels que la virtualisation et de l'architecture *multi-tenant*. En effet, les solutions de sécurité actuelles proposent des solutions contre des menaces provenant de l'extérieur afin de protéger le réseau intérieur, mais cette approche

n'est plus viable avec le cloud [15] [4]. En effet, le système ne peut plus avoir confiance dans son réseau interne, car un utilisateur malicieux peut être présent sur le réseau interne. Des nouvelles attaques ont émergées suite à l'arrivé du cloud [17]. La sécurité dans le cloud fait alors appel à une série de politiques de sécurité et de technologies dans le but de protéger les données, informations, applications et infrastructures permettant d'assurer la sécurité du cloud en validant certains propriétés :

Disponibilité : les ressources du cloud sont toujours accessible aux utilisateurs.

Integrité : la modification ou la suppression par des utilisateurs non autorisés est impossible et le cloud permet de vérifier que les données n'ont pas été altérées.

Confidentialité : les données des utilisateurs sont secrètes et aucun autre utilisateur n'a de droit sur les données d'un autre utilisateur.

Contrôle : gestion des ressources du cloud.

Authentification : le cloud est capable de garantir que l'utilisateur est bien celui qu'il prétend lorsqu'un utilisateur agit sur les ressources du cloud et les utilise.

Non-répudation : le cloud est capable de garantir les actions effectués sur des ressources par un utilisateur.

Dans un environnement cloud, le vecteur d'attaque principal est le réseau. Il apparaît crucial de sécuriser les communications internes et externes. Par conséquent, nous avons séparer les communications en deux catégories : les communications intra-cloud et inter-cloud. Les communications intra-cloud impliquent de nouveaux concepts et ont générés des nouveaux problèmes en terme de sécurité. Le partage des communications sur une même infrastructure et les réseaux virtuels sont des nouveaux challenges dues aux nouveaux concepts du cloud. ([2]). Un routeur virtuel est une instance d'un routeur physique, et plusieurs routeurs virtuels peuvent s'exécuter en même temps sur un même routeur physique. On obtient alors la possibilité de déployer plusieurs réseaux, que l'on appelle des réseaux virtuels, sur un même réseau physique. Un réseau virtuel est un regroupement d'un ensemble de routeurs virtuels compatibles. Les routeurs virtuels utilisent les liaisons physiques pour s'interconnecter. Tous ces routeurs virtuels doivent être isolés les uns des autres et cela implique une sécurité des réseaux implacable à ce niveau. En effet, comme les *Virtual Machine* (ou appelés également machines virtuelles) partagent la même infrastructure réseau physique et cela ouvre notamment la porte à des attaques de type cross-tenant [17]. L'isolation des réseaux protège également contre des attaques globales. Les communications inter-cloud sont similaires aux communications sur Internet et donc, les différents problèmes de sécurité sont les mêmes.

Dans ce document, nous allons nous intéresser à la sécurité du cloud au niveau du réseau au travers notamment des communications internes et externes du cloud, car l'intégrité et la confidentialité des données peuvent être menacés à chaque communication. Les solutions aujourd'hui pour assurer la sécurité du cloud informatique passent notamment par l'utilisation de firewalls, des IDS et des IPS. Sécuriser le réseau du cloud permet d'éviter de nombreuses attaques. Tout d'abord, le balayage de port qui se définit comme une attaque réseau visant à récupérer l'état des services réseaux d'une machine. L'attaque par déni de service, appelée *Denial of Service* (DoS), est une attaque réseau pouvant cibler différents composants d'un système (processeur, mémoire) et visant à dégrader les performances du système. On parle maintenant d'attaque DDoS (*Distributed Denial of Service*), car l'attaque est menée par plusieurs systèmes. L'attaque *cross-tenant* exploite la mémoire cache L3, ce cache étant l'image miroir de la mémoire système et est partagé entre tous les utilisateurs sur un cloud *multi-tenant*. Une personne malveillante va alors manipuler l'état de cette mémoire cache et attendre l'activité d'un utilisateur. En examinant les sections du cache modifiées, la personne malveillante peut trouver l'adresse mémoire contenant les données de l'utilisateur [17]. Le

spoofing se définit par l'usurpation d'identité visant à se faire passer pour quelqu'un d'autre dans le but de commettre différents actes malveillants. Enfin, on terminera par le *sniffing* qui se caractérise par une écoute clandestine d'un réseau visant à lire et capturer les données qui transitent.

Nous allons tenter de répondre au problème suivant : comment obtenir un réseau sécurisé tout en garantissant un niveau de performance élevé et une confiance limitée à tous les niveaux. Nous allons tenter, dans ce rapport, de trouver les meilleures solutions possibles actuelles et de comprendre quels éléments peuvent être rajoutés à cette configuration pour obtenir une configuration idéale. Le but de ce document sera donc de créer une solution cohérente qui inclut le maximum de besoins et d'exigences en sécurité dans l'environnement cloud, mais cette solution doit être en balance entre les enjeux de sécurité et les enjeux de performance et de disponibilité du cloud. Ce document se base principalement sur l'étude effectuée par Mazhar et al. [2].

Nous allons poser différentes hypothèses liées à notre problématique auxquels nous allons chercher à répondre par des solutions concrètes :

Hypothèse de stockage : aucune confiance, les données du cloud ne doivent jamais être rendues publiques ou être victime d'une violation de données.

Hypothèse d'utilisateur : confiance limité avec l'utilisateur, ce dernier peut provoquer des fuites de données.

Hypothèse du réseau : aucune confiance au réseau, les données ne doivent pas être visibles et lues par une personne malveillante écoutant le réseau.

La figure 1.1 décrit les différents challenges au niveau de la sécurité et souligne les parties, en rouge, dont nous allons nous préoccuper principalement. Ce modèle fait écho aux hypothèses que nous avons posé et nous allons nous baser notamment sur l'étude effectuée [2]) qui découpe la sécurité du cloud en plusieurs sections distinctes et propose différentes solutions pertinentes.

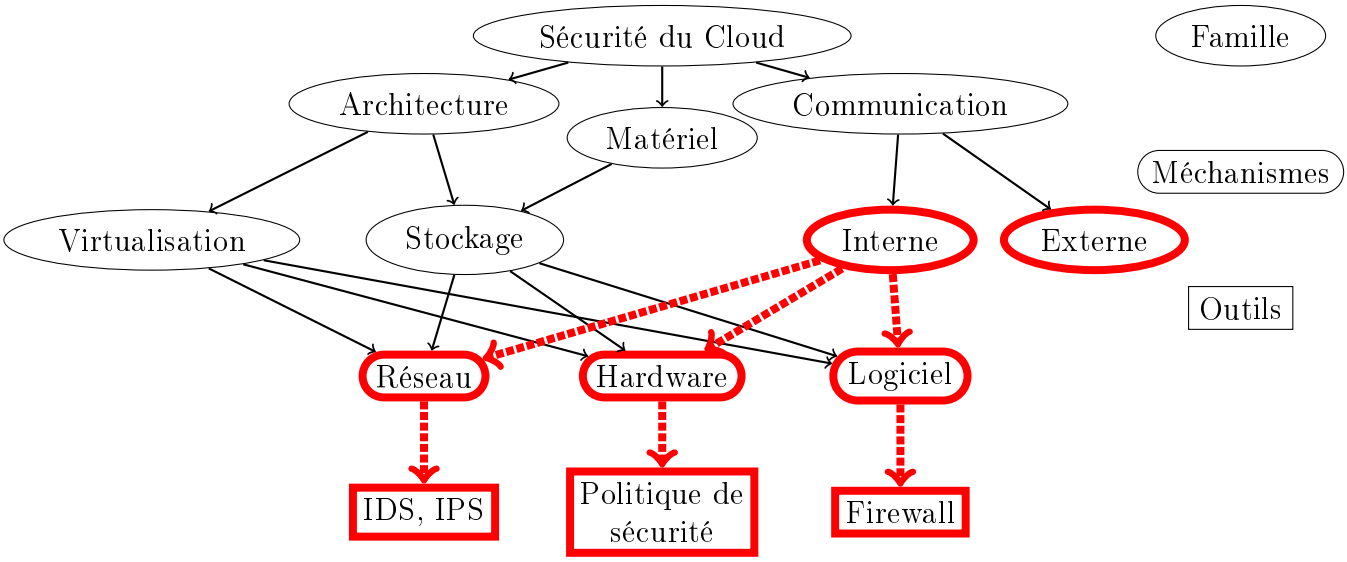

Figure 1.1: Challenges de la sécurité du cloud

Ce document est organisé de la manière suivante : le chapitre 2 contient un état de l'art de la sécurité dans le cloud informatique structuré autour de trois axes : la stockage 1, la virtualisation 2, et l'authentification et le contrôle d'accès 3. Le chapitre 3 analysera différentes méthodes de la littérature pour les communications réseaux dans le cloud, avec une séparation entre les communications inter-cloud 1 dans un premier temps, puis les communications intra-cloud 2.

CHAPTER 2

ÉTAT DE L'ART

Ce chapitre va s'intéresser en premier lieu à l'hypothèse du stockage sécurisé dans un environnement cloud. Puis, le chapitre va s'intéresser à de potentielles solutions des challenges posés au niveau de la virtualisation en terme de sécurité. Enfin, on discutera d'une solution garantissant une authentification sécurisée de plusieurs acteurs d'un même système

1 | Stockage

Le stockage est l'une des trois hypothèses posées dans l'introduction. Pour pouvoir nous focaliser sur les communications dans un environnement cloud, il est capital d'établir un schéma avec un stockage sécurisé. Le stockage est un élément central dans l'environnement cloud. Le cloud n'offre pas tous les droits à l'utilisateur sur les données. En effet, un utilisateur n'a qu'un droit de contrôle sur la VM sur laquelle dont il est l'utilisateur. Les fournisseurs de service de cloud (appelé CSP pour *Cloud Service Provider*), eux, ont le rôle de gérer les données et les serveurs. Mais, ce niveau de contrôle doit être maîtrisé et vérifié, car l'environnement cloud (architecture *multi-tenant* et virtualisation) a généré de nouvelles menaces. Le stockage doit faire face différentes menaces : une confidentialité et une intégrité des données non maîtrisées, une vulnérabilité sur la récupération des données (données d'un utilisateur ayant eu la même image précédemment), un mauvais nettoyage des supports (changement de disque nécessaire ou données stockées inutilisées), un *backup* des données non sécurisé, des vulnérabilités via les applications Web ([1]) ou encore une authentification et un contrôle d'accès insuffisamment efficace.

Dans cet optique, nous avons suivi les principales recommandations du CSA (*Cloud Security Alliance*) en terme de gestion des clés et de stockage des données ([16]) : Premièrement, la gestion des clés doit être effectuée par les organisations/utilisateurs eux-mêmes ou bien par un service cryptographique de confiance avec un service crédible et neutre. Deuxièmement, les meilleurs pratiques pour la gestion des clés et de chiffrement provenant de source fiable doivent être utilisées. Troisièmement, il est recommandé d'utiliser une technologie standardisée dans la mesure du possible. Quatrièmement, la portée d'une clé ne doit pas dépasser un utilisateur ou un groupe d'utilisateur. Enfin, l'utilisation d'algorithmes standards est recommandée et les algorithmes de chiffrement propriétaires sont déconseillés.

Sur la base de l'étude [2], il en ressort plusieurs solutions. Nous avons décidé d'en retenir deux dans le but d'offrir des approches différentes selon le stockage souhaité. Ces deux solutions ont été choisies, car elles mettent en place les recommandations de sécurité du CSA [16]. De plus, les solutions sont présentées dans le papier Mazhar et al [2] et ont l'avantage d'être des solutions récentes pour sécuriser le stockage dans le cloud : FADE ([21]), SecCloud ([24]).

1.1 FADE

FADE (*File Assured Deletion*) est un protocole permettant une gestion de clés cryptographiques et utilisant des techniques de cryptographie symétriques et asymétriques. Ce protocole garantit une confidentialité et une intégrité des données dans un environnement cloud en fonctionnant avec des groupes de gestion de clés (KM pour *key manager*) agissant comme un *Trusted third party* (*Trusted Party Third*) et générant des clés publiques/privées. Différents cryptages sur les mêmes données permettent de sécuriser ces dernières et d'effectuer un contrôle d'accès implicite des données. Le fonctionnement du protocole est décrit sur la figure 2.1 : pour encrypter un fichier F de l'utilisateur, une clé de donnée aléatoire K est utilisée. Ce fichier F, encrypté par K, sera encrypté par une clé symétrique S. Enfin, la paire de clé générée par KM encrypte S. Un ficher de politique d'accès P permet de vérifier qui peut accéder au fichier.
Pour télécharger un fichier stocké cloud, un utilisateur doit effectuer une requête en envoyant P auprès du KM afin qu'il lui délivre une paire de clé publique/privée. La clé publique sera transmise à l'utilisateur qui encryptera le fichier F en encryptant F avec K et K avec S. Enfin, la clé publique, délivrée par KM, encryptera S. Le fichier F est stockée dans le cloud avec un haut niveau de cryptage et avec le fichier P. Pour décrypter un fichier stocké dans le cloud, il est nécessaire que l'utilisateur envoie S à KM afin de décrypter le fichier, puis décrypter F. Enfin, un écrasement sécurisé des clés cryptographiques et d'un fichier de politique (politique vis-à-vis des accès au fichier stocké sur le cloud par un utilisateur) permet de rendre les données inaccessibles et donc assurément supprimées.
Une solution retenue pour le stockage dans le cloud est FADE, car elle permet un contrôle d'accès et une suppression définitive des données, tout en proposant un scalabilité élevé, un cryptage des données avec l'aide d'un TPA et garantit l'intégrité et la confidentialité des données.

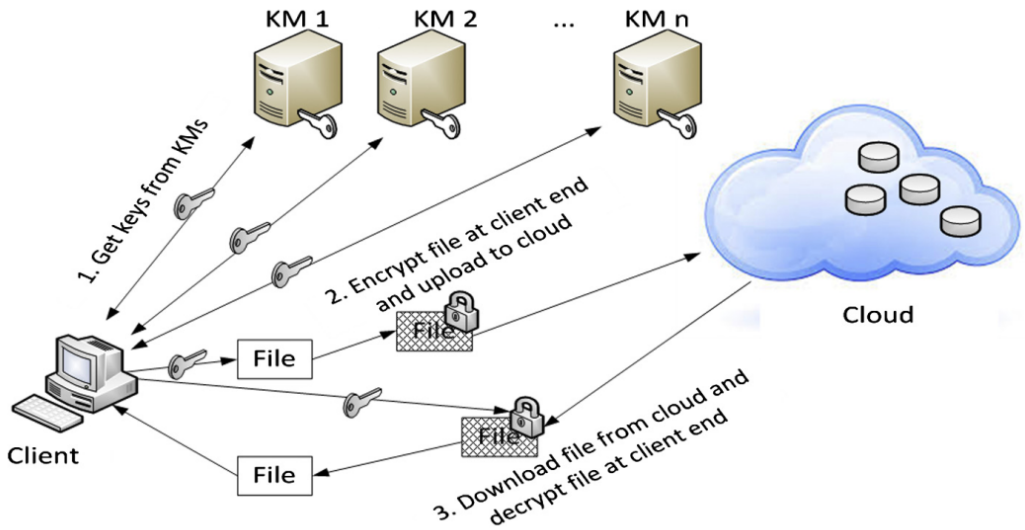

Figure 2.1: Fonctionnement du protocole FADE [[2]]

1.2 SecCloud

SecCloud est un protocole cherchant à protéger les données stockées par un utilisateur, mais également les calculs effectués sur ces données. Le protocole se base sur l'utilisation d'une cryptographie à base de couplage pour permettre de générer des clés pour les utilisateurs, le CSP et un TPA. Un utilisateur peut alors stocker ses données dans l'espace de stockage fourni par un CSP et ces données sont signées par un TPA. Les données sont alors envoyées au cloud avec la signature du TPA et encryptées avec une clé de session. Le cloud va alors

décrypté ces données, vérifié la signature du TPA et stocké les données dans les partitions désignées dans le but d'empêcher toute violation des données. L'algorithme Merkle Hash Tree, reconstruit par le TPA, permet de vérifier les calculs effectués sur les données. Le TPA a une place importante, car il a le devoir de vérifier l'accessibilité et l'intégrité des données sur requête du client.
SecCloud est une solution pertinente pour le stockage dans le cloud. L'utilisation d'un TPA, la vérification de signature et l'utilisation d'une cryptographie à base de couplage pour crypter les données sont des fonctionnalités alléchantes de cette solution. Elle assure la confidentialité et l'intégrité des données via ces fonctionnalités.

2 | Virtualisation

La virtualisation permet l'utilisation des mêmes ressources physiques pour plusieurs utilisateurs. Une VM est instanciée pour chaque utilisateur et permet alors à l'utilisateur d'avoir une machine complète et opérationnelle. Plusieurs machines virtuelles peuvent être sur les mêmes ressources physiques et créer un environnement multi-tenant. Tout comme le stockage, la virtualisation soulève de nouveaux défis, car toutes les ressources sont virtualisées (serveurs, stockage, équipements de réseau, etc.). En particulier, la connexion aux serveurs dans le cloud doit s'effectuer par le biais d'un réseau virtuel mis en place dans ce but et où les routeurs virtuels se déplacent en fonction des emplacements des ressources du CSP. Aujourd'hui, selon le type de menace, on peut recenser des solutions différentes. Dans cette partie, nous dresserons des tableaux comparatifs proposant des solutions adéquates à chaque problème de virtualisation.

2.1 Migration

La migration d'une VM est une opération courante effectuée dans le cloud afin d'équilibrer les charges au sein du cloud, remplacer des VMs tombées dû à des pannes physiques, économiser de l'énergie ou encore effectuer des mises à jour matérielles ou logicielles. La relocalisation d'une VM sur une autre machine physique sans éteindre la VM ouvre la porte à des problèmes d'intégrité et de confidentialité des données, car les données de la VM sont exposées au réseau. De plus, la migration elle-même est sujet aux attaques, comme une relocalisation d'une VM sur un serveur malveillant dans le but de prendre le contrôle total de la VM. La migration est une opération délicate qui nécessite une sécurité accrue. Le tableau comparatif 2.2 présente diverses solutions permettant de migrer une VM. De ce tableau, on peut retenir la solution appelée *Framework for secure live VM migration* qui permet un cryptage des données lors de la migration, mais également un contrôle d'accès basé sur les rôles (*Role-Based Access Control*). Malgré une scalabilité moyenne, les apports supplémentaires de cette solution nous amènent à la plébisciter. En effet, elle apporte une sécurité contre les migrations inutiles, mais également contre les attaques *VM hopping* (attaque appelée également *VM jumping*).

2.2 Partage d'image

Les images de VM permettent d'initialiser une VM. Pour ne pas corrompre une VM, elle doit être parfaitement intègre et sécurisée. Une fois en place dans le cloud, une VM infectée est capable de gérer son trafic entrant/sortant et peut endommager les données d'autres utilisateurs et créer des problèmes d'intégrité et de confidentialité. La plupart du temps, les images des VMs sont utilisées par divers utilisateurs et certains utilisateurs malicieux seront tentés d'infecter certaines images en injectant un *malware*. Autre possibilité, un utilisateur malicieux va chercher une faille et un point d'attaque dans le code public des images et si il est capable d'en trouver une, mettre en place une attaque contre ce type d'images. Le tableau

Proposed scheme	Basic theory	Privacy	Integrity	Scalability	Other features
Secure and trust preserving VM migration mechanism	• Trusted computing • Remote auditing	✓	✓	Medium	Novel credentials for trust level quantification
Protocol for vTPM based VM migration	• Trusted computing • Remote auditing • Tunneled communication channel	✓	✓	Medium	Data freshness
Protocol for VM-vTPM migration	Trusted Computing	✓	✓	Medium	Migration initiation authenticity
Framework for secure live VM migration	• Trusted computing • Role based access control • Cryptography	✓	✓	Medium	Security against • VM hopping • Useless migrations
Framework for security context and VM migration	Migration of security context to ensure security	✓	✓	High	-

Figure 2.2: Comparaison de diverses solutions permettant une migration de VM sécurisée [[2]]

comparatif 2.2 présente diverses solutions garantissant l'utilisation d'images sécurisées de VM. Ce tableau comparatif révèle que la solution *EVDIC* est la plus adéquate. C'est la seule solution permettant une intégrité et une confidentialité des données. En terme de sécurité, il est préférable de protéger les données et de les préserver. On observe que les autres solutions apportent différentes fonctionnalités, mais dans le cadre d'un cloud public, la solution *EVDIC* paraît la plus apte.

Proposed scheme	Handled images type	Privacy	Integrity	Access control	Outdated software detection	Leftover owner's data removal	Malware protection	Scalability	Other features
Mirage, a VM image management system	Dormant	✗	✗	✓	✓	✓	✓	High	Auditability
EVDIC, for VM image's privacy and integrity	Dormant	✓	✓	✓	✗	✗	Dormant images only	Medium	-
A scheme for patch management for VM images	Running and dormant	✗	✗	✗	✓	✗	✗	High	Reports CSP about vulnerable VMs
ImageElves, for patch management for VM images	Running and dormant	✗	✗	✗	✓	✗	✗	Low	Automatic updating of outdated software
OPS-offline, for patch management for VM images	Dormant	✗	✗	✗	✓	✗	✓	Low	Automatic updating of outdated software

Figure 2.3: Comparaison de diverses solutions offrant des images de VM sécurisée [[2]]

2.3 Exécution

Lors de l'exécution d'une VM, cette dernière doit pouvoir offrir des garanties sur la sécurité de son environnement. Premièrement, l'isolation des VMs les unes des autres sur le même matériel (ou *hardware*) physique est obligatoire, car elles sont sujet à des attaques *cross-tenant* ou encore à l'apparition de brèches de données. Durant l'exécution, des actions sont également sensibles à des attaques. Par exemple, des *rollback* sont possibles sur une VM, mais ils peuvent poser des problèmes de sécurité : des autorisations données à certains utilisateurs non autorisés avant *rollback* ou une modification de la configuration sont des exemples de problèmes de sécurité générés par des *rollback*. Le tableau comparatif 2.4 présente diverses solutions permettant de sécuriser l'exécution d'une VM. Dans la continuité de nos propos, la solution *HyperCoffer* est la plus adaptée à sécuriser l'exécution d'une VM. La séparation de la sécurité et du management des VMs couplée au cryptage des données assure une sécurité satisfaisante pour l'exécution d'une VM. Ce cryptage doit rester interne à la VM, car les données échangées dans le réseau virtuel doivent être en claires. Nous reviendrons sur ce

point dans la section 2. Enfin, en plus de fournir une haute scalabilité, elle permet d'effectuer des *rollbacks* sécurisés.

Proposed scheme	Basic theory	Privacy	Integrity	Kernel rootkit	Scalability	Other features
Secure runtime environment for VM	• Cryptography • Access control	✓	✓	✗	Low	Availability
CloudVisor, Secure runtime environment for VM	• Decoupling of security and VM management tasks • Nested virtualization • Trusted computing • cryptography	✓	✓	✗	High	Security of Cloudvisor itself
HyperCoffer, Secure runtime environment for VM	• Decoupling of security and VM management tasks • Trusted computing • cryptography	✓	✓	✗	High	Security against VM rollback
CloudSec, an approach to detect and prevent memory based kernel rootkits	• Bridging of semantic gap between external and internal VMI • Construction of KSD externally	-	-	✓	Low	Live migration of VM in certain situations
Exterior, A dual VM architecture to secure VM execution	• VMI • Use of dual-VM for program execution	✓	-	✓	Low	• Intrusion detection • Removal of malicious code

Figure 2.4: Comparaison de diverses solutions garantissant une exécution sécurisée d'une VM [[2]]

2.4 Sécurité de l'hyperviseur

Si l'Hyperviseur, appelé aussi *Virtual Machine Monitor* (ou VMM) est compromise, toutes les VMs contrôlées par la VMM sont alors sous le contrôle d'une personne malveillante. En effet, une VMM tourne en mode privilégié. Si une VMM est compromise, un utilisateur malveillant est capable de s'attribuer des droits non autorisés. C'est pourquoi, notamment, les données des VMs transitant par la VMM sont en position critique posant des problèmes de confidentialité et d'intégrité. Le tableau comparatif 2.5 présente diverses solutions garantissant la sécurité de l'hyperviseur. Pour ce dernier tableau comparatif, on retiendra *DeHype*. La protection des VMs est assurée. Cette solution sépare également la VMM et le système d'exploitation donnant lieu à un risque d'attaque amoindri provenant du système d'exploitation. Enfin, la réduction des droits privilégiés freine fortement les actions possibles si la VMM est sous le contrôle d'une personne malveillante.

Proposed scheme	Basic theory	VM protection	Scalability	Other feature(s)
HyperCheck, a hardware assisted integrity monitor	• Hypervisor state monitoring through third party • Secure transmission of VMM state	✗	Low	• Data and code integrity • Security against Rootkit DoS, and evasion attacks
DeHype, a technique to reduce hypervisor attack surface	• Least privilege principle • Dependency decoupling between VMM and host OS • Reduction of TCB	✓	Medium	Prevents data leakage from kernel to user space
HyperLock, for isolating hypervisor from host OS	• Shadow hypervisor for every VM • Controlled access to host system • Reduction of TCB	✓	Medium	Exclusion of QEMU from
SplitVisor, for reducing root mode code	• Reduced functionality in root mode • Modern hardware virtualization • Reduction of TCB	✓	Medium	-
NoHype, for virtualization without hypervisor	• Elimination of hypervisor • Pre-allocation of Resources • Use of virtualized I/O only • No indirections	✗	Medium	-

Figure 2.5: Comparaison de diverses solutions garantissant la sécurité de l'hyperviseur [[2]]

3 | Authentification et contrôle d'accès

Comme précédemment, le CSA ([16]) recommande certaines mesures de sécurité. Tout d'abord, l'utilisation de standards libre (OAuth, SAML [16]) est recommandée. Deuxièmement, les sources des attributs doivent être aussi proche possible de la *master* source et les attributs doivent être validés par le *master* ou équivalent. Troisièmement, les caractéristiques des entités doivent avoir un niveau de confiance identifié. Quatrièmement, une confiance bidirectionnelle est souhaitée entre deux entités afin d'avoir une relation et des transactions sécurisées. Enfin, les services doivent avoir la capacité d'exporter/importer avec des standards (*eXtensible Access Control Markup Language* (XACML) [25] [16]).
La solution retenue pour cette gestion d'identité et ce contrôle d'accès est *Hierarchical Attribute-Set-Based Encryption* (HASBE)[23]. Tout d'abord, elle a été retenue, car elle respecte les recommandations proposées par le CSA. De plus, dans l'étude effectuée [2], un tableau comparatif des solutions de gestion d'identité et de contrôle d'accès permet de constater que c'est HASBE qui offre le plus de sécurité tout en offrant une scalabilité performante.
Pour introduire HASBE, il faut évoquer *Attribute Based Encryption* (ABE) [19] qui a été utilisé pour fournir un contrôle d'accès dans un environnement cloud. Les messages encryptés sont associés aux attributs des utilisateurs. Ainsi, pour décrypter un message, il est nécessaire que l'utilisateur possède cet attribut. Puis, une extension de ABE, appelée *Attribute Set Based Encryption* (ASBE) [5] a introduit un arrangement récursif des attributs en les séparant par catégories permettant à l'utilisateur de pouvoir appliquer des contraintes dynamiques sur la manière dont les attributs remplissent la politique de contrôle d'accès. Enfin, HASBE, étant une extension de ASBE, pose une structure hiérarchique des utilisateurs en introduisant des autorités de confiance et des autorités de domaine. La figure 2.6 démontre le fonctionnement de HASBE. L'autorité de confiance administre les autorités de domaine qui gèrent à leur tour des autorités de domaine au niveau suivant ou des utilisateurs du domaine. L'autorité de confiance génère et délivre les paramètres systèmes (permettant de créer des groupes notamment) et la clé racine aux autorités de domaine. Les clés sont générées à l'aide de groupes multiplicatifs bilinéaires et les clés asymétriques sont délivrées aux utilisateurs par les autorités de domaine. Ces clés asymétriques sont des *Tree structure based key* c'est-à-dire des structures d'arborescentes hiérarchiques ou chaque élément étant un attribut ou un ensemble d'attribut. Le contrôle d'accès est défini comme une arborescence hiérarchique et il est assuré uniquement pour les données dans le cloud. Enfin, les données sont cryptées avec une clé de cryptage, clé protégée avec le HASBE utilisant la structure d'accès des clés qui spécifie les politiques et attributs de contrôle d'accès. L'accès au déchiffrement est possible pour les utilisateurs possédant les attributs et les accès dans la structure des clés d'accès. HASBE a été retenue pour couvrir la sécurité de l'authentification et du contrôle d'accès. HASBE fournit un chiffrement par attributs, un accès aux données contrôlée et une mise en place d'une hiérarchie de confiance. Supportant une haute scalabilité, HASBE se voit être la meilleure solution pour répondre à l'hypothèse d'authentification sécurisée de plusieurs acteurs d'un même système.

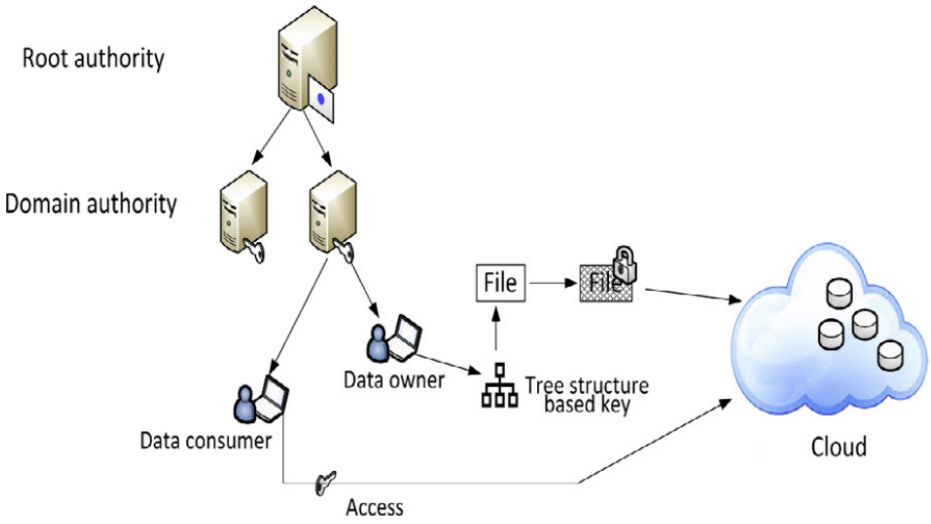

Figure 2.6: Fonctionnement de HASBE[[2]]

CHAPTER 3

COMMUNICATIONS CLOUD

Pour commencer, il est primordial de rappeler que la plupart du temps, les services du cloud sont accessibles par les utilisateurs via Internet [31]. Le protocole IP et ses mécanismes sont utilisés pour permettre la communication entre l'utilisateur et le cloud [20] afin de délivrer les services du cloud et assurer l'échange de données.

1 | Communications inter-cloud

Les communications inter-cloud sont similaires aux autres communications sur Internet et posent les mêmes problématiques en terme de sécurité. Ces problèmes de sécurité interviennent au plus haut de la couche à la plus basse de la pile TCP/IP et chacune ajoute de l'information dans les paquets envoyés. Heureusement, des solutions existent et la sécurité est présente dans plusieurs couches du modèle TCP/IP. Toutefois, il n'est pas toujours nécessaire de crypter et sécuriser chaque couche.

Couche application : utilise des mécanismes de sécurité pour une application donnée, séparés des autres couches. *Pretty Good Privacy* (PGP) [7] est un exemple d'utilisation sur la couche application. L'envoi de courrier électronique sécurisés est possible grâce aux fonctionnalités de PGP : chiffrer des textes et les signer.

Couche transport : des contrôles de sécurité permettent de protéger les données en transit entre deux hôtes distants. *Transport Layer Security* [8] et *Secure Sockets Layer* [10] sont des protocoles cryptographiques permettent de sécuriser les échanges sur Internet via la couche transport. Ils permettent l'authentification de l'hôte distant et la confidentialité et l'intégrité des données échangées. Des modifications peuvent être nécessaire à certaines applications pour utiliser ces protocoles.

Couche réseau : toutes les communications entre deux hôtes ou avec le réseau peuvent être protégées avec cette couche sans modification spécifique. IPSec [22] est un ensemble de protocoles utilisant des algorithmes garantissant des échanges sécurisés en authentifiant et chiffrant chaque paquet entre deux hôtes distants.

La sécurité au niveau de la couche réseau est la solution la plus souvent utilisée pour protéger l'ensemble des applications et les informations IP. IPSec est utilisé majoritairement pour fournir un *Virtual Private Networking* (VPN) [9], plus précisément un VPN site-à-site. Un VPN est un réseau privé virtuel, construit sur un réseau physique existant et fournissant un lien direct entre deux hôtes distants et offrant un échange de données sécurisé. En effet, en isolant le trafic entre les deux hôtes distants, un VPN facilite la sécurité de

la communication sur des réseaux publics. Un VPN site-à-site permet d'établir plusieurs connexions sécurisés avec plusieurs hôtes situés à différentes locations entre eux afin de former un groupe d'utilisateurs fermés, appelé *Closed User Group* (CUG). Aujourd'hui, via le service IPSecVPN, les connexions inter-cloud permettent d'offrir des garanties de sécurité satisfaisantes. Cependant, le nombre d'utilisateurs se connectant aux CSP va augmenter considérablement et vont, sans doute, utiliser plusieurs services de cloud différents. Ainsi, le nombre de réseaux clouds connectés via IPSec VPN va augmenter considérablement également. Il est nécessaire donc de proposer une solution durable et performante basé sur IPSecVPN.

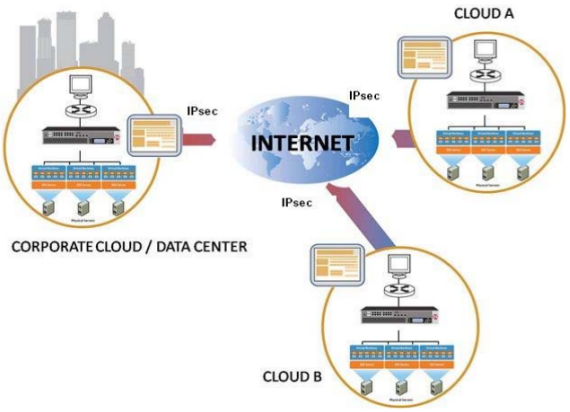

Figure 3.1: Architecture général des communications inter-cloud [13]

Dayananda M. S. et Ashwin K. proposent une architecture pour les communications inter-cloud utilisant IPSecVPN [13]. Il existe deux types d'architecture IPSecVPN principales : *Full Mesh* et *Hub-and-Spoke*. L'architecture *Full Mesh* se caractérise par une connections de tous les réseaux entre eux et les réseaux vont communiqués en utilisant IPSec. L'architecture *Hub-and-Spoke* est le modèle le plus utilisé avec un *hub* placé entre tous les réseaux et les connectant. Un seul lien IPSec est nécessaire, c'est le lien entre le réseau et le *hub*. Les deux architectures présentent chacune des problèmes de performance, par exemple, lorsque le nombre de CUG augmente ou encore lorsque le trafic entre les CUG augmente. Pour résoudre le problème de performance, il se base sur l'architecture textitHub-and-Spoke afin de mettre en place un système de IPSecVPN qui créer dynamiquement les tunnels entre les différents clouds en fonction de la demande.
L'architecture proposée permet tout d'abord d'établir des connexions IPSec dynamiques via des accès VPN multi-points en combinant le chiffrement d'IPSec, des tunnels *Generic Routing Encapsulation* (*Generic Routing Encapsulation*) et le protocole *Next Hop Resolution* (*Next Hop Resolution Protocol*). Il est nécessaire que chaque routeur (autre que le *Hub*) possède une interface point-à-point afin de pouvoir créer le tunnel vers le *Hub*. On oblige alors tout le trafic entre les routeurs, appelé *Spoke*, a transité par le *Hub*. De plus, si un *Spoke* transmet des données, le *Hub* agit alors comme serveur NHRP afin de déterminer dynamiquement l'adresse de destination. Il est à noter que lorsque le *Hub* agit comme serveur NHRP, il permet à GRE de configurer et déterminer les adresses de destination les plus courtes des autres pairs. Les deux routeurs, source et destination, vont alors lancer IPSec entre eux et ainsi créer dynamiquement un tunnel GRE point-à-point, mais également commencer les négociations des sessions IPSec directement sans configuration traditionnelle (gain de temps considérable). Ce tunnel sera automatiquement tombé après une certaine période d'activité. Ensuite, autre avantage de ce type de configuration est que les *Spoke* peuvent utiliser une adresse IP dynamique, car NHRP permet de déterminer l'adresse IP de l'interface du *Spoke* distant via le routeur *Hub*. Enfin, dernier avantage, la configuration du *Hub* est grandement simplifiée puisqu'elle n'a pas besoin de configurer les informations GRE ou IPSec des *Spoke*, car elles sont apprises dynamiquement via NHRP. De plus, la configuration permet d'ajouter un nouveau *Spoke* sans configuration supplémentaire, car ce dernier va s'enregistrer auprès du *Hub*

dynamiquement et les différentes informations du nouveau *Spoke* seront propagées vers les autres pairs et inversement. Pour finir, l'architecture proposée, sur la figure 3.2, apporte également une autre fonctionnalité : lorsqu'une nouvelle connexion IPSec/IKE est établie entre deux *Spoke*, les règles de filtrage de *Hub* ne sont pas appliqués entre eux. L'utilisation d'une extension du *Traffic Selector* (TS) permet de placer les différents filtres dans la création de la session IPSec/IKE afin d'établir les règles de filtrage dans l'échange IPSec/IKE entre les deux *Spoke*.

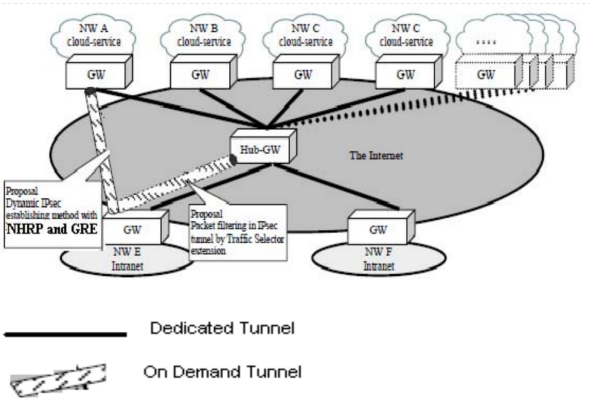

Figure 3.2: Architecture proposé pour les communications inter-cloud [13]

En conclusion, l'architecture proposé, appelé Dynamic Multipoint VPN, permet aux utilisateurs (ici, les fournisseurs de cloud) de mieux adapter les tunnels IPSecVPN en combinant le chiffrement d'IPSec, des tunnels GRE (*Generic Routing Encapsulation*) et le protocole NHRP. Les différents problèmes rencontrés dans les autres topologies IPSec VPN, Full-Mesh et Hub-and-Spoke, sont résolues avec les solutions proposées par cette architecture (voir 3.3).

Requirements	Current architecture		Proposed architecture
	Full-Mesh IPsecVPN	Hub-and-Spoke IPsecVPN	
1. Numbers of CUG members		✓	✓
2. Traffic between CUG members	✓		✓
3. Addition of CUG members		✓	✓
4. Minimum delay	✓		✓
5. Traffic control		✓	✓

✓ indicates architecture meets the requirement

Figure 3.3: Comparaison des architectures IPSec VPN [13]

Le cloud implique de nouveaux problèmes du à l'utilisation de l'architecture *multi-tenant*. Une VM peut être corrompue et peut agir de manière malveillante. Un hôte interne du réseau est alors infecté et cible alors les machines du cloud. On parle alors d'une attaque interne impliquant les communications intra-cloud. La deuxième partie du problème est donc de pouvoir sécuriser le cloud de possibles attaques internes. La démonstration de ce type d'attaque a été démontrée lors de la conférence DefCon 18 par Bryan et Anderson ([6]). Une attaque DDoS a été mise en place depuis le cloud EC2 d'Amazon et ils ciblaient une autre machine virtuelle, sur le même réseau.

2 | Communications intra-cloud

Aujourd'hui, les solutions actuelles, mises en place dans les cloud pour le sécuriser, utilisent des firewalls, des IDS et des IPS. Dans cette partie, nous allons tenter de proposer une solution performante en se basant sur le papier de Mazhar et al [2]. De plus, on ajoutera que le CSA ([16]) recommande l'utilisation de combinaisons de différents *Local Area Network* virtuels, d'IDS, d'IPS et d'un firewall afin de protéger les données en transit dans le cloud.

2.1 Solutions NIDS pour le cloud

2.1.1 SnortFlow

SnortFlow [29] est une solution proposée visant à améliorer les performances de *Snort*. *Snort* est un NIDS (Network IDS) orienté IDS/IPS : ce sont des solutions permettant de surveiller en temps réel un système et de détecter des activités suspectes violant la politique de sécurité du système. *Snort* permet d'analyser le trafic en temps réel et enregistrer les paquets (log) au-dessus du protocole IP. Mais, les services proposées ne s'arrêtent pas là : recherche et correspondance de contenu, inhiber les prise d'empreinte de la pile TCP/IP, dépassement de *buffer*, scans, attaque CGI (Common Gateway Interface), sondes SMB (Server Message Block). *Snort* propose trois modes d'action du NIDS : le mode *Sniffer* lit les paquets et les affiche, le mode *Packet Logger* permet d'obtenir une log des paquets sur le disque et enfin, le mode *Network Intrusion Detection System* qui gère le trafic et l'analyse avec les règles fixées par l'administrateur. Toutefois, le trafic ne doit pas être crypté, sinon le NIDS n'est plus capable de l'analyser. Le fonctionnement de *Snortflow* se base sur *Snort* en ajoutant les fonctionnalités d'*OpenFlow*. *OpenFlow* présente une nouvelle architecture pour fournir un environnement de réseau virtuel. L'idée sous-jacente est la séparation physique des plans de contrôle et de données. C'est la raison pour laquelle différents éléments exécutent la procédure de transfert de paquets (plan de données) et la procédure de contrôle de réseau (plan de contrôle). Le transfert est effectué grâce à une table de transfert partagée qui représente le plan de données, tandis que tous les plans de contrôle sont centralisées dans un nœud appelé contrôleur ou *Controller*, faisant fonctionner les applications qui contrôlent le réseau virtuel. Ce contrôleur est un élément central du réseau. Il peut communiquer avec tous les nœuds pour configurer les tables de flux. Il travaille comme une interface entre les applications du réseau et les nœuds de transfert. Dans *SnortFlow*, *OpenFlow* est associé à POX. Chaque plan de contrôle est composé d'un jeu d'application tournant sur POX. C'est la raison pour laquelle, dans *OpenFlow*, un réseau virtuel est défini par son plan de contrôle, qui est un jeu d'applications fonctionnant sur le contrôleur, et par ses flux associés, provenant du *Cloud Cluster* (voir 3.4).

De plus, *SnortFlow* se repose sur *glsXenServer*. Xen est un hyperviseur ou *Virtual Machine Monitor* qui fonctionne sur des plate-formes matérielles standards. En plus du VMM, situé lui sur le matériel physique, l'architecture Xen est composée de plusieurs domaines tournant de manière simultanée sur l'hyperviseur, appelés machines virtuelles. Chaque machine virtuelle peut avoir son propre système d'exploitation et ses applications. Le VMM contrôle l'accès au matériel des domaines multiples et gère le partage des ressources des différents domaines. Il existe deux types de domaines : Le domaine 0, abrégé Dom0, est le domaine de gestion appartenant au domaine administratif du cloud. Le domaine, abrégé DomU, est le domaine hôte des VMs des utilisateurs. Le Dom0 a des privilèges spéciaux comparé aux DomU et dispose, par exemple, d'un accès total au matériel de la machine physique. Les DomU possèdent des pilotes virtuels et opèrent comme s'ils pouvaient directement accéder au matériel. Cependant, ces pilotes virtuels communiquent avec le Dom0 afin d'avoir accès au matériel physique. Donc, toutes les ressources de DomU sont gérés par Dom0, mais c'est

OpenFlow Switch (OFS) qui permet l'interconnexion les ressources sur différents serveurs clouds dans *SnortFlow*. La communication entre les VMs doit donc obligatoirement passer par l'OVS, donc toutes les communications entre VMs sont analysés par le *SnortFlow Agent* comme on peut le constater sur la figure 3.4. Le composant permettant de gérer et vérifier la sécurité du réseau est la partie *SnortFlow Server*, car les communications analysées par *SnortFlow Agent* envoient des alertes lorsqu'un incident de sécurité apparaît tel qu'une violation de la politique de sécurité ou des données douteuses sont échangées. *SnortFlow daemon* permet de collecter toutes les informations des alertes envoyées du *SnortFlow Agent* et les envoyées à l'*alert interpreter*. Ce dernier va analyser toutes les alertes et cibler les trafics suspects repérés. Le *rules generator* va alors générer les règles qui vont être injectés dans le dispositif *OpenFlow* où les nouvelles règles seront appliquées au réseau afin de ne plus laisser passer les trafics suspects repérés auparavant. Ces règles seront injectées dans une table appelés *Flow Table*. Cependant, les faux positifs présents dans l'*alert interpreter* sont nombreux et l'un des défis de *SnortFlow* et des IDS en général est de minimiser au maximum l'apparition de ces faux positifs. Pour terminer, *SnortFlow* propose une dernière fonctionnalité, appelé *Network Reconfiguration*. Cette fonctionnalité va finaliser le processus des systèmes IPS. Elle octroie la possibilité de reconfigurer les caractéristiques du réseau tels que les paramètres de qualité de service ou encore la topologie. En ajoutant SDN au système des réseaux virtuels du cloud, cette fonctionnalité permet notamment d'appliquer des changements pour construire les contre-mesures des systèmes IPS dans les réseaux virtuels. En conclusion, lorsqu'une anomalie est détecté, le cloud va se défendre et mettre à jour ses données en enclenchant différents mécanismes.

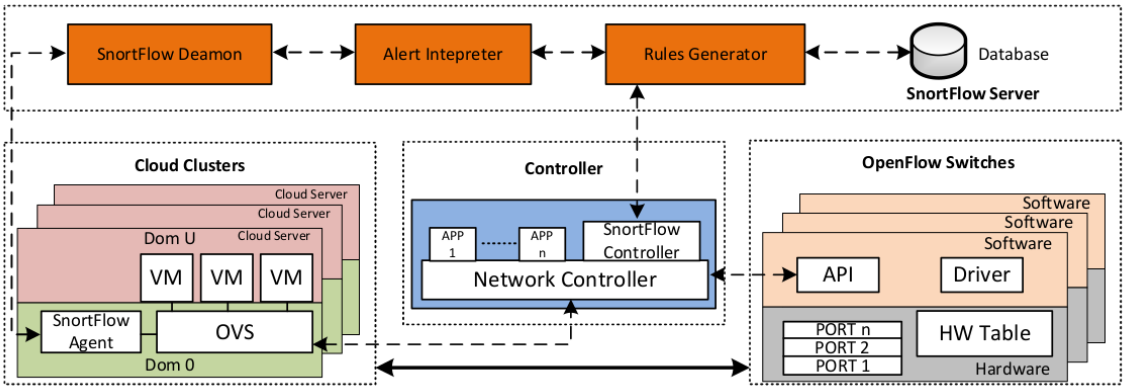

Figure 3.4: Schéma fonctionnel de Snortflow [[29]]

2.1.2 SDNIPS

SDNIPS [30] est une solution reprenant tous les concepts de *SnortFlow* en tentant de répondre à certains paramètres non présents, d'après les auteurs de SDNIPS, dans *SnortFlow* tels que : l'établissement d'un IDS efficace basé sur SDN et d'une architecture réseau permettant aux mécanismes défensifs des IDS/IPS d'être plus efficace. En effet, le fonctionnement de *SnortFlow* est repris, ainsi que la majorité des composants utilisés (*OpenFlow*, *XenServer*), comme on peut le constater sur la figure 3.5.

La principale différence entre les deux NIDS se traduit par une différence d'architecture. La figure 3.6 permet de visualiser l'architecture proposée par SDNIPS. *Open vSwitch* est le *switch OpenFlow* et se compose de deux éléments dans l'espace utilisateur : *ovsdb-server* et *ovs-switchd*. Une des principales différences proposées est le déploiement direct de *Snort* dans Dom0. Dom0 permet la détection et le traitement des paquets. En injectant *Snort* dans Dom0, la détection des chemins virtuels vers les *Virtual Interface* (VIF) s'effectue

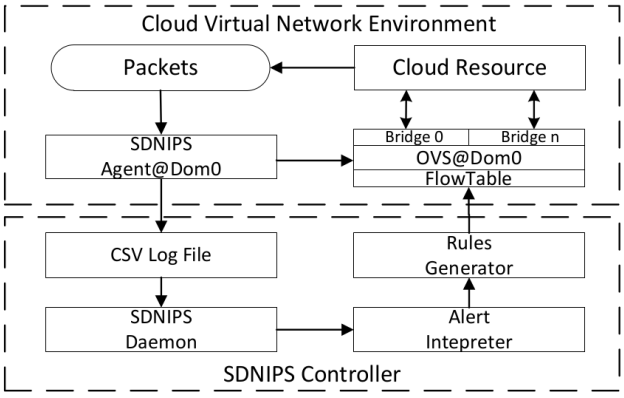

Figure 3.5: Fonctionnement de SDNIPS [30]

nativement dans OVS et offre une meilleure performance, car *Snort* analyse et gère tous les réseaux virtuels dans le domaine privilégié.

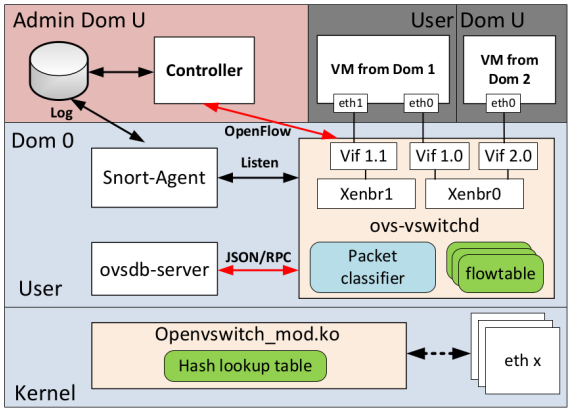

Figure 3.6: Architecture de SDNIPS [30]

2.1.3 Comparaison de Snortflow et SDNIPS

SnortFlow et SDNIPS sont tous les deux des systèmes IPS/IDS basés sur SDN générant dynamiquement des contre-mesures contre des attaques sur le réseau. Cependant, la faiblesse du papier de *SnortFlow* est son manque d'évaluation et de comparaison. Les comparaisons effectuées n'apportent pas d'informations supplémentaires sur l'efficacité de *SnortFlow*. Au contraire, SDNIPS apporte une comparaison entre le NIDS proposé et l'IPS "traditionnel" (*Snort*/IPTables). Pour cette comparaison, ils ont inondé le réseau d'attaque DoS et ICMP. Il apparaît que SDNIPS détecte tous les paquets malveillants jusqu'à une limite de 15 000 paquets par seconde. L'IPS "traditionnel", lui, ne permet pas de détecter tous les paquets pendant une courte période et les auteurs notent que seuls 13,72% des paquets ICMP sont détectés comme malveillants. On constate donc une différence importante de performance entre les deux outils. SDNIPS renforce la sécurité du cloud en détectant les anomalies, qu'elles soient nombreuses ou non. Ce type de comparaison est absente du papier présentant *SnortFlow*. En conclusion, les deux papiers présentent des solutions similaires, mais l'absence d'une réelle évaluation de *SnortFlow* justifie la préférence de mettre en avant SDNIPS comme NIDS dans ce rapport.

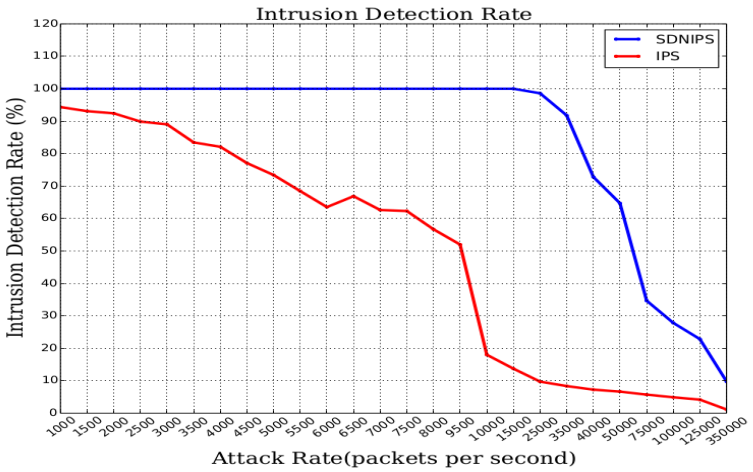

Figure 3.7: Évaluation du taux de détection d'intrusion entre un Snort/IPTable et SDNIPS [30]

2.2 Firewall avec arbre de règle

Pour conclure sur la sécurité des échanges intra-cloud, on ne pourra pas se passer d'un *firewall*. La solution proposée par Xiangjian He et al. propose un *firewall* sécurisé et performant. Il apparaît différent des *firewalls* traditionnels, car il fonctionne via un arbre de règles et non pas une liste de règles (respectivement, *Tree-rule* et *Listed-rule*). Il propose un arbre et un ensemble de règles hiérarchiques, apparu dans les manuels de *Cisco Services Switch* (CSS). Les *Listed-rule firewall* apparaissent limité, car ce type de *firewall* pose des problèmes de performance, de sécurité, mais également une difficulté à exploiter correctement ce type de *firewall*. Dans ce papier, Xiangjian He et al. nous démontrent les différentes failles d'un *Listed-rule firewall*. En premier lieu, la présence de *shadowed rule* qui est l'une des grandes failles, car c'est une règle qui ne correspond à aucune entrée et pouvant créer ainsi des problèmes de sécurité et de performance. Autre inconvénient, les changement des positions des règles (dans le but d'affecter une règle plutôt qu'une autre pour un paquet) peut engendrer la modification de la politique de sécurité du *firewall* et des failles de sécurité peuvent apparaître. De plus, des règles redondantes sont possibles et peuvent impacter la performance d'un *firewall*. Un autre point négatif est la configuration d'un *Listed-rule firewall*. En effet, un administrateur réseau a le devoir de localiser les plus grosses adresses réseaux aux plus petites (*difficult to use*). Et enfin, les recherches séquentielles (ici recherche séquentielle des règles) pose des problèmes de rapidité et de performance.

La figure 3.8 permet de visualiser la forme du *firewall* présenté dans le papier de Xiangjian He et al. Le *firewall* lit les données contenues dans l'en-tête des paquets et va comparer les informations du paquet avec les règles des différents nœuds. On peut constater, sur la figure 3.8, que à chaque nœud, des attributs différents sont comparés (Destination IP, Destination Port, Source IP). On peut donc spécifier une action spécifique pour certains paquets ou certains utilisateurs. De plus, les attributs présentés sont des exemples, on peut également travailler avec d'autres attributs ou bien même interchanger la place des différents attributs, c'est-à-dire placer la *Destination Port* dans le premier nœud et la *Destination IP* dans le deuxième nœud. Ce *Tree-rule firewall* résout tous les problèmes posés par un *Listed-rule firewall*. En effet, le *firewall* proposé ne pose plus de problèmes de sécurité : les utilisateurs n'ont plus besoin d'effectuer des changements de position des règles et les *shadowed rule* et les règles redondantes n'existent pas dans ce type de *firewall*. Les paquets reçus passent les différents nœuds et suivent un chemin. Si aucun chemin n'est trouvé pour un paquet reçu, c'est-à-dire qu'il ne *match* pas avec une règle contenue dans les nœuds pour un certain attribut, il est alors supprimé(*Deny*). Sinon, on affecte une action spécifique (*Accept*).

Ensuite, l'exploitation et la configuration d'un *Tree-rule firewall* est facile, il suffit d'intégrer les différentes informations dans les différentes colonnes des nœuds. Enfin, la complexité d'un arbre de règle est bien moins coûteuse qu'une liste de règles. Selon les auteurs, cette comparaison revient à comparer un arbre binaire de recherche et une recherche linéaire.

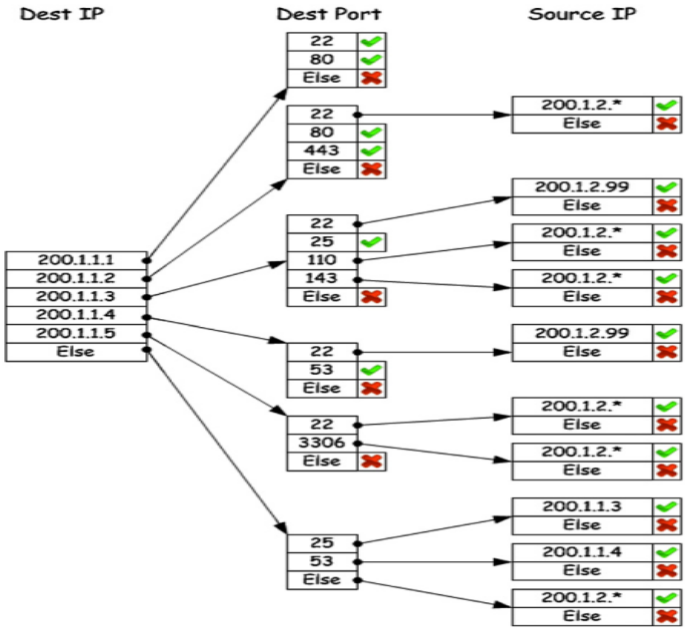

Figure 3.8: Structure d'un *Tree-rule firewall* [11]

De plus, IPTables apporte la plupart des fonctionnalités de base d'un *firewall* (filtrage dynamique, translation de port et d'adresse, filtrage au niveau 2), mais lorsque le nombre de règles commence à être très grand, IPTables n'est plus capable d'apporter des performances satisfaisantes : moins de la moitié des règles sont atteintes avec IPTables quand le nombre de règles est de 5000. De plus, le CPU approche les 100% d'utilisation. *Tree-rule firewall* apporte toujours une excellente performance, malgré un nombre de règle important. Enfin, le CPU n'atteint même pas les 5% d'utilisation. On peut donc affirmer que les performances du *Tree-rule firewall* sont largement supérieur à IPTables.
Pour conclure sur ce *Tree-rule firewall*, les auteurs proposent l'utilisation du *firewall* sur un environnement cloud et proposent une comparaison entre différents *firewall*. On constate via la figure 3.9 que le *Tree-rule firewall* apporte des performances supplémentaires par rapport aux autres *firewall*. La sécurité et les performances apportées par le *Tree-rule firewall* sont importantes et vérifiées par les différentes comparaisons effectuées dans ce papier.

	ESXi firewall	5Nine vFirewall	IPTABLES	Tree-Rule firewall
Can protect hypervisor	Yes	Yes	Yes	Yes
Can protect internal VMs	No	Yes	Yes	Yes
Can prevent attacking between VMs	No	Yes	Yes	Yes
Stable	Yes	No	Yes	Yes
Fast packet decision	No	No	No	Yes
Support more than 5000 rules	No	No	No	Yes
No shadowed/redundant/confliction rules	No	No	No	Yes
No down time for adding new VMs	No	Yes (only on Hyper-V)	No	No

Figure 3.9: Comparaison des fonctionnalités de différents*firewall* dans un environnement cloud [11]

CHAPTER 4

CONCLUSION

Le but de ce chapitre est de résumer les contributions de ce Travail d'Étude et de Recherche, à savoir une classification des différentes catégories de problèmes de sécurité dans le cloud informatique, une étude des différentes solutions de la littérature, et une analyse des points qui pourraient être améliorés. La section 1 résume les différentes contributions de ce document, et la section 2 énumère les potentielles contributions futures, comme par exemple les points faibles des architectures actuelles.

1 | Contributions

Dans ce sujet du Travail d'Étude et de Recherche, nous nous sommes intéressés à la sécurité de systèmes de *workflow*. Nous nous sommes limités à un cas d'étude particulier : le cloud computing. Pour sécuriser ce système, il est nécessaire d'établir la surveillance globale du trafic, afin de détecter l'ensemble des attaques perpétrées, y compris les attaques internes à la structure. Les solutions de sécurité traditionnelles réalisent généralement leurs analyses en ne considérant que le trafic qui transite localement. De même, les traitements réalisés par ces équipements estiment que les attaques proviennent exclusivement de l'extérieur du réseau. Cette vision n'est malheureusement plus applicable pour des réseaux comme le cloud, où un utilisateur malicieux peut facilement obtenir des ressources puissantes, et à bas coût, pour réaliser des attaques provenant du réseau interne. Cependant, le cloud ne pose pas que des problèmes de sécurité au niveau du réseau. Le stockage et la virtualisation posent également de nouveaux problèmes de sécurité.

Pour palier à ces problèmes, ce document propose différentes pistes à établir dans un cloud dans le but d'améliorer la sécurité du réseau principalement, mais également au niveau de la virtualisation et du stockage. Tout d'abord, nous avons proposé deux solutions différentes pour améliorer le stockage dans le cloud : FADE et SecCloud. Ensuite, nous avons évoqué différentes pistes pour sécuriser la virtualisation et permettre son fonctionnement. Par la suite, une solution a été proposée pour permettre une authentification sécurisée et un contrôle d'accès efficace. Puis, nous nous sommes intéressés au réseau, plus précisément aux communications inter-cloud et intra-cloud. Une architecture DMVPN (*Dynamic Multipoint VPN*) garantit l'utilisation d'IPSecVPN et résout les différents problèmes posés par les architectures inter-cloud actuelles. La partie intra-cloud se compose par l'utilisation d'un NIDS et d'un *firewall*. Deux NIDS sont proposés, SnortFlow et SDNIPS. Cependant, seul SDNIPS propose une véritable application et différentes comparaisons prouvant l'efficacité du NIDS dans un environnement cloud. Enfin, le *Tree-rule firewall* est mis en avant. Ce dernier se différencie des *firewalls* classiques, appelés *Listed-rule firewall*, car il résout tous les problèmes de performance et de sécurité associés aux *Listed-rule firewall*.

2 | Perspectives

Tout d'abord, les premières perspectives envisagées par ce document sont l'établissement et l'intégration des pistes étudiées dans un environnement cloud, plus précisément dans un cloud public. De plus, les différents pistes abordées présentent également des défauts et peuvent faire l'objet d'évolutions. Par exemple, le *Tree-rule firewall* ne traite pas encore IPv6, NAT (*Network Address Translation*) ou encore les VPN. Si l'intégration du VPN s'effectue, il peut paraître très pertinent d'associer l'architecture inter-cloud proposée avec des règles de *firewall*.

Malheureusement, le cloud évolue sur plusieurs nouvelles couches réseaux et des nouvelles technologies continuent d'apparaître et d'évoluer à une vitesse vertigineuse. Il existe un vrai décalage entre la sécurité et les technologies actuelles ce qui implique un décalage entre les menaces actuelles de sécurité et les solutions et le périmètre de sécurité établis.

GLOSSARY

Black Hats Spécialiste en informatique qui recherche différents moyens de contourner la sécurité mise en place au niveau logiciel ou matériel dans le but de nuire à autrui, faire du profit ou encore obtenir des informations. La plupart des actions effectués par ces spécialistes sont illégales.. 1

Generic Routing Encapsulation Protocole de mise en tunnel entre deux réseaux qui permet d'encapsuler n'importe quel paquet de la couche réseau. [26]. 12, 13

Local Area Network Réseau informatique restreint où les différents acteurs s'échangent les données au niveau de la couche liaison, c'est-à-dire sans accèder à Internet.. 14

Next Hop Resolution Protocol Protocole permettant à une source d'envoyer des données à une destination en utilisant et apprenant les routes les plus directes permettant de joindre cette destination. [27]. 12

Role-Based Access Control concept de sécurité du réseau selon lequel le réseau accorde des droits aux utilisateurs en fonction de leur rôle dans l'entreprise. 6

Service Level Agreement Contrat définissant une qualité de service, prestation prescrite entre un fournisseur de service et un client.. 1

Trusted third party Entité facilitant les interactions entre deux parties qui ont tous les deux confiance en cette entité.. 5

VM hopping Attaque visant à prendre le contrôle d'une machine virtuelle et par la suite, tenter de prendre le contrôle d'une autre. Si plusieurs machines virtuelles sont présentes sur le même hôte, alors elles sont des cibles de l'attaque.. 6

Virtual Machine Une machine virtuel (Virtual Machine en anglais, abrégé VM) est un environnement d'applications ou un système d'epxloitation installé par un logiciel ou instancée par un hyperviseur permettant l'émulation d'un matériel dédié.. 2

eXtensible Access Control Markup Language **(XACML)** Standard pour définir les contrôles d'accès et les autorisations.. 8

firewalls Un pare-feu (ou firewall en anglais) est un ensemble de composants placé entre deux réseaux filtrant l'ensemble du trafic provenant du réseau externe vers le réseau interne et réciproquement. De plus, en fonction de la politique de sécurité mise en place, un firewall filtre également le trafic non autorisé et il doit être impénétrable et inattaquable. [3] . 2

Hyperviseur Outil permettant à plusieurs systèmes d'exploitation de tourner sur la même machine physique en même temps.. 7

IDS Un *Intrusion Detection System* est un dispotif ou une application qui surveille en temps réel un réseau ou un(des) système(s) d'activités malicieuses ou d'une violation de la politique de sécurité mise en place. Équipé d'un système d'alarme, ils ont pour but de prévenir l'utilisateur des activités malicieuses et d'offrir des informations sur ces dernières via un système de log. . 2

IPS Un *Intrusion Prevention System* est un dispotif ou une application ayant les mêmes fonctionnalités qu'un IDS, mais peut mettre en place des contre-mesures afin de stopper une activité malveillante dans un délai court. . 2

OpenFlow OpenFlow est un protocole de communication qui propose une architetcure Software-defined networking (SDN). . 14

POX POX is an open source development platform for Python-based software-defined networking (SDN) control applications, such as OpenFlow SDN controllers. [18]. 14

prise d'empreinte Une prise d'empreinte de la pile TCP/IP est un procédé permettant de déterminer l'identité du système d'exploitation utilisé sur une machine distante en analysant les paquets provenant de cet hôte. [28]. 14

TCP/IP Architecture réseau en 4 couches dans laquelle les protocoles TCP et IP jouent un rôle prédominant.. 11

BIBLIOGRAPHY

[1] Open Web Application Security Project Top 10-2013. The ten most critical web application security risks. `https://www.owasp.org/index.php/Top10/OWASP>`, Mis en ligne en 2013, consulté le 03 Mars 2019.

[2] Mazhar Ali, Samee U. Khan, and Athanasios V. Vasilakos. Security in cloud computing: Opportunities and challenges. *Information Sciences*, 305:357–383, June 2015.

[3] S.M. Bellovin and W.R. Cheswick. Network firewalls. *Communications Magazine*, 32:50–57, September 1994.

[4] Bitweasil. Cryptohaze cloud cracking. In *Defcon 20*, 2012.

[5] R. Bobba, H. Khurana, and M. Prabhakaran. Attribute-sets: a practically motivated enhancement to attribute-based encryption. page 587–604, 2009.

[6] David Bryan and Michael Anderson. Cloud computing : A weapon of mass destruction ? In *DEFCON 18*, 2010.

[7] J. Callas, L. Donnerhacke, H. Finney, D. Shaw, and R. Thayer. *OpenPGP message format*, November 2007.

[8] T. Dierks and E. Rescorla. *The Transport Layer Security (TLS) Protocol*, August 2008.

[9] Sheila Frankel, Karen Kent, Ryan Lewkowski, Angela Orebaugh, Ronald Ritchey, and Steven Sharma. *Guide to IPsec VPNs*, December 2005.

[10] A.O. Freier, P. Karlton, and P.C. Kocher. *The Secure Sockets Layer (SSL) Protocol Version 3.0*, August 2011.

[11] X. He, T. Chomsiri, P. Nanda, and Z. Tan. Improving cloud network security using the tree-rule firewall. *Future Generation Computer Systems*, 30:116–126, 2014.

[12] M. Balachandra Reddy Kandukuri, Ramakrishna Paturi V, and Dr. Atanu Rakshit. Cloud security issues. *IEEE International Conference on Services Computing*, pages 517– 520, September 2009.

[13] Dayananda S. M. and Kumar A. Architecture for inter-cloud services using ipsec vpn. *Proceedings of Advanced Computing and Communication Technologies (ACCT)*, pages 463–467, 2012.

[14] Peter Mell and Timothy Grance. The nist definition of cloud computing. September 2011.

[15] Haroon Merr, Marco Slaviero, and Nicholas Arvanitis. Clobbering the cloud ! In *Defcon 17*, 2009.

[16] Rich Mogull, James Arlen, Francoise Gilbert, Adrian Lane, David Mortman, Gunnar Peterson, and Mike Rothman. Security guidelines for critical areas of focus in cloud computing v4.0. 2017.

[17] Kaustubh Pande. Is cross-tenant cloud attack a legitimate threat? `https://www.mindtree.com/blog/cross-tenant-cloud-attack-legitimate-threat`. Dernière modification du site le 16 Juillet 2018, consulté le 5 Mars 2019.

[18] Margaret Rousse. Pox. `https://searchnetworking.techtarget.com/definition/POX`. Dernière modification du site en Mars 2013, consulté le 27 Avril 2019.

[19] A. Sahai and B. Waters. Fuzzy identity-based encryption. page 457–473, September 2005.

[20] S. Subashini and V. Kavitha. A survey on security issues in service delivery models of cloud computing. *J. Netw. Comput. Appl.*, 34:1–11, 2011.

[21] Y. Tang, P.P. Lee, J.C.S. Lui, and R. Perlman. Secure overlay cloud storage with access control and assured deletion. *Transactions on Dependable and Secure Computing*, 9:903–916, 2012.

[22] R. Thayer, N. Doraswamy, and R. Glenn. *IP Security Document Roadmap*, November 1998.

[23] Z. Wan, J. Liu, and R.H. Deng. Hasbe: a hierarchical attribute-based solution for flexible and scalable access control in cloud computing. *Transactions on Information Forensics and Security*, 7:743–754, 2012.

[24] L. Wei, H. Zhu, Z. Cao, X. Dong, W. Jia, Y. Chen, and A.V. Vasilakos. Security and privacy for storage and computation in cloud computing. *Informations Sciences*, 258:371–386, 2014.

[25] Wikipedia. extensible access control markup language (xacml). `https://en.wikipedia.org/wiki/XACML`. Dernière modification du site le 14 Mars 2019, consulté le 27 Avril 2019.

[26] Wikipedia. Generic routing encapsulation. `https://en.wikipedia.org/wiki/Generic_Routing_Encapsulation`. Dernière modification du site le 15 Mars 2019, consulté le 12 Mars 2019.

[27] Wikipedia. Next hop resolution protocol. `https://en.wikipedia.org/wiki/Next_Hop_Resolution_Protocol`. Dernière modification du site le 7 Juin 2018, consulté le 12 Mars 2019.

[28] Wikipedia. Tcp ip stack fingerprinting. `https://en.wikipedia.org/wiki/TCP/IP_stack_fingerprinting`. Dernière modification du site le 18 Février 2019, consulté le 07 Avril 2019.

[29] T. Xing, D. Huang, L. Xu, C. Chung, and P. Khatkar. Snortflow: a openflow-based intrusion prevention system in cloud environment. page pp. 89–92, 2013.

[30] T. Xing, Z. Xiong, D. Huang, and D. Medhi. Sdnips: Enabling software-defined networking based intrusion prevention system in clouds. *International Conference on Network and Service Management (CNSM) and Workshop*, page 308–311, November 2014.

[31] Y.A. Younis, M. Merabti, and K. Kifayat. Secure cloud computing for critical infrastructure: a survey. 2013. Technical Report, Liverpool John Moores University, United Kingdom, Tech. Rep. ISBN: 978-1-902560-27-4.