Ministère de l'Enseignement Supérieur de la Recherche et de l'Innovation

Université Assane SECK de Ziguinchor
UFR Sciences et Technologies
Département Informatique

Mémoire de fin d'études

Pour l'obtention du diplôme de Master
Mention : Informatique ; Spécialité : Génie Logiciel

Sujet :

SDN : Etat de l'art et Perspectives pour une solution contre le DoS

Présenté par: M. Amadou Diouhé DIALLO

Soutenance le 04/07/2020

Sous la direction de : Dr Youssou FAYE et Mme Marius DASYLVA

Sous la supervision du Pr. Salomon SAMBOU

Membres du jury

Salomon SAMBOU	Professeur	Président	UASZ
Youssou FAYE	Maître de Conférence Titulaire	Encadreur	UASZ
Marius DASYLVA	Enseignant-Chercheur	Encadreur	UASZ
Thierno Ahmadou DIALLO	Maître de conférences Assimilé	Rapporteur	UASZ
Mohamadou GAYE	Maître de conférences Assimilé	Rapporteur	UASZ

Année universitaire 2018/2019

Remerciements

Au terme de ce mémoire, je tiens à exprimer ma profonde gratitude et mon immense respect à Monsieur Youssou FAYE et Madame Marius DASYLVA pour m'avoir encadré et guidé tout au long de la réalisation de ce travail.

Avec beaucoup d'égard, je ne manquerai pas d'exprimer ma grande reconnaissance à tous les membres du jury (et plus particulièrement au président du jury Pr Salamon SAMBOU) pour avoir accepté d'évaluer ce modeste travail.

De même, je souhaite transmettre l'expression de ma reconnaissance et ma profonde gratitude à mes camarades de classe pour leur aide aussi bien morale que technique.

Mes vifs remerciements à mes parents, mes frères et sœurs, mes camarades de chambres, mon oncle qui est à Thiès, les professeurs qui m'ont soutenu et encouragé pour achever ce travail.

Mes remerciements vont enfin à toute personne qui a contribué, de près ou de loin à l'élaboration de ce travail.

SDN : état de l'art et perspective pour une solution contre le DoS

Résumé

Le paradigme SDN (Software Defined Networking) est une nouvelle technologie qui permet la programmation nécessaire pour permettre aux opérateurs réseau de gérer leurs infrastructures de façon dynamique. Le réseau SDN est une nouvelle approche qui sépare le plan de contrôle du plan de données pour une meilleure gestion. Les nombreux avantages que présentent les SDN, poussent de nos jours plusieurs opérateurs à y pencher.

Cependant, malgré tous les avantages que présentent ces derniers, les réseaux SDN sont vulnérables à plusieurs types d'attaques. Parmi les attaques, nous pouvons noter les attaques de déni de service qui sont les plus populaires. Ces attaques surchargent facilement le contrôleur, les tables de commutations et même la communication entre le contrôleur et les commutateurs. Ces surcharges entrainent un disfonctionnement des performances du réseau.

Pour résoudre ces problèmes, nous proposons d'utiliser les systèmes de détection d'intrusion pour contrôler les flux. Ils seront en mesure de détecter tous les trafics malveillants et d'en alerter. Cependant, la gestion de ces attaques avec seulement les IDS présente des limites. Pour pallier ces limites nous allons procéder par l'échantillonnage de trafic. Cet échantillonnage est possible grâce à l'outil d'échantillonnage sFlow.

Mots clé : SDN, OpenFlow, Déni de service (Dds), IDS, échantillonnage de trafic, cloud Computing.

Summary :

The SDN paradigm is a new technology that allows the programming necessary to allow network operators to manage their infrastructures dynamically. SDN is a new approach that separates the control plane from the data plane for better management.

The many advantages of SDN are pushing several operators to consider them these days. However, despite all the advantages of these, SDN networks are vulnerable to several types of attack. Among the attacks, we can note the most popular denial of service attacks.

These attacks easily overload the controller, switch tables and even the communication between the controller and the switches. These overloads cause a malfuncton of network performance.

To solve these problems, we propose to use intrusion detection systems to control flows. They will be able to detect and alert all malicious traffic. However, managing these attacks with only IDS has limitations. To overcome these limits we will proceed with traffic sampling. This sampling is possible thanks to the sFlow sampling tool.

Keywords : SDN, OpenFlow, Denial of Service, IDS, Traffic Sampling, Cloud Computing.

Table des matières

Liste des figures

Liste des tableaux

Liste des abréviations

API : Application Programming Interface

ARP : Address Resolution Protocol

BGP : Border Gateway Protocol

CAM : Content Addressable Memory

CPU : Central Processing Unit

Dds : Déni de service

DDds : Déni de service distribué

ForCES : Forwarding and Control Element Separation

HIDS : Host based Intrusion Detection System

ICMP : Internet Control Message Protocol

IDPS : Intrusion Detection and Prevention System

IDS : Intrusion Detection System

IP : Internet Protocol

MAC : Media Access Control

NetConf : Network Configuration Protocol

NIDS : Network based Intrusion Detection System

NIPS : Network based Intrusion Prevention System

OF : OpenFlow

ONF : Open Networking Foundation

OSPF : Open Shortest Path Firs

OVS : Open VSwitch

QoS : Quality of Service

REST : Representational State Transfer

SDN : Softeware Defined Networking

sFlow : Sampled Flow

SSH : Secure Shell

STP : Spaning Tree Protocol

TCAM : Tenary Content Addressable Memory

TCP : Transmission Controcol Protocol

UDP : User Datagram Protocol

VLAN : Virtual Local Area Network

VNF: Virtual Network Function

Introduction générale

Avec la croissance de l'utilisation de la technologie de l'information, nous assistons à une énorme augmentation du trafic qui circule dans les réseaux. Il y a un grand nombre de périphériques et d'applications modernes qui sont interconnectés. Les innovations technologiques récentes telles que la virtualisation, le cloud computing et tant d'autres poussent les réseaux à montrer facilement leurs limites. En effet, les administrateurs réseau doivent gérer une gamme de données, de type de services et de périphériques différents. Cette gestion devient très difficile avec les outils des réseaux traditionnels qui n'ont pas été conçus pour faire face à des topologies évolutives à grande échelle. Les architectures IP traditionnelles sont d'une part, complexes à configurer à cause de la nature distribuée des protocoles et d'autre part, difficile à faire évoluer en raison du couplage qui existe entre le plan de contrôle et le plan de données des équipements existants.

Le problème est de nos jours, le manque d'innovation dans les réseaux qui pose des contraintes importantes aux déploiements des applications réseau. Les chercheurs se sont penchés sur des recherches pour apporter aux entreprises et aux fournisseurs de services une solution. Cette dernière tente de résoudre la flexibilité et une gestion plus simple. Les principaux domaines d'innovations au cours de ces dernières années sont à rechercher du côté d'un contrôle centralisé, la programmabilité et la virtualisation.

Pour pallier à ces innovations un nouveau paradigme a vu le jour : SDN (Software Defined Networking). L'architecture des réseaux SDN permet une gestion dynamique des infrastructures face aux changements de comportement du trafic dans le réseau. SDN permet la séparation entre le plan de contrôle (responsable de la prise de décision du routage des flux dans le réseau et gère les applications et les services offerts par ce dernier) et le plan de données ou transmission (se charge de transférer les flux suivant les directives fournis par le plan de contrôle).

SDN est une approche qui centralise puis simplifie la gestion du réseau. Ce qui permet aux administrateurs réseau d'orchestrer et d'automatiser à travers une interface de contrôle logicielle sans accéder aux composants physiques (commutateur, routeur, …). Les administrateurs peuvent facilement gérer le trafic en modifiant une règle de routage via le contrôleur. Cette simplicité fait des SDN une approche promoteuse.

Le paradigme SDN gagne du terrain et plusieurs fournisseurs d'internet et d'administrateurs de centre de données l'adoptent progressivement, alors on s'intéresse de plus en plus au problème

de sécurité. La sécurité des réseaux SDN reste de nos jours un souci majeur. Une étude récente sur cette menace de sécurité classifie les types d'attaques par catégorie comme la modification des données, le déni de service, les applications malicieuses, l'accès non autorisé, les problèmes de configuration. Les résultats de ces attaques ont montré que le plan de contrôle et celui des données s'infectent simultanément dans la plupart des attaques.

Dans ce travail, nous nous intéressons aux attaques de déni de service (DdS). Elles ont pour but de surcharger un réseau, un serveur donné en l'inondant de manière agressive avec une grande quantité de trafic pour le rendre incapable de répondre aux attentes des utilisateurs légitimes.

De nos jours, les attaques de DdS sont inévitables et restent les plus populaires pour les SDN comme elles l'ont toujours été pour les réseaux classiques. Dans un réseau SDN, les attaques de DdS entrainent les problèmes suivants :

- ✓ **La surcharge des commutateurs** (dépassement de la mémoire TCAM Tenary Content Addressable Memory du commutateur) : chaque commutateur SDN possède une mémoire de stockage qui est bien déterminée pour le stockage des règles de flux dans les tables de commutation. Lors d'une attaque de DdS, un attaquant va envoyer un grand nombre de flux. Cela causera la saturation des tables de commutation du commutateur donc sa mémoire TCAM due à sa capacité limité de stockage. Une fois ce stade atteint, les commutateurs seront forcés à ajouter et supprimer continuellement des entrés des flux dans leurs tables et d'envoyer beaucoup de demande au contrôleur.
- ✓ **La surcharge du contrôleur** : lors d'une attaque de DdS, un énorme nombre de paquets est envoyé par l'attaquant, donc nécessite des demandes de règle de flux pour joindre une destination. Par conséquent le contrôleur sera surchargé.
- ✓ **L'épuisement de la bande passante entre contrôleur et commutateur** : lors d'une attaque DdS, il y a une communication excessive entre les commutateurs et le contrôleur. La liaison d'un commutateur vers le contrôleur peut être congestionnée et certaines demandes de règles peuvent être perdues. Cela peut créer un retard dans la décision de routage pour les flux en attentes.

L'objectif principal de ce travail est de faire un état de l'art des réseaux SDN. Ensuite faire une perspective pour un début de solution qui permet d'empêcher les attaques de déni de service. Notre solution devrait être capable de réduire la surcharge du contrôleur, les tables de commutation et la congestion de la liaison entre commutateur et contrôleur. Ainsi, nous proposons d'utiliser un IDS BroFlow qui se base sur l'outil Bro qui est un autre IDS pour la

détection d'intrusion et une technique d'échantillonnage qui utilise l'outil sFlow pour ne pas surcharger l'IDS.

La méthodologie que nous proposons pour traiter ce sujet et atteindre notre objectif est de subdiviser en six chapitres répartis sur trois parties.

Dans la première partie, notre premier chapitre porte sur l'introduction des réseaux, ce qui donne une vision globale des réseaux et les limites qu'ont ces derniers. Le deuxième chapitre introduit les réseaux SDN, là aussi nous donnons une approche générale des SDN.

Dans la deuxième partie, nous traitons l'aspect sécurité des SDN. Le troisième chapitre porte sur la sécurité et les vulnérabilités des réseaux SDN.

Dans la troisième partie, nous apportons notre contribution. Cette partie compte deux chapitres : le chapitre quatre parle des IDS et le chapitre cinq traite la solution contre le déni de service en utilisant les outils déjà disponible.

Partie1 : Les Réseaux SDN

Chapitre 1 : Introduction aux réseaux

Introduction

Un[1] réseau est le résultat de la connexion de plusieurs machines entre elles, afin que les utilisateurs et les applications qui fonctionnent sur ces derniers puissent échanger des informations. Le réseau informatique est un ensemble d'ordinateurs (ou périphériques) autonomes interconnecté entre eux pour assurer une communication. Il est de nos jours incontournable. Pour ce chapitre, nous allons commencer par rappeler l'historique des réseaux, donner une classification ensuite l'évolution des réseaux qui nous conduira à la virtualisation et en fin donner les limites de ces réseaux classiques.

I. Historique

Nous allons essayer de rappeler en quelques mots l'historique des réseaux. A partir des années 60 il y'a eu le[2] système de télétraitement, les multiplexeurs et concentrateurs. Vers les années 70, les réseaux ont évolué avec peu d'ordinateurs avec de petite puissance, coûteux, des liaisons bas débit et des technologies propriétaires incompatibles. Au années 80, c'est la multiplication des ordinateurs et moyen coûteux, adoption des technologies de transmission haute débit et des normalisations. Aux années 90, on souligne l'apparition du Web et des débits supérieurs à cent giga bits. Les années 90 jusqu'à nos jours sont marquées par la puissance de transmission avec des débits supérieurs à des giga bits.

Les réseaux sont nés d'un besoin d'échanger des informations de manière simple et rapide entre des machines. Lorsque l'on travaillait sur une même machine, toutes les informations nécessaires au travail étaient centralisées sur la même machine. Presque tous les utilisateurs et les programmes avaient accès à ces informations. Les informations devaient alors être dupliquées sur les différentes machines du même site. Cette duplication était plus ou moins facile et ne permettait pas toujours d'avoir des informations cohérentes sur les machines. On est arrivé donc à relier les machines entre elles ; ce fût l'apparition des réseaux locaux. Ces réseaux étaient souvent des réseaux domestiques. Plus tard on a éprouvé le besoin d'échanger des informations entre des sites distants. Les réseaux, moyennes et longues distances, commencèrent à voir le jour. Ces réseaux étaient souvent propriétaire. Aujourd'hui les réseaux se retrouvent à l'échelle planétaire. Le besoin d'échange d'information est en pleine évolution. Ceci étant, ils s'articulent sur deux axes orthogonaux que sont : la qualité de communication (fiabilité, débit, disponibilité, sécurité, …) et la réduction des coûts (partage des ressources entre utilisateurs et partage des ressources entre usagers).

II. Architecture des réseaux

1. Classification des réseaux

PAN[3] (Personnal Area Network) : il désigne un réseau restreint d'équipements informatiques habituellement utilisés dans le cadre d'une utilisation personnelle.

LAN (Local Area Network) : c'est un réseau informatique à une échelle géographique relativement restreinte, par exemple une salle informatique, une habitation particulière, un bâtiment ou un site d'entreprise.

MAN (Metropolitan Area Network) : ou réseaux urbains sont à mi-chemin entre les réseaux locaux et les réseaux étendus. Ils s'étendent de deux kilomètres à des dizaines de kilomètres.

WAN (Wide Area Network) : permet l'interconnexion de plusieurs LAN ou WAN sur de grandes distances géographiques, à l'échelle d'un pays ou mondiale.

2. Topologies des réseaux

Un[3] réseau informatique est constitué d'ordinateurs reliés entre eux grâce à du matériel (câblage, carte réseau, ainsi que d'autres équipements permettant d'assurer la bonne circulation des données). L'arrangement physique (type de câble) de ces éléments est appelé topologie physique. Il en existe généralement :

Bus : c'est l'organisation la plus simple. Ici tous les ordinateurs sont reliés à une même ligne de transmission par l'intermédiaire de câble, généralement coaxial. Les connecteurs sont des connecteurs en T ou Vampire. Lorsqu'on émet, la trame parcourt tout le bus jusqu'à ce qu'elle arrive à destinataire. A chaque extrémité, le réseau est terminé par une résistance appelée bouchon. Si le câble tombe en panne alors tout le réseau ne fonctionne pas. La figue 1 donne une parfaite illustration.

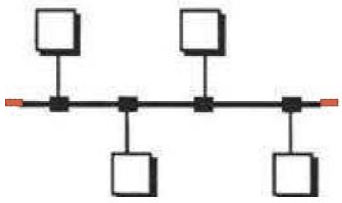

Figure 1**[2]**: topologie en bus

Anneau: il s'agit d'un réseau local dans lequel les nœuds sont reliés à un répartiteur appelé MAU (Mutistation Access Unit). Les données circulent sur un anneau de nœud à l'autre.

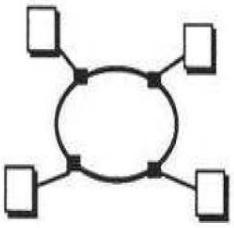

Figure 2[1]: topologie en anneau

Etoile : pour cette topologie, chaque nœud du réseau est relié à un nœud central. Ce dernier se charge de la transmission des signaux à leur destination.

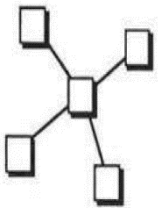

Figure 3[1]: topologie en étoile

Arbre : connu sous le nom de hiérarchique. Le sommet, de haut niveau, est connecté à plusieurs nœuds de niveau inférieur dans la hiérarchie.

Maillé : cette topologie correspond à plusieurs liaisons point à point. Chaque terminal est relié à tous les autres.

3. Modèle de référence OSI&TCP/IP

Un[3] modèle de référence est utilisé pour décrire la structure et le fonctionnement des communications réseaux. On distingue deux modèles : OSI et TCP/IP. La figure 4 l'illustre.

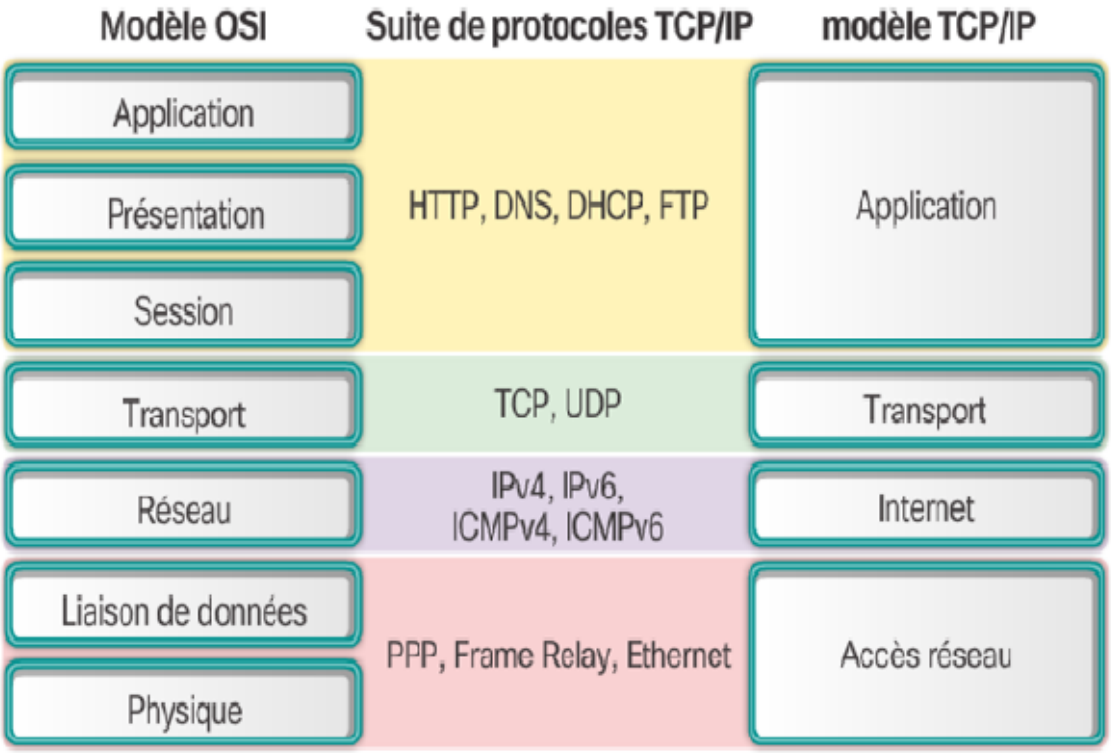

Figure 4[3]: Modèle de référence OSI&TCP/IP

Nous allons expliquer les différentes couches du modèle OSI.

La couche physique : décrit les caractéristiques physiques de la communication, comme le média utilisé et tous les détails associés comme les connecteurs, les types de codages, le niveau des signaux, … et les distances maximales. Elle assure la transmission des bits de la trame de la couche supérieure sur le réseau physique.

La couche liaison de données : spécifie comment les paquets de la couche supérieure seront transportés. Elle assure la mise en trames, leurs acheminements sans erreurs et la méthode d'accès au réseau physique.

La couche réseau : résout le problème de l'acheminement des paquets à travers un réseau. Elle permet de transférer des données pour de nombreux protocoles de plus haut niveau.

La couche transport : cette couche est responsable du transport des données de bout en bout au travers du réseau.

La couche session : elle établit une connexion entre émetteur et récepteur en assurant l'ouverture et la fermeture des sessions.

La couche présentation : à son tour met en forme les informations échangées pour les rendre compatible avec l'application destinatrice, dans le cas de dialogue entre système hétérogène.

La couche application : en fin elle va apporter les services de base offerts par le réseau pour les logiciels.

4. Couplage du plan de données du plan de contrôle

La représentation simplifiée d'un routeur est décrite par un plan de contrôle qui décide de la route et d'un plan de donnée pour prendre en charge le trafic. Pour les équipements de réseau traditionnels, le plan de contrôle et le plan de données sont localisés dans le même équipement. La souplesse apportée par les algorithmes de routage permet d'allouer dynamiquement les routes optimales en fonction de l'évolution de charge du trafic. Pour une bonne compréhension de ce que nous venons de dire, nous allons apporter la figure 5 qui va l'illustrer.

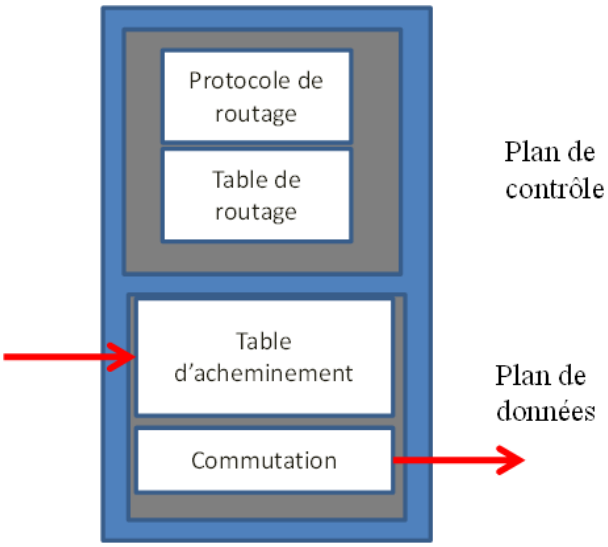

Figure 5[3]: Architecture d'un routeur

III. Exemple d'architecture des réseaux

1. Ethernet

C'est un[4] réseau local créé par Xerox pour connecter des ordinateurs et autres équipements terminaux entre eux. Il est normalisé par l'institut IEEE sous la norme IEEE 802.3. C'est le premier réseau local à être créé. Il se présente aujourd'hui sous plusieurs normes offrant des débits allant de 10 Mbit/s à 100 Gbit/s. il est soit partagé ou commuté.

Figure 6[1]: architecture ethernet

2. Token Ring

C'est un[4] réseau créé par IBM pour éviter les conflits générés par l'accès par compétition d'Ethernet. Il est caractérisé par un droit d'émission conditionné par la possession d'un jeton. Token Ring est normalisé par IEEE sous la norme IEEE 802.5 et assure des débits de 4 ou 16 Mbit/s.

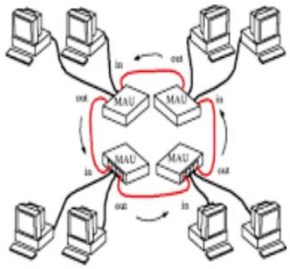

Figure 7[1]: architecture Token Ring

3. FDDI

C'est un réseau à fibre optique utilisé à la fois pour les MAN et les LAN. FDDI offre des débits de 100 Mbit/s.

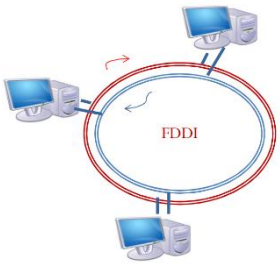

Figure 8[1]: Architecture FDDI

4. Wifi

Le wifi est une technologie de réseau local permettant de relier sans câble des équipements. Cette technologie utilise les ondes radio pour la transmission des données par le biais d'antennes sur une portée de quelques dizaines de mètres. Elle est normalisée par IEEE 802.11 et offre des débits allant de quelques Mbit/s à des Gbit/s sur une portée de quelques dizaines de mètres.

Figure 9[2]: Architecture Wifi

5. Pourquoi les WAN ?

Les[5] LAN sont des réseaux géographiquement limités. On les utilise pour assurer la connectivité entre des utilisateurs et le partage des ressources dans une zone limitée. Les WAN quant à eux, sont des réseaux qui peuvent s'étendre sur de vastes zones, permettant ainsi d'interconnecter des LAN ou des utilisateurs distants. Ils sont une propriété d'un opérateur de télécoms pour le transport d'informations entre réseaux ou utilisateurs éloignés. Les WAN fonctionne principalement au niveau des couches 1 et 2 du modèle OSI et repose généralement sur une architecture à commutation. On peut noter deux catégories de WAN : les WAN privés (Ligne louée, RTC, RNIS, Frame Relay, ATM) et les WAN d'accès au réseau public ou internet (xDSL, Câble, 3G et 4G).

IV. Evolution des réseaux

Les réseaux ne cessent d'évoluer, c'est pourquoi les besoins se multiplient. Pour permettre à ses abonnés l'accès au réseau internet, le réseau WAN du fournisseur d'accès doit être connecté à d'autres réseaux. Cette interconnexion est rendu possible grâce au protocole IP. Le réseau IP permet donc l'interconnexion des réseaux ayant des technologies de transmissions différentes. Sur un réseau IP, la mise en relation entre une machine et un serveur s'appuie sur des équipements de routage et des équipements de commutation. Le rôle du routeur est d'acheminer chaque paquet au nœud suivant de la machine au serveur. Avant toute utilisation, les routeurs doivent d'abord être configurés. Ce qui permet de définir les adresses IP des interfaces

physiques et virtuels du routeur et d'informer le routeur des protocoles de routage à appliquer pour acheminer les paquets. Les routeurs échangent entre eux des informations sur leur table de routage en fonction du protocole de routage. La table de routage permet de définir le réseau de destination pour chaque paquet.

Malgré ces avantages que présentent les réseaux classiques, ils en présentent beaucoup de limites par rapport à l'attente des usagers. Nous pouvons noter le couplage du plan de données et celui du contrôle qui entraine une évolution très complexe. Ces réseaux présentent aussi un manque de flexibilité donc nouvelle installation très compliquée. L'adaptabilité du réseau de transport par rapport à la charge de trafic cause problème. Ils présentent d'énormes difficultés pour l'équilibrage de charge, des pare-feu et le déploiement de nouveau services. Tous ceux-ci s'expliquent par la multiplication des usagers et des besoins. Ainsi, pour répondre aux attentes, il faut obligatoirement revoir la façon de concevoir ces derniers. C'est pourquoi des études nous mènent vers de nouvelles approches qui sont la virtualisation et la mise en réseau par logiciel (SDN).

V. La virtualisation

1. Introduction

Pour définir la[6] virtualisation, elle correspond à l'ensemble des techniques matérielles et/ou logicielles qui permettent de faire fonctionner sur une seule machine plusieurs systèmes d'exploitations et/ou plusieurs applications, séparément les uns des autres, comme s'ils fonctionnaient sur des machines distinctes.

Nous allons donner le principe de la virtualisation, les avantages qu'elle offre, ses limites et terminer par les limites des réseaux classiques.

2. Principe

Le principe de la virtualisation consiste à séparer le matériel du logiciel et en émulant le matériel par l'aide de logiciel. Pour qu'un système d'exploitation fonctionne au-dessus du système d'exploitation primaire on fait appelle à la virtualisation et aux machines virtuelles. Une machine virtuelle peut être définie comme étant un fichier de données déplaçable et pouvant être copié sur une autre machine physique comme un fichier de données normal. Les machines virtuelles fonctionnent comme des machines physiques. Elles sont isolées les unes des autres malgré l'utilisation du même substrat physique.

La séparation des couches matériels et logiciels est obtenu à l'aide d'une couche supplémentaire appelée hyperviseur. Un hyperviseur ou autrement moniteur de machine virtuelle (virtual machine monitor : VMM) est un logiciel qui permet de créer et exécuter des machines virtuelles. Il permet d'instancier plusieurs machines virtuelles qui fonctionnent chacune avec son propre système d'exploitation mais qui partagent les mêmes ressources physiques qui sont présentes sur la couche matérielle (CPU, cartes réseau, RAM, …).

La virtualisation est représentée par une architecture composée de trois parties. L'hyperviseur[7] joue le rôle d'intermédiaire entre la couche physique et les machines virtuelles. Alors, chaque machine virtuelle à son propre système d'exploitation et une couche applicative qui lui est propre en fonction du besoin.

La figure 10 ci-dessous donne une parfaite illustration de ce que nous venons de dire.

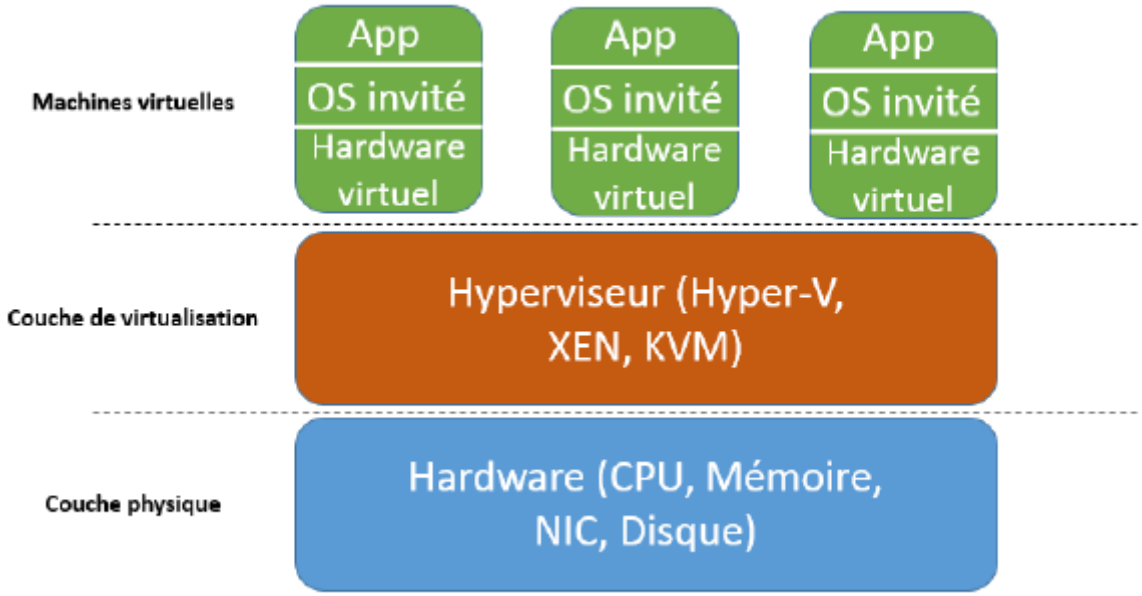

Figure 10[8]: architecture de la virtualisation

3. NFV (Network Function Virtualization) ou virtualisation des fonctions réseau

NFV[8] est une architecture réseau qui fait appel aux technologies de virtualisation pour pouvoir virtualiser des classes entières de fonction de nœuds de réseau dans des blocs qui peuvent se connecter pour créer des services de communication. Une fonction de réseau

virtualisée ou NFV est constituée d'une ou plusieurs machines virtuelles en cours d'exécution avec des logiciels et processus différents, au-dessus des serveurs, switchs et unité de stockage de grande taille.

Son principal but est de découpler les fonctions réseau du matériel physique. Il permet aux services de réseau qui sont dans les routeurs, les pare-feu et autres périphériques dédiés à être hébergés sur des machines virtuelles. Après que toutes les fonctions réseau soient sous contrôle de l'hyperviseur alors les services qui nécessitaient auparavant un matériel dédié peuvent être lancés sur des serveurs x86 standard. Ceci étant, les administrateurs réseau n'ont plus besoin d'acheter du matériel dédié pour construire une topologie réseau complète. Ainsi la capacité d'un serveur peut être agrandie de façon logicielle tout en évitant le surdimensionnement matériels. Si une application qui fonctionne sur une machine virtuelle nécessite plus de bande passante alors l'administrateur peut déplacer la machine vers un autre serveur physique ou créer une autre machine sur le serveur d'origine pour prendre une partie de la charge.

L'obtention de cette souplesse permet de répondre d'une manière très intéressante à l'évolution des objectifs des entreprises ainsi que des demandes des services réseau. Nous allons illustrer ces arguments par les différentes possibilités de virtualisation par une image.

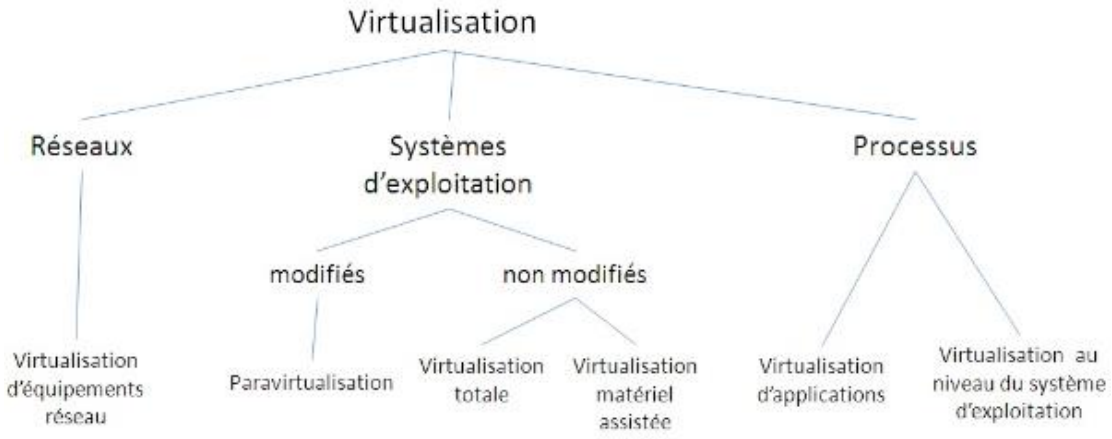

Figure 11**[6]**: virtualisation possible

4. L'hyperviseur Xen

a. Définition

L'hyperviseur[7] Xen est une plate-forme de virtualisation qui permet à plusieurs système d'exploitation de travailler sur la même machine physique et simultanément. Il se base sur le concept de micronoyau fournissant des services permettant à plusieurs systèmes d'exploitations d'être exécutés sur le même matériel et en même temps. Xen est responsable de la gestion de

la mémoire et l'ordonnancement du CPU de toutes les machines virtuelles. Il est responsable aussi du lancement du domaine le plus privilégié : le « Dom0 » .L'hyperviseur peut être géré depuis le Dom0, c'est aussi à partir de ce domaine qu'on peut lancer les domaines non privilégiés.

b. Architecture Xen

L'architecture de Xen est composée de plusieurs couches qui communiquent les unes avec les autres. Une instance de machine virtuelle[9] est appelée domaine ou DomU. Un domaine Dom0 est un domaine spécial qui contient les pilotes pour tous les périphériques du système. Nous allons présenter la figure 12 et sur laquelle nous allons faire un commentaire pour plus d'illustration.

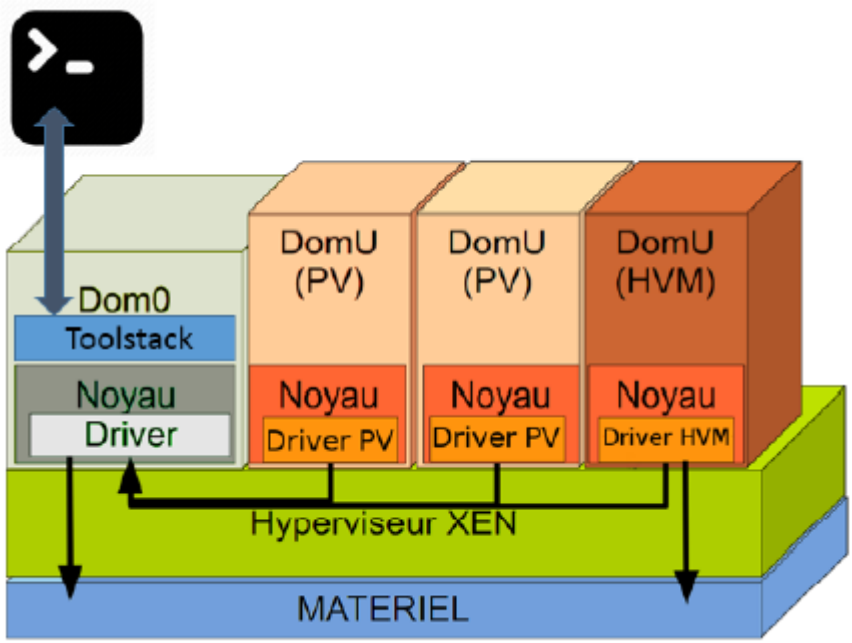

Figure 12[8]: Architecture Xen

Nous allons apporter quelques explications sur cette figure pour mieux comprendre cette architecture.

- ✓ La couche matérielle: c'est dans cette couche que nous avons les I/O (Input/Output), la mémoire, le CPU et les interfaces réseaux.
- ✓ L'hyperviseur Xen : un logiciel qui fonctionne directement sur le matériel. Il est responsable de la gestion de la mémoire, du CPU ainsi que les interruptions.

- ✓ Les DomU : ou machines virtuelles sont des environnements virtualisés. Chacun exécute son propre système d'exploitation et application. L'hyperviseur prend en charge deux modes de virtualisations : la para-virtualisation (PV) et la virtualisation complète (HVM). Ces deux types de modes peuvent être utilisés sur le même hyperviseur.
- ✓ Le dom0 : ou encore domaine de contrôle est une machine virtuelle qui possède des privilèges spéciaux. Ces privilèges sont l'accès direct au matériel, la gestion de tous les accès aux fonctions d'I/O du système et l'interaction avec les autres machines virtuelles.
- ✓ Le toolstack : c'est une couche de contrôle qui est présente dans le Dom0. Il permet à un utilisateur de gérer la création, la destruction et la configuration des machines virtuelles. Il s'appuie sur son interface qui peut être pilotée par une console de ligne de commande, une interface graphique ou un outil d'orchestration.

c. La virtualisation dans Xen

Nous avons dit précédemment que l'hyperviseur supporte deux modes de virtualisations : la para-virtualisation et la virtualisation complète.

- ✓ **La para-virtualisation ou virtualisation partielle**

C'est une technique de virtualisation très efficace. Elle ne nécessite pas d'extension de virtualisation depuis le processeur de l'hôte. Tous les clients para-virtualisés requièrent un noyau apte à la para-virtualisation et des drivers de sorte qu'ils soient conscients de l'hyperviseur.

- ✓ **La virtualisation complète**

C'est une technique qui utilise des extensions de virtualisation du presseur de l'hôte pour virtualiser les domaines invités. Elle nécessite des extensions matérielles. Ces dernières sont utilisées pour améliorer les performances de l'émulation.

5. Les bienfaits de la virtualisation et ses limites

Le fait de passer à la virtualisation est une décision très lourde en conséquence sur l'infrastructure réseau. La virtualisation répond à beaucoup de problématiques réelles afin de relever les nouveaux défis auxquels l'informatique fait face. Nous pouvons en citer le besoin de souplesse, évolutivité et l'isolation. La virtualisation présente beaucoup d'avantages mais a aussi des inconvénients.

a. Les bienfaits de la virtualisation

Les gens ont tendances à donner comme avantage de la virtualisation la réduction des coûts de l'entreprise. Pour notre cas nous allons au-delà de cet argument.

- ✓ **Réductions des coûts** : l'un des principaux avantages de la virtualisation est qu'elle nécessite moins de matériels pour exécuter le même type et la même quantité de logiciel. Elle permet une meilleure utilisation du réseau en utilisant les ressources disponibles.
- ✓ **Redéploiement rapide et continuité des services** : un autre avantage, c'est la récupération des données simplifiées. Exemple : si le serveur virtuel tombe en panne, il suffit de le supprimer et le restaurer à partir de sa sauvegarde virtuelle. Donc un gain de temps et moins d'efforts sont fournis.
- ✓ **Le testing** : une plate-forme est fournie qui permet de tester différentes configurations logicielles et sur différentes plateformes avant le déploiement.
- ✓ **Ecologique** : une baisse de la consommation d'énergie vue qu'on utilise moins de matériels informatique pour accomplir le même type de travail.
- ✓ **Sécurité et fiabilité** : la démarcation de la couche physique et de la couche logicielle permet une amélioration de la sécurité du système et de sa fiabilité.
- ✓ **La migration** : la machine virtualisée peut être déplacée d'un emplacement à un autre sans perte de temps.

b. Les limites de la virtualisation

La virtualisation est une technologie à double tranche, ce qui veut dire que malgré les avantages qu'elle offre, elle présente des inconvénients voir des limites.

- ✓ **Les machines virtuelles reposent sur un substrat physique** : même si c'est rare, les défaillances physiques peuvent être dévastatrices. Exemple : si le disque dur principal qui contient toutes les données virtuelles et physiques est soudainement volé, brisé, brûlé, endommagé alors tous les serveurs à la fois physiques et virtuels doivent être restaurés.
- ✓ **Nécessite du matériel puissant** : la virtualisation est une technologie qui dépend de la puissance de calcul et de la mémoire. Il faut alors prendre en compte beaucoup plus de mémoire et de puissance de calcul.
- ✓ **Nécessite une expertise pour la maintenance :** si un problème survient dans un système virtualisé, cela nécessite dans certains cas un dépannage complexe.

- ✓ **L'overhead** : il est important de penser à l'impact qu'aura une infrastructure virtualisée par rapport à une infrastructure physique. Les machines virtualisées ont besoin d'un temps particulier pour s'autogérer.

Conclusion

En résumer, on peut dire que les réseaux classiques facilitent le transport et le traitement d'information. Mais la multiplication des utilisateurs ainsi que leur besoins nous conduit à minimiser l'offre de ces réseaux classiques. Il faut impérativement revoir la façon de concevoir le réseau pour répondre à l'attente des usagers. C'est ce que nous allons voir dans le chapitre suivant qui va introduire les réseaux SDN.

Chapitre 2: Introduction aux réseaux SDN

Introduction

Les réseaux classiques ont joué un rôle primordial depuis plusieurs années. Cependant avec la multiplication des utilisateurs ainsi que les multitudes de services sollicités, ces derniers tardent à répondre à temps aux usagers. Ce chapitre porte sur l'approche SDN. Pourquoi la venue des SDN ? Nous allons donner une présentation générale des SDN, ensuite les avantages et impact et en fin les défis.

I. C'est quoi SDN

1. Introduction

L'idée du Software Defined Networking est de rendre le réseau programmable, c'est-à-dire de permettre aux applications d'interagir directement avec le réseau. Il vise donc à séparer le logiciel du matériel. Dans une telle agitation, les administrateurs réseau sont les plus souvent perdus. Ils se posent les questions suivantes : c'est quoi SDN ?, qu'est-ce qu'un contrôleur ?, qu'elle est son apport ?

Dans ce chapitre nous commencerons par donner une définition, rappeler l'historique, présenter l'architecture et ses composants, donner ses avantages et impacts.

2. Historique des SDN

Avant l'apparition des réseaux SDN tels que nous les connaissons aujourd'hui, plusieurs idées et travaux ont été proposés auparavant, notamment la programmation du réseau et la séparation des plans de contrôle et de données. Nous donnons dans cette section un bref aperçu de ces travaux, qui peuvent être considérés comme des ancêtres de SDN.

La[10] première idée de programmation de réseaux a été développée en 1996 sous le nom de «réseaux actifs» (AN, Active Network). Ces réseaux injectent des programmes parmi les données du paquet. Quand un noeud du réseau reçoit ces paquets, il extrait et exécute les programmes à partir des données du paquet et déclenche par conséquent des actions de transmission, de modification ou de suppression du paquet. Avec cette approche, de nouveaux mécanismes de services et de routage du réseau peuvent être implémentés sans modification des équipements de transmission.

Plusieurs études ont été menées sur les ANs, en particulier sur les paquets intelligents, ANTS et switchWare. Puisque les paquets des ANs peuvent transporter des programmes malveillants, une alternative aux ANs appelée « réseaux programmables »(PN) a été proposée en 1999. Les

PNs injectent des programmes à l'intérieur des noeuds du réseau. Ces noeuds exécutent les programmes uniquement après une phase de signalisation et de vérification, pour renforcer la sécurité. Les ANs et les PNs ont cherché à introduire la programmabilité dans les réseaux à travers des paquets et des switchs programmables. Ces approches n'ont pas réduit la complexité de l'infrastructure réseau.

D'autres parts, certains projets ont essayé de séparer le plan de contrôle du plan de données pour simplifier l'architecture réseau et fournir une abstraction sur l'infrastructure physique. Par exemple, le projet de recherche dénommé DCAN (Devolved Control of ATM Networks) de l'université de Cambridge dont l'objectif était de développer l'infrastructure réseau de manière à ce que les fonctions de contrôle et de gestion de plusieurs équipements (les switchs ATM dans le cas du DCAN) soient découplées de l'équipement physique et déléguées aux entités externes telles que le gestionnaire. Mais DCAN avait besoin d'un protocole de communication entre le gestionnaire et l'infrastructure réseau. En 1998, le groupe de travail OPENSIGN a commencé une série d'ateliers dédiés à rendre les réseaux d'ATM, d'Internet, et de mobiles plus ouverts, extensibles, et programmables. OPENSIG établit une distinction entre l'équipement physique qui transporte les paquets, le système de contrôle et de gestion du réseau. Le coeur de leur proposition était de fournir un accès à l'équipement physique via des interfaces réseaux programmables et ouvertes, puis de laisser au fournisseur de services le choix de manipuler le réseau en utilisant des applications intermédiaires. Motivé par ces idées, un groupe de travail de l'IETF (Internet Engineering Task Force) a pu proposer un protocole général de gestion des switchs GSMP (General switch Management Protocol). Un autre projet appelé 4D a été lancé en 2005 pour séparer les décisions de routage et les protocoles qui gouvernent les interactions entre les équipements du réseau. Les plans de diffusion et de découverte collectent les informations sur le réseau et les envoient au plan de décision, qui a une vision globale sur le réseau, afin de contrôler la transmission du trafic qui circule dans le plan de données.

Le début des[11] réseaux SDN a commencé avec le projet Ethane, lancé en 2006 à l'université de Stanford. En effet, le projet Ethane définit une nouvelle architecture pour les réseaux d'entreprises. L'objectif d'Ethane était d'avoir un contrôleur centralisé pour gérer les règles (policies) et la sécurité dans le réseau. Ethane utilise deux composantes : un contrôleur pour décider si un paquet doit être transmis, et un switch Ethane composé de table et d'une chaîne de communication entre les deux. Ethane était une source d'inspiration pour un système d'exploitation consacré aux réseaux dénommé Nox, et pour un nouveau concept appelé aujourd'hui « réseaux programmés par logiciel » (SDN, Software-Defined Networking). Nous notons que les chercheurs d'Ethane sont derrière Nox et SDN.

II. Présentation de l'architecture SDN

Selon l'ONF (Open Network Fundation), SDN est une architecture qui sépare le plan de contrôle du plan de données, et unifie le plan de contrôle dans un software de contrôle externe appelé « Contrôleur », pour gérer plusieurs éléments du plan de données via des APIs (Application Programming Interface). Dans cette section, nous présentons l'architecture du SDN et décrivons ses principales composantes.

1. Architecture

Traditionnellement, un réseau informatique est composé d'équipements réseaux interconnectés tels que des switchs et des routeurs. Dans chaque équipement, nous trouvons un mécanisme de transmission dans la couche de transmission et un plan de contrôle qui incorpore le système d'exploitation et les applications. Dans ce modèle, les équipements réseaux sont fermés et nous n'avons aucune possibilité d'ajouter une nouvelle application. L'installation de nouveaux protocoles dépend des cycles de production qui peuvent être longs. Afin d'ouvrir les équipements réseaux et de séparer les applications de la couche de transmission, l'architecture de « réseau programmé par logiciel », ou SDN[12], a vu le jour. Il est composé de trois parties : le plan de données, le plan de contrôle et le plan d'application. Nous l'illustrons par la figure 13.

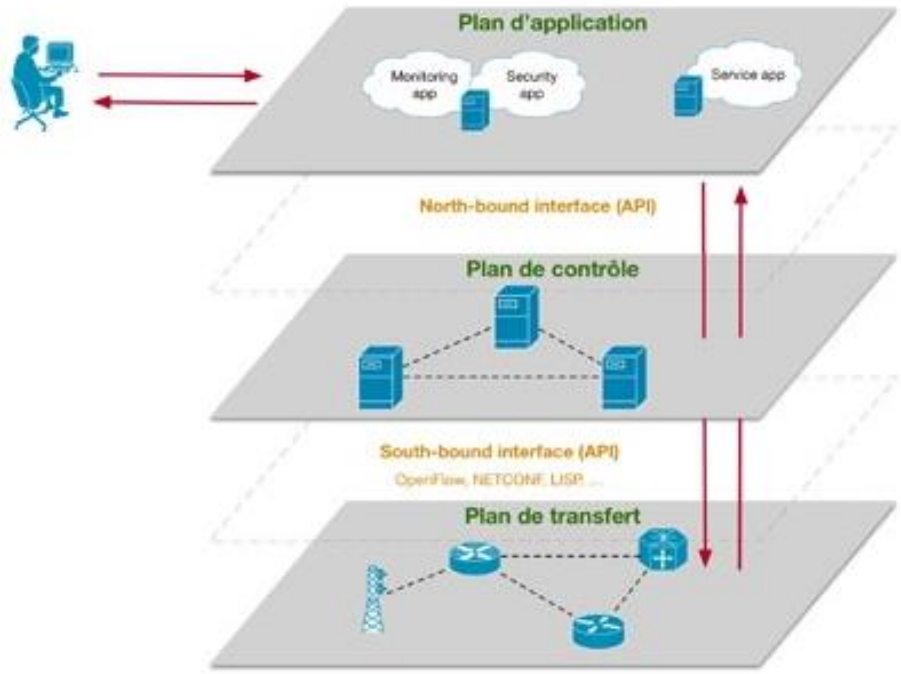

***Figure 13**[13]: Architecture SDN*

Cette figure présente l'approche SDN avec ses trois parties. Nous allons expliquer le rôle de chaque partie de cette architecture.

2. Les composants d'une architecture SDN

Le plan de données : c'est[14] la couche de transmission. Il contient les équipements de transmission tels que les switchs physiques et virtuels. Son rôle principal est de transmettre les données, surveiller les informations locales et collecter les statistiques.

Le plan de contrôle : il est constitué d'un ou plusieurs logiciels de contrôle (contrôleur). Ces contrôleurs utilisent des interfaces sud ouvertes pour contrôler le comportement des équipements et communiquent via des APIs nord avec la couche supérieure pour superviser et gérer le réseau.

Le plan d'application : héberge les applications qui peuvent introduire des nouvelles fonctionnalités réseau, comme la sécurité, la configuration dynamique et la gestion. Il exploite la vue globale et distance du réseau offerte par le plan de contrôle pour fournir des directives appropriées au plan de contrôle.

Deux[15] types d'interfaces permettent aux contrôleurs de communiquer avec leur environnement : interfaces sud et interfaces nord.

Interfaces sud : ce sont des interfaces de communication permettant au contrôleur d'interagir avec les switchs/routeur du plan de données. Plusieurs protocoles ont vus le jour mais le plus utilisé est le protocole OpenFlow.

Interfaces nord : servent à programmer les éléments de transmission en exploitant l'abstraction du réseau fourni par le plan de contrôle. Selon l'ONF (Open Networking Fundation), plusieurs niveaux d'abstraction et différents cas d'utilisation peuvent être caractérisés, ce qui signifie qu'il peut y avoir plusieurs interfaces nord pour servir tous les cas d'utilisation.

Interfaces Est/Ouest : ce sont des interfaces de communication qui permettent la communication entre les contrôleurs SDN dans une architecture à plusieurs contrôleurs pour synchroniser le réseau. Ces architectures sont récentes et aucun standard de communication entre les contrôleurs n'est encore disponible. C'est un projet qui est toujours en cours.

Après explication des différents plans et interfaces, nous allons attaquer le contrôleur qui se situe au niveau du plan de contrôle.

3. Le contrôleur

Le[16] contrôleur SDN permet d'implémenter rapidement un changement sur le réseau en traduisant une demande globale en une suite d'opérations sur les équipements réseau. Le contrôleur communique avec le plan de données par les APIs sud et le plan d'application via les APIs nord. Nous présentons ci-dessous une liste non exhaustive des contrôleurs SDN:

- ✓ **Nox**: le premier contrôleur disponible au public écrit en langage C, l'utilisation d'un seul thread dans son coeur limite son déploiement. Plusieurs dérivations du NOX ont été proposées plus tard, notamment le Nox dans sa version multithread appelée NOX-MT, le QNOX, NOX supportant la qualité de service (QoS), FortNOX, une extension du NOX implémentant un analyseur de détection des conflits dans les règles de flux causées par les applications, et enfin, le contrôleur POX, un contrôleur basé sur NOX en langage python, dont le but est d'améliorer les performances du contrôleur original NOX.
- ✓ **Maestro**: utilise la technologie du multithreading pour effectuer le parallélisme au bas niveau, en gardant un modèle simple de programmation pour les développeurs d'applications. Il atteint ses performances à travers la distribution des tâches du cœur aux threads disponibles et la minimisation de la mémoire consommée. En plus, Maestro peut traiter les demandes issues de multiples flux par une seule tâche d'exécution, ce qui augmente son efficacité.
- ✓ **Beacon**: développé en java par l'université de Stanford, il est aussi basé sur les technologies de multithreads et est multiplateforme. Son architecture modulaire permet au gestionnaire d'exécuter uniquement les services désirés.
- ✓ **SNAC**: utilise une application web pour gérer les règles du réseau. Un langage de définition des règles flexibles et des interfaces faciles à utiliser ont été intégrés pour configurer les équipements réseaux et contrôler leurs évènements.
- ✓ **Floodlight**: est une variante de Beacon caractérisée par sa simplicité et sa performance.
- ✓ Il a été testé avec les commutateurs OpenFlow physiques et virtuels. Il est aujourd'hui supporté et amélioré par une large communauté de développeurs, comprenant des industriels comme Intel, Cisco, HP, Big switch et IBM.
- ✓ **McNettle**: c'est un contrôleur SDN programmé par Nettle, qui est un DSL (Domain Specific Language) intégré dans Haskell, et permet la programmation des réseaux en utilisant OpenFlow. McNettle opère dans des serveurs multi-coeurs qui partagent leur mémoire pour atteindre une visibilité globale, une haute performance et une faible latence.

- ✓ **RISE**: conçu pour les expérimentations des réseaux à grande échelle, RISE est un contrôleur basé sur Trema. Ce dernier est un Framework programmé en Ruby et C. Trema fournit un environnement intégré de test et de débogage, incluant un ensemble d'outils pour le développement.
- ✓ **Open Daylight** : Alors que le consortium ONF s'est penché sur l'API OpenFlow permettant le contrôle du plan de données, le consortium Open Daylight a été créé pour se pencher sur la problématique du contrôleur, qui est un élément prépondérant de nombreuses architectures SDN. L'objectif de ce consortium est de regrouper les efforts industriels et académiques pour développer un contrôleur réellement utilisable dans des environnements de production. Ce projet collaboratif, placé au sein de la Linux Foundation, a permis le développement d'un contrôleur.

Nous pouvons noter la quantité d'interfaces sud (OpenFlow, OVSDB, Netconf, BGP, PCEP, SNMP, HTTP, Opflex…) mettant en évidence qu'une seule API focalisée sur le plan de données ne suffit pas quand il est nécessaire d'assurer de nombreux services. Pour assurer l'intégration simple d'une nouvelle API sud, une couche d'abstraction appelée SAL (Service Abstraction Layer) a été conçue et reste un élément clé du contrôleur Open Daylight, permettant un développement de nouveaux services sans lien direct avec les API utilisées pour les implémenter. Pour ce protocole le projet est toujours en cours.

La liste n'étant pas exhaustive, on peut y ajouter : MUL, Maestro, Trema, Jaxon, Helios, SNAC, NodeFlow, Flowvisor, Ovs-controller, RouteFlow.

4. OpenFlow

OpenFlow[17] dans sa version 1.4 est défini par L'ONF comme étant un protocole de communication qui permet d'accéder directement au plan de données des périphériques réseau tels que les routeurs et les switch, à la fois physiques et virtuels à travers le réseau. Autrement dit, il permet de contrôler le comportement du plan de données d'une façon centralisée, dynamique et programmable.

Dans un premier temps on va étudier le fonctionnement d'un Switch OpenFlow. Cette partie couvrira les différents composants et fonctionnalités basiques de ce dernier, ainsi que sa gestion à travers un contrôleur distant.

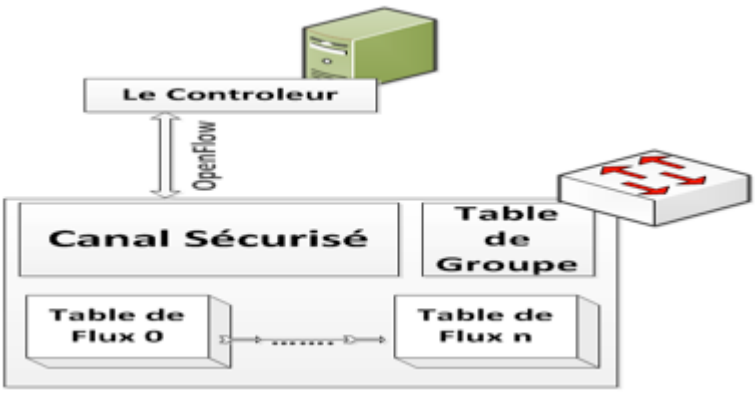

***Figure 14**[18]: Architecture OpenFlow*

a. Fonctionnement de OpenFlow

OpenFlow fonctionne d'une manière très simple. C'est avec un jeu d'interaction. Lorsqu'un paquet arrive à un commutateur, ce dernier vérifie s'il y a une entré dans la table de flux qui correspond à l'en-tête du paquet. Si tel est le cas alors le commutateur exécute l'action correspondante dans sa table de flux. Et si c'est le cas contraire, qui veut dire qu'il y a pas une entré correspondante, dans ce cas le commutateur notifie un message au contrôleur puis le contrôleur décide selon sa configuration une action pour le paquet et envoie une nouvelle règle de transmission au commutateur. En fin la table de flux du commutateur est actualisée pour prendre en compte la nouvelle règle. La figure 15 suivante décrit la transmission d'un paquet avec OpenFlow.

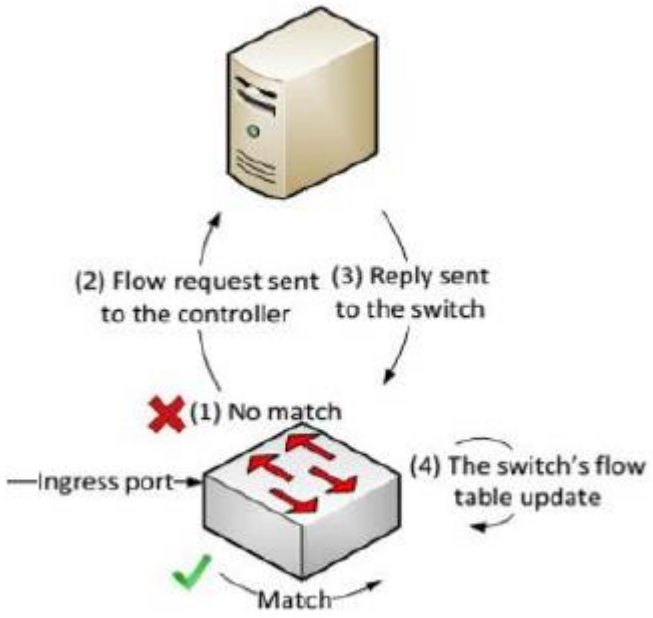

***Figure 15**[18]: Transmission d'un paquet*

b. Processus de transmission d'un paquet avec OpenFlow

Un Switch[19] Open Flow contient une ou plusieurs tables de Flux et une Table de Groupe, qui traitent les paquets entrants et les commutent vers la destination, il s'appuie sur un Canal sécurisé pour communiquer avec le contrôleur externe.

Le contrôleur gère le Switch en utilisant le protocole OpenFlow, qui lui permet d'ajouter, d'effacer et de mettre à jour les entrées dans la ou les Table de Flux.

La correspondance des paquets commence par la première entrée et peut continuer ainsi, dans le cas où plusieurs Tables de Flux existent. Les entrées font la correspondance selon la priorité, si une correspondance est trouvée les instructions associées à cette entrée sont exécutées.

S'il n'y a pas de correspondance dans la table de Flux :

- Le paquet est commuté vers le Contrôleur via le canal sécurisé OpenFlow.
- Le paquet sera Abandonné.
- Le paquet va continuer vers la table de Flux qui suit (Traitement Pipeline)

Les instructions associées à chaque entrée décrivent l'acheminement du paquet, la modification, et les traitements dans le pipeline.

Les instructions permettent lors d'un traitement par pipeline de:

- ✓ Envoyer le paquet vers la prochaine Table de Flux et la communication d'information entre elles sous forme de Meta-data.
- ✓ Le traitement s'arrête quand l'ensemble des instructions associées à une entrée ne spécifie plus de table, généralement arrivé à ce stade le paquet est modifié et acheminer vers la destination.

c. Les Tables OpenFLow

Cette section décrit les composants de tables OpenFLow à savoir, les Table de Flux, les Tables de groupes, ainsi que les mécanismes de correspondance.

- ✓ **Les Tables de Flux**

Chaque Table de Flux dans un Switch contient un ensemble d'entrées.

Champ de Correspondance	Instructions	Compteurs

Les principaux éléments d'une entrée dans une table de Flux :

Le champ de correspondance « Match Fields »: c'est la partie sur laquelle le contrôleur se base pour faire correspondre les paquets, cela consiste à vérifier le port d'entrée ou l'entête du paquet afin d'y appliquer une action X.

Compteurs : Il est possible de disposer d'un certain nombre de statistiques dont on sert pour la gestion des entrées dans les tables de flux.

Instructions : c'est une opération pouvant contenir en elle-même un ensemble d'actions à appliquer sur le paquet.

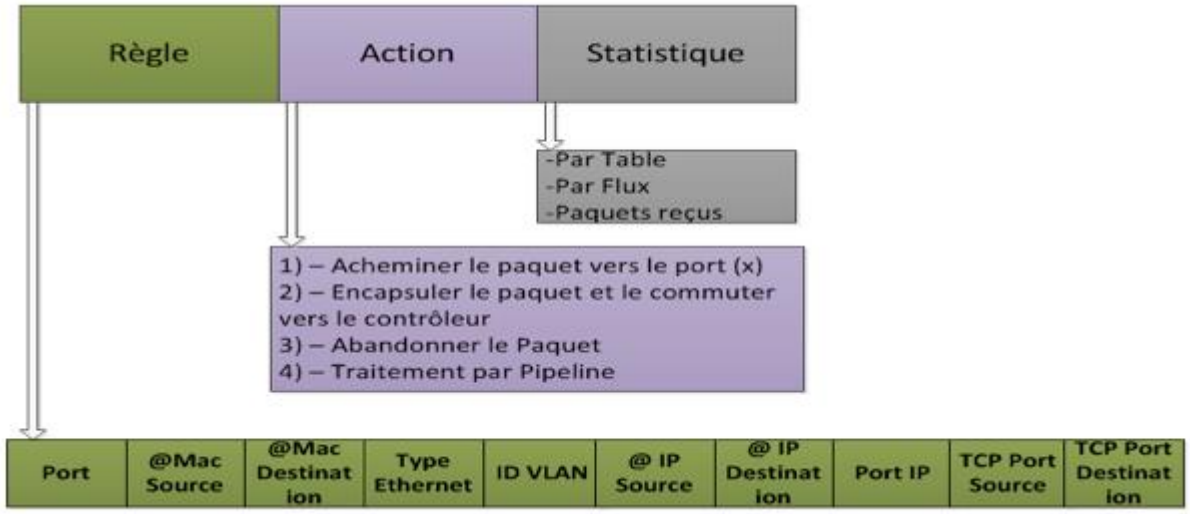

Figure 16[20]: Correspondance de paquet

Processus de traitement d'un paquet dans un Switch OpenFLow

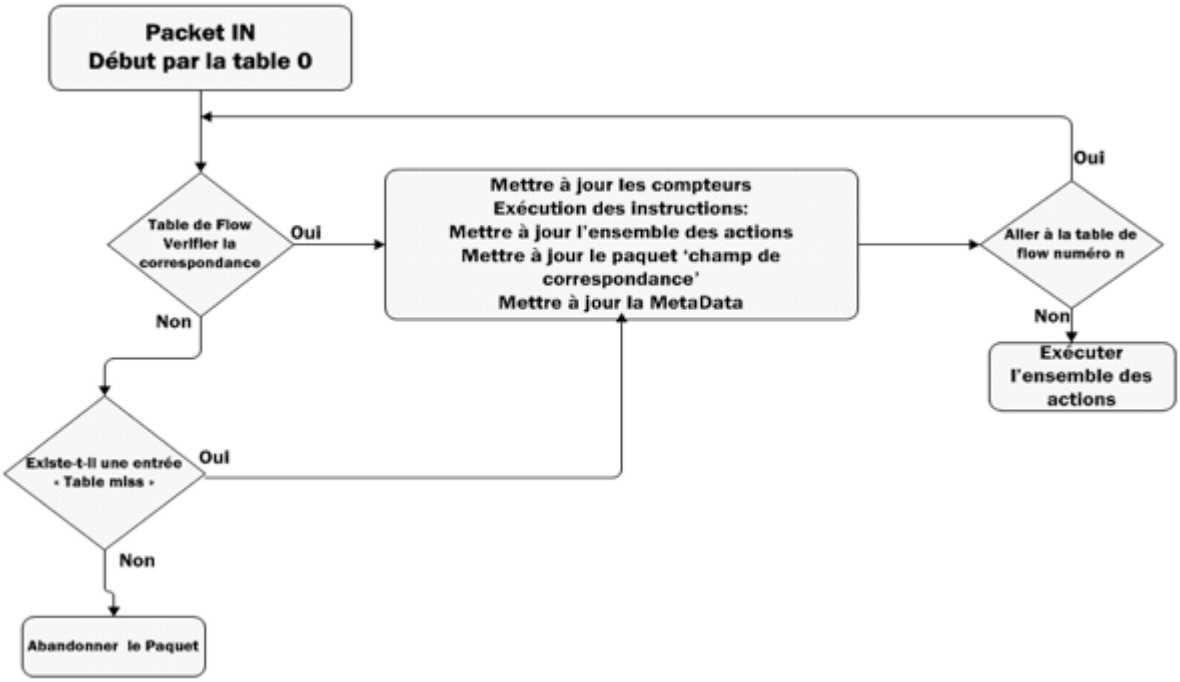

Figure 17[18]: Processus de traitement d'un paquet

A la réception d'un paquet, le Switch OpenFlow réalise les fonctions montrées dans les figures précédentes. Le Switch commence par faire une recherche dans la première table

de Flux, et, en se basant sur le traitement par pipeline, il peut aussi effectuer une recherche dans les autres tables de Flux.

Les instructions

Chaque Entrée dans la table de flux contient un ensemble d'instructions qui seront exécutées quand un paquet correspond à cette entrée. Parmi les instructions supportées on trouve :

- Apply-Action : appliquer l'action spécifique immédiatement, sans aucun changement du Action set. Cette instruction peut être utilisée pour modifier le paquet entre deux tables de flux ou pour exécuter plusieurs actions du même type. Les actions sont spécifiées comme un action set.
- Clear-Actions : Effacer toutes les actions dans l'action set immédiatement.
- Write-Actions: Fusionner les actions dans l'action set actuel, si une des actions du même type existe déjà dans l'action set actuel, l'écraser, autrement l'ajouter.
- Write Metadata : Ecrire la valeur dans le champ Meta-data.
- Goto-Table : Indique la prochaine table de Flux dans le traitement pipe-line.

Action set

Un ensemble d'actions associées qui s'accumulent au fur et à mesure que le paquet avance dans la chaine de traitement. Pipeline est traité par chaque table de flux. Quand une instruction ne contient pas un **Goto-Table**, le traitement pipeline s'arrête et les actions dans le set actions sont exécutées.

Ce set est vide par défaut. Une entrée dans la table de Flux peut modifier l'action set en utilisant l'instruction Write-Action ou Clear-Action associée à un match. Quand plusieurs actions du même type sont requises, l'instruction Apply-Actions peut être utilisée.

Les actions dans une action set sont appliquées dans l'ordre spécifié ci-dessous :

- ✓ Copy TTL inwards
- ✓ Pop
- ✓ Push
- ✓ Copy TTL outwards
- ✓ Decrement TTL
- ✓ Set
- ✓ Qos
- ✓ Group
- ✓ Output

Traitement par Pipeline

Pipeline OpenFlow : chaque Switch contient plusieurs tables de Flux, et chaque table contient plusieurs entrées. Le traitement pipeline OpenFlow décrit et définit comment les paquets interagissent avec les Tables de Flux

Les tables de Flux d'un Switch OpenFlow sont séquentiellement énumérées, commençant par 0.

Le traitement par Pipeline commence toujours avec la première table de Flux : on fait correspondre le paquet selon les entrées de la table de Flux 0. D'autres tables de Flux peuvent être utilisées selon le résultat obtenu lors de la première correspondance faite dans la première table de Flux.

Si le paquet correspond à une entrée dans la table de Flux, l'instruction en question sera exécutée. Les instructions dans les tables de Flux peuvent explicitement diriger le paquet vers une autre table de Flux utilisant l'instruction Goto.

Une entrée x dans une Table de Flux n peut enchainer le traitement du paquet en l'envoyant vers une autre table de Flux si seulement cette dernière dispose d'un numéro n supérieur à celui de la table où l'entrée x se trouve. Ainsi la dernière table du pipeline ne peut inclure l'instruction Goto. Si l'entrée dans la table de Flux ne redirige pas le paquet à une autre table de Flux, le traitement pipeline s'arrête. Arrivé à ce stade le paquet est traité avec l'action associée.

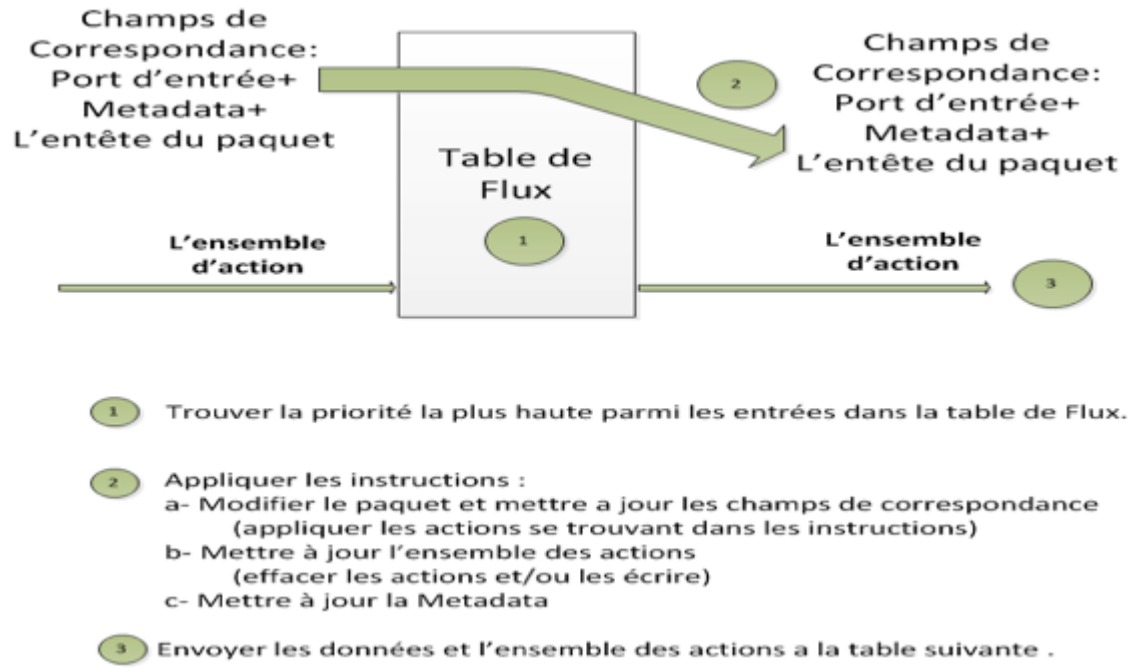

Figure 18[19]: Traitement d'un paquet

- ✓ **Les Tables de groupe**

La table de groupe contient des entrées, et chaque entrée une liste d'actions appelée Conteneur D'actions. Les actions d'un ou plusieurs **Conteneurs d'actions** sont appliquées sur les paquets envoyés au groupe.

Chaque entrée dans la table de groupe contient :

Identifiant de Groupe	Type de Groupe	Compteurs	Conteneurs D'actions

L'identifiant de groupe : c'est un entier de 32 bits.

Le Type de groupe : sert à déterminer le type du groupe « All – Select – Indirect – Fast Failover »

Les compteurs : mis à jour quand un paquet est traité par un groupe.

Conteneur d'action : un ensemble d'actions et de paramètres associés, qui sont définies pour les groupes.

d. Message OpenFLow

Le[21] protocole OpenFLow supporte trois types de messages. Messages Contrôleur vers Switch, messages asynchrone et messages symétrique, chaque type a une sous-catégorie.

Les Messages depuis le Contrôleur vers Switch

Sont initiés par le contrôleur, ils servent à gérer ou vérifier l'état du switch, ces types de messages peuvent ou non demander une réponse de la part du switch.

- ✓ **Features** : lors du Handshake le contrôleur peut demander l'identité et les capacités d'un switch en envoyant une requête.
- ✓ **Modify-State** : Ce type de message est envoyé pour gérer l'état dans les Switch. Sa fonction primaire est d'ajouter, modifier ou effacer les entrées dans les tables OpenFlow Flow/Groupe.
- ✓ **Read-state** : Ces messages sont utilisés par le contrôleur pour collecter différentes informations du switch, comme sa configuration actuelle, des statistiques et des capacités.
- ✓ **Packet-Out** : sont utilisés pour transférer les paquets reçus par les messages Packet-In. Soit ils contiennent le paquet en entier soit l'id du buffer faisant référence au paquet stocké dans le switch. Ils doivent contenir aussi une liste d'actions à appliquer, s'il n'y a pas d'action définie le paquet sera détruit.

Les Messages asynchrones

Les messages asynchrones sont envoyés par le switch vers le contrôleur pour indiquer un changement d'état ou l'arrivé d'un paquet.

- ✓ **Packet-In** : Avec ce type de message le switch transfère l'arrivé d'un paquet au contrôleur.
- ✓ **Flow-removed** : informe le contrôleur de la suppression d'une entrée dans la table de Flux
- ✓ **Port-Status** : Informe le contrôleur d'un changement sur un port du switch.

Les Messages symétriques

Les messages symétriques sont envoyés sans aucune sollicitation ni du switch ni du contrôleur.

- ✓ **Echo** : ont comme utilité la vérification de la connectivité entre switch et contrôleur.
- ✓ **Hello** : Ces messages sont échangés entre les deux une fois la connexion établie.
- ✓ **Error** : Utilisé pour signaler de part et d'autre des problèmes de connexion.

Test de connectivité

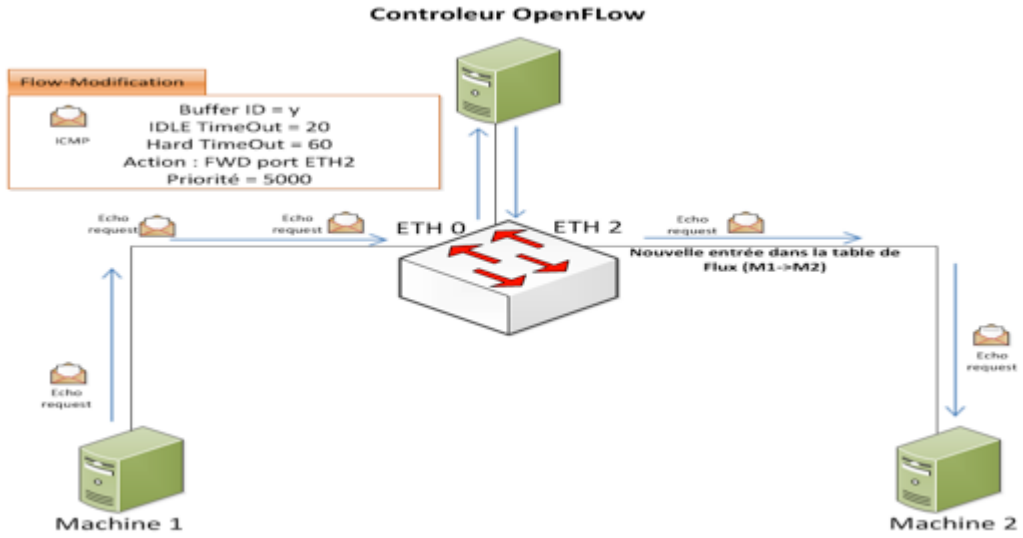

Figure 19[17]: Processus de traitement

III. Opportunités et défis des SDN

1. Opportunités

L'approche[22] SDN présente de multiples avantages pour faciliter et améliorer les tâches des administrateurs réseau. Nous allons détailler quelques uns.

Réseau programmable : avec SDN, il est plus simple de modifier les stratégies réseaux. La centralisation de la logique dans un contrôleur est entièrement personnalisé avec des connaissances globales et une puissance de calcul élevée, simplifient le développement des fonctions sophistiquées.

La flexibilité : SDN apporte également une grande flexibilité dans la gestion du réseau. Il devient facile de rediriger le trafic, d'inspecter des flux particuliers, de tester de nouvelles stratégies ou de découvrir des flux intermédiaires.

Le routage : SDN gère les informations de routage de manière centralisée en déléguant le routage et en utilisant une interface pour le contrôleur.

Politique unifiée : SDN garanti une politique réseau unifiée et à jour grâce à son contrôleur car le contrôleur est responsable de l'ajout des règles dans les commutateurs.

Gestion du cloud : SDN gère de manière très simple une plateforme cloud. La dynamique apportée par SDN traite des problèmes spécifiques au cloud tels que l'évolutivité, l'adaptation, ou des mouvements de machines virtuels.

Simplification matérielle : SDN a tendance à utiliser des technologies standards et de bases pour contrôler les équipements du réseau, tandis que la puissance de calcul n'est requise qu'au niveau du contrôleur. Ainsi, les équipements de réseau deviendront des produits à bas prix offrant des interfaces standard. Avec ce type de matériel, il serait également simple d'ajouter de nouveaux périphériques, puisqu'ils ne sont pas spécialisés, de les connecter au réseau et de laisser le contrôleur les gérer conformément à la politique définie. Ainsi le réseau devient facilement évolutif dès que le contrôleur est évolutif.

2. Les défis des réseaux SDN

Les[14] réseaux SDN ont connus plusieurs défis que ce soit sur le plan de données ou sur le plan de contrôle. Pour notre travail, nous allons présenter des défis au niveau du plan de contrôle. Ces derniers comprennent la performance, la scalabilité, la fiabilité et la sécurité.

La performance

La performance des contrôleurs SDN reste un domaine très important. Depuis l'arrivée des SDN, les chercheurs essayent toujours d'améliorer les performances des contrôleurs. La technique de SDN est basée sur les flux, du coup ces performances sont mesurées en fonction de deux métriques : le temps nécessaire pour instaurer un nouveau flux dans les commutateurs et le nombre de flux que le contrôleur peut traiter par seconde.

La scalabilité

La scalabilité ou l'évolutivité du réseau SDN est un autre défi que présente ce paradigme. Comme nous le savons, plus la taille du réseau augmente, plus des demandes sont envoyées au contrôleur. Donc à un moment donné, le contrôleur peut être incapable de traiter toutes les demandes.

La fiabilité

On fait allusion aux premiers déploiements des SDN. Un seul contrôleur est utilisé, et ce dernier est responsable de tout le réseau. S'il tombe en panne ou devient défaillant alors c'est tout le réseau qui devient indisponible.

La sécurité

Les SDN présentent un défi majeur pour la sécurité. Le fait que toute l'intelligence du réseau soit centralisée en un point qui est le contrôleur peut rendre ce dernier vulnérable. Si le contrôleur devient indisponible ou compromis tous les aspects du réseau serons endommagés c'est pourquoi il représente un point critique. Le réseau SDN est soumis à plusieurs problèmes de sécurité tels que le déni de service, l'usurpation d'identité, l'élévation des privilèges, la falsification, la répudiation, etc.

IV. Impacte des réseaux SDN

Malgré tous ses avantages, l'approche SDN a des impacts importants dans l'utilisation des réseaux. On ne peut dire des inconvénients mais des changements qui nécessitent un coût. Nous allons en citer quelques un :

Un problème de dotation qui veut dire une reformation du personnel ou recrutement de nouvel agent pour bien se familiariser à son utilisation.

Il faut une très grande réorganisation, des partages, d'information entre entreprises, d'applications, serveurs et équipes réseautages.

Comme nous l'avons évoqué ci-dessus, intervient le problème de coût pour le recyclage, une réorganisation, de nouvelle licence, perte de continuité des activités lors du déploiement initiale.

Et pour la sécurité, on peut noter que c'est une nouvelle technologie, avec de nouveau protocoles qui peut aboutir à des faiblesses et de vulnérabilité.

Et pour finir Le SDN ne permet pas de : réduire la dépendance vis-à-vis de la topologie spécifique du matériel physique sous-jacent, virtualiser toutes les fonctions et tous les composants réseau, déployer des réseaux en parallèle avec des ressources de calcul et de stockage virtualisés.

Conclusion

L'essor et l'adoption du SDN illustrent la nécessité de s'affranchir des solutions orientées matériel exigeant une configuration individuelle de chaque périphérique pour adopter des

solutions orientées logiciel, capables de fournir un véritable contrôle centralisé des services et périphériques réseau.

Cependant, il faut penser à sécuriser les réseaux SDN. La suite de notre travail va porter sur la sécurité des réseaux SDN.

Partie2: Sécurité dans les SDN

Chapitre 3 : Sécurité et vulnérabilité des réseaux SDN

Introduction

La[23] sécurité de SDN vise à protéger les actifs de SDN contre les attaquants. Un SDN sécurisé préserve la dimension sécuritaire de ses actifs. Il interdit tout accès non autorisé à ses composants, ressources, et services. Il assure l'identité de ses composants dans les différentes couches. Ces identités doivent être uniques pour éviter toute usurpation d'identité. Par exemple, un adversaire qui peut prétendre être un contrôleur légitime sera en mesure de détruire le réseau. En outre, SDN doit suivre et enregistrer les actions et événements de tous ses actifs. Ces informations sont précieuses pour surveiller le réseau et pour construire une vision globale du réseau. Ce chapitre a pour objectif d'évoquer comment la sécurité est gérée dans SDN, et d'évoquer les éventuels vulnérabilités auxquelles sont exposés ces derniers.

I. La sécurité SDN

Le[24] concept de sécurité de SDN utilise les avantages de SDN pour améliorer sa sécurité. Exemple : la centralisation donne aux applications de sécurité des connaissances globales. Cette connaissance globale peut être utilisée pour optimiser les opérations de sécurité, pour renforcer les politiques de sécurité et de réagir efficacement aux attaques du réseau. La programmabilité permet aux applications réseau de déployer leur comportement automatiquement sur les périphériques réseau. La fédération et l'externalisation fournissent des moyens pour les applications de sécurité de collaborer avec les composants SDN. En conséquence, la configuration des politiques de sécurité et la gestion devient plus simple car les administrateurs utilisent des interfaces standardisées communes de haut niveau pour exprimer leurs politiques. Le contrôleur interprète les règles de bas niveau appropriées selon l'interface appropriée vers le sud. De plus, la séparation permet au contrôleur de vérifier la cohérence et l'exactitude des politiques de sécurité pour les valider. SDN ne se contente pas sur un protocole spécifique, ni à une architecture particulière. Cette séparation permet la cohabitation de plusieurs solutions de sécurités. De plus, les applications de sécurité s'adaptent aux changements qui se produisent dans le réseau grâce à SDN. Exemple : SDN permet aux applications de sécurité de fonctionner à différents niveaux granulaires dans le réseau. Une application de sécurité peut se comporter différemment avec un trafic différent à plusieurs niveaux de sécurités au lieu d'appliquer le même comportement à tout le trafic à chaque fois.

1. Simplification et indépendance de SDN pour la sécurité

La conception et le déploiement de nouvelles solutions de sécurité sont plus simples et utiles pour la couche application. Un développeur implémente son application de sécurité dans n'importe quel langage de programmation et indépendamment du code source du contrôleur. L'API nord cache la complexité du réseau. L'application a seulement besoin d'intégrer l'interface appropriée pour communiquer avec le contrôleur.

Grâce à l'indépendance offerte par SDN, les destins des composants SDN sont séparés. Chaque actif est développé séparément des autres tandis que dans l'architecture classique, le développement était basé sur l'intégration verticale entre les trois couches du réseau. En outre, le composant SDN s'exécute dans un système et un matériel différents. Cette diversité élève la résilience de l'architecture SDN. Par exemple, une erreur dans le code source d'un composant SDN ne peut pas générer une autre erreur dans le code source d'un autre composant SDN. D'ailleurs, un attaquant qui détecte une vulnérabilité de la technologie logicielle dans un actif SDN ne pourra pas exploiter la même vulnérabilité dans d'autres actifs SDN. Il doit étudier et analyser les codes sources de tous les actifs SDN. Ainsi, l'indépendance du SDN rend ses tâches coûteuses et lourdes.

2. Agilité pour la sécurité de SDN

L'agilité SDN est la capacité de s'adapter dynamiquement aux changements du réseau. Il tire parti de l'application de sécurité avec de nombreuses fonctionnalités. SDN permet le déploiement d'applications de sécurité par la création dynamique de réseaux virtuels. Ces réseaux virtuels prennent en charge la mobilité et les flux des machines virtuelles. Il contrôle les flux de manière dynamique pour les diriger vers l'application de sécurité appropriée. De plus, SDN fournit des moyens pour adapter et déployer dynamiquement des politiques de sécurité dans des éléments de réseau. Grâce à SDN, il est désormais possible de déployer dynamiquement des applications de sécurité à la source de l'attaque, puis diriger uniquement le trafic suspect vers cette application. Les avantages de l'adaptabilité SDN sont la personnalisation des solutions de sécurité selon l'état du réseau et leur déploiement efficace de la sécurité.

3. Connaissance globale du SDN pour la sécurité

SDN permet aux applications de sécurité de programmer des éléments de réseau. La sécurité des applications s'appuient sur des interfaces SDN pour collecter l'état et les événements du réseau. Ces données sont nécessaires pour analyser le trafic et détecter les attaques de sécurité. La connaissance holistique du SDN améliore les opérations d'applications de sécurité qui

partagent la connaissance globale sans attendre la convergence et le temps d'adapter leur comportement. Par exemple, si une attaque est détectée, l'alerte est partagée sur toutes les applications de sécurité. Ce partage de connaissances peut être utilisé pour adapter la sécurité d'un comportement global en fonction de l'état du réseau.

Contrairement à SDN, les applications de sécurité dans l'architecture conventionnelle n'ont que des vues partielles du réseau. Elles ne collaborent pas entre elles. En conséquence, leurs fonctions sont limitées par leurs portées étroites. De plus, leur connaissance partielle affecte la cohérence de la sécurité du réseau pour deux raisons. Le comportement d'une application de sécurité peut se chevaucher avec celle d'une autre application de sécurité telle qu'une application de sécurité qui traite le trafic qui a déjà été traité par un autre. L'autre raison concerne les conflits de comportement. Les politiques des applications de sécurité peuvent contredire les politiques de sécurité d'une autre.

La connaissance globale de SDN résout ces problèmes. Le contrôleur utilise ses connaissances holistiques pour vérifier, valider et renforcer les politiques de sécurité. L'utilisation des connaissances mondiales pour les politiques est très utile pour les pare-feu et les applications de surveillance de la sécurité.

4. Convergence SDN pour la sécurité

SDN automatise la sécurité. Il permet aux applications de sécurité de reprogrammer les éléments du réseau de manière proactive en fonction de leurs comportements de sécurité. Dans ce cas, une application de sécurité installe dynamiquement son comportement sur les éléments du réseau avant les événements qui le déclenchent. Outre, les applications de sécurité réagissent efficacement aux événements du réseau car le contrôleur interprète leur actions aux règles de sécurité et les installe sur les éléments de réseau à côté de la source de l'attaque. Le résultat de cette automatisation est une résolution rapide des attaques réseau.

De plus, SDN propose aux applications de sécurité des éléments de réseau hautement reprogrammables et la capacité de rediriger dynamiquement le trafic réseau. En conséquence, la configuration et le déploiement des politiques de sécurité dans le plan de données deviennent moins sujettes aux erreurs et efficaces. Contrairement aux réseaux classiques, dans SDN, l'utilisateur exprime ses politiques de sécurité, puis le contrôleur les propage automatiquement sur les entités de réseau appropriées. Une application de sécurité peut surveiller le trafic, puis met à jour les politiques de sécurité pour améliorer la sécurité du réseau.

Lorsque l'administrateur modifie les politiques, le contrôleur effectue automatiquement les mises à jour sans avoir besoin d'interventions manuelles sur chaque périphérique réseau, comme

dans le réseau classique. Cette flexibilité des politiques réduit les temps d'arrêt et améliore le diagnostic des problèmes de sécurité.

5. Pare-feu dans SDN

Un pare-feu est un mécanisme de sécurité qui protège un réseau contre tout accès non autorisé. Il filtre le trafic en faisant correspondre les en-têtes des paquets avec un ensemble de politiques de sécurité pour autoriser uniquement le trafic actuel pour accéder au réseau. SDN élève les pare-feu avec toutes ses fonctionnalités. Sa simplification permet aux administrateurs de gérer les politiques de pare-feu sans se soucier de la complexité de l'infrastructure sous-jacente. Son agilité adapte dynamiquement les règles de pare-feu en fonction des états du réseau. La connaissance globale permet aux applications de pare-feu de diffuser leurs règles tout autour du réseau. L'interopérabilité permet aux applications de pare-feu d'installer leurs règles et de collaborer via une API commune vers le nord. Enfin, l'automatisation permet à une application de pare-feu d'installer ses règles de façon autonome et de les gérer sans l'intervention des administrateurs.

Cependant, avec la centralisation du contrôleur, les réseaux SDN sont très vulnérables à plusieurs niveaux. La suite de cette partie va porter sur les vulnérabilités les plus populaires.

II. Vulnérabilité des réseaux SDN

La[20] sécurité de SDN vise à protéger les actifs de SDN contre les attaquants. Un SDN sécurisé préserve la dimension sécuritaire de ses actifs. Il interdit tout accès non autorisé à ses composants, ressources, et services. Il assure l'identité de ses composants dans les différentes couches. Ces identités doivent être uniques pour éviter toute usurpation d'identité. Par exemple, un adversaire qui peut prétendre être un contrôleur légitime sera en mesure de détruire le réseau. En outre, SDN doit suivre et enregistrer les actions et événements de tous ses actifs. Ces informations sont précieuses pour surveiller le réseau et pour construire une vision globale du réseau.

En outre, les informations sont précieuses pour la sécurité. SDN doit également protéger ses actifs informationnels en les gardant secrets pour les entités non autorisées. SDN doit empêcher un attaquant d'abuser de l'intégrité de ses composants, ses informations et ses ressources.

Les communications dans SDN doivent être sécurisées. Toutes ses interfaces et leurs protocoles doivent protéger leurs communications.

La question n'est pas de savoir si l'organisation va se faire attaquer mais quand et comment cette attaque aura-t-elle lieu ? C'est pourquoi il est très important de trouver les possibilités de

défaillances. De nos jours les réseaux sont sources de menace à plusieurs niveaux. Dans la suite de cette partie nous allons apporter les vulnérabilités les plus connues de ce nouveau paradigme.

1. Les menaces basées sur l'externalisation SDN

La[13] séparation du plan de contrôle et du plan de données introduit de nouvelles menaces sur l'accès au contrôleur, aux communications SDN et à la distribution du contrôleur. L'externalisation du contrôleur ouvre l'accès à la couche de contrôle contrairement au réseau classique où le contrôle a été scellé et couplé dans un périphérique réseau. Un attaquant identifie et pénètre aux composants de la couche de contrôle sans passer par la complexité de toutes les couches comme dans l'architecture de réseau classique.

Dans SDN, l'attaquant peut étudier et attaquer la couche contrôle sans passer par la complexité du plan de données, dans l'architecture classique ses attaques doivent au moins prendre en compte le plan de données. Elle introduit des vulnérabilités qui peuvent conduire à des attaques importantes telles que la corruption du contrôleur, son usurpation d'identité, ou même briser l'ensemble du réseau. Dans les réseaux classiques, l'attaque des liens nécessite un accès physique aux périphériques réseaux. Cependant, dans SDN, grâce à l'externalisation, ces liens se dévoilent. Ils ne sont plus internes et cachés. Ils deviennent canaux réseau qui sont vulnérables à de nombreuses attaques de sécurité. Par exemple, Un attaquant peut voler des données sur ces liens en les identifiant et en y accédant à distance.

La distribution des contrôleurs élargit aussi le paysage de menace de SDN. L'externalisation du contrôleur permet à l'attaquant d'étudier le contrôleur pour se faire passer pour lui.

2. Les menaces basées sur la centralisation

La centralisation du contrôleur SDN réduit la résilience du réseau et introduit de nombreuses vulnérabilités (perturbations) du contrôleur. Cette centralisation fait du contrôleur un point unique point d'échec en raison de la forte dépendance des composants SDN. L'attaquant voit le contrôleur comme cible facile qui lui donne des privilèges. Bien que la centralisation donne aux applications réseaux des connaissances globales. Un attaquant peut collecter des informations pour forger une solide connaissance de ses cibles potentielles. Dans un autre cas, un attaquant qui arrive à falsifier ses informations va corrompre la vue du contrôleur et les opérations des composants SDN dans les autres couches. Si ce dernier arrive à interrompre le contrôleur centralisé alors il peut provoquer la panne de tout le réseau. Il peut aussi utiliser la dépendance et la concentration pour transformer les autres actifs de SDN en vecteurs d'attaque contre le contrôleur centralisé.

3. Les menaces basées sur la fédération

L'unification des communications entre les différentes couches de SDN introduit de nouvelles vulnérabilités dans SDN. L'homogénéisation des interfaces SDN réduit la complexité des attaques qui les ciblent ou en abusent, tandis que dans le réseau existant, leur hétérogénéité améliore leur résilience.

Grâce à la fédération, l'attaquant cible une interface ouverte et standardisée qui est partagée par de nombreux composants SDN. Il devient plus facile pour l'attaquant d'étudier l'interface, détecter ses vulnérabilités et les exploiter pour réussir ses attaques. Par ailleurs, la fédération crée un grand vecteur d'attaque qui connecte chaque point de vulnérabilité des interfaces.

Ce lien entre toutes les couches SDN peut être utilisé par un attaquant comme passage souterrain pour attaquer les composants SDN tout en se cachant derrière leurs pairs légitimes. Grâce à l'interopérabilité apportée par la fédération dans SDN, l'attaquant est désormais en mesure de parler à tous les composants SDN de n'importe quelle couche. Il s'appuie sur les interfaces pour traduire ses politiques malveillantes sur n'importe quelle couche SDN. En outre, l'attaquant dans SDN peut abuser de la collaboration entre les applications SDN et les contrôleurs pour interrompre, corrompre et voler les composants SDN. Il peut même mal utiliser une interface pour empoisonner un autre composant SDN. Par exemple, il peut injecter les informations de fausse topologie de l'API vers le nord pour une application SDN.

4. Les menaces basées sur la programmabilité SDN

La mise en réseau par logiciel permet de programmer le réseau avec des applications logicielles. En conséquence, un attaquant peut corrompre une application SDN (ou injecter une application malveillante) pour reprogrammer le comportement des éléments du réseau. La programmabilité peut ajouter du carburant aux flammes de l'attaquant parce que l'adversaire peut renverser les opérations de d'autres applications dans SDN. Par exemple, il programme le réseau avec des politiques contradictoires qui peuvent contredire les règles juridiques. Les attaques peuvent devenir persistantes et dynamiques car la programmabilité SDN remplace entièrement la gestion du réseau par l'automatisation. Un attaquant peut injecter du code malveillant dans une application légitime pour réagir automatiquement aux événements du réseau.

L'application malveillante reprogramme dynamiquement les éléments du réseau lorsqu'elle observe les événements de réseau. Même si ses politiques disparaissent, son comportement se manifestera à nouveau en observant les événements du réseau. De plus, les pannes dans les applications peuvent affecter l'ensemble du réseau car les applications peuvent programmer des éléments de réseau et manipuler le trafic automatiquement. Le contrôleur devient un point de

défaillance unique. Un attaquant peut utiliser cette vulnérabilité pour épuiser les ressources du contrôleur avec des stratégies malveillantes via l'API vers le nord. Il peut également amplifier son comportement malveillant sur tout le réseau, les éléments qui sont connectés au contrôleur en les programmant à la volée. Le moteur de décision est réparti entre le contrôleur et le réseau périphériques en mode déclaratif. Un attaquant, dans ce cas, peut détourner les éléments du réseau sans que le contrôleur en soit conscient. Il peut abuser des éléments du réseau comme des zombies pour attaquer les composants SDN ou empoisonner la connaissance holistique du contrôleur. Ce dernier inclut toutes les informations des règles OpenFlow existantes dans les éléments du réseau.

III. Les attaques

1. Attaques de reconnaissance

Les attaquants utilisent des attaques de reconnaissance SDN pour collecter des informations sur les composants SDN. Ces derniers exploitent ces informations pour détecter les vulnérabilités de SDN et attaquer le système avec des attaques actives. Ils collectent des données réseau classique et des informations spécifiques au SDN. C'est permis par l'état du port, les informations de QoS, la topologie et d'autres informations. L'attaquant rassemble des informations spécifiques au SDN. Ces informations peuvent être de la taille de la table de flux, le contenu de la table de flux, le type de contrôleur, la liste des différentes interfaces SDN et autres données SDN. L'attaquant envoie des sondes de balayage (trafic falsifié) vers des éléments du réseau. Il surveille le trafic de retour des sondes et déduit le contenu des tables de flux dans les éléments du réseau. L'attaquant cible le canal de communication entre le contrôleur et les éléments réseau. Il surveille leurs interactions pour en savoir quels paquets déclenchent l'installation de nouvelles règles OpenFlow. Pour y remédier une solution a vu le jour. Il s'agit de la contre-mesure. C'est un mécanisme de retardement de paquet. Les premiers paquets de règles OpenFlow installées sont retardés à l'intérieur des éléments du réseau. A partir de ce niveau, la précision de l'attaque dimunie et les performances du réseau sont alertées. La solution est une méthode de numérisation appelée méthode de numérisation de changement d'en-tête pour déduire le comportement des éléments réseau en observant les temps de réponse de ses paquets de sondages. La contre-mesure appliquée est d'activer les caractères génériques des règles OpenFlow pour compresser le nombre de flux gérés par chaque règle. Dans ce cas, les temps de réponse des paquets de sondage deviennent stables. Cette solution s'appuie sur deux techniques. La première est l'empreinte digitale. Cette technique déduit les valeurs de timeout et le temps de traitement du contrôleur. Ensuite, elle compare les valeurs des temps des

différents contrôleurs à l'aide d'une base de données de référence pour trouver le contrôleur. La deuxième technique déduit de la connaissance du paquet pour identifier le type de contrôleur. Elle analyse le contenu et les intervalles LLDP qui se différencient d'un contrôleur à un autre. Ces informations sont comparées avec la base de données du contrôleur de référence pour déduire le type de contrôleur.

2. Attaques d'accès

Les attaques d'accès SDN abusent des dimensions d'authentification et de contrôle d'accès de SDN. Ils utilisent des techniques telles que le déguisement, Man-in-the-middle, élévation des privilèges, confiance et attaques par mot de passe. Les détails de ces attaques SDN sont les suits.

- ✓ Attaques de camouflage SDN: dans les attaques de camouflage, l'attaquant prétend être un Composant SDN. Ensuite, il se fait passer pour l'identité d'un contrôleur SDN, d'un élément de réseau ou d'une application SDN pour corrompre le SDN.
- ✓ Attaques SDN Man-in-the-middle: elles visent les communications SDN via les interfaces SDN. L'attaquant se place à l'intérieur de l'interface SDN entre deux composants SDN tout en permettant à ces actifs de communiquer. Ensuite, il écoute ou modifie les communications à l'intérieur de l'interface.
- ✓ Attaques d'élévation de privilèges SDN: elles permettent à l'attaquant de modifier ses privilèges d'accès et ses limites de confiance au sein des composants SDN. Elles exploitent le manque de confiance des mécanismes dans SDN et les vulnérabilités dans la station centrale. Ces attaques donnent à l'attaquant le contrôle total du SDN.
- ✓ Attaques de confiance et de mot de passe: dans les attaques de confiance, un attaquant compromet une confiance de composant SDN. Puis, il abuse de ses frontières de confiance pour pénétrer le domaine SDN. Les attaques par mot de passe brisent les mots de passe des utilisateurs. Ensuite, ils utilisent ces mots de passe pour accéder aux composants SDN. Ils exploitent les vulnérabilités du domaine d'administration. La mise en œuvre de mécanismes de contrôle d'accès SDN réduit la surface d'attaque SDN. Cependant, la littérature manque d'ouvrages qui proposent des solutions de contrôle d'accès pour les composants SDN. Certains chercheurs discutent en général sur l'introduction du contrôle d'accès et de l'authentification dans la conception d'architecture SDN et mettre en œuvre leurs idées. D'autres travaux se concentrent sur le contrôle d'accès des Applications SDN pour protéger le contrôleur. Opération Checkpoint est une solution de contrôle d'accès SDN qui autorise uniquement les

applications SDN autorisées pour accéder aux ressources du contrôleur. Pour chaque application SDN, la solution définit une liste d'autorisation des opérations du contrôleur. Opération Checkpoint contrôle l'accès en direction de l'API nord selon cette liste. Si une application tente d'accéder à des ressources non autorisées, la solution empêche l'accès et enregistre l'événement pour profiler l'application.

3. Attaques de perturbation SDN

Les attaques de perturbation constituent une boîte de pandore dans la couche de contrôle car le contrôleur est un point de défaillance unique. Ces attaques de perturbation peuvent avoir trois aspects dans SDN tels que les données mortes, le refus d'accès et les inondations. Lors de la première attaque, un attaquant exploite les vulnérabilités du code source des composants SDN pour trouver lequel est mal formé. Les données provoquent la corruption du comportement d'un composant SDN ou sa terminaison. Les attaques par déni de services refusent l'accès légitime d'un composant SDN aux ressources des autres composants SDN ou du réseau. L'attaquant effectue ces attaques en neutralisant les données de la victime. De plus, ces attaques peuvent également manipuler les politiques du réseau. Cette attaque transforme les éléments du réseau à une forte inondation qui épuisent le contrôle de la liaison de données et les ressources du contrôleur.

4. Les attaques par téléportation

Elles[13] consistent à exploiter le contrôleur pour relayer une information d'un nœud à un autre dans le plan de transfert. Pour ces attaques, deux nœuds du plan de transfert peuvent trouver un moyen de s'échanger des informations sans utiliser les liens du plan de transfert. Ainsi ces informations qui seront partagées peuvent avoir des impacts négatifs comme :

Coordination et rendez-vous : elles peuvent être utilisées comme protocole de prise de rendez-vous pour se découvrir et faire une synchronisation pour la mise en place d'une attaque.

Exfiltration : c'est une possibilité d'un transfert d'information d'un réseau à un autre qui dispose pas de connectivité dans le plan de données donc qui ne devraient pas pouvoir communiquer.

Mise à l'écoute et altération des données : utilisée en cas de collision entre un hôte et un commutateur. C'est une attaque qui est particulièrement puissante car un commutateur et un hôte peuvent utiliser l'attaque « main-in-the-middle » contre les hôtes bénignes, avec des pages web malicieuses par exemple. La figure ci-dessous illustre cette attaque.

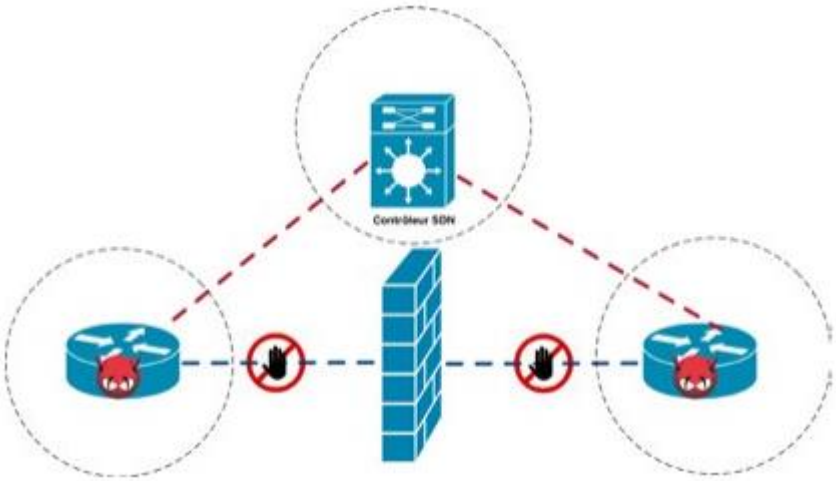

Figure 20[13]: Architecture de la téléportation

Trois techniques ont été identifiées pour mettre en place une attaque par téléportation : (re-)configuration de flots, identification des commutateurs et la transmission hors bande.

a. (Re-) configuration des flots

Nous savons que le contrôleur doit réagir à différents évènements du plan de données, comme l'arrivées de nouveaux flots, les pannes etc., et ce afin de (re-)configurer le routage des flots du réseau.

Path-update : il s'agit de la mise à jour des chemins qui est une fonctionnalité fondamentale et qui est présente dans la plupart des contrôleurs industriels. Elle supporte des fonctionnalités comme la mobilité, la migration de machines virtuelles. Son fonctionnement est le suivant : un contrôleur maintient typiquement des associations entre hôtes et ports correspondants du commutateur. Si un hôte apparait soudainement derrière un autre commutateur, le contrôleur installe de nouvelles règles pour l'hôte dans le nouveau commutateur et efface les règles correspondantes dans le commutateur précèdent. C'est alors dit une mise à jour de chemin dite « path-update » qui est déclenchée. Elle fait usage de message OpenFlow de type Packet-in, Flow-mod et Packet-out. Les commutateurs malicieux peuvent utiliser le « path-update » à plusieurs reprises pour se transmettre une information via la génération de « path-update ». Pendant un « path-update » le Packet-out est envoyé à la destination indiquée par le packet-in, ce qui peut générer du trafic au plan de données.

Path-reset : pour fournir de la haute disponibilité, un contrôleur doit surveiller les états du réseau et peut mettre en œuvre des ré-optimisations comme re-router des flots ou effacer des

règles au niveau des commutateurs. Un contrôleur peut apprendre qu'une partie d'un chemin assigné n'est plus disponible suite à la réception d'un packet-in et peut donc initier la reconfiguration ou réparation du chemin. Le path-reset implique des messages OpenFlow de type Packet-in, Flow-mod et Packet-out. Des commutateurs malicieux peuvent donc utiliser le path-reset pour une téléportation implicite : si le contrôleur réinitialise un chemin complet entre deux hôtes quand il reçoit un packet-in d'un commutateur qui ignore la règle sur le flot, une information peut donc être communiquée. En répétant telle opération plusieurs fois, un commutateur malicieux peut téléporter de l'information à d'autres commutateurs au long du chemin.

b. Identification de commutateurs

Une autre particularité dans les réseaux SDN, c'est que les commutateurs sont responsables de s'identifier de façon unique auprès du contrôleur avec le OpenFlow DatapathID (DPID) de 64 bits : typiquement, les premiers 48 bits sont personnalisables. A cet effet, un commutateur peut inférer un (mauvais) usage d'un même DPID par un autre commutateur. En utilisant un ou des DPID déjà connus, une paire de commutateurs malicieux peuvent donc mettre en œuvre une téléportation. Exemple : le commutateur 1 dit au contrôleur que son DPID est 1. Plus tard, le commutateur 2 dit lui aussi que son DPID est 1. A ce moment le contrôleur ne permet pas au commutateur 2 de se connecter avec un tel DPID : l'information contenue par DPID peut ainsi être transmise du commutateur 1 à 2 via le contrôleur, et ainsi de suite avec d'autres DPID utilisés par le commutateur. On peut donc imaginer un transfert d'information aussi en exploitant les 16 bits personnalisables du DPID, outre simplement utiliser ce canal de signalisation pour des signaux binaires (DPID utilisé ou non).

c. Transmission hors-bande

Le[13] contrôleur SDN a une fonctionnalité qui répond à des évènements venant du plan de données avec une règle de traitement de flots. Cette fonctionnalité peut être exploitée par la téléportation. Un paquet, en prévenance d'un commutateur peut potentiellement atteindre d'autres commutateurs dans le réseau via le plan de contrôle en mettant donc en œuvre une transmission hors-bande (out-of-band) dans le sens où la transmission ne se fait pas au niveau des liens du plan de données. Il existe plusieurs modes travail selon lesquels un packet-in venant d'un commutateur vers le contrôleur génère d'autres messages OpenFlow du contrôleur vers d'autres commutateurs. Il s'agit là de l'installation de règles en mode proactif : quand un nouveau flot arrive au sein d'un commutateur, le packet-in qui arrive au contrôleur génère des

Flow-mod vers le commutateur source mais aussi vers les commutateurs qui se trouvent sur le chemin calculé pour le flot par le contrôleur.

5. Les attaques par fabrication de liens

Ce[13] sont des attaques qui sont récentes et qui visent à faire croire au contrôleur qu'un lien direct existe entre deux commutateurs alors qu'il n'en existe pas en réalité. La majorité des contrôleurs SDN apprennent les connexions entre les commutateurs en utilisant le protocole LLDP. Les commutateurs voisins observeront ces messages LLDP et les renverront au contrôleur, c'est ce qui lui permet de connaitre le lien en commutateur. C'est un processus de découverte qui attire les attaquants. Donc un attaquant peut abuser de ce processus en générant, répliquant relayant des messages LLDP. Un attaquant avec une connexion à plusieurs commutateurs SDN peut observer des messages LLDP diffusés et relayer le message d'un commutateur à un autre. Et si jamais il arrive que ce relais ait lieu, le contrôleur croira qu'un lien existe entre les commutateurs. Le danger réside sur l'interception du trafic qui sera à travers ce lien et l'attaquant agira en tant que « main-in-the-middle ». Pour la réalisation d'une telle attaque, il suffit qu'un hot soit connecté à deux commutateurs. L'hôte pourra donc bridger ses deux interfaces. Exemple : deux hôtes malicieux qui communiquent avec une interface sans fil, un attaquant peut se passer aussi de la contrainte physique d'avoir deux interfaces filaires vers des commutateurs SDN, avec donc une grande distance entre les deux commutateurs. La figure ci-dessous illustre cette attaque.

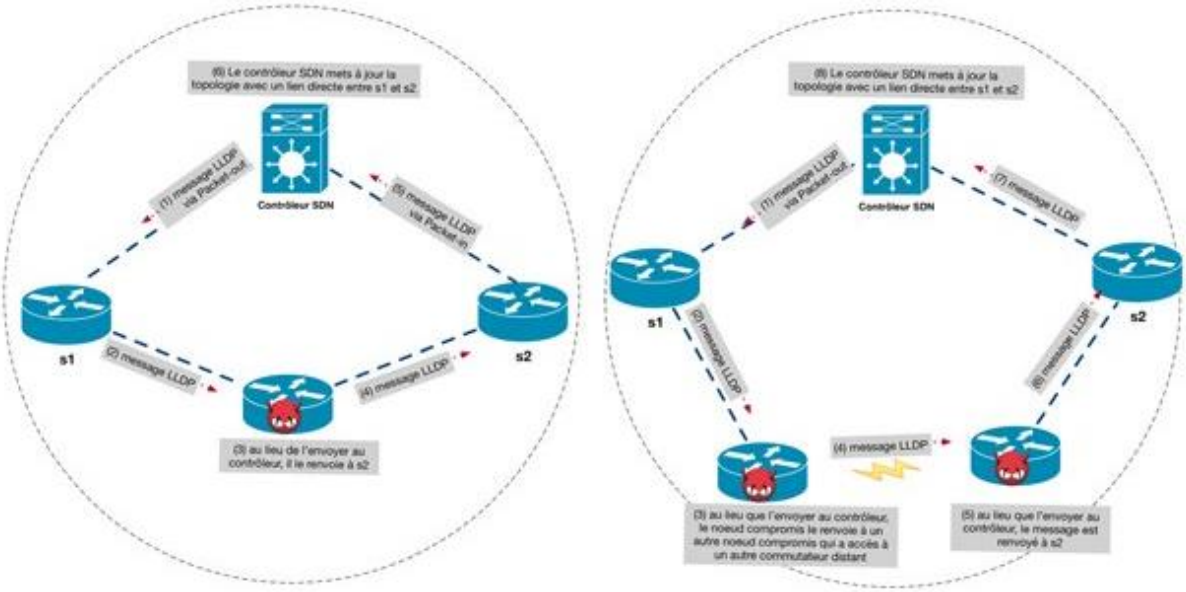

Figure 21[13]: Architecture de la fabrication de lien

IV. Les attaques de déni de services

1. Présentation

C'est en principe une[20] attaque qui vise à rendre une application, un site web, un serveur etc. incapable de répondre à la demande de ses utilisateurs légitime. Pour ce faire, l'attaquant va inonder sa cible avec beaucoup de trafic de flux jusqu'à ce qu'elle ne parvient plus à répondre. Une illustration de cette attaque est au niveau de cette image suivante.

Figure 22[20]: Déni de service

On peut noter aussi les dénis de services distribués. Dans ce cas d'attaque, l'attaquant exploite à distance plusieurs ordinateurs infectés via un cheval de Troie ou autres pour inonder sa cible simultanément. Cette attaque est très complexe, il est très difficile de repérer la source de l'attaque et de faire la différence entre une requête légitime ou non.

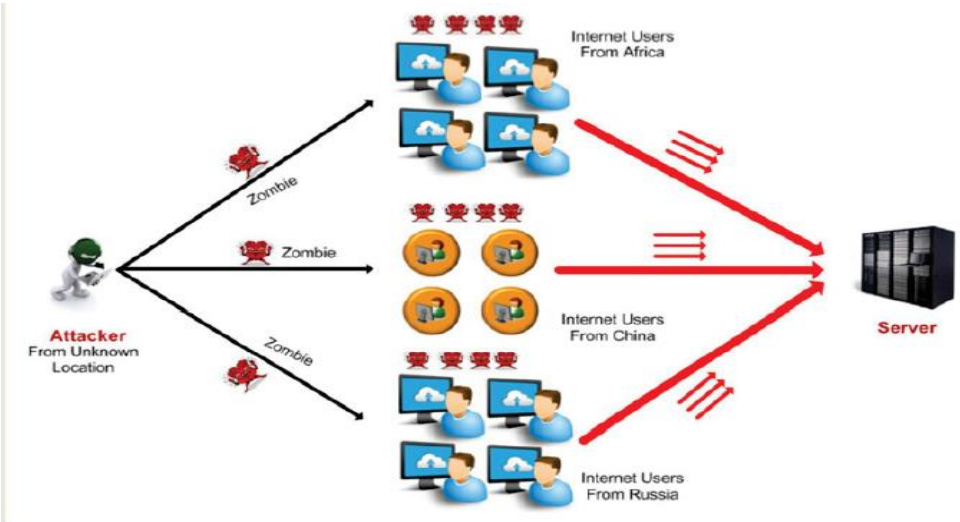

Figure 23[20]: Déni de service distribué

Le déni de service est une attaque qui impact au niveau de trois parties dans les réseaux SDN : c'est soit sur le réseau, au niveau du plan de données ou au niveau du plan de contrôle.

2. Impact des attaques de déni de service

Vue la centralisation du contrôleur dans le plan de contrôle, il s'avère très grave si les attaquants parvenaient à paralyser ce dernier. Dans ce cas, ces attaques pourraient affecter le contrôleur, la liaison entre le contrôleur et les commutateurs et le plan de données (commutateurs et liens du réseau).

a. Impact des attaques de déni de service sur le plan de contrôle

Que ce soit avec le déni de service ou déni de service distribué, l'attaquant va envoyer une énorme quantité de flux. Et à chaque fois que les commutateurs reçoivent de nouveaux paquets, font appel au contrôleur pour une demande de règles à appliquées. Par conséquent, le contrôleur va recevoir beaucoup de demandes venant des commutateurs et donc sera surchargé. Si le contrôleur est surchargé, alors on peut noter une lenteur au niveau de la réponse des demandes de règles, dans ce cas la décision de routage ne sera bien prise en charge.

b. Impact des attaques de déni de service sur le plan de données

Les commutateurs enregistrent toujours les règles de commutation dans leurs tables de commutations et les utilisent jusqu'à l'expiration des temporisateurs (idle timeout et hard timeout). Si le commutateur reçoit une quantité importante de flux, alors il sera obligé de la stocker dans sa mémoire TCMA. Une fois cette mémoire remplie, le commutateur sera dans l'obligation d'ajouter, de supprimer et d'envoyer continuellement les règles de flux et d'envoyer plus de demande au contrôleur.

c. Impact des attaques de déni de service sur la liaison contrôleur-commutateur

Il existe une communication importante entre le contrôleur et les commutateurs demandant des décisions de routage. La liaison entre le contrôleur et les commutateurs sera exténuée et congestionnée. Ceci peut causer la perte de plusieurs messages et même un retard de réponse des messages échangés entre ces derniers.

Conclusion

Après cette longue étude des vulnérabilités qu'ont ces réseaux SDN, nous avons orienté notre travail sur celles basées sur les dénis de services. C'est une menace populaire qui suscite

beaucoup de débat. Pour contourner ces attaques, nous allons faire appel aux systèmes de détections d'intrusions. Le chapitre suivant va porter sur l'étude des IDS.

Partie 3 : Contribution

Chapitre 4 : Les IDS et le DdS

Introduction

Les SDN, comme toutes nouvelles approches, restent vulnérables aux menaces des attaquants. Pour contourner ces attaques, nous allons solliciter les services des systèmes de détection

d'intrusion pour identifier les flux et reconnaitre ceux qui sont légitimes. Ce chapitre porte sur la première partie de la solution qui porte sur les IDS. Nous allons introduire les IDS, étudier leur fonctionnement puis quelques-uns et en fin faire un choix d'outil de travail.

1. Le Déni de service dans les SDN

Les[20] attaques par déni de service sont inévitables et restent les plus populaires comme elles l'ont toujours été pour les réseaux classiques. Dans un réseau SDN, les attaques de déni de service (DdS) entrainent les problèmes suivants :

- ✓ La surcharge des commutateurs: pour les réseaux SDN, chaque commutateur possède une mémoire de stockage qui est bien déterminé pour le stockage des règles de flux dans les tables de commutation. Lors d'une attaque de déni de service, un attaquant va envoyer un grand nombre de flux pour inonder les commutateurs. Vu que les commutateurs ont une capacité de stockage alors la mémoire sera remplie. Une fois ce stade atteint, les commutateurs seront forcés à ajouter et supprimer continuellement des entrés des flux dans leurs tables et d'envoyer d'énorme demande au contrôleur.
- ✓ La surcharge du contrôleur : ce nouveau paradigme qui est SDN est conçu de tel sorte que seul le contrôleur peut être responsable de la décision du routage des flux dans le réseau. C'est pourquoi, quand le commutateur reçoit un flux alors il vérifie dans sa table de flux et si une règle de correspondance n'est pas trouvée donc il fait appel au contrôleur pour une demande de règle. Pendant une attaque de déni de service, un nombre énorme de paquets est envoyé par l'attaquant, donc nécessitant des demandes de règles de flux pour joindre une destination. Par conséquent le contrôleur sera surchargé. Les messages de demande de règles de flux vont saturer la file d'attente au niveau du contrôleur. Arrivé à ce stade, il sera difficile de prendre une décision de routage. Dans ce cas, les flux sans règle seront supprimés ou bloqués dans la file d'attente des commutateurs, soit dirigés vers une route par défaut.
- ✓ L'épuisement de la bande passante entre contrôleur et commutateur : pendant les attaques de déni de service, il y a une communication excessive entre les commutateurs et le contrôleur. Pour les nouvelles connexions, nous notons beaucoup de demandes de règles de flux pour envoyer les paquets à destination. Donc la liaison entre commutateur et contrôleur peut être congestionnée et des demandes de règles peuvent être perdues. Cela perturbe le réseau et des décisions de routage pour les flux seront toujours mises en attente, ou même perdues.

2. Eviter la propagation de requêtes non autorisées

Dans les SDN, ils arrivent que la même requête légitime soit soumise à plusieurs reprises pour être traitée. On peut noter également, la transmission et le traitement de requêtes illégitimes

- ✓ **Les requêtes légitimes** : quand le contrôleur reçoit une requêtes légitimes et que la réponse tarde à être envoyée (du fait que le contrôleur est occupé), la même requête peut être renvoyée à plusieurs reprise, créant ainsi une saturation aussi bien qu'au niveau des commutateurs qu'au niveau du contrôleur.
- ✓ **Les requêtes illégitimes:** pour une requête illégitimes, le commutateur ne disposant pas à priori la règle, va l'acheminer vers le contrôleur. Et pour un attaquant bien averti, l'objectif sera d'envoyer de fausses requêtes qui seront retransmises vers le contrôleur, qui à son tour va les traiter avant de les supprimer. Ce qui va entrainer la retransmission et traitement de fausses requêtes.

La figure suivante donne une parfaite illustration de ce que nous venons de dire. Les flèches en bleu représentent les requêtes légitimes alors que les requêtes illégitimes sont représentées en rouge.

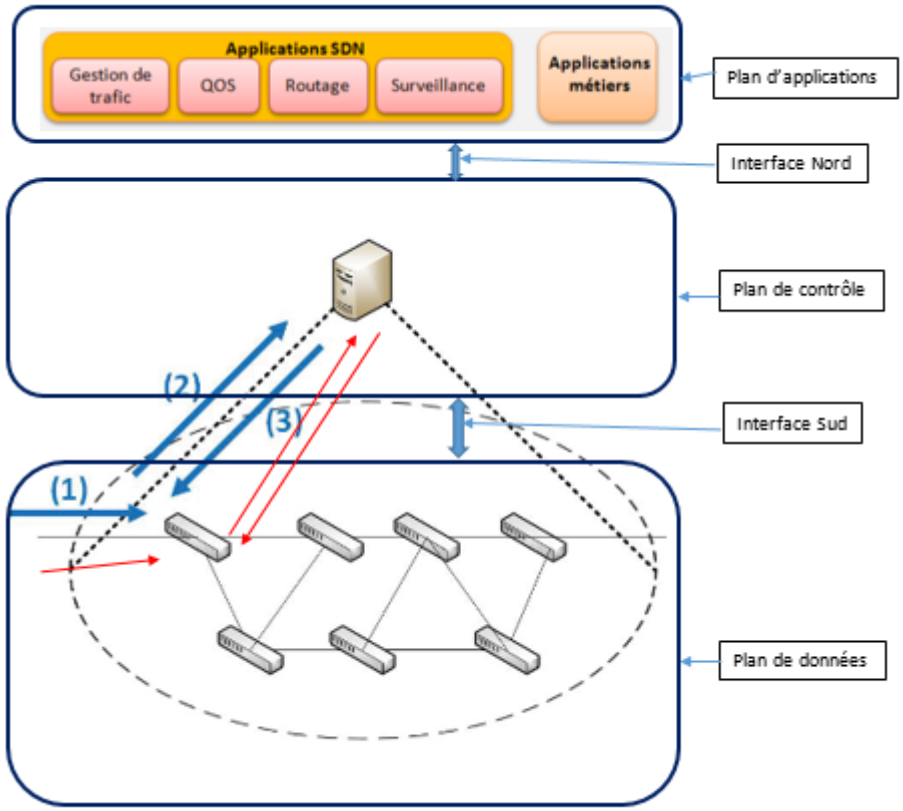

Figure 24: Schéma de la problématique

Notre objectif est d'éviter tout traitement redondant d'une requête légitime, et toute retransmission ou traitement d'une requête illégitime

Pour empêcher la perturbation des SDN par ces attaques de déni de services nous avons fait recourt à d'autres technologies. Premièrement nous avons sollicité les services des IDS (système de détection d'intrusion) qui sont parvenu à résoudre les problèmes. Cependant, les IDS ont présenté des limites relatives à la capacité de traitement des requêtes. Pour pallier à ses limites nous nous sommes penchés sur l'échantillonnage de trafic en utilisant des outils intégrés. La suite de ce mémoire porte sur les IDS

I. Les IDS

1. Définition

IDS (intrusion detection system) est un ensemble de composant matériels et logiciels qui permettent de détecter des activités anormales au sein du réseau. Une fois mis en place, un IDS assure la prévention des anomalies.

2. Principe de fonctionnement des IDS

La[25] détection d'intrusion se base sur des calculs. Elle se repose sur des principes. Nous allons l'illustrer par une image.

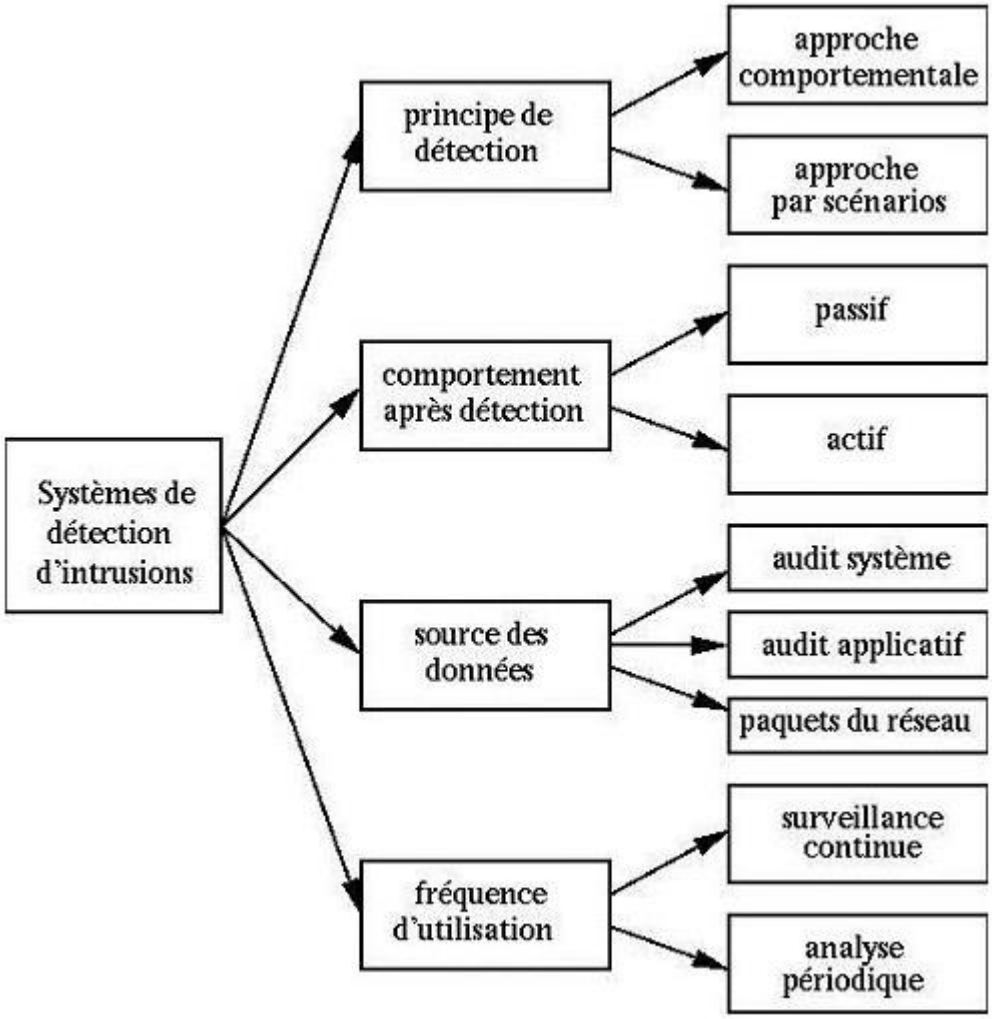

Figure 25: principe de fonctionnement de la détection

3. Principe de détection des intrusions

Un système de détection d'intrusion ou IDS se base sur deux approches. Il s'agit de l'approche par scénario appelée aussi signature et l'approche comportementale.

a. Approche par scénario ou signature

C'est une technique qui s'appuie sur la connaissance des méthodes utilisées par les pirates pour en faire une déduction. Elle utilise des signatures d'attaques existantes. Cette approche se base sur :

- ✓ **La recherche de motifs (pattern matching)**

Cette méthode est plus connue et est plus facile à comprendre. Elle a comme point fort la recherche de motif au sein des flux. L'IDS a une base de données qui contient un ensemble de signature où chaque signature contient les protocoles et les ports utilisés par une attaque spécifique et un motif pour reconnaitre les paquets suspects.

Pour cette méthode, seules les attaques reconnues par les signatures seront détectées. Il est donc obligatoire de mettre à jour régulièrement la base de données des signatures. Une petite différence dans la base des signatures laisse passer une intrusion.

- ✓ **Recherche de motifs dynamiques**

Comme son nom l'indique, elle est pareille à la précédente. La seule différence est qu'elle évolue dynamiquement. Cette approche est dotée de fonctionnalités qui facilitent l'adaptation et l'apprentissage.

- ✓ **Analyse de protocole**

C'est une méthode qui se base sur une vérification de conformité des flux et sur une observation des champs et paramètres suspects. Cette approche permet de détecter des attaques inconnues.

- ✓ **Analyse heuristique et détection d'anomalie**

Cette approche procède à une analyse intelligente qui facilite la détection d'une activité suspecte ou toute autre anomalie.

b. Approche comportementale

C'est une approche qui permet la détection d'intrusion en se basant sur le comportement de l'utilisateur. Pour cette technique, un profil utilisateur est défini au préalable et une alerte est déclenchée lors d'un changement inhabituel. C'est une technique qui s'applique sur les utilisateurs et les applications. Cette méthode possède des inconvénients :

Peu fiable : tout changement dans le profil provoque une alerte

Il faut une période de non fonctionnement pour la mise en œuvre des mécanismes

Beaucoup plus d'attention dans l'établissement du profil pour éviter les fausses alertes.

Toutes ces approches permettent la détection d'intrusion de types déni de service. Le choix de notre approche va dépendre du type d'IDS que nous allons adopter.

II. Différents types d'IDS

Un IDS est conçu en fonction du besoin. Compte tenu des composants d'un système, il peut se situer à plusieurs niveaux. On note différents types d'IDS :

- ✓ NIDS Network IDS, système de détection d'intrusion réseau ;
- ✓ HIDS Host IDS, système de détection d'intrusion de type hôte ;

- ✓ Les autres types d'IDS sont des dérivés de ces deux grandes familles. Il s'agit des IDS hybrides et IPS.

Les IDS affichent une disponibilité sous deux formats. Ils sont soit logiciels ou Appliance.

1. NIDS

Un[26] NIDS permet une analyse passive des flux en transite au sein du réseau et détecte les intrusions en temps réel. Il a comme principe d'écouter tout le réseau puis de l'analyser et de générer des alertes au cas où des paquets semblent être dangereux.

2. HIDS

Les[27] HIDS, quant à eux sont placés le plus souvent sur des machines sensibles, qui risquent de subir des attaques et qui contiennent des données sensibles pour la structure.

Pour ne pas faillir à sa mission, un HIDS a besoin d'un système qui lui est sain afin de vérifier l'intégrité des données.

3. IDS Hybrides

Ils sont le plus souvent utilisés dans des environnements décentralisés, les IDS Hybrides permettent de réunir des informations provenant d'un NIDS et d'un HIDS.

4. IPS

C'est une technique qui va au-delà des IDS. C'est des outils aux fonctions actives qui permettent d'empêcher toute activité suspecte détectée au sein du système. Les IPS assurent la détection et tentent de bloquer les intrusions.

NIPS

Un NIPS a pour rôle principal de surveiller le trafic d'un réseau. Il permet de bloquer des attaques réseau. Son fonctionnement se repose sur une étude de la correspondance entre les champs des paquets avec celle des empreintes d'intrusions déjà contenus dans les règles.

III. Inconvénient des IDS

La mise en place d'un IDS nécessite une bonne connaissance en sécurité. Pour configurer un IDS et l'administrer, il faut beaucoup de temps et une bonne maitrise de la sécurité. Malgré tous les avantages que présentent ces IDS, ils[26] présentent des inconvénients :

Les faux positifs : ce sont des alertes provenant d'un IDS ou IPS et qui ne correspond pas à une attaque

Faux négatif : c'est quand une intrusion passe non aperçu par l'IDS ou l'IPS.

1. Pollution et ou surcharge

Un[28] IDS ou IPS peut être pollué. Cela est possible par une génération importante de trafic. Dans ce cas nous assisterons à une saturation des ressources.

2. Consommation de ressources

La détection d'intrusion est un mécanisme très gourmand en ressources. Il doit toujours générer des journaux de compte rendus des activités.

3. Perte de paquets

Il arrive que les vitesses de transmissions soient supérieures aux vitesses d'écritures des disques durs ou du traitement des processeurs. C'est pourquoi des paquets seront perdus.

IV. Etude des IDS

L'interconnexion des systèmes ainsi que leurs ouvertures au réseau internet permettent facilement aux malfaiteurs d'avoir une main mise sur un système donné. Nous notons des attaques nombreuses et diversifiées. Outre les pare-feu et les systèmes d'authentification, de nos jours il est nécessaire de mettre en place un système de détection/prévention d'intrusion. Les IDS sont indispensables pour la sécurité du réseau. Ils permettent la détection d'intrusion en se basant sur une signature ou l'apprentissage d'un comportement pour les flux présents dans le réseau. Plusieurs outils sont de nos jours disponibles. Nous allons définir quelques un et porter notre choix sur l'un d'entre eux. Dans ce travail, nous nous intéressons qu'au mode NIDS.

1. Snort

C'est un[29] système de détection et de prévention des intrusions (NIDS/NIPS), créé par Martin Roesch en 1998. C'est le système de détection d'intrusion le plus utilisé. Grace à sa version commerciale, Snort détient une très bonne réputation auprès des entreprises. Il est capable d'effectuer une analyse du trafic réseau en temps réel. Il[30] est aussi doté de plusieurs technologies de détection d'intrusions. Snort se base d'un langage de règle permettant de décrire le trafic qui doit être collecté et accepté. Il peut fonctionner en trois modes :

- ✓ Sniffer des paquets : ici Snort observe les paquets et les affiche d'une façon continue comme *tcpdump* ;

- ✓ Logger des paquets : là aussi, il enregistre les paquets dans des répertoires sur le disque. Ce mode est utilisé pour limiter les logs avec certains critères comme une plage d'adresse IP ou un protocole ;
- ✓ NIDS : dans ce mode, Snort analyse le trafic réseau, compare ce trafic à des règles déjà définies par l'administrateur du réseau et établit des actions à exécuter.

La figure suivante permet une parfaite illustration du fonctionnement de Snort.

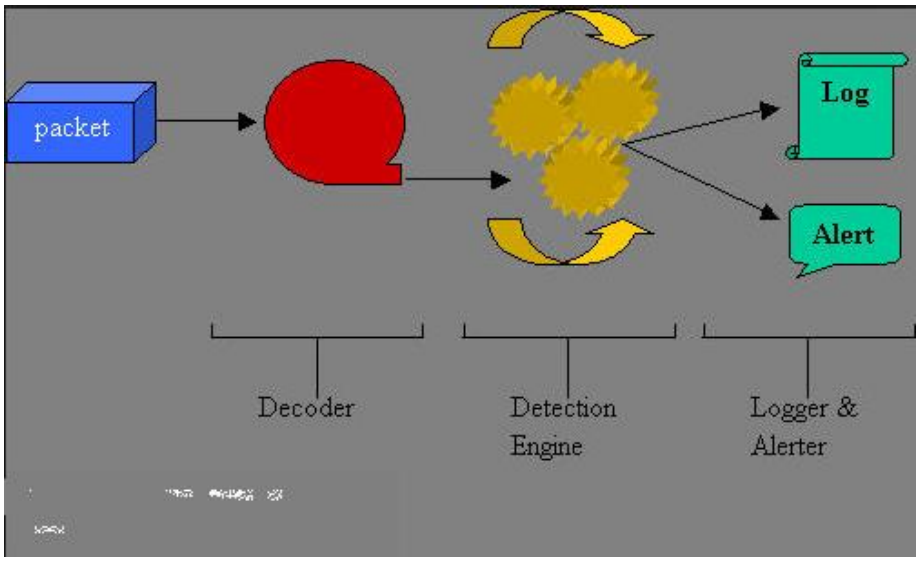

Figure 26: Architecture de Snort

2. Bro

C'est un[31] système de détection d'intrusion réseau (NIDS) open source qui est basé sur UNIX et qui a été fondé par Vern Paxson en 1998. Il surveille passivement le trafic réseau afin de trouver des flux malveillants. Il[32] analyse le trafic réseau pour détecter les intrusions, puis exécute des analyseurs orientés événements qui comparent l'activité avec des motifs jugés malveillants. L'analyse du trafic suspect comprend la détection des attaques spécifiques (signature et évènement) et des activités inhabituelles (anormales). Bro est capable de donner une analyse détaillée aux évènements qui décrivent l'activité observée. Il contient un ensemble de scripts de politique qui sont conçus pour détecter les attaques internes les plus courantes et tout en limitant le nombre de faux positifs. Ces scripts sont des programmes écrits en langage Bro. Ils[33] contiennent des règles qui décrivent le type du trafic ou les activités qui sont considérés comme malveillantes. La figure suivante illustre le mode de fonctionnement de Bro.

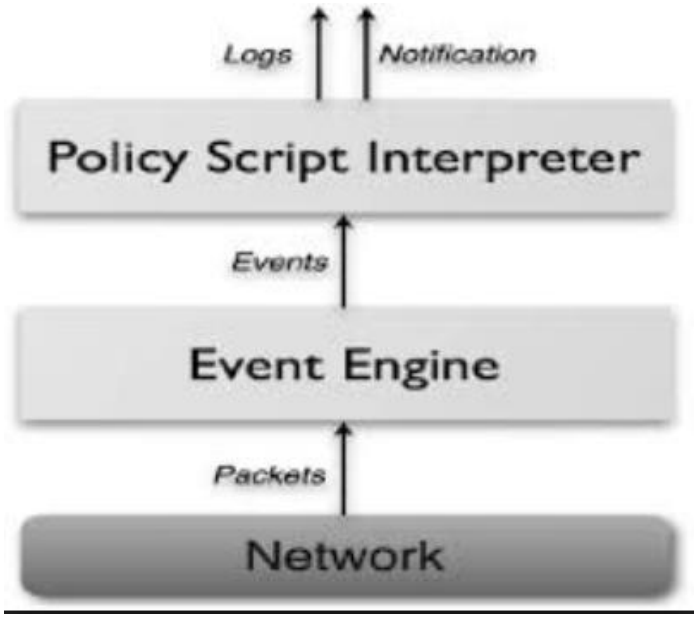

Figure 27[33]: Architecture de Bro

3. Suricata

C'est un[34] système de détection et de prévention des intrusions. Il est open source et est développé par l' « open information security foundation – OISF ». La version bêta a été diffusée en décembre 2009 alors que la version stable a eu lieu en juillet 2010. Suricata a été créé pour apporter de nouvelles idées et technologies au domaine de détection d'intrusion. L'OISF fournit à Suricata un ensemble de règles de détection et de prévention des intrusions. Ce[35] système est capable d'utiliser des règles à partir d'autres systèmes tels que Snort pour fournir le meilleur ensemble de règles possibles. Suricata surveille le trafic réseau comme tous les autres systèmes de détections d'intrusion puis crée des alertes et des journaux d'évènements lorsque le trafic malveillant est détecté.

La liste n'est pas exhaustive, mais nous allons nous limiter là et faire une comparaison entre Snort et Bro pour le choix de notre outil.

4. Tableau comparatif entre Snort et Bro

Paramètres	Bro	Snort
Signatures contextuelles	Oui	Non
Personnalisation du site	Haute	Moyenne
Capacité réseau à grande vitesse	Haute	Moyenne
Grande communauté d'utilisation	Non	Oui
Analyse graphique	Quelques	Beaucoup

Compatibilité du système d'exploitation	Unix	Tout

Tableau 1:Tableau : tableau comparatif entre Snort et Bro

Nous portons notre choix sur Bro. En plus des résultats du tableau comparatif, Bro est compatible avec les règles de Snort grâce à un convertisseur nommé snort2bro. Cependant, Bro seul ne peut pas résoudre à tous les problèmes. Nous allons faire appel à d'autres technologies pour une meilleure gestion de ces menaces. Nous penchons notre choix sur l'outil BroFlow qui utilise Bro et l'échantillonnage de trafic.

Conclusion

Pour résumer, ce chapitre nous donne une très bonne information sur les IDS. Ayant une connaissance solide de ces outils, nous pouvons facilement faire un choix selon nos besoins. Cependant ces outils présentent des limites, c'est pourquoi le chapitre suivant va porter sur la capture et l'analyse avec BroFlow.

Chapitre 5 : Solution contre le déni de service

Introduction

Les IDS, à eux seulement, ne permettent pas d'apporter une solution complète contre les attaques de déni de service. Pour une bonne maitrise de ces attaques nous avons fait appel aux services de BroFlow qui se base sur l'outil Bro. Dans ce chapitre, nous allons présenter

BroFlow. Il s'agit de la présentation de son architecture et les différentes fonctionnalités offertes.

I. Présentation générale de Broflow

BroFlow[36] est un mécanisme basé sur l'API OpenFlow et sur l'analyseur de trafic réseau Bro. Il implémente différents algorithmes de détection d'anomalie contre les attaques de déni de services. BroFlow effectue les contre-mesures pour bloquer les attaques. Il utilise un commutateur programmable logiciel OpenvSwitch (OVS) qui est un commutateur OpenFlow. OVS présente une table de transfert qui pourrait être mise à jour par un contrôleur OpenFlow. BroFlow dispose des capteurs répartis dans le réseau. Pour la détection d'attaque, il est possible d'établir un nombre réduit de capteurs au lieu de placer des capteurs dans chaque commutateur. Cet avantage est accordé en combinant l'outil Bro avec la vue globale du réseau fournie par OpenFlow. Le choix de Bro se justifie par sa description élevée des politiques à travers le langage. Avec le langage Bro, l'utilisateur peut définir ses propres politiques. De plus, Bro inspecte le trafic réseau en temps réel, en créant des rapports et alarmes lorsqu'une politique de sécurité est menacée. La figure 28 suivante permet une parfaite illustration de ce texte.

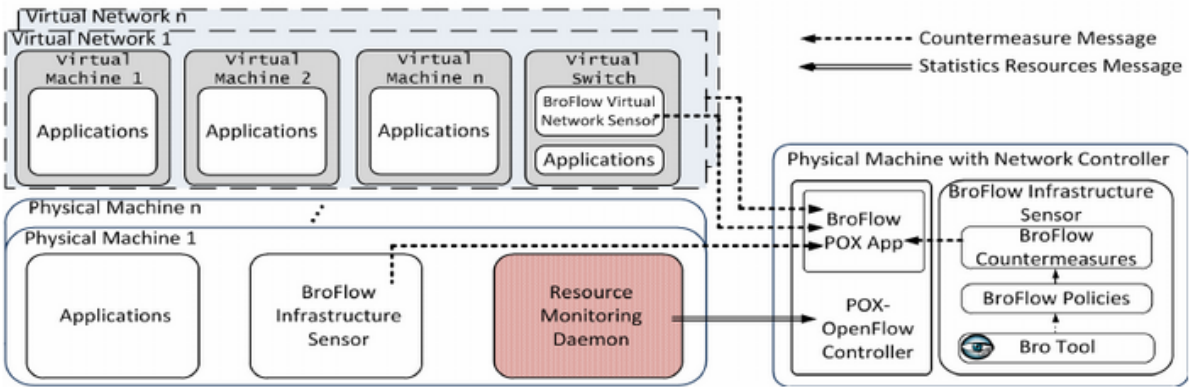

Figure 28[37]: Architecture BroFlow

1. Les capteurs BroFlow

Le[38] système BroFlow possède deux types de capteurs : le BroFlow Virtual Network (VN) pour les réseaux virtuels et le capteur d'infrastructure BroFlow. Chaque capteur exécute un outil Bro avec une consommation minimale de ressources. Les capteurs VN surveillent les commutateurs virtuels ou des hôtes spécifiques. Ils doivent être répartis géographiquement entre les commutateurs virtuels. Dans chaque capteur VN, des politiques spécifiques sont établis pour chaque réseau virtuel. Tous les capteurs BroFlow ont une application de détection

d'attaque et de répartition d'alarmes qui sont définies par deux modules : module de politique et de contre-mesure.

Le capteur d'infrastructure BroFlow quant à lui fonctionne parallèlement avec le contrôleur afin de protéger le réseau physique. Il détecte facilement et empêche les attaques de type inondation ARP. Les paquets capturés sont envoyés au moteur d'évènements qui, les vérifie, les ordonne et les convertit en évènements. Ensuite ces évènements sont envoyés à l'interpréteur de règles. Le moteur d'évènement et l'interpréteur de règles appartiennent à l'outil d'analyseur de trafic Bro. Le module de politique BroFlow décide sur les évènements générés par Bro celui représentant une attaque et dans ce cas, quelle action l'application doit prendre.

2. Les politiques BroFlow

Les[39] politiques de sécurité sont composées de deux modules : Network Event Inspection (NEI) et Attack Detection (AD). Le module NEI analyse en temps réel les informations pertinentes fournies par l'outil Bro sur les flux établis lors de la réception des paquets. Toutes les politiques de détection d'attaques de déni de services sont écrites en langage Bro. Ainsi, chaque fois qu'un paquet lié aux évènements décris dans les politiques est détecté par le module NEI, le module AD est appelé. Le module AD implémente des algorithmes, résumés dans des politiques de langage Bro.

3. Les contre-mesures BroFlow

Le[36] module de contre-mesure effectue la communication avec l'application BroFlow. Ce module traduit les informations générées par les capteurs BroFlow et transmet les messages d'alarme à l'application BroFlow. Lorsqu'une attaque est détectée par le module politique, elle envoie un message d'alarme au contrôleur. Pour garantir une communication authentifiée et cryptée, un canal Secure Socket Layer (SSL) est utilisé au niveau des interfaces réseau dédiées. Les messages sont sous format JSON et incluent les informations de flux, les adresses IP et ports, source et destination, l'adresse MAC de destination et la contre-mesure à prendre. Ces champs sont tous des informations que l'outil Bro obtient à partir d'un paquet. Comme ces champs ne composent pas un flux complet d'OpenFlow alors OpenFlow remplit les autres champs avec des valeurs génériques.

4. Application BroFlow

L'application[37] BroFlow s'exécute au sommet du contrôleur. Cette application reçoit des alarmes dérivées de plusieurs capteurs BroFlow et exécute les contre-mesures requises pour répondre à ces alarmes. Lorsqu'un message d'alarme est reçu du module de contre-mesure des

capteurs, l'application BroFlow vérifie dans son tableau quels flux correspondent au contenu du message d'alarme. Après cela, l'application indique le contrôleur la contre-mesure à prendre dans tous les commutateurs réseau. Chaque fois qu'une contre-mesure est appliquée, une minuterie est activée. Quand une minuterie éclate, la contre-mesure est nettoyée et toutes les analyses sont rétablies. Il est possible de détecter si l'attaque se termine, cessant d'utiliser les ressources systèmes.

a. Module de gestion des flux

Pour éviter la surcharge des capteurs il faut gérer les flux. Un module de gestion des flux est créé dans l'application BroFlow. Il est responsable de la distribution des flux entre les capteurs BroFlow. Pour la mise en miroir de paquets entre les capteurs BroFlow, le tunnel Generic Routing Encapsulation (GRE) est utilisé. Bien que GRE augmente la quantité totale de données dans le réseau, cette approche allège les charges d'inspection de chaque machine. L'inspection des paquets se fait après la décapsulation, assurant l'intégrité des paquets. Cette mise en miroir permet de placer les capteurs dans différents réseaux virtuels hébergés sur différentes machines physiques. Donc, la distribution des flux prend en compte les ressources du système disponible dans chaque machine virtuelle et la source du paquet. Un flux d'une nouvelle source est alloué dans le traitement le plus bas et les flux provenant de la même source sont alloués ensemble dans la même machine empêchant les attaques de passer inaperçues.

b. Module gestion des ressources

Situé dans le cadre privilégié de machine physique, ce module surveille les ressources du système tel que la bande passante, le traitement et la mémoire de chaque machine physique. Les statistiques de toutes les machines physiques sont agrégées dans le contrôleur. Ainsi, le contrôleur dispose des informations sur la disponibilité des ressources de chaque machine analysée. En cas de surcharge, ce module analyse les ressources disponibles dans les machines physiques et décide où instancier un nouveau capteur BroFlow. De même, toutes les machines physiques contenant les capteurs BroFlow sont analysées ensemble, afin de détecter quand une redistribution de flux est possible, permettant de désactiver une machine.

Cependant, malgré tous les avantages que présente BroFlow, il a des limites qui peuvent l'empêcher de répondre à nos attentes. Pour en citer ces limites, nous allons nous référer dans la partie inconvénients des IDS énoncée précédemment. Ces inconvénients sont valables pour tous IDS. En plus de ses inconvénients, BroFlow ne prend pas en compte la défaillance d'un capteur. Ceci étant, il faut songer à une politique de capture des flux à analyser. La suite de ce document va porter sur l'échantillonnage de trafic.

BroFlow permet de contourner les attaques de déni de services. Cependant, il y a beaucoup de trafic à analyser. Pour identifier le trafic en provenance d'une source légitime ou non, il faut tout analyser. A un moment donné, le trafic à analyser peut dépasser les capacités de stockages des disques des capteurs ou la puissance de calcul. C'est pourquoi il faut palier cette faiblesse. Dans la partie suivante, nous allons aborder l'échantillonnage de trafic.

II. Echantillonnage comme solution : SFLOW

Avec la grande quantité d'information capturée par l'IDS, il s'avère très difficile à analyser et donner des résultats en temps réel. C'est pourquoi il est important de filtrer le trafic qui doit être capturé. Ce chapitre porte sur l'échantillonnage de trafic.

1. Classification du trafic

Pour répondre aux besoins des utilisateurs, il est impératif de connaitre la nature des trafics des différentes activités dans le réseau. Pour ces raisons, la classification du trafic d'internet suscite un intérêt particulier. En effet la classification du trafic constitue une activité cruciale dans toutes les activités d'ingénierie de trafic et de gestion des réseaux. Pour répondre adéquatement aux différentes exigences d'applications circulant dans le réseau, ces mécanismes doivent se référer à la classification du trafic afin de pouvoir assigner chaque flot à la classe de service appropriée et lui attribuer ainsi une priorité adéquate. Les flots appartenant à une classe sont traités différemment par rapport aux autres selon les niveaux de qualité de services prédéfinis. La classification est une composante essentielle dans le système de sécurité, en particulier dans les mécanismes de détection d'intrusion, d'utilisation injustifiée des ressources réseau ou trafic malicieux ainsi que les fonctions de sécurité conventionnelle tels que les pare-feu.

L'identification et la classification du trafic ont connu plusieurs stades d'évolution. Au début, ces activités étaient faciles et précises du fait que la plupart des applications utilisaient des numéros de ports bien connus. Toutefois cette situation n'a pas duré très longtemps dû à l'apparition et la popularisation rapide des applications utilisant des ports dynamiques ou non standard. A partir de ce moment, la classification est devenue incontournable.

2. Classification basée sur les ports

Les[40] premières solutions de classification de trafic se basent sur les numéros de ports pour classifier les applications. C'est une approche très simple et rapide. Elle classifie efficacement les applications standards vue qu'elles utilisent des ports assignés. Bien que cette approche

présente des avantages, elle est de nos jours inefficace avec la présence des applications non standards telle que le trafic P2P (Peer to Peer).

3. Classification par l'inspection de charge

Pour pallier la limite précédente, une autre approche dite classification par l'inspection de charge a vu le jour. Elle consiste à examiner la charge utile de chaque paquet dans sa recherche d'un indice ou la signature de l'application. Cette approche se décline en deux branches : Inspection profonde de paquets (DPI pour Deep Packet Inspection) et inspection stochastique de paquets (SPI Stocastic Packet Inspection).

La[40] première branche repose sur une inspection mécanique de la charge utile de chaque paquet lors de la recherche d'une expression ou d'un mot clé caractérisant une application donnée. De plus, elle est susceptible d'être utilisée dans un processus de classification de trafic en ligne car la signature peut être déduite à partir des premiers paquets des flux. Cette approche est implémentée dans plusieurs solutions, telle que la détection d'intrusion, les pare-feu. Néanmoins, elle souffre de plusieurs problèmes, en particulier celui du trafic encrypté.

La deuxième branche reprend le même principe de base qui est l'inspection de la charge, mais d'une manière statistique de façon à chercher les propriétés distinctives de chaque application. C'est une approche qui vise à combler certaines lacunes de la première branche. Pour ce faire, elle utilise des méthodes automatiques pour former des modèles distinctifs. Bien que les méthodes SPI permettent de distinguer la nature de plusieurs types de contenu de trafic, y compris le contenu chiffré ce qui est utile pour prioriser un trafic par rapport à un autre. Ces méthodes héritent plusieurs problèmes des méthodes DPI et elles sont incapables d'identifier le type de trafic du fait que certaines applications peuvent utiliser plusieurs types de contenus en même temps.

4. Approche comportementale

C'est[40] une approche qui se base sur le comportement des hôtes et de la distribution des connexions pour pouvoir déduire le type de trafic avec le but d'identifier les applications actives sur hôte donné. Elle examine le comportement du trafic en observant un certain nombre de paramètres, tels que le nombre d'hôtes qui y sont connectés, le nombre de ports et les protocoles utilisés. L'idée derrière une telle approche, est que, différentes applications génèrent des comportements différents.

5. Approche statistique

C'est[40] une approche qui s'appuie d'un côté sur les techniques d'apprentissage machine et d'un autre côté sur le fait que les différents types de trafics possèdent différentes caractéristiques ou métadonnées telles que la taille des paquets, la taille des flux, le temps inter arrivé des paquets, la durée des flux, etc.

III. Echantillonnage de trafic

L'analyse et la supervision des réseaux sont deux tâches indispensables dans le processus et la gestion des réseaux. Elles constituent une activité incontournable. L'analyse se base essentiellement sur la collecte et l'extraction des informations contenues dans les paquets acheminés sur le réseau. Il est à noter que le volume du trafic véhiculé sur les réseaux modernes devient de plus en plus important. Le coût de la classification et de la visualisation du trafic devient aussi important.

L'évolution d'internet et l'apparition de nouvelles technologies permettent d'utiliser des réseaux haut débit. Ainsi de nouveaux enjeux apparaissent, notamment la gestion des données massives. Avec ces changements, l'analyse du trafic en utilisant l'approche classique n'est plus possible à cause des quantités de données. Ceux-ci peuvent occasionner des perturbations comme la saturation des équipements. En effet, la collecte de tout le trafic n'est pas seulement coûteuse durant le processus de traitement, mais il peut consommer beaucoup bande passante. Les équipements peuvent être surchargés et la bande passante significativement affectée si les données collectées doivent être envoyées vers un serveur distant.

Plusieurs techniques d'échantillonnage ont été abordées par la littérature. Nous pouvons citer une liste non exhaustive en particulier, l'échantillonnage systématique, l'échantillonnage aléatoire et l'échantillonnage aléatoire adaptatif.

1. Echantillonnage systématique

C'est[40] une technique qui suit une règle très simple. Les échantillons sont sélectionnés d'une manière périodique selon une fonction déterministe. En effet, chaque $k^{ème}$ élément est sélectionné à partir d'un point de départ choisi au hasard entre 1 et k. cette technique peut se faire de manières :

- ✓ Soit à base de paquets, dans ce cas l'échantillon est formé par la sélection de $k^{ème}$ paquet ;

- ✓ Soit à base du temps, dans ce cas la sélection d'un paquet est effectué chaque période de temps.

L'échantillonnage systématique est facile à effectuer et assure une bonne précision. C'est une technique qui assure une bonne répartition des éléments dans l'ensemble des échantillons. Par contre elle risque de biaiser les données à cause de la périodicité.

2. Echantillonnage aléatoire

a. Echantillonnage aléatoire simple

Cette[40] technique consiste à choisir des paquets (ou flux) du transfert de tel sorte que tous les éléments ont la même chance de figurer dans l'échantillon. C'est une technique qui est parfois appelée n-out-of-N. Les éléments sont divisés en un ensemble de groupe de N. Et pour chaque groupe, n éléments sont sélectionnés aléatoirement.

b. Echantillonnage probabiliste

C'est une technique qui choisit les éléments en fonction d'une probabilité prédéfinie. Pour un échantillonnage probabiliste dit uniforme, chaque paquet est sélectionné indépendamment avec une probabilité fixe. L'autre échantillonnage probabiliste dit non uniforme, c'est quand la probabilité dépend de l'entrée.

c. Echantillonnage stratifié

C'est une technique qui consiste à diviser les éléments en groupes homogènes (strates) puis à sélectionner à partir de chaque strate des échantillons indépendants. Toutes les techniques d'échantillonnages citées précédemment peuvent être utilisées ici. La méthode d'échantillonnage peut varier d'une strate à une autre.

d. Echantillonnage aléatoire adaptatif

Cette technique a pour but de combler les lacunes des techniques d'échantillonnages citées précédemment. En effet, un constat est fait, la plupart des techniques sont utilisées d'une façon statique. Les éléments de l'échantillon sont sélectionnés périodiquement selon un pas d'échantillonnage prédéfini. Ce qui peut occasionner un sous-échantillonnage qui rend la détection de certains éléments du réseau plus difficile voire impossible sinon une génération d'une grande quantité de données qui peut saturer les équipements ou la transmission.

Bien que toutes ces techniques[40] d'échantillonnage sont destinées à réduire le nombre de données collectées et traitées et tout en garantissant un certain niveau de performance dans le processus d'analyse et d'estimation des caractéristiques de flux, ces techniques ne prennent pas

en compte les contraintes liées à la capacité des équipements et le compromis nécessaire entre le taux d'échantillonnage adaptatif nécessaire. Des solutions d'échantillonnage adaptatif du trafic sont proposées dans la littérature. C'est des solutions qui reposent sur les variations des ressources réseau pour adapter le taux d'échantillonnage tel le taux d'utilisation de CPU (Central Processing Unit) et la prédiction de la charge du trafic dans le prochaine intervalle de temps.

Notre solution se base sur les méthodes standards d'échantillonnage et de collecte de trafic. Pour faire un bon choix nous avons eu à faire une étude comparative entre deux outils de collecte et d'analyse. Il s'agit de NetFlow et sFlow.

IV. Les outils d'échantillonnage

1. sFlow

Le[41] sFlow (ou Sampled Flow) est définie comme étant un standard pour la surveillance des réseaux à haute vitesse. C'est une technologie qui s'installe dans les équipements réseaux et offre une visibilité complète sur l'activité du réseau en permettant une gestion énorme et un contrôle remarquable. sFlow est conçu pour être implementé dans la majorité des dispositifs réseau et pour fournir des statistiques d'une manière périodique. C'est un standard qui est basé sur l'échantillonnage aléatoire de paquets. Le commutateur capture un paquet parmi n paquets successifs par interface et l'envoi vers le collecteur. Ce taux d'échantillonnage est configurable en choisissant un nombre maximum, si aucune configuration n'est spécifiée, le protocole utilise les taux d'échantillonnage selon la vitesse de l'interface.

Ce[42] standard est composé essentiellement de deux composantes dont la première est l'agent sFlow qui est incorporé dans les commutateurs ou routeurs sujets de surveillance et la deuxième est le collecteur sFlow. La figure suivante suivante donne une parfaite illustration.

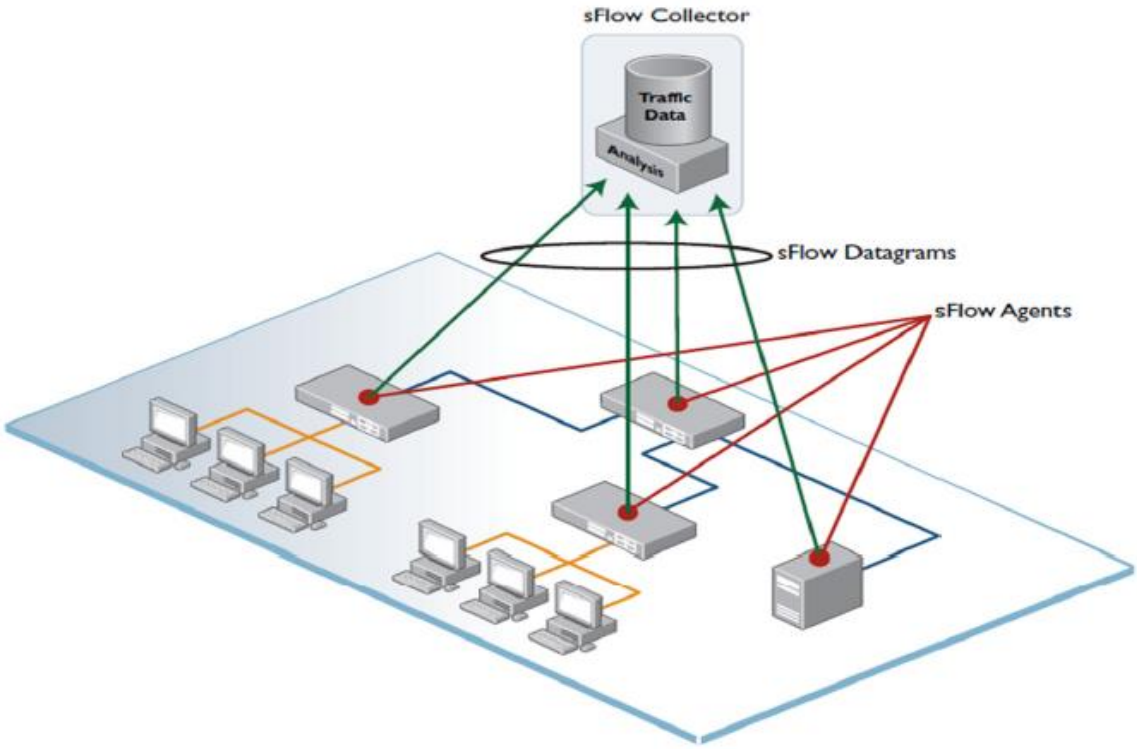

Figure 29[43]: Architecture sFlow

a. Agent sFlow

Il est incorporé dans un commutateur ou routeur. Il s'occupe de la capture du trafic par le biais d'échantillonnage selon la vitesse du lien si aucun taux n'est précis. L'échantillonnage de paquet se fait pratiquement à base de la méthode 1-out-of-N et il est possible de personnaliser le taux de prélèvement sur chaque port. Les données collectées et les statistiques de chaque port sont encapsulées dans le datagramme sFlow. Les paquets sont encapsulés en fonction du taux d'échantillonnage. Les statistiques de chaque interface sont prélevées dans des périodes maximums dites polling interval de 20 s par défaut. Les agents sFlow envoient les datagrammes sFlow au collecteur en fonction du taux d'échantillonnage et de la taille maximale de datagramme qui est de 1400 octets par défaut.

Le datagramme est composé de trois champs :

- ✓ *L'entête* qui contient des informations ordinaires permettant d'acheminer le datagramme vers le collecteur où on trouve les adresses IP sources (agent sFlow), adresse destination (collecteur) et la version du protocole sFlow ;
- ✓ *Packets samples* contient les entêtes des paquets échantillonnées. Il peut contenir aussi une partie de la charge des paquets.

- ✓ *Interface counter* permet d'envoyer les statistiques relatives à chaque interface. Il contient l'indice de l'interface, son statut, le nombre d'octets entrant et sortant, etc.

b. Collecteur sFlow

Il a pour objectif de stocker les statistiques reçues d'un ou plusieurs agents sFlow. Le collecteur peut présenter une analyse du trafic à l'administrateur du réseau s'il est doté d'une fonction d'analyse. La plupart des collecteurs utilisent SNMP pour configurer les agents et pour résoudre les numéros des index des interfaces de manière à pouvoir présenter les informations relatives aux noms des interfaces physiques associées.

2. NetFlow

C'est[44] une architecture de surveillance des réseaux développé par Cisco afin de collecter des informations sur les flux de réseau générés par les commutateurs et routeurs compatible avec NetFlow. Il définit un format d'exportation de ces informations nommé NetFlow services export format (format d'exportation des services NetFlow). Une architecture NetFlow intégrale consiste en trois composantes essentielles dont la première est nommé NetFlow Exporter. Elle permet de former des flux IP en se basant sur l'observation des paquets transitant à travers le commutateur ou routeur. Cette composante maintient une cache de flux utilisée pour sauvegarder les statistiques de trafic basées sur les flux. Les statistiques de chaque flux actif sont maintenues dans la cache et sont mise à jour lorsque les paquets de chaque flux sont commutés. La seconde composante est le NetFlow Collector. Les flux expirés sont exportés, périodiquement, par le biais du protocole (UDP) et SCTP (Stream Control Transmission Protocol datagrammes), vers le NetFlow Collector. Le NetFlow Collector reçoit et sauvegarde les enregistrements des flux envoyés par le NetFlow Exporter vers la troisième composante qu'est le storage. La figure suivante permet une parfaite illustration de cette littérature.

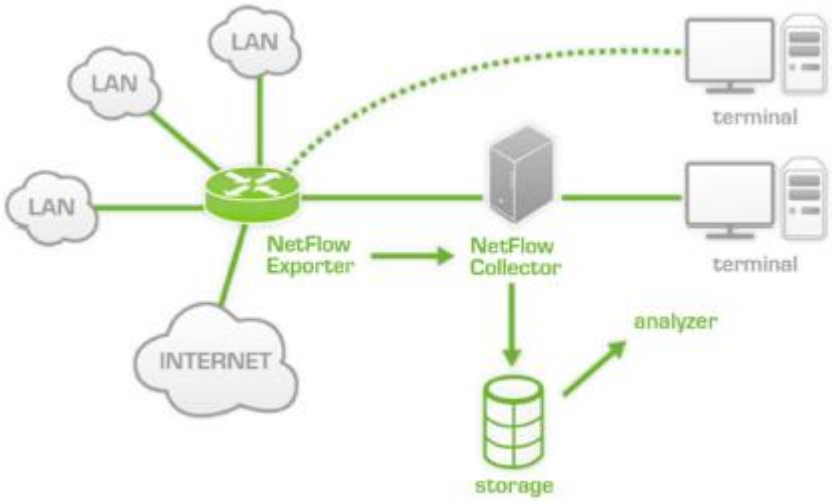

Figure 30[45]: Architecture de NetFlow

NetFlow et l'échantillonnage

NetFlow[46] est protocole qui est initialement conçu pour traiter tous les paquets qui traversent les routeurs ou commutateurs. Le fait de créer des flux par le biais de traitement de tous les paquets permet de fournir des statistiques plus précises. Quand le réseau devient plus large et génère plus de données alors l'implémentation peut occasionner des perturbations du réseau à cause de l'utilisation importante de ressource pour collecter le trafic. Pour remédier aux limitations des premières versions, Cisco a conçu une autre version qui permet de prendre en considération la surcharge du réseau. C'est le NetFlow v9. Il est possible dans cette version de configurer une méthode d'échantillonnage pour sélectionner certains paquets parmi l'ensemble des paquets commutés, à partir des échantillons captures. Elle permet une création de table de flux. Cisco a introduit plusieurs techniques d'échantillonnages, celle de sFlow en particulier.

3. Comparaison entre sFlow et NetFlow

Pour mener à bien notre comparaison, nous allons définir dans un tableau les fonctions permises par chacun des deux protocoles. Le tableau suivant donne un ensemble de possibilité pour chacun d'eux.

Possibilité	**NetFlow**	**sFlow**
Capture de paquet	Non	Partiel
Compteurs d'interfaces	Non	Oui
Echantillonnage	Partiel	Oui
En-tête de paquets	Non	Oui
Ethernet 802.3	Non	Oui
IP/ICMP/UDP/TCP	Oui	Oui
Interfaces entrées/sorties	Oui	Oui
Priorité entrées/sorties	Non	Oui
Input/Output Vlan	Non	Oui
Source sous-réseau/préfixe	Oui	Oui
Destination sous-réseau/préfixe	Oui	Oui
Prochain saut	Oui	Oui
Configuration sans SNMP	Oui	Oui
Configuration avec SNMP	Non	Oui

Taux d'échantillonnage par interface	Non	Oui
Faible coût	Non	Oui
Evolutif	Non	Oui

Tableau 2: Tableau comparatif entre NetFlow et sFlow

Après étude de ce tableau nous notons que sFlow présente plus d'avantage comparé à NetFlow. C'est pourquoi notre choix porte sur la technologie sFlow.

Après étude de l'IDS de traitement des requêtes qui est BroFlow et celui du filtrage pour la capture, nous allons traiter les requêtes.

V. Traitement des requêtes : BroFlow

1. Architecture

Nous avons utilisé des outils existants pour une solution qui agit sur deux parties de l'architecture globale des SDN. C'est au niveau du plan de données et du plan de contrôle. L'outil d'échantillonnage sFlow au niveau du plan de données pour la capture des éléments et l'IDS BroFlow qui agit parallèlement avec le contrôleur dans la partie plan de contrôle.

Chacun des deux outils joue un rôle primordial. Les deux outils se complètent pour bien maitriser les attaques de déni de services. La figure suivante fait un zoom sur les éléments de contribution apportée. C'est une architecture SDN avec les trois plans. Nous allons juste représenter les éléments de la contribution de chaque plan. Notre contribution intervient dans le plan de données et dans le plan de contrôle. Seuls les éléments apportés sont représentés. La figure suivante donne une parfaite illustration de tout ce que nous venons de dire.

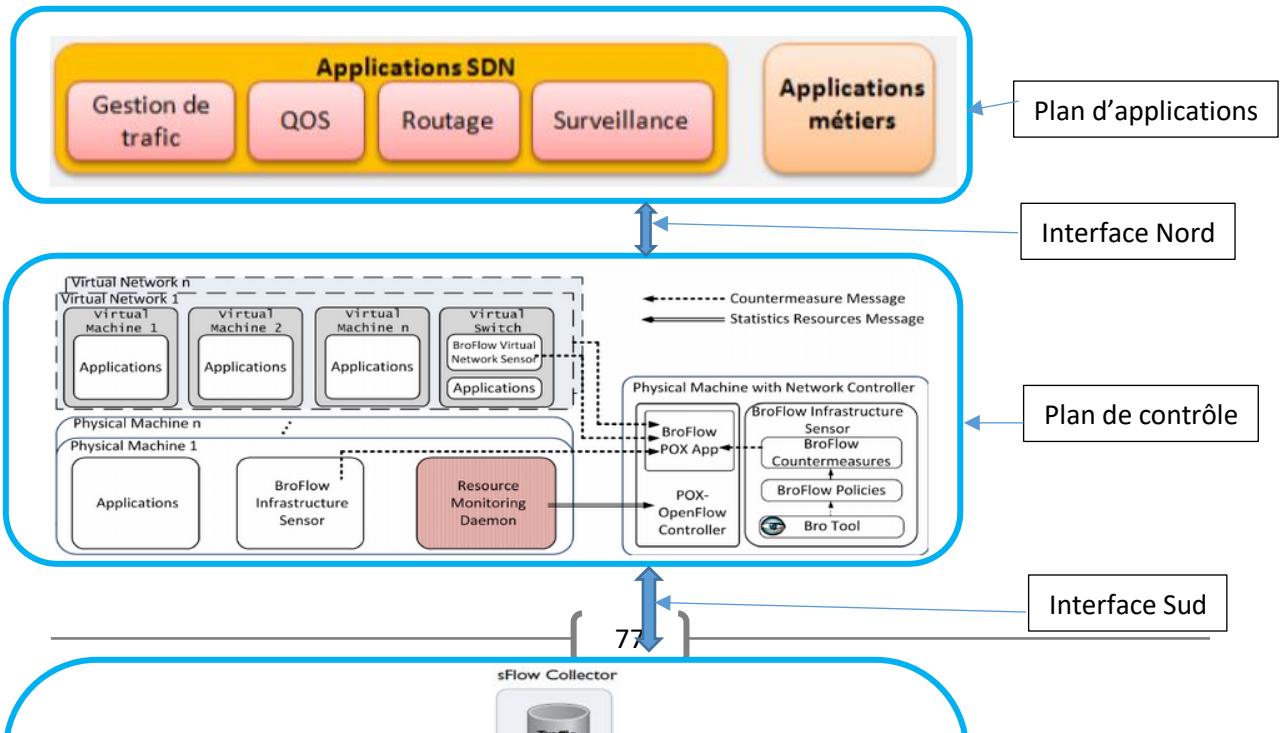

Figure 31: Architecture de la solution

2. Les problèmes réglés

a. Plan de données

A chaque fois qu'il y a une nouvelle connexion, les commutateurs SDN vont solliciter le contrôleur pour pouvoir prendre une décision. Nous savons que les attaques de déni de services vont tenter de remplir les mémoires des tables des commutateurs. Pour résoudre les problèmes que causent ces attaques, nous utilisons la technique d'échantillonnage. Notre solution se base sur l'outil d'échantillonnage sFlow. L'outil a des composants appelés agents sFlow intégrés dans les commutateurs. Ils se chargent de la collecte de données. Une fois les données collectées, les agents sFlow les envoient au sFlow collector. Ce dernier se charge de mémoriser les sources, les types de paquets, la taille ainsi que les statistiques. Combien de fois la même information est envoyée ?

Pour éviter de solliciter une règle pour la même information à plusieurs reprises, les sFlow collector vont échantillonner le trafic pour identifier les sources des données. Pour la même source et le même paquet, une demande ne sera envoyé qu'une seule fois au niveau du contrôleur. Dans ce cas, peu de demandes seront envoyées au contrôleur. Le contrôleur va répondre très rapidement. Ce qui va éviter de remplir les mémoires des commutateurs.

Notre solution permet d'empêcher aux commutateurs la suppression forcée des données dans leurs tables. Ici y aura pas risque de suppression des informations d'un utilisateur légitime. Cet

échantillonnage contribue à la résolution d'une partie énoncé dans la problématique qui est la surcharge de la communication entre commutateurs et contrôleur. Vu que peu de données sont envoyées au contrôleur pour des demandes de règles alors la communication peut ne pas être saturée.

b. Plan de contrôle

Dans les réseaux SDN, seul le contrôleur est habilité à prendre des règles de routage. C'est pourquoi il demeure le point critique des réseaux SDN. Toute connexion n'ayant pas de règle de routage va forcer les commutateurs à aviser le contrôleur pour une prise de décision. Pour ne pas saturer le contrôleur notre solution se base sur BroFlow comme expliquée précédemment. Une fois une demande de règle envoyée au contrôleur, avec notre solution, celle-ci n'atteint pas directement le contrôleur. La demande passe d'abord par BroFlow qui fonctionne parallèlement avec le contrôleur.

BroFlow va analyser la demande en utilisant l'outil d'analyse de trafic Bro comme indiqué dans l'architecture précédente. L'outil est capable de distinguer la bonne information en appliquant les techniques de détections que nous avons évoqué dans le chapitre intitulé introduction aux IDS. En fonction du résultat obtenu après analyse, l'outil va y joindre la contre-mesure à appliquer. A partir de ce moment, il l'envoi à l'application BroFlow. Cette application est intégrée au sommet du contrôleur. Ayant toutes les informations nécessaires, cette application avise le contrôleur pour la décision à prendre. Dans ce cas, le contrôleur répond automatiquement en appliquant la contre-mesure que lui a indiquée l'application BroFlow.

Notre solution allège les tâches au commutateur. Au lieu d'analyser toutes les demandes, BroFlow s'en charge de cette partie. Si le contrôleur parvient à répondre très rapidement alors cela va automatiquement impacter au niveau du plan de données. Ceci est dû à la liaison étroite entre le plan de données et le plan de contrôle.

BroFlow résout lui aussi une partie de la saturation de la liaison entre contrôleur et commutateurs. Le contrôleur répond à temps et une règle va être appliquée pour ne rien saturer, que ce soit sur la liaison ou sur les différents plans.

c. Liaison contrôleur et commutateurs

Comme nous l'avons énoncé dans les deux plans précédents, notre solution permet une meilleure gestion de la saturation de la liaison entre contrôleur et commutateurs. Grâce à l'échantillonnage de trafic, toutes les demandes ne seront pas envoyées au contrôleur. Avec peu de demandes envoyées au contrôleur, il n'y a pas un risque de saturation de la liaison. En plus,

la réponse automatique du contrôleur, permet une réaction automatique au niveau du plan de données. Donc les commutateurs vont appliquer les règles que les ont donné le contrôleur pour une meilleur gestion du trafic.

Conclusion

Notre solution a permis de contourner les attaques de déni de service. L'allégement des tâches du contrôleur nous donne une meilleure gestion du trafic dans les SDN. Ceci grâce l'outil d'échantillonnage qui collecte et envoi moins de demande de règle. Les demandes ne vont pas directement atteindre le contrôleur mais passe par BroFlow pour une analyse et la prise de la contre-mesure. C'est la contre-mesure qui va être transmise au contrôleur pour répondre au commutateur demandant. C'est cette gestion automatique des demandes qui permet la maitrise des attaques de déni de services.

Nous projetons, dans le futur, de faire une simulation et d'évaluer les performances de la contribution. Ce qui va nous permettre de donner les pourcentages de réduction. Cette simulation nous permettra d'étudier la performance au niveau de chaque partie où intervient la contribution et d'en donner des statistiques.

Conclusion et perspectives

La mise en réseau par logiciel (SDN) a récemment émergé comme une nouvelle technologie de réseau. Elle annonce des changements très importants qui permettent aux opérateurs de réseau de gérer dynamiquement leurs infrastructures. Ceci grâce à l'ouverture, la virtualisation et l'orchestration offerte par ce nouveau paradigme qu'est le SDN. Le réseau SDN est un concept qui sépare la partie contrôle de la partie infrastructure (ou données) du réseau. Ce nouveau paradigme présente de nos jours plusieurs avantages dans la gestion des réseaux. Il attire l'attention de plusieurs entreprises ainsi que des chercheurs.

Cependant, les réseaux SDN subissent des attaques. Parmi ces dernières, les attaques de déni de service (Dds) sont considérées comme la menace majeure pour les SDN. Etant donné la centralisation de la gestion dans l'architecture SDN, ces attaques peuvent facilement surcharger le contrôleur, les tables de commutation des commutateurs ainsi que la liaison entre contrôleur et commutateurs. Le déni de service est un fléau qui trouble la sécurité et le bon fonctionnement du réseau. Dans ce mémoire, nous nous sommes intéressé à contourner les attaques de déni de

service. Pour atteindre notre objectif nous avons fait appel aux systèmes de détection d'intrusion qui était insuffisant que nous avons complété avec l'échantillonnage de trafic.

Il existe plusieurs directions prometteuses que nous pouvons poursuivre dans le futur. Nous prévoyons de faire une simulation sur la solution proposée. Cette simulation va porter sur la mise en œuvre des IDS (l'outil BroFlow qi se base sur Bro) et de l'échantillonnage du trafic (en se basant sur l'outil sFlow). Nous allons utiliser le logiciel MININET pour le déploiement d'un réseau SDN. Nous allons implémenter les outils utilisés pour faire le test. Cette simulation va permettre d'obtenir la fiabilité de la solution. Jusqu'à quel niveau elle apporte de l'importance.

Bibliographie

[1] G. Pujolle, *Les réseaux*. Editions Eyrolles, 2014.

[2] A. S. Tanenbaum, D. Wetherall, M.-C. Baland, E. Burr-Campillo, and S. Pauquet, *Réseaux*, vol. 4. Pearson Education, 2003.

[3] A. Johnson, "Scaling Networks v6 Companion Guide and Lab ValuePack," *Indianap. U. S. Am.*, 2017.

[4] Z. Mammeri, "Réseaux sans fils Caractéristiques et principaux standards," *M1 Info Cours Réseaux IRIT Univ. Paul Sabatier Toulouse Httpwww Irit Fr∼ Zoubir MammeriChap6WLAN Pdf*.

[5] J.-P. Georges, "Systèmes contrôlés en réseau: évaluation de performances d'architectures Ethernet commutées," PhD Thesis, 2005.

[6] A. El Bardai, "Virtualisation d'une plateforme de gestion de contexte," masters, École de technologie supérieure, Montréal, 2015.

[7] P. Primet, O. Mornard, and J.-P. Gelas, "Évaluation des performances réseau dans le contexte de la virtualisation XEN," INRIA, report, 2007. Accessed: Jul. 08, 2020. [Online]. Available: https://hal.inria.fr/inria-00203461.

[8] N. Triki, "Gestion de ressources dans les infrastructures de virtualisation de réseaux," masters, École de technologie supérieure, 2013.

[9] F. Diakhaté, "Contribution à l'élaboration de supports exécutifs exploitant la virtualisation pour le calcul hautes performances," phdthesis, Université Sciences et Technologies - Bordeaux I, 2010.

[10] E. Haleplidis, K. Pentikousis, S. Denazis, J. H. Salim, D. Meyer, and O. Koufopavlou, "Software-defined networking (SDN): Layers and architecture terminology," in *RFC 7426*, IRTF, 2015.

[11] J. Matias, J. Garay, N. Toledo, J. Unzilla, and E. Jacob, "Toward an SDN-enabled NFV architecture," *IEEE Commun. Mag.*, vol. 53, no. 4, pp. 187–193, 2015.

[12] J.-M. Sanner, "Architecture du plan de contrôle SDN et placement de services réseaux dans les infrastructures des opérateurs," PhD Thesis, Université de Rennes 1 [UR1], 2019.

[13] S. SECCI, "QUELQUES VULNÉRABILITÉS DU SDN." Mai/Juin 2018, [Online]. Available: https://www.miscmag.com.

[14] D. V. Bernardo and B. B. Chua, "Introduction and analysis of SDN and NFV security architecture (SN-SECA)," in *2015 IEEE 29th international conference on advanced information networking and applications*, 2015, pp. 796–801.

[15] J. Durand, "Le SDN pour les nuls," *Cisco Syst. P8 JRES*, 2015.

[16] I. Choukri, M. Ouzzif, and K. Bouragba, "Software Defined Networking (SDN): Etat de L'art," 2019.

[17] M. M. Bahnasy, "OpenFlow protocol extension for optical networks," 2014.

[18] B. Fouad and M. Mouad, "Etude des Performances des Architectures du Plan de Contrôle des Réseaux 'Software- Defined Networks.'"Jan. 2017, [Online]. Available: https://www.researchgate.net/publication/316620880.

[19] E. Canceres, *Le Protocole OpenFlow dans l'Architecture SDN*. Consulté sur: http://www. efort. com/r_tutoriels/OPENFLOW_EFORT. pdf, 2016.

[20] L. Dridi, "Mitigation des attaques de déni de service dans les réseaux définis par logiciel," PhD Thesis, École de technologie supérieure, 2017.

[21] H. BOUIDA née SAIDI, "Etude et mise en oeuvre d'une solution SDN .," Thesis, 11-03-2018, 2017.

[22] T. Issa, K. B. Médard, and A. Ferdinand, "Etude du nomadisme dans un Cloud éducatif administré par la technologie SDN/OpenFlow," 2016.

[23] F. Benamrane, "Etude des Performances des Architectures du Plan de Contrôle des Réseaux 'Software-Defined Networks,'" 2017.

[24] T. Probst, "Évaluation et analyse des mécanismes de sécurité des réseaux dans les infrastructures virtuelles de cloud computing," PhD Thesis, 2015.

[25] W. Larbi-Mezeghrane, K. Bir, and Y. Saoudi, ". Mise en place d'un système de détection d'intrusion," PhD Thesis, universite abderrahmane mira Bejaia, 2017.

[26] N. Dagorn, "Détection et prévention d'intrusion: présentation et limites," 2006.

[27] K. Müller, K. M. alias' Socma, and G. Tarbouriech, "IDS-Systèmes de Détection d'Intrusion, Partie I," *LinuxFocus Artic.*, no. 292, 2003.

[28] M. E.-S. Gadelrab, "Évaluation des Systèmes de Détection d'Intrusion," PhD Thesis, Université de Toulouse, Université Toulouse III-Paul Sabatier, 2008.

[29] S. Chakrabarti, M. Chakraborty, and I. Mukhopadhyay, "Study of snort-based IDS," in *Proceedings of the International Conference and Workshop on Emerging Trends in Technology*, 2010, pp. 43–47.

[30] K. J. Cox and C. Gerg, *Managing Security with Snort & IDS Tools: Intrusion Detection with Open Source Tools*. O'Reilly Media, Inc., 2004.

[31] S. Gupta and S. G. Kaur, "A Graphical User Interface Framework for Detecting Intrusions using Bro IDS," PhD Thesis, 2012.

[32] B. Chen, J. Lee, and A. S. Wu, "Active event correlation in Bro IDS to detect multi-stage attacks," in *Fourth IEEE International Workshop on Information Assurance (IWIA'06)*, 2006, p. 16–pp.

[33] I. D. S. Bro, *Homepage: http://www. bro-ids. org*. July, 2008.

[34] K. Nam and K. Kim, "A study on sdn security enhancement using open source ids/ips suricata," in *2018 International Conference on Information and Communication Technology Convergence (ICTC)*, 2018, pp. 1124–1126.

[35] O. Eldow, P. Chauhan, P. Lalwani, and M. Potdar, "Computer network security ids tools and techniques (snort/suricata)," *Int J Sci Res Publ*, vol. 6, no. 1, p. 593, 2016.

[36] M. A. Lopez, D. M. F. Mattos, and O. C. M. Duarte, "An elastic intrusion detection system for software networks," *Ann. Telecommun.*, vol. 71, no. 11–12, pp. 595–605, 2016.

[37] M. B. Duarte, "Martin Andreoni Lopez, Diogo Menezes Ferrazani Mattos & Otto Carlos," *Ann Telecommun*, vol. 71, pp. 595–605, 2016.

[38] M. A. Lopez and O. C. M. Duarte, "Providing elasticity to intrusion detection systems in virtualized software defined networks," in *2015 IEEE International Conference on Communications (ICC)*, 2015, pp. 7120–7125.

[39] B. Mandal, S. Sarkar, S. Bhattacharya, U. Dasgupta, P. Ghosg, and D. Sanki, "A Review on Cooperative Bait Based Intrusion Detection in MANET," *Available SSRN 3515151*, 2020.

[40] M. ELBAHAM, "Visualisation de trafic de réseau en temps réel." LE JUIN 2017.

[41] P. Phaal, S. Panchen, and N. McKee, "InMon corporation's sFlow: A method for monitoring traffic in switched and routed networks," 2001.

[42] Y. Lu and M. Wang, "An easy defense mechanism against botnet-based DDoS flooding attack originated in SDN environment using sFlow," in *Proceedings of the 11th International Conference on Future Internet Technologies*, 2016, pp. 14–20.

[43] M. Afaq, S. Rehman, and W.-C. Song, "Large flows detection, marking, and mitigation based on sFlow standard in SDN," 멀티미디어학회논문지, vol. 18, no. 2, pp. 189–198, 2015.

[44] I. Pepelnjak, *DefenseFlow NetFlow and SDN based DDoS attack defense*. .

[45] Y. Afek, A. Bremler-Barr, S. Landau Feibish, and L. Schiff, "Sampling and large flow detection in SDN," in *Proceedings of the 2015 ACM Conference on Special Interest Group on Data Communication*, 2015, pp. 345–346.

[46] J. Suárez-Varela and P. Barlet-Ros, "Towards a NetFlow implementation for OpenFlow software-defined networks," in *2017 29th International Teletraffic Congress (ITC 29)*, 2017, vol. 1, pp. 187–195.