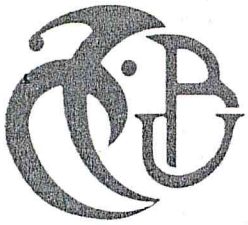
Université Saad Dahlab Blida1



Faculté des Sciences

Département d'Informatique

Projet de fin d'étude pour l'obtention du diplôme Master

Spécialité :
Sécurité des Systèmes d'Information

Thème :

Conception et implémentation d'un gestionnaire de VPN dans un réseau SDN(Software Defined Networking)

Réalisé par :

Kessas Aissa

t proposé et encadré par : Dr Mohamed Amine Bouabid

ıbres de jurés:

Guessoum Président

ɒuya Examinateur

Soutenu le: 31/10/2017

2016/2017

Remerciements

Je tiens à remercier tout d'abord Monsieur le Recteur Abadlia Mohammed Tahar,M. Senouci sid Ahmed le vice-recteur et M.Hammouda Mohammed, le chef du département informatique de l'Université et, de l'Université Sâad Dahleb de Blida pourl'accueil Chaleureux au sein de cet établissement.

Je remercier mon directeur de thèse, Dr Bouabid Mohammed Amine, pour sa patience, et surtout pour sa confiance, ses remarques et ses conseils, sa disponibilité et sa bienveillance.

Je tiens à remercier M.Chaib Slimène, M.Frendi Mohammed, pour leurs soutiens et encouragements.

Je voudrais également remercier les membres du jury pour avoir accepté d'évaluer ce travail et pour toutes leurs remarques et critiques, ainsi que le personnel et les enseignants du département de l'informatique de l'université Saad Dahleb de Blida.

A tous mes enseignants, qui m'ont initié aux valeurs authentiques.

Merci à vous tous

Dédicace

À la mémoire de ma chère mère.

À mon père que dieu le protège.

À ma femme qui a toujours était à mes côtés.

À mes chères enfants : Nour, Ritedj et Amir.

À tous mes frères et soeurs, ainsi que leurs enfants.

À tous mes amis et collègues de travail.

À tous les étudiants de la promotion 2016/2017

Option : SSI.

A tous ceux qui, par un mot, m'ont donné la force de continuer .....

Summary:

The rapid development of computer and telecommunication technologies makes network infrastructures vulnerable to attack, exposing deployed platforms to a significant number of threats, taking advantage of weaknesses in communication protocols or in deployed assets and software. where the need to accompany this revolution by setting up the necessary means to fight against these attacks.

The appearance of Software Defined Networking (SDN) in the virtualization world with the deployment of the Openflow protocol requires us to implement a set of security platforms and scenarios to ensure reliable data transfer.

Our work is focused on two main axes, the first one is the management and the supervision of the network flows in an SDN architecture, it is the processing and management of the Openflow protocol, the second is the security of the flows through the VPN deployment (IPSEC or SSL) or by the segmentation of the network by VLANs, these two axes will be orchestrated by the development of a mini-controller (application) which is based on the same principle of the SDN controllers, adding to that a set of scenarios and encryption rules, segmentation and monitoring of the traffic in question.

Résumé

Le développement rapide des technologies en Informatique et en Télécommunication rend les infrastructures réseaux vulnérables aux attaques, ce qui expose les plateformes déployés à un nombre important de menaces, profitant des faiblesses dans les protocoles de communication ou dans les équipements actifs et logiciels déployés, d'où la nécessité d'accompagner cette révolution par la mise en place de Moyens nécessaires afin de lutter contre ces attaques.

L'apparition des réseaux définis par logiciel (Software Defined Networking) ou SDN dans le monde de virtualisation avec le déploiement du protocole Openflow nous oblige de mettre en œuvre un ensemble de plateformes et scénarios de sécurisation afin de garantir un transfert de donnée fiable.

Notre travail est concentré sur deux principaux axes, le premier est la gestion et la supervision du flux dans une architecture SDN, c'est les traitement et gestion du flux Openflow, le deuxième est la sécurisation du flux à travers le déploiement de VPN(IPSEC ou SSL) ou par la segmentation du réseau par VLANs, ces deux axes seront orchestrés par le développement d'un mini-contrôleur (application) en se basant sur le même principe des contrôleurs SDN, en ajoutant à ça un ensemble de scénarios et de règles de cryptage, de segmentation et de monitoring du trafic en question.

Table des matières

LISTE DES FIGURES

INTRODUCTION GENERALE

La sécurité est une nécessité et une obligation pour protéger l'intégrité, la disponibilité et la confidentialité de toutes les ressources connectées. Les réseaux SDN doivent être protégés, Le Software Defined Networking (SDN) est une approche de la mise en réseau qui sépare le plan de contrôle du plan de donnée. La plupart des d'architecture SDN définit ont trois couches: une couche inférieure de périphériques réseau SDN, une couche intermédiaire de contrôleurs SDN et une couche supérieure qui inclut les applications et les services qui demandent ou configurent le SDN.

Les attaques peuvent se produit sur plusieurs niveaux dans l'architecture SDN, des attaques de type DDOS(Distributed Denial of Service) qui peuvent affecter la disponibilité du contrôleur, des attaques de type MIM(Men in the Middle) se substituent le rôle du contrôleur comme ils peuvent sniffer le trafic réseau.

Notre objectif dans cette étude est de sécuriser un trafic réseau dans une architecture SDN par l'utilisation et l'implémentation d'un ensemble d'outils de cryptographie et de segmentation et par la mise en œuvre d'une application de gestion, de supervision et de sécurisation de flux SDN, ce document est organisé comme suit :

Dans le premier Chapitre nous présentons un état de l'art des réseaux SDN, avec présentation leurs architectures, fonctionnements et leurs déploiement dans le monde des réseaux.

Dans le deuxième chapitre nous présentons les deux piliers des réseaux virtualisés: le Switch Openflow virtuel (OVS) et le protocole Openflow, leurs principes, les principaux composants et leurs déploiements dans les réseaux SDN.

Le troisième chapitre est un aperçu sur les réseaux privés virtuels (VPN), les différentes architectures et les cas d'usages des VPNs avec la présentation de l'ensemble des protocoles assurant ce principe.

Notre contribution est présentée dans les deux derniers chapitres, étudier les différentes architectures de sécurisation de flux dans un SDN basé sur OpenvSwitch tels que les VLANs et les VPNs et de concevoir et de développer une application dédiée à la gestion de la sécurité des communications en réseau en utilisant le protocole OpenFlow et le système OVS.

L'utilisateur ou l'administrateur, aura l'occasion, via cette application de déterminer le ou les flux du trafic réseau à sécuriser à travers le plus grand nombre de critères possibles, de choisir les moyens de sécurisation et enfin d'appliquer et de superviser le trafic en question.

CHAPITRE I Le SDN : Un état de l'art

I.1.Définition

SDN signifie littéralement Software-Defined Networking, c'est-à-dire le réseau défini par l'application. On comprend donc immédiatement que le sujet est vaste et qu'il va être difficile d'avoir une définition unique. De plus, la définition même du SDN a changé avec le temps.

La définition académique, qui a depuis largement évolué, consistait à voir le SDN comme une architecture qui découplait les fonctions de contrôle et de transfert des données du réseau (data plane) afin d'avoir une infrastructure physique complètement exempte de tout service réseau (figure 1).

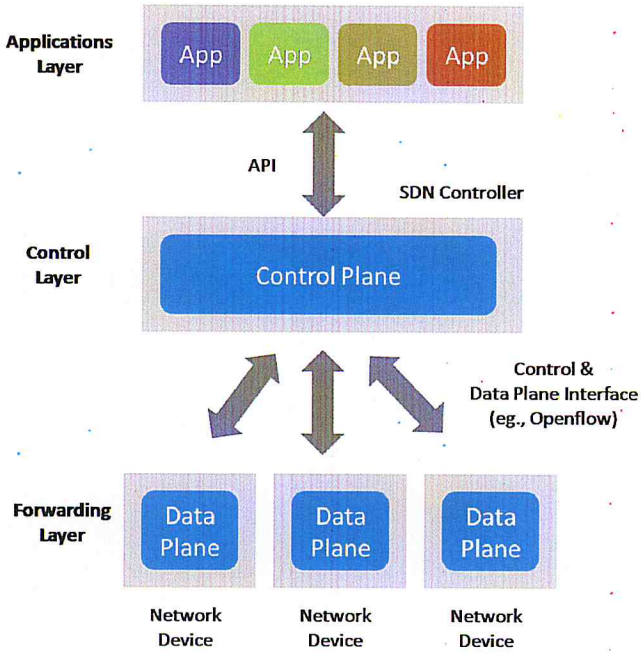


Figure 1. Architecture SDN

Le marché s'est rapidement approprié le SDN comme ensemble de solutions/architectures permettant de supprimer les frontières existantes entre les mondes des applications et du réseau. Alors que le déploiement d'applications est toujours plus aisé et dynamique, notamment grâce à la virtualisation et au Cloud, le réseau reste parfois perçu comme un frein.

Le SDN est donc plus globalement reconnu aujourd'hui comme une architecture permettant d'ouvrir le réseau aux applications. Cela intègre les deux volets suivants :

- ✓ permettre aux applications de programmer le réseau afin d'en accélérer le déploiement ;
- ✓ permettre au réseau de mieux identifier les applications transportées pour mieux les gérer (qualité de service, sécurité, ingénierie de trafic...).

Le SDN est donc différent d'Openflow :

La plupart du temps, il y a une confusion entre SDN et Openflow, dans l'architecture SDN initiale, des règles de traitement des flux de données sont injectées par un contrôleur intelligent dans des équipements réseau sans fonction de contrôle. Openflow est le protocole défini par l'ONF (Open Networking Foundation) pour transférer ces règles. Il permet par exemple à un contrôleur d'injecter des règles sur des Switchs ou des routeurs.

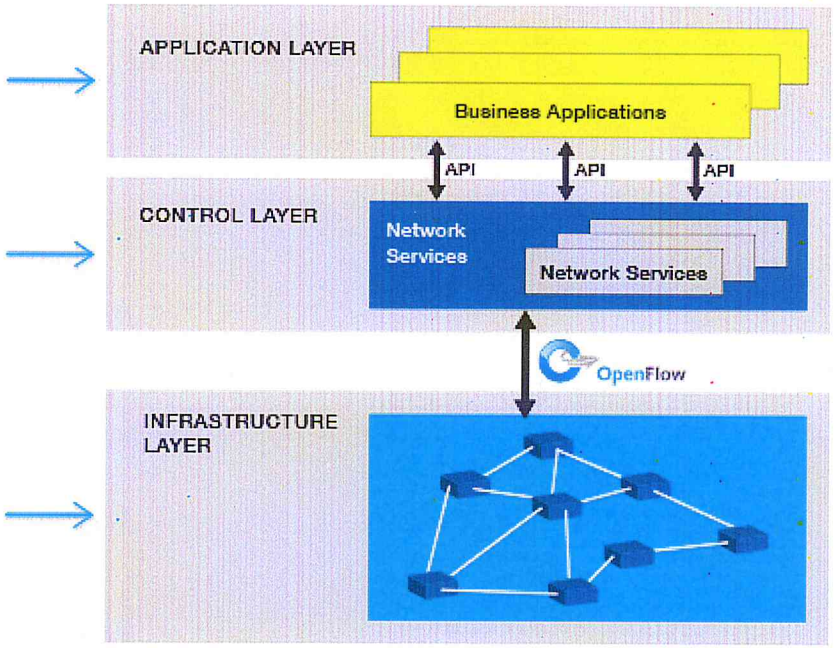


Figure 2.Openflow dans l'architecture SDN

Openflow transmet des ordres au niveau du plan de données, par exemple :

- ✓ Transférer les trames à destination de l'adresse MAC X vers l'interface Y en rajoutant le VLAN Z dans l'entête 802.1Q.
- ✓ Jeter les paquets ayant pour source ou destination l'adresse IPv6 a encapsulé en GRE à destination de l'adresse IPv4 C tous les paquets IPv4 provenant de l'adresse D.

Comme cela a été clarifié précédemment, le SDN va au-delà de la capacité d'injecter des règles, et donc des états sur les équipements pour chaque action nécessaire. Openflow en soi ne suffit donc pas à répondre aux problématiques réelles des organisations. Aussi, cela pose de nombreuses questions quant à la capacité de résister au facteur d'échelle quand le nombre de règles/flux s'accroit du fait de l'augmentation du nombre d'utilisateurs, de terminaux et d'applications.

Openflow est donc une composante du SDN mais c'est loin d'être la seule puisque cette technologie ne répond pas directement aux problématiques rencontrées. Rob Sherwood, CTO de Big Network, acteur historique sur le marché du SDN avec une solution de contrôleur Openflow.
Depuis 2013, on constate d'ailleurs assez peu d'avancées sur Openflow. Il semble que l'ONF se cherche une nouvelle orientation à travers des sondages envoyés à la communauté.

Le SDN est une architecture permettant d'ouvrir le réseau aux applications. Cela intègre les deux volets suivants :

- ✓ permettre aux applications de programmer le réseau afin d'en accélérer le déploiement ;
- ✓ permettre au réseau de mieux identifier les applications transportées pour mieux les gérer (qualité de service, sécurité, ingénierie de trafic...).

Openflow transmet des ordres au niveau du plan de données, par exemple :

- ✓ Transférer les trames à destination de l'adresse MAC X vers l'interface Y en rajoutant le VLAN Z dans l'entête 802.1Q.
- ✓ Jeter les paquets ayant pour source ou destination l'adresse IPv6 a encapsulé en GRE à destination de l'adresse IPv4 C tous les paquets IPv4 provenant de l'adresse D.

I.2) Différents modèles pour le SDN

Le SDN englobe toutes les solutions permettant une programmation du réseau, afin de mieux interagir avec les applications. Diverses solutions coexistent, adaptées selon les besoins des utilisateurs.
On peut noter différents modèles de programmation :
1) Programmation individuelle de chaque équipement (via une API sud comme le protocole OpenFlow). Dans ce modèle une application interagit directement avec chaque équipement via des API. L'application est centralisée ou peut être localisée directement sur l'équipement réseau pour réaliser des tâches spécifiques.
2) Programmation via un contrôleur (via une API nord comme OpenDayLight). Dans ce modèle, une application donne un ordre abstrait et global à un contrôleur, qui à son tour traduit cette requête en une suite d'ordres auprès des équipements du réseau concerné. Ce modèle est certainement le plus populaire puisqu'il permet de simplifier le réseau. Le contrôleur masque la complexité du réseau. On peut distinguer plusieurs cas selon le type d'ordres échangés entre le contrôleur et les équipements. Si, au

départ, il était question d'avoir une Programmation du plan de données (via Openflow par exemple), des modèles plus récents implémentent des architectures dans lesquels des ordres plus abstraits sont donnés aux équipements, ces derniers restants libres de les implémenter au mieux. On parle dans ce cas d'un modèle « policy-intent ».

3) Création d'un réseau virtuel au-dessus du réseau physique. Dans ce modèle, les applications créent leur propre réseau « overlay », s'affranchissant des contraintes du réseau physique sous-jacent. Ce dernier n'a pour mission que la simple connectivité entre les noeuds d'extrémité des tunnels, et le réseau d'overlay assure l'intégralité des services. On parle également de virtualisation des fonctions réseau (NFV –Network Function Virtualization) quand les routeurs, Switchs, firewalls, etc. sont des éléments virtuels sur des serveurs.[6]

I.3) Les contrôleurs SDN

I.3.1. Définition :

Le contrôleur SDN permet d'implémenter rapidement un changement sur le réseau en traduisant une demande globale (par exemple : prioriser l'application X) en une suite d'opérations sur les équipements réseau, les ordres sont donnés au contrôleur par une application via une API dite « Northbound » ou nord. Les éditeurs logiciels de contrôleurs publient la documentation de l'API afin de permettre d'interfacer des applications.

Le contrôleur communique avec les équipements via une ou plusieurs API dites « Southbound » ou sud (figure 3). Openflow se positionne comme une API sud agissant directement sur le plan de données. D'autres API permettent d'agir sur le plan de management ou de contrôle. Netconf est par exemple une API sud permettant au contrôleur de configurer un équipement. Un contrôleur pourra même parler directement en CLI avec un équipement pour actionner une fonctionnalité.

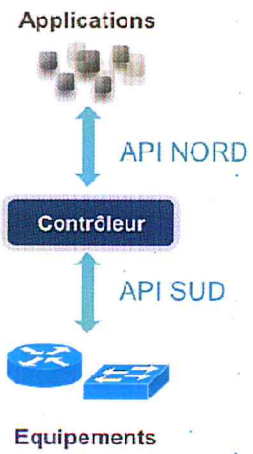


Figure 3–APIs Nord et Sud

Afin de pouvoir interagir avec le réseau, le contrôleur a besoin d'une vue précise de ce dernier. C'est ainsi que le concept de NIB (Network Information Base) a vu le jour. Cette NIB est construite au niveau du contrôleur et permet à ce dernier de savoir comment implémenter chaque ordre abstrait, trouver les équipements qui doivent être reconfigurés, s'assurer de la capacité de ces derniers à implémenter une directive, les API supportées par l'équipement.[4]

La plupart des équipementiers principaux travaillent aujourd'hui sur des contrôleurs SDN. Certaines start-ups, la plus connue étant sans doute Big Network, se sont créées pour se focaliser sur cette tâche.

I.3.2.Contrôleur SDN un outil de management

Le contrôleur pouvant agir sur le plan de management, la frontière avec les outils de management traditionnels peut s'avérer bien mince.

Nous noterons les principales différences suivantes :

Le contrôleur est conçu pour exécuter un changement sur le réseau. Il n'a pas pour objectif principal d'assurer une traçabilité du réseau ni de vérifier de l'état de chaque nœud. L'outil de management en revanche a pour première mission de vérifier que le réseau fonctionne correctement et générer des alarmes en cas de besoin. Afin de bien mettre en évidence cette première distinction essentielle, nous conviendrons qu'un outil de management restera indispensable, ne serait-ce que pour s'assurer que le contrôleur garde le réseau dans un état opérationnel !

Le contrôleur agit tant sur le plan de management que sur les plans de contrôle et de données. L'outil de management reste exclusivement sur le plan de management (configuration de l'équipement).

Dans un modèle SDN à base de contrôleur, il y a un découplage entre l'application, qui donne un ordre abstrait, et le contrôleur, qui va implémenter cet ordre au niveau du réseau. Dans un outil de management, il n'y a pas de dissociation entre ces deux éléments. L'outil de management peut très bien évoluer pour devenir une des applications utilisant les services d'un contrôleur. Dans ce cas, l'outil de management se concentre sur la GUI et des opérations globales abstraites, et laisse le/les contrôleur(s) réaliser les changements sur le réseau.

Un contrôleur est ouvert puisqu'il est là pour exécuter des ordres fournis via une API Nord. Les outils de management ne sont pas toujours ouverts et ne peuvent réaliser que les missions que pour lesquelles ils ont été conçus. La mission du contrôleur va donc bien au-delà du management puisque les applications peuvent être très variées : gestion de la QoS par un serveur de téléphonie, loadbalancing contrôlé directement par l'application, auto-configuration.

I.4) Architecture et fonctionnement du SDN :

Le but de SDN est de fournir des interfaces ouvertes qui permettent le développement de logiciels qui peuvent contrôler la connectivité fournie par un ensemble de ressources réseau et le flux du trafic réseau, ainsi que l'inspection et la modification possible du trafic qui peut être réalisée.

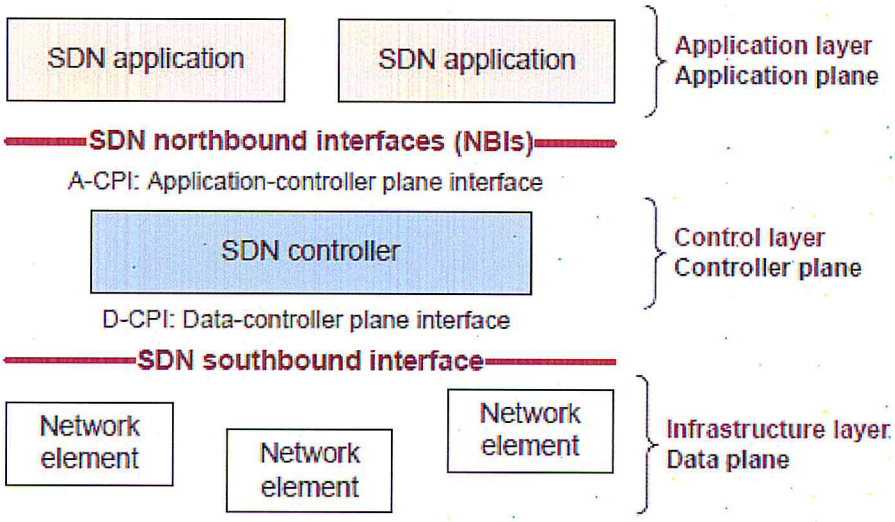


Figure 4- composants de base SDN

La figure 4 présente les composants de base SDN, la vue initiale infrastructure, constituée de couches de contrôle et d'application (texte rouge), qui sont désigné sous forme de plan de données, plan de contrôle et plan d'application (texte noir).

La couche d'infrastructure (plan de données) comprend des éléments de réseau, qui exposent les paquets reçus à la couche de contrôle (plan de contrôle) par l'intermédiaire de l'interface sud de contrôleur (Southbound), on appelle cette interface le plan de contrôle de données. Les applications SDN existent dans la couche d'application. [4]

1.5. Conclusion

Le SDN est une architecture regroupant une panoplie de composants physiques et logiciels, protocoles de communication et APIs dédiés à divers types de contrôleurs et applications, y compris la communication inter-contrôleurs. L'objectif premier du SDN étant de séparer le plan de contrôle du plan de transfert (Data) afin de pouvoir programmer le réseau selon les besoins des applications (d'où son nom) en se libérant des contraintes des réseaux traditionnelles gérés de manière unitaire (et non pas globale) exclusivement par des équipes indépendantes des équipes de développement et d'administration des systèmes et bases de données. Le SDN est un ensemble de technologies soutenant la tendance DevOps dont l'objectif est de séparer les frontières dans un même système d'information.
L'architecture SDN a dépassé son objectif premier afin de répond à des besoins de plus en plus diversifiées ciblant différentes applications. Parmi les composant du SDN, le protocole OpenFlow occupe une place centrale, au point où ces deux concepts sont souvent confondus l'un à l'autre. La majorité des constructeurs réseaux implémentent aujourd'hui (à des proportions différentes) ce protocole dans leurs équipements et logiciels à coté de leurs propres composants. Une des meilleures implémentations de ce protocole est sans doute le switche virtuel open source OVS. Ceci explique la présente de ce produit dans la plupart des offres de plateformes de virtualisation et de Cloud (y compris les solutions propriétaires) OpenFlow et OVS constituent donc deux briques clés dans toute solution SDN d'où leur choix dans notre projet. Le chapitre suivant est dédié à la présentation de ces deux composants.

II.1. Introduction

Dans le fonctionnement actuel des réseaux IP, chaque équipement de réseau (en particulier les routeurs et les Switchs Ethernet) exécute deux fonctions afin de pouvoir acheminer le trafic reçu vers sa destination :

- Le plan de contrôle: qui contient toute l'intelligence de l'équipement à travers les algorithmes de calcul de tous les chemins possibles entre les sources et destinations connues par l'équipement. Pour les switchs il s'agit du calcul de la table de correspondance entre un flux et le port sortie. Pour les routeurs il y a en plus les algorithmes et protocoles de routage statiques et dynamiques qui permettent de calculer les chemins vers un ensemble de destinations. La table de correspondance à ce niveau peut contenir plusieurs entrées pour la même source et/ou destination.
- Le plan de données : cette partie ne contient table d'acheminement de données unique contenant la correspondance entre les différents flux de trafic et les ports de sortie. L'objectif du plan de données est d'acheminer le plus rapidement possible le trafic reçu vers sa destination. La plan de donnée est configurée par le plan de contrôle qui veille à y insérer un seul chemin par source/destination, le meilleur parmi les chemins possibles. Lorsque le plan de données reçoit du trafic inexistant dans sa table, il l'envoi au plan de contrôle, qui lui affectera (le cas échéant) une entrée dans la table d'acheminement.

Le SDN (Software Defined Network) propose de créer un point central qui gère le plan de contrôle, tandis que les Switchs/routeurs physiques n'ont plus à prendre en charge que le plan de données. La communication entre le plan de contrôle centralisé et les switchs s'effectue via un protocole conçu à cet effet : OpenFlow. OpenFlow est spécifié et publié par l'Open Networking Foundation (ONF). c'est un standard utilisé par le contrôleur pour transmettre au Switch des instructions qui permettent de programmer leur plan de données et d'obtenir des informations de ces Switchs afin que le contrôleur puisse disposer d'une vue globale logique (abstraction) du réseau physique. Cette vue est utilisée pour toutes les décisions que doit prendre le plan de contrôle (routage, filtrage de trafic, partage de charge, traduction d'adresse, etc).[8]

De son coté le contrôleur SDN fournit une API aux « applications SDN » de niveau supérieur qui expose cette vue globale. Ces applications implémentent, via le contrôleur, des services tels que le routage de flux, la mise en œuvre de politiques de QoS pour les flux, la sécurité de bout en bout des flux, le filtrage de flux, etc. Cette approche permet une mise en œuvre flexible et rapide de nouvelles applications réseau qui émulent les équipements réseaux.

La figure 5 ci-dessous, illustre la position du protocole OpenFlow dans une architecture SDN typique. OpenFlow permet en particulier, de configurer les tables de correspondances du plan de donnée de tous les switchs gérés par le contrôleur. Nous verrons dans la suite la nouvelle

structure de ces tables d'acheminement qui permettent d'opérer des traitements intéressants sur le trafic.

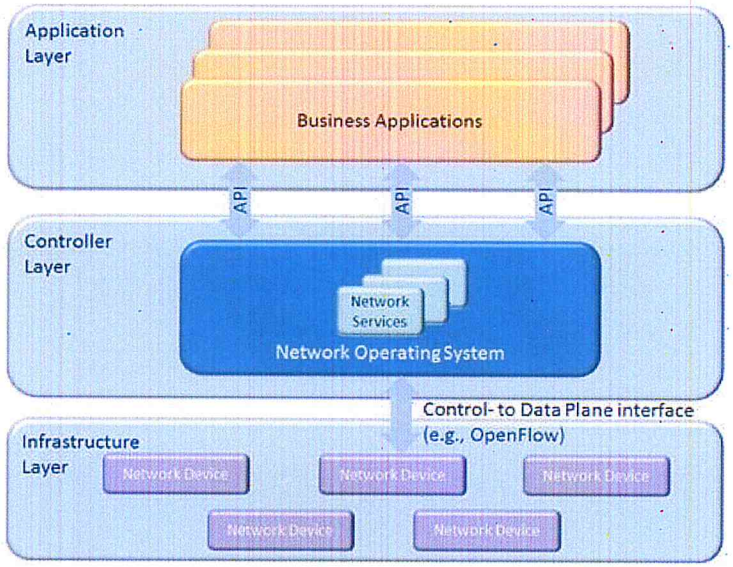


Figure 5. Openflow dans une architecture SDN

II.2. Structure d'un Switch OpenFlow

Les Switchs Ethernet et les routeurs les plus modernes contiennent des tables de flux qui sont utilisées pour effectuer des fonctions de transfert selon les couches 2,3 et 4 du modèle OSI, indiquées dans les entêtes de paquets. Bien que chaque fournisseur ait des tables de flux différentes, il existe un ensemble commun de fonctions pour une large gamme de Switchs et de routeurs. Cet ensemble commun de fonctions est mis à profit par OpenFlow, protocole entre un contrôleur central OpenFlow et un Switch OpenFlow et qui, comme indiqué, peut être utilisé pour programmer la logique de transfert ou d'acheminement du Switch.

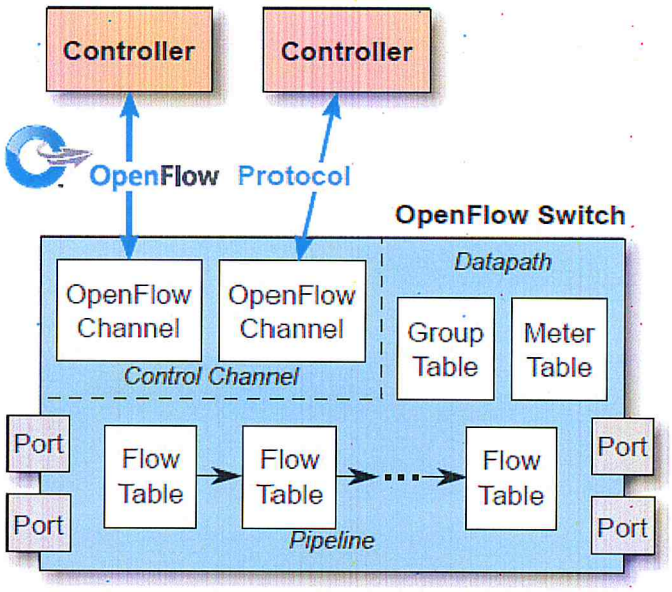


Figure 6.Principaux composants d'un Switch OpenFlow.

La figure ci-dessus décrit les fonctions réalisées par les équipements de réseau du plan de données (data plane) aussi appelés Switchs au sens générique. Les principales fonctions des Switchs sont les suivantes :

- Fonction de support du contrôle (Control support function) : Interagit avec la couche de contrôle SDN afin de supporter la programmation via les interfaces de contrôle de ressource. Le Switch communique avec le contrôleur et le contrôleur gère le Switch avec le protocole OpenFlow, qui peut être utilisé aussi bien pour du contrôle que pour de la gestion.
- Fonction d'acheminement des données (Data forwarding function) : Accepte les flux de données entrants provenant d'autres équipements de réseau et des systèmes de terminaison et les relaie sur un chemin de commutation qui a été calculé et établi à partir des règles définies par les applications SDN, passées au contrôleur avant d'être être descendues au Switch.

Ces règles d'acheminement des données (data forwarding rules) sont présentes dans les tables d'acheminement (forwarding tables). Ces règles indiquent pour des catégories de paquets données quel doit être le prochain saut sur la route. Le Switch peut, par ailleurs, modifier l'en-tête du paquet avant son acheminement, ou rejeter le paquet. Comme montré à la figure 5, les paquets arrivant sont placés dans une file d'attente en entrée, attendant leurs traitements par le Switch; les paquets acheminés sont placés dans une file d'attente en sortie, avant d'être transmis.

Le Switch à la figure 7 dispose d'au moins trois ports d'entrée/sortie: Un port dédié au contrôle par un contrôleur SDN, et les autres ports sont destinés à recevoir et transmettre les paquets de données. Il s'agit d'un exemple simple. Le Switch peut disposer de plusieurs ports pour communiquer avec plusieurs contrôleurs SDN, et de plus de deux ports pour les paquets de données entrant et sortant du Switch.[1]

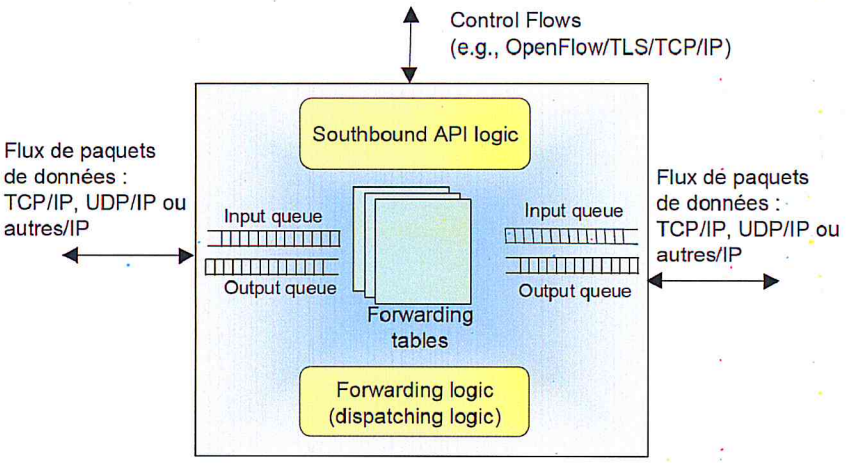


Figure 7.Principaux fonctions d'un Switch OpenFlow.

Les flux de données consistent en des flux de paquets IP. Il peut être nécessaire pour la table d'acheminement (forwarding table) de définir des entrées sur la base de champs d'entête de protocole de couche supérieure, tel que TCP, UDP, SCTP ou protocole d'application. Le Switch analyse l'en-tête IP et si nécessaire d'autre en-têtes dans chaque paquet et prend une décision pour son acheminement.

L'autre flux de données important est via l'interface sud (southbound) consistant en le protocole OpenFlow ou tout protocole équivalent même si OpenFlow est le protocole de référence.

La figure 6 décrit les fonctions de bases du Switch OpenFlow et sa relation avec le contrôleur.

Comme attendu dans un Switch de paquet, la fonction de base est de recevoir les paquets qui arrivent sur un port (Chemin X sur port 2 sur la figure 8) et les relayer via un autre port (Port N sur la figure 8) en réalisant toute modification nécessaire sur les paquets durant son chemin. La fonction de correspondance de paquet (packet-matching function) est très importante dans le Switch OpenFlow. La table adjacente est une table de flux, La flèche grisée sur le chemin commence dans la logique de décision, montre une correspondance avec une entrée particulière dans la table de flux, et dirige le paquet pour lequel une correspondance a été trouvée à une case action sur la droite.[1]

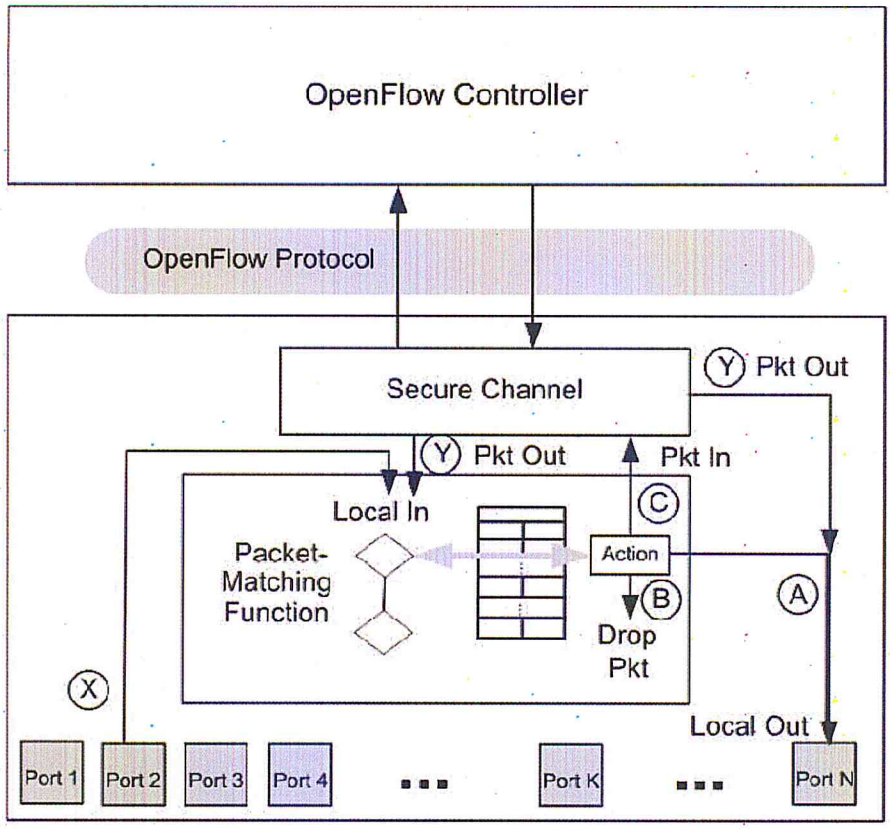


Figure 8.Fonctions du Switch OpenFlow

La case action a trois options de base pour le traitement du paquet arrivé :

A. Relayer le paquet sur un port de sortie, avec auparavant la possibilité de modifier certains champs d'en-tête du paquet.

B. Supprimer le paquet

C. Passer le paquet au contrôleur. Le paquet est encapsulé dans un message OpenFlow PACKET_IN.

Ces trois chemins fondamentaux pour le paquet sont illustrés ci-dessus. Dans le cas du chemin C, le paquet est passé au contrôleur via un canal sécurisé montré à la figure 8 Si le contrôleur a soit un message de contrôle (e.g. mettre hors service un port du Switch) ou un paquet de données à fournir au Switch, le contrôleur utilise ce même canal sécurisé dans le sens inverse. Lorsque le contrôleur a un paquet de données à relayer via le Switch, il utilise le message OpenFlow PACKET-OUT. Sur la figure 8, un paquet de données provenant du contrôleur peut suivre deux chemins, dénotés Y via la logique OpenFlow. Dans le cas du chemin Y de droite; le contrôleur spécifie directement le port de sortie et le paquet est passé à ce port N. Dans le cas du chemin Y de gauche, le contrôleur indique qu'il souhaite déléguer la décision de transfert du paquet à la logique de correspondance de paquet. Le contrôleur stipule alors le port de sortie TABLE pour le traitement du paquet par la fonction de correspondance de paquet et l'identification par ce traitement du port de sortie.

Un Switch OpenFlow peut être uniquement OpenFlow ou hybride. Dans ce dernier cas, le Switch peut aussi commuter les paquets selon son mode traditionnel. Le cas hybride peut être vu comme un Switch OpenFlow qui réside à côté d'un Switch traditionnel et indépendant de lui. Ce type de Switch hybride requiert un mécanisme de classification qui soit diriger les paquets au contrôleur OpenFlow, soit les traiter de manière traditionnelle.

La spécification OpenFlow définit le concept de port OpenFlow. Il s'agit d'un port physique dans OpenFlow 1.0. Pendant de nombreuses années, les Switchs ont supporté de nombreuses files d'attente par port physique.

Ces files d'attente sont servies par des algorithmes d'ordonnancement qui contribuent à fournir différents niveaux de QoS pour différents types de paquet.

OpenFlow prend en compte ce concept et permet au flux d'être envoyé sur une file d'attente déjà définie sur un port de sortie. La sortie du paquet sur un port N peut inclure le numéro de file d'attente du port N dans laquelle placer le paquet.[1]

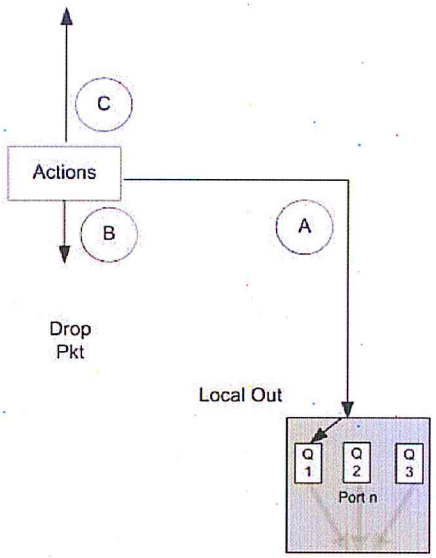


Figure 9.Ports physique et file d'attente OpenFlow

II.2.1 Table de flux

Chaque table de flux du Switch contient un ensemble d'entrées qui présentent les règles d'acheminement des paquets par flux.

Une entrée de flux est composée de (Figure 10):

- ✓ **Match fields :** champs de correspondance qui définit le modèle du flux de paquets à travers un masque couvrant 11 champs d'en-tête allant de la couche Ethernet à la couche Transport
- ✓ **Counters :** des compteurs sur les paquets ayant déclenchés la règle en question
- ✓ **Actions :** Actions à appliquer aux paquets qui correspondent à l'entrée de flux.[1]

Figure 10.Structure d'une entrée de flux

Chaque entrée de flux est associée à zéro ou plusieurs actions qui dictent la façon dont le Switch gère les paquets correspondants. Si aucune action n'est présente, le paquet est supprimé. Les listes d'actions présentes dans les entrées de flux doivent être traitées dans l'ordre spécifié. Cependant, il n'y a pas d'ordre de sortie garanti au paquet vers un port donnée.

Par exemple, une liste d'actions peut résulter en deux paquets envoyés à deux différents VLANs sur un seul port. Ces deux paquets peuvent être arbitrairement réordonnés, mais les corps des paquets doivent correspondre à ceux générés par une exécution séquentielle des actions.

Un Switch peut rejeter une entrée de flux si elle ne peut pas traiter la liste d'actions dans l'ordre spécifié, auquel cas il doit immédiatement retourner un message d'erreur « flux non pris en charge » au contrôleur. Un Switch n'a pas à supporter tous les types d'action - seulement celles marquées " Actions obligatoires ". Lors de la connexion au contrôleur, un Switch indique lesquelles des actions optionnelles il supporte.

II.2.2. Champs de correspondance

Les champs d'entêtes couverts par le Match fields spécifié dans OpenFlow 1.0 sont :

- **Ingress Port :** port d'entrée sur lequel est reçu le paquet
- **Ethernet source address (48) :** adresse Ethernet source
- **Ethernet destination address (48):** adresse Ethernet destination. Il est à noter que seule la couche Ethernet est considérée au niveau liaison de données.
- **Ethernet frame type (16) :** Type de trame Ethernet : Ethernet II, LLC ou NSAP.
- **VLAN id (12) :** Identificateur de VLAN dans l'en-tête VLAN si présent.
- **VLAN priority (3) :** Priorité VLAN dans l'en-tête VLAN si présent.
- **IPv4 source address (32) :** adresse IPv4 source du paquet
- **IPv4 destination address (32) :** adresse IPv4 destination du paquet
- **IP protocol (8) :** Protocole contenu dans IP, e.g., OSPF, TCP, UDP, etc.
- **IP ToS bits (6) :** QoS spécifiée dans le paquet via le champ ToS (Type of Service)
- **Transport source port /ICMP Type :** Port source (couche transport)
- **Transport destination port /ICMP code :** Port destination

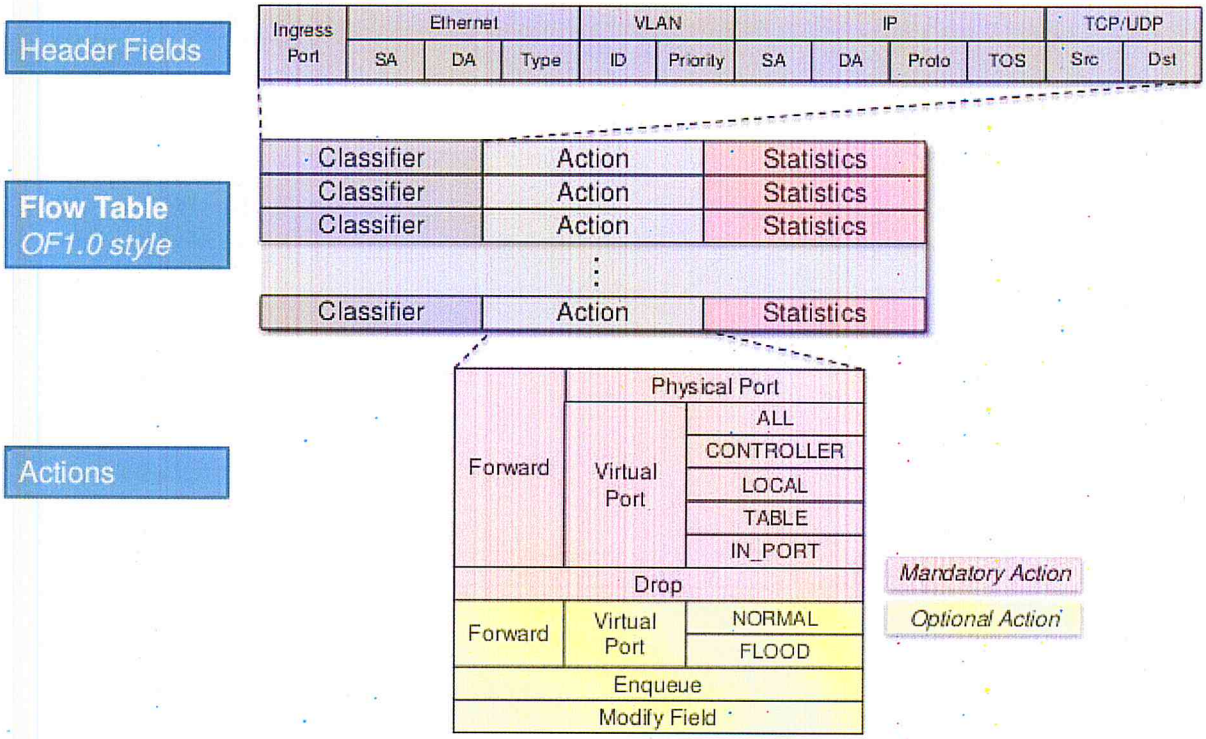


Figure 11.Les champs d'entêtes et de la table de flux

II.2.3. Compteurs

Les compteurs peuvent être maintenus pour chaque table de flux, entrée de flux, port, file d'attente. La liste ci-dessus décrit les compteurs.

Par Flow Table :

- **Reference Count (active entries) :** Nombre d'entrées actives dans la table.
- **PacketLookups :** Nombre de paquets soumis à la table.
- **Packet Matches :** Nombre de paquets pour lesquels une des entrées de la table a pu s'appliquer.

Par Flow entry :

- **ReceivedPackets :** Nombre de paquets reçus par l'entrée pour lesquels la recherche de correspondance a réussi.
- **Received Bytes :** Nombre d'octets de paquets reçus par l'entrée pour lesquels la recherche de correspondance a réussi.
- **Duration (seconds) :** Durée en secondes pendant laquelle l'entrée a été active.
- **Duration (nanoseconds) :** Durée en nanosecondes pendant laquelle l'entrée a été active au-delà de la durée en secondes.

Par Port :

- **ReceivedPackets :** Nombre de paquets reçus sur ce port
- **TransmittedPackets :** Nombre de paquets transmis par ce port

- **ReceivedBytes :** Nombre d'octets reçus par ce port
- **Transmitted Bytes :** Nombre d'octets transmis par ce port
- **Receive Drops :** Nombre de paquets reçus et rejetés par ce port
- **Transmit Drops :** Nombre de paquet rejetés en transmission
- **ReceiveErrors :** Nombre de paquets reçus en erreur
- **Transmit Errors :** Nombre de paquet transmis en erreur
- **Receive Frame AlignmentErrors :** Nombre de paquet reçus avec erreur d'alignement
- **ReceiveOverrunErrors :** Nombre de paquets reçus et rejetés faute de mémoire dans les files d'attente en entrée du port
- **Receive CRC Errors :** Nombre de paquets reçus avec un CRC erroné
- **Collisions :** Nombre de paquets rejetés suite à une collision

Par Queue (file d'attente)

- **Transmit Packets :** Nombre de paquets transmis sur cette file d'attente
- **Transmit Bytes :** Nombre d'octets transmis sur cette file d'attente
- **Transmit OverrunErrors :** Nombre de paquet transmis sur cette file d'attente être jetés faute de mémoire sur la file.
- Lorsqu'un compteur arrive à sa valeur maximum, il repasse à 0 sans autre indication. Si un compteur n'est pas disponible, sa valeur doit être positionnée à -1.[1]

II.2.4. Actions

Il existe des actions obligatoires que doit supporter tout Switch OpenFlow et des actions optionnelles.

Action (Obligatoire): Forward. Les Switchs OpenFlow doivent supporter l'acheminement de paquets aux ports physiques ainsi qu'aux ports virtuels suivants :

ALL: Est utilisé pour inonder un paquet sur tous les ports du Switch à l'exception du port d'entrée ; ceci permet d'offrir une capacité broadcast rudimentaire au SwitchOpenFlow.

CONTROLLER: Encapsuler le paquet et l'envoyer au contrôleur (Message PACKET_IN du Switch au contrôleur). Cette action peut être indiquée dans l'entrée de flux. Aussi, si aucune entrée ne correspond au paquet à acheminer, le Switch émet par défaut ce paquet au contrôleur.

LOCAL: Envoyer le paquet au contrôleur local embarqué dans le Switch (à ne pas confondre avec le traitement normal qui est indiqué par l'action NORMAL décrit ci-dessous).

TABLE: S'applique uniquement aux paquets que le contrôleur émet au Switch. Ces paquets arrivent via le message PACKET_OUT du contrôleur, incluant la liste d'actions. Cette liste d'actions va généralement contenir une action de sortie, qui spécifie un numéro de port. Le contrôleur peut vouloir directement spécifier le port de sortie pour ce paquet de données, ou

vouloir que le port soit déterminé par le traitement de paquet OpenFlow normal; dans ce dernier cas, il stipule TABLE comme port de sortie.

IN PORT: Envoyer le paquet sur le port d'entrée. Cette action est nécessaire lorsque l'émetteur et le récepteur du paquet sont joignables via le même port du Switch (e.g., émetteur et récepteur présents sur le même hotspot WiFi).

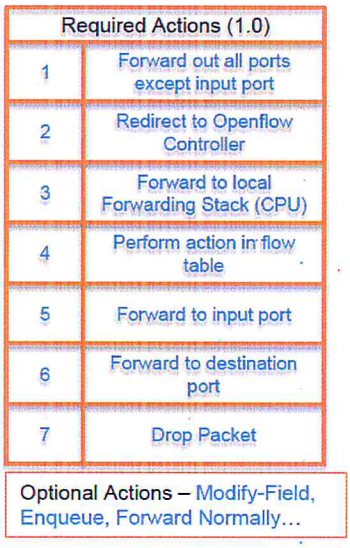


Figure 12.Liste des actions dans la table de flux

Action (Optionnelle) Forward : Le Switch peut en option supporter le port virtuel NORMAL qui revient à traiter le paquet selon la méthode de commutation traditionnelle. Le paquet doit donc être soumis à la logique d'acheminement traditionnelle du Switch. Il faut distinguer le port virtuel NORMAL du port virtuel LOCAL, qui signifie que le paquet doit être passé au traitement du contrôle OpenFlow local. De même, les paquets pour lesquels l'entrée correspondante indique NORMAL comme port de sortie doivent induire l'interrogation des tables d'acheminement qui sont mises à jour par le plan de contrôle non-OpenFlow local. L'utilisation de NORMAL a du sens uniquement si le Switch est hybride (Switch à la fois OpenFlow et traditionnel).

Action (Obligatoire)Drop : Une entrée de flux sans action indique que tous les paquets correspondants doivent être supprimés.

Action (Optionnelle) Enqueue : Cette action relaie un paquet via une file d'attente associée à un port. Le comportement d'acheminement est dicté par la configuration de la file d'attente et permet de fournir un support pour une qualité de service.

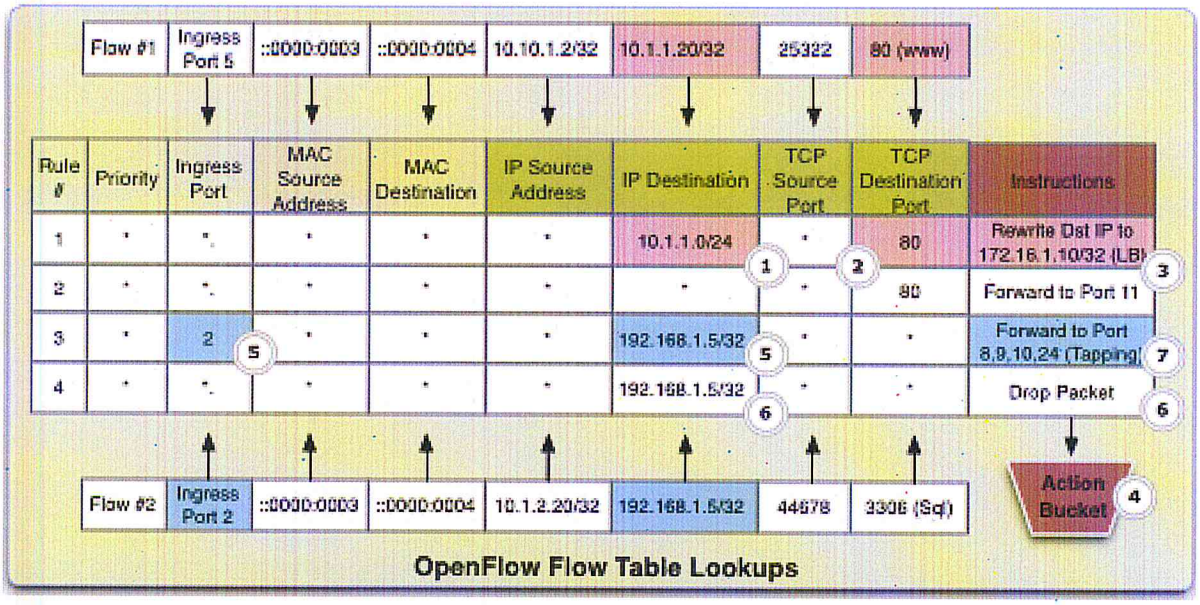


Figure 13.table de flux

Action (Optionnelle)Modify Field : Les actions indiquées ci-dessous apportent de la valeur à une implantation OpenFlow; elles permettent de modifier des champs d'en-tête du paquet avant son acheminement sur un port de sortie, en particulier, les en-têtes VLAN, les adresses Ethernet source et destination, les adresses Pv4 source et destination, et les numéros de port TCP ou UDP.

Set VLAN ID (12 bits) : Si aucun en-tête VLAN n'est présent, un nouvel en-tête est ajouté avec le VLAN ID et une priorité à 0. Si un en-tête VLAN est présent, le VLAN-ID est remplacé avec la valeur spécifiée.

Set VLAN priority (3 bits) : Si aucun en-tête VLAN n'est présent, un nouvel en-tête est ajouté avec la priorité spécifiée et un VLAN ID positionné à 0. Si un en-tête VLAN est déjà présent, le champ priorité est remplacé par la priorité spécifiée.

Strip VLAN header : supprimer l'en-tête VLAN si présent.

Modify Ethernet source MAC address (48 bits) : Remplace l'adresse MAC source avec la nouvelle valeur.

Modify Ethernet destination MAC address (48 bits) : Remplace l'adresse MAC destination avec la nouvelle valeur.

Modify IPv4 source address (32 bits) : Remplace l'adresse source IPv4 avec la nouvelle valeur et recalculer le checksum IP.

Modify IPv4 destination address (32 bits) : Remplace l'adresse destination IPv4 avec la nouvelle valeur et recalculer le checksum IP.

Modify IPv4 ToS bits (6 bits): Remplace le champ IPv4 ToS avec la nouvelle valeur.

Modify transport source port (16 bits) : Remplace le numéro de port source TCP ou UDP avec la nouvelle valeur et recalculer le checksum TCP ou UDP.

Modify transport destination port (16 bits) : Remplace le numéro de port destination TCP ou UDP avec la nouvelle valeur et recalculer le checksum TCP ou UDP.

La figure 14 présente quelques exemples d'entrée de table de flux. La première entrée concerne la commutation Ethernet, la seconde, le filtrage de trafic et la troisième, le routage du trafic.[1]

Switching

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	00:1f:..	*	*	*	*	*	*	*	port6

Firewall

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	*	*	*	22	drop

Routing

Switch Port	MAC src	MAC dst	Eth type	VLAN ID	IP Src	IP Dst	IP Prot	TCP sport	TCP dport	Action
*	*	*	*	*	*	5.6.7.8	*	*	*	port6

Figure 14.Exemples d'entrées de table de flux

II.3. Ports OpenFlow

Les ports OpenFlow sont les interfaces réseau pour passer des paquets entre le traitement OpenFlow et le Reste du réseau. Les Switchs OpenFlow se connectent entre eux via leurs ports OpenFlow, un paquet peut être transmis d'un Switch OpenFlow à un autre uniquement via un Port de sortie OpenFlow sur le premier Switch et un port OpenFlow d'entrée sur le deuxième Switch.

Un Switch OpenFlow rend disponible un certain nombre de ports pour le traitement OpenFlow. L'ensemble des Les ports OpenFlow peuvent ne pas être identiques à l'ensemble des interfaces réseau fournies par le matériel du Switch, Certaines interfaces réseau peuvent être désactivées pour OpenFlow, et le protocole OpenFlow peut définir des paramètres supplémentaires à ses Ports.

Les paquets OpenFlow sont reçus sur un port d'entrée et traités par le pipeline OpenFlow (figure 15) Qui peut les transmettre à un port de sortie. Le port d'entrée de paquet est une propriété du paquet Dans tout le pipeline OpenFlow et représente le port OpenFlow sur lequel le paquet a été reçu Dans le Switch OpenFlow. Le port d'entrée peut être utilisé lors de l'appariement des paquets. L'Open-Flow pipeline peut décider d'envoyer le paquet sur un port de sortie à l'aide de l'action de sortie, qui Définit comment le paquet retourne au réseau.[8]

II.3.1. Ports physiques

Les ports physiques OpenFlow sont des ports qui correspondent à des interfaces matérielles dans le Switch. Par exemple, sur un Switch Ethernet, les ports physiques se correspondent individuellement aux interfaces Ethernet.

Dans certains déploiements, le Switch OpenFlow peut être virtualisé sur le matériel du Switch. Dans ces cas, un port physique OpenFlow peut représenter une tranche virtuelle de l'interface matérielle correspondante au Switch.

II.3.2. Ports logiques

Les ports logiques OpenFlow sont des ports qui ne correspondent pas directement à une interface matérielle dans le Switch. Les ports logiques sont des abstractions de niveau supérieur qui peuvent être définies dans le Switch En utilisant des méthodes non OpenFlow (par exemple, groupes d'agrégation de liens, tunnels, interfaces de bouclage).

Les ports logiques peuvent inclure l'encapsulation de paquets et peuvent correspondre à différents ports physiques. Le traitement fait par le port logique dépend de l'implémentation et doit être transparent pour le traitement OpenFlow et ces ports doivent interagir avec le traitement OpenFlow comme les ports physiques OpenFlow.

Les seules différences entre les ports physiques et les ports logiques sont qu'un paquet associé à un port logique peut avoir un champ de pipeline supplémentaire appelé Tunnel-ID associé à celui-ci et lorsqu'un paquet reçu sur un port logique est envoyé au contrôleur, son port logique et son port physique sous-jacent sont signalés à la fois au contrôleur.

II.3.3. Ports réservés

Les ports réservés OpenFlow sont définis par cette spécification. Ils spécifient un renvoi générique des actions telles que l'envoi au contrôleur, la diffusion ou le renvoi en utilisant des méthodes non OpenFlow, comme dans le traitement de commutation "normal".

II.3.4. Modifications du port

Une configuration de commutation, par exemple en utilisant le protocole de configuration OpenFlow, peut ajouter ou supprimer des ports à partir du Switch OpenFlow à tout moment. Le Switch peut changer l'état du port en fonction de mécanisme de port sous-jacent, par exemple si le lien est en baisse, le contrôleur ou la configuration du Switch peut modifier la configuration du port. Des changements de ce genre à l'état du port ou la configuration doit être communiquée au contrôleur OpenFlow.

II.3.5.Recirculation du port

Les ports logiques peuvent éventuellement être utilisés pour insérer un service réseau ou un traitement complexe dans OpenFlow. Le plus souvent, les paquets envoyés aux ports logiques ne reviennent jamais au même Switch OpenFlow,Ils sont soit consommés par le port logique, soit éventuellement envoyés sur un port physique. Dans d'autres cas,les paquets envoyés à un port logique sont recyclés dans le Switch OpenFlow après le port logique en traitement.[1]

II.4.Traitement des flux OpenFlow

Cette section décrit les traitements des tables de flux et des tables de groupe, ainsi que le mécanisme de correspondance et gestion de l'action.

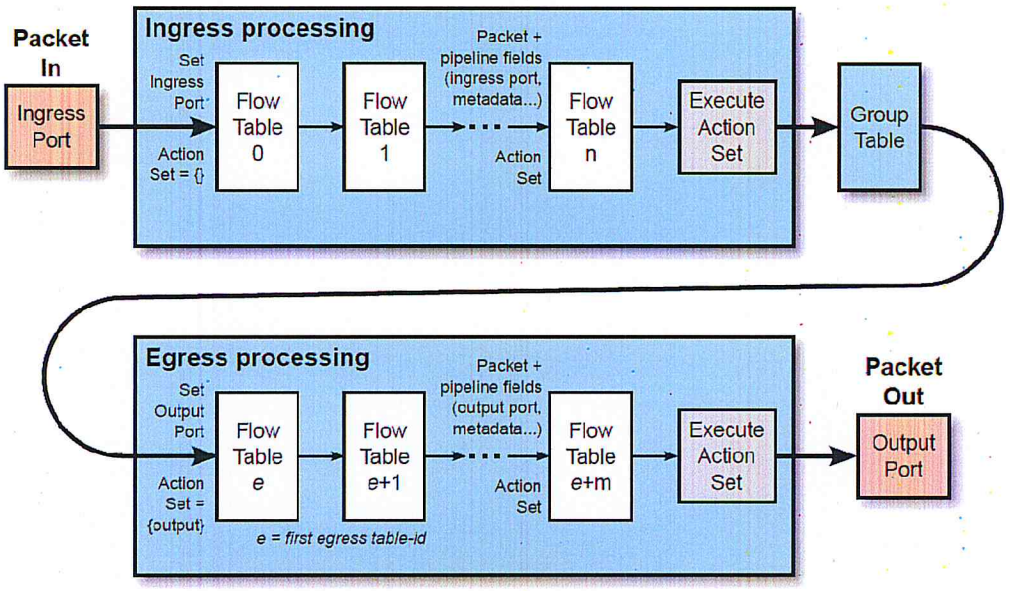


Figure 15. Le traitement de flux dans le pipeline.

Le pipeline OpenFlow de chaque Switch logique OpenFlow contient une ou plusieurs tables de flux, chaque table de flux contenant plusieurs entrées de flux. Le traitement du pipeline OpenFlow définit comment les paquets interagissent avec ces tables de flux (voir la figure 15). Un contrôleur OpenFlow est requis pour avoir au moins un flux d'entrée table, et peut éventuellement avoir plus de tables d'écoulement. Un Switch OpenFlow avec une seule table de flux est valide, dans ce cas, le traitement des pipelines est grandement simplifié.

Les tables de flux d'un Switch OpenFlow sont numérotées séquentiellement, à partir de 0. Le traitement du pipeline se déroule en deux étapes, le traitement de l'entrée et le traitement des sorties. La séparation des deux étapes est indiquée par la première table de sortie, toutes les tables avec un nombre inférieur à la première table de sortie doit être utilisé comme table d'entrée, et aucune table avec un nombre supérieur ou égal à la première table de sortie peut être utilisé comme table de saisie.

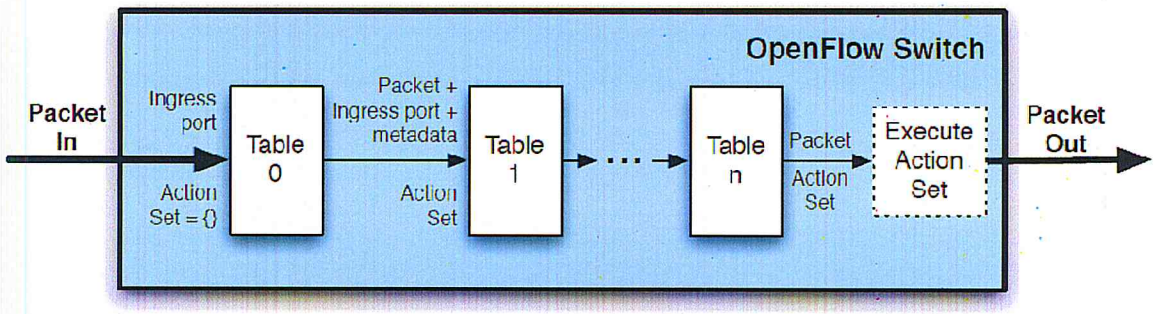


Figure 16.Flux de données détaillant le flux d'un paquet

Le traitement du pipeline commence toujours par le traitement de l'entrée à la première table de flux: le paquet doit être d'abord adapté aux entrées de flux de la table d'écoulement 0 (voir la figure 16). D'autres tables de flux d'entrée peuvent être utilisées en fonction du résultat de la correspondance (Matching) dans la première table. Si le résultat du traitement de l'entrée est pour transférer le paquet vers un port de sortie, le Switch OpenFlow peut effectuer un traitement de sortie dans le contexte de ce port de sortie. Le traitement de sortie est facultatif, un Switch peut ne pas supporter les tables de sortie ou peut ne pas être configuré pour les utiliser. Si aucune table de sortie valide n'est configurée comme première table de sortie, le paquet doit être traité par le port de sortie, et dans la plupart des cas, le paquet est transmis vers le switch. Si une table de sortie valide est configurée comme première table de sortie, le paquet doit correspondre aux entrées de flux de cette table de flux, et d'autres tableaux de flux de sortie peuvent être utilisés en fonction du résultat du matching dans cette table de flux.

Lorsqu'il est traité par une table de flux, le paquet est adapté aux entrées de flux de la table de flux pour sélectionnez une entrée de flux. Si une entrée de flux est trouvée, l'ensemble d'instructions inclus dans cette entrée de flux est réalisé. Ces instructions peuvent ordonner explicitement le paquet vers une autre table de flux (en utilisant l'instruction Goto-Table , où le même processus est répété à nouveau). Une entrée de flux ne peut que diriger un paquet vers un numéro de table de flux qui est supérieur à son propre numéro de table de flux, autrement dit le traitement dans le pipeline ne peut avancer . Les entrées de flux du dernier tableau d'un pipeline ne peut pas inclure l'instruction Goto-Table. Si l'entrée de flux correspondante ne dirige pas de paquets à une autre table de flux, l'étape actuelle du traitement de pipeline s'arrête à ce tableau, le paquet est traité avec son jeu d'action associé et généralement renvoyé.

Si un paquet ne correspond pas à une entrée de flux dans une table de flux, il s'agit d'une erreur de table (appelé table-miss). Le comportement à une table-miss dépend de la configuration de la table. Les instructions incluses dans l'entrée « table-miss flow » dans la table de flux peut spécifier comment traiter des paquets inégalés, les options utiles incluent le déclenchement eux, en les passant dans une autre table ou en les envoyant aux contrôleurs sur le canal de commande via les messages de paquets.

Il existe peu de cas où un paquet n'est pas entièrement traité par une entrée de flux et le traitement de pipeline s'arrête sans traiter l'ensemble d'actions du paquet ou le diriger vers un autre tableau. Si aucune entrée de débit table-miss n'est présente, le paquet est supprimé. Si une TTL invalide est trouvée, le paquet peut être envoyé au contrôleur.

Le pipeline OpenFlow et diverses opérations OpenFlow traitent des paquets de type spécifique en conformité avec les spécifications définies pour ce type de paquet, à moins que la spécification actuelle ou la configuration OpenFlow spécifie autrement. Par exemple, la définition d'en-tête Ethernet utilisée par Open-Flow doit être conforme aux spécifications IEEE, et la définition d'en-tête TCP / IP utilisée par OpenFlow doit être conforme aux RFCs associés. De plus, la réorganisation des paquets dans un Switch OpenFlow doit être conforme aux exigences des spécifications IEEE, à condition que les paquets soient traités par le même flux d'entrés.

II.5. Messages OpenFlow

Les messages OpenFlow entre le contrôleur et le Switch sont transmis via un canal sécurisé, implanté via une connexion TLS/SSL sur TCP. Le Switch initie la connexion TLS/SSL lorsqu'il connaît l'adresse IP du contrôleur. Chaque message entre le Switch et le contrôleur commence avec l'en-tête OpenFlow. Cet en-tête spécifie le numéro de version OpenFlow, le type de message, la longueur de message, et l'identificateur de transaction du message. Il existe trois catégories de message : symmetric, controller-switch et async.

II.5.1. Messages symétriques

Les messages symétriques peuvent être émis indifféremment par le contrôleur ou le Switch sans avoir été sollicité par l'autre entité. Les messages HELLO sont échangés une fois que la canal sécurisé a été établi afin de déterminer le numéro de version OpenFlow le plus élevé supporté par le Switch et le contrôleur; Le protocole spécifie que la plus petite des deux versions doit être utilisée pour la communication contrôleur –Switch sur le canal sécurisé les messages ECHO sont utilisés par n'importe quel entité (contrôleur, Switch) pendant le fonctionnement du canal pour s'assurer que la connexion est toujours en vie et afin de mesurer la latence et le débit courants de la connexion. Chaque message ECHO_REQUEST doit être acquitté par un message ECHO_REPLY.[8]

Les messages VENDOR sont disponibles pour des améliorations et pour une expérimentation spécifique au vendeur.

II.5.2. Messages asynchrones

Les messages asynchrones sont émis par le Switch au contrôleur sans que le Switch ait été sollicité par le contrôleur. Le message PACKET_IN est utilisé par le Switch pour passer les paquets de données au

contrôleur pour leur prise en charge (lorsque par exemple aucune entrée de flux ne correspond au paquet entrant ou lorsque l'action de l'entrée correspondante spécifie que le paquet doit être relayé au contrôleur). Le trafic du plan de contrôle (e.g., paquet de routage OSPF) sera généralement relayé au contrôleur via ce message PACKET_IN. Si le Switch dispose d'une mémoire suffisante pour mémoriser les paquets qui sont envoyés au contrôleur, les messages PACKET-IN contiennent une partie de l 'en-tête (par défaut 128 octets) et un buffer ID à utiliser par le contrôleur. Les Switchs ne supportant pas de mémorisation interne (ou ne disposant plus de mémoire) émettent le paquet entier au contrôleur dans le message PACKET-IN.

Le Switch peut informer le contrôleur qu'une entrée de flux a été supprimée de la table de flux via le message FLOW_REMOVED. Cela survient lorsqu'aucun paquet entrant n'a de correspondance avec cette entrée pendant un temporisateur spécifié par le contrôleur lors de la création de cette entrée au niveau de la table de flux du Switch.

Le message PORT_STATUS est utilisé afin de communiquer un changement de configuration du port (port désactivé par un usager) ou changement d'état du port (le lien est hors service). Finalement le Switch utilise le message ERROR pour notifier des erreurs au contrôleur, à titre d'exemple : l'ajout d'une entrée de flux par le contrôleur contenant des actions non supportées par le Switch.

II.5.3. Messages contrôleur-Switch

Les messages contrôleur-Switch représentent la catégorie la plus importante de messages OpenFlow. Ils sont divisés en cinq sous-catégories : switch configuration, command from controller, statistics, queue configuration, et barrier.

Les messages « switch configuration consistent » en un message unidirectionnel et deux paires de messages requête-réponse. Le contrôleur émet le message unidirectionnel SET_CONFIG afin de positionner les paramètres de configuration du Switch. Le contrôleur utilise la paire de message FEATURES_REQUEST et FEATURES_REPLY afin d'interroger le Switch au sujet des fonctionnalités (notamment optionnelles) qu'il supporte. La paire de message GET_CONFIG_REQUEST et GET_CONFIG_REPLY est utilisée afin d'obtenir la configuration du Switch.

Les messages command à partir du contrôleur sont au nombre de trois. PACKET-OUT est analogue à PACKET_IN mentionné précédemment. Le contrôleur utilise PACKET_OUT afin d'émettre des paquets de données au Switch pour leur acheminement via le plan usager (Plan de données), le contrôleur modifie les entrées de flux existantes dans le Switch via le message FLOW_MOD, PORT_MOD est utilisé pour modifier l'état d'un port OpenFlow.

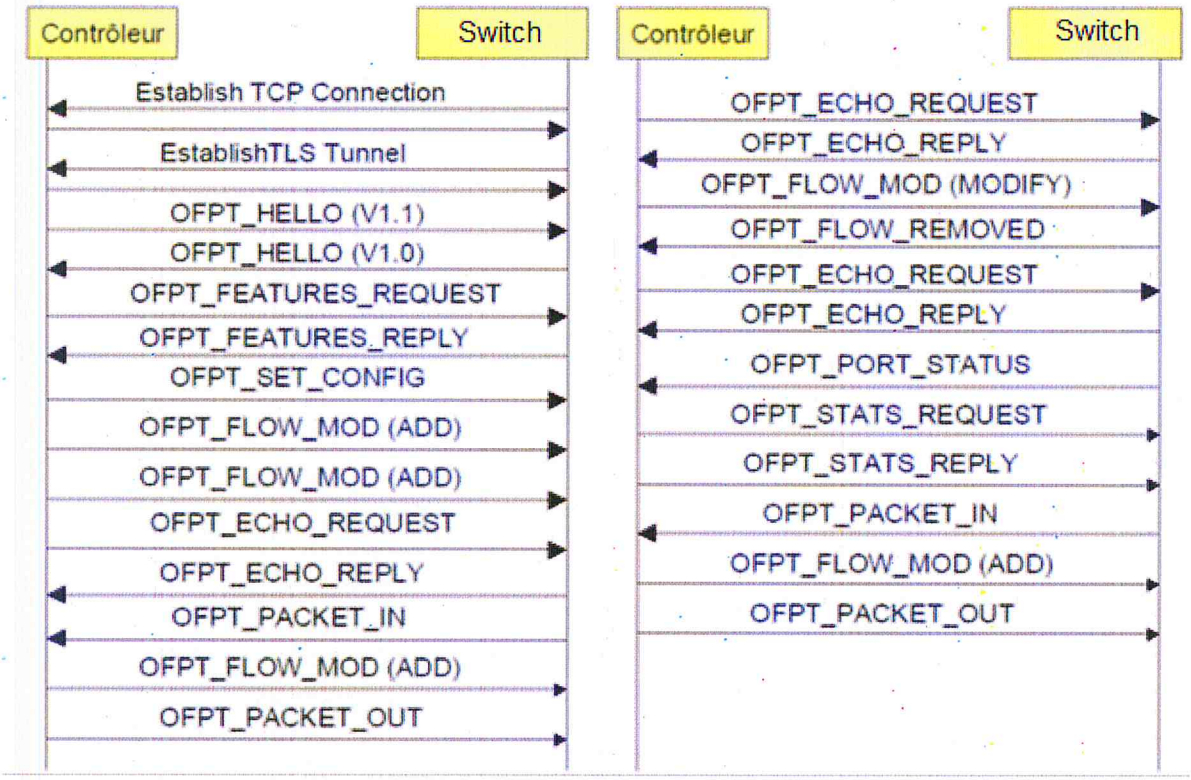


Figure 17.Echanges Openflow entre contrôleur et Switch

Des statistiques sont obtenues du Switch par le contrôleur via la paire de message STATS_REQUEST et STATS_REPLY La configuration de files d'attente associées à un port n'est pas spécifiée par OpenFlow. Un autre protocole de configuration doit être utilisé pour ce faire. Toutefois le contrôleur peut interroger le Switch via QUEUE_GET_CONFIG_REQUEST acquitté par QUEUE_GET_CONFIG_REPLY pour apprendre quelle est la configuration des files d'attente associées à un port afin de pourvoir acheminer des paquets sur des file s d'attente spécifiques et ainsi fournir à ces paquets un niveau de QoS désiré.

Le message BARRIER_REQUEST est utilisé par le contrôleur pour s'assurer que tous les messages OpenFlow émis par le contrôleur et qui ont précédé cette requête ont été reçus et traités par le Switch. La confirmation est retournée par le Switch via la réponse BARRRIER_REPLY.

II.6.Canal OpenFlow et Canal de contrôle

La chaîne OpenFlow est l'interface qui relie chaque Switch logique OpenFlow à un contrôleur OpenFlow. Grâce à cette interface, le contrôleur configure et gère le Switch, reçoit des événements à partir du Switch, et envoie des paquets au Switch. Le canal de commande du Switch peut supporter un seul canal OpenFlow avec un seul contrôleur, ou plusieurs canaux OpenFlow permettant plusieurs contrôleurs pour partager la gestion du Switch.

Entre le parc de données et le canal OpenFlow, l'interface est spécifique à la mise en œuvre, cependant tous les messages de canal OpenFlow doivent être formatés selon le protocole de Switch OpenFlow. Le canal OpenFlow est généralement crypté à l'aide de TLS, mais peut être exécuté directement sur TCP.[8]

II.7.Conclusion

Le protocole OpenFlow représente une brique essentiel dans l'architecture SDN au point où les deux concepts sont souvent confondus (par abus de langage). Il s'agit du protocole de communication dédié au contrôle des switchs (dits OpenFlow) par un contrôleur. OpenFlow est présent dans les switchs virtuels Open Source comme OpenvSwitch (OVS) mais également supportés de plus en plus par les constructeurs de switchs traditionnels où se trouve embarqué à coté de leurs fonctions propriétaires. Un switch typique est constitué de ports OpenFlow et de ports non-OpenFlow. Les ports OpenFlow peuvent être de différents types: physiques, logiques, réservés etc.

OpenFlow définit l'ensemble des messages nécessaires aussi bien à la gestion des switchs qu'à la gestion des flux de trafic reçus par ces derniers. Entre le contrôleur et le switch, des messages sont définis afin de prendre en charge tous les cas traités par les switchs traditionnels.

Un switch OpenFlow implémente un traitement souple des paquets à travers au moins un pipeline constitué d'une succession de tables de traitement de flux. Ces tables définissent le traitement à appliquer pour chaque flux correspondant à une entrée dans la table situé à l'entrée du pipeline. OpenFlow offre plusieurs actions applicables au trafic allant de la simple retransmission, de filtrage et au changement de l'entête. Les flux non reconnus sont envoyées au contrôleur pour prendre la décision adéquate.

Chapitre III : Présentation des Réseaux Privés Virtuels - VPN

III.1 - Introduction

Les applications et les systèmes distribués font de plus en plus partie intégrante du paysage d'un grand nombre d'entreprises. Ces technologies ont pu se développer grâce aux performances toujours plus importantes des réseaux locaux. Mais le succès de ces applications a fait aussi apparaître un de leur écueil. En effet si les applications distribuées deviennent le principal outil du système d'information de l'entreprise, comment assurer leur accès sécurisé au sein de structures parfois réparties sur de grandes distances géographiques ? Concrètement comment une succursale d'une entreprise peut-elle accéder aux données situées sur un serveur de la maison mère distant de plusieurs milliers de kilomètres? Les VPN ont commencé à être mis en place pour répondre à Ce type de problématique. Mais d'autres problématiques sont apparues et les VPN ont aujourd'hui pris une place importante dans les réseaux informatique et l'informatique distribuées. Nous verrons ici quelles sont les principales caractéristiques des VPN à travers un certain nombre d'utilisation type. Nous nous intéresserons ensuite aux protocoles permettant leur mise en place.[9]

III.2 - Principe de fonctionnement

III.2.1 - Principe général

Un réseau VPN repose sur un protocole appelé "protocole de tunneling". Ce protocole permet de faire circuler les informations de l'entreprise de façon sécurisé d'un bout à l'autre du tunnel. Ainsi, les utilisateurs ont l'impression de se connecter directement sur le réseau de leur entreprise.

Le principe de tunneling consiste à construire un chemin virtuel après avoir identifié l'émetteur et le destinataire. Par la suite, la source chiffre les données et les achemine en empruntant Ce chemin virtuel. Afin d'assurer un accès aisé et peu coûteux aux intranets ou aux extranets d'entreprise.

Les réseaux privés virtuels d'accès simulent un réseau privé, alors qu'ils utilisent en réalité une infrastructure d'accès partagée, comme Internet, les données à transmettre peuvent être prises en charge par un protocole différent d'IP. Dans Ce cas, le protocole de tunneling encapsule les données en ajoutant un en-tête. Le tunneling est l'ensemble des processus d'encapsulation, de transmission et de dés encapsulation.

III.2.2 - Fonctionnalités des VPN

Il existe 3 types standards d'utilisation des VPN. En étudiant ces schémas d'utilisation, il est possible d'isoler les fonctionnalités indispensables des VPN.

III.2.2.1 - Le VPN d'accès

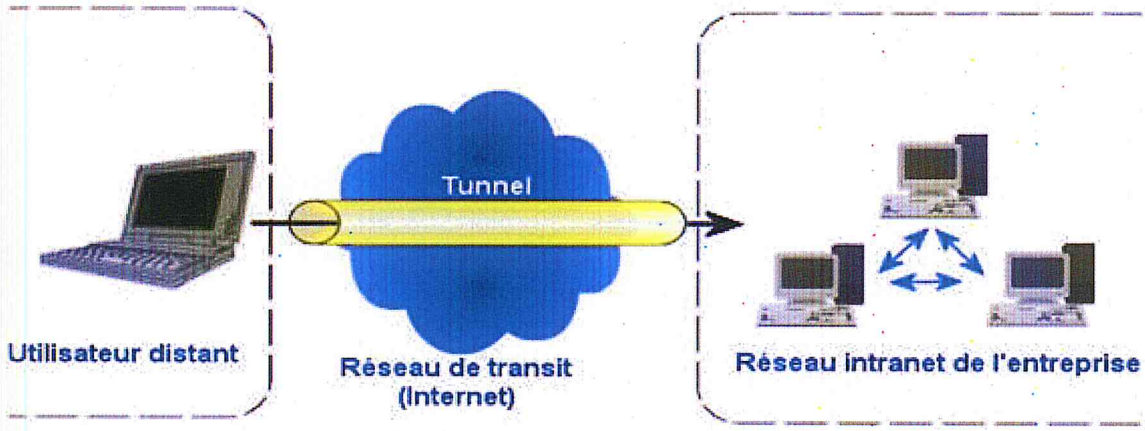


Figure 18.VPN d'accès

Le VPN d'accès est utilisé pour permettre à des utilisateurs itinérants d'accéder au réseau privé. L'utilisateur se sert d'une connexion Internet pour établir la connexion VPN. Il existe deux cas:

- ✓ L'utilisateur demande au fournisseur d'accès de lui établir une connexion cryptée vers le serveur distant : il communique avec le NAS (Network Access Server) du fournisseur d'accès et c'est le NAS qui établit la connexion cryptée.
- ✓ L'utilisateur possède son propre logiciel client pour le VPN auquel cas il établit directement la communication de manière cryptée vers le réseau de l'entreprise.

Les deux méthodes possèdent chacune leurs avantages et leurs inconvénients :

- ✓ La première permet à l'utilisateur de communiquer sur plusieurs réseaux en créant plusieurs tunnels, mais nécessite un fournisseur d'accès proposant un NAS compatible avec la solution VPN choisie par l'entreprise. De plus, la demande de connexion par le NAS n'est pas cryptée Ce qui peut poser des problèmes de sécurité.
- ✓ Sur la deuxième méthode Ce problème disparaît puisque l'intégralité des informations sera cryptée dès l'établissement de la connexion. Par contre, cette solution nécessite que chaque client transporte avec lui le logiciel, lui permettant d'établir une communication cryptée. Nous verrons que pour pallier Ce problème certaines entreprises mettent en place des VPN à base de SSL, technologie implémentée dans la majorité des navigateurs Internet du marché.

Quelle que soit la méthode de connexion choisie, Ce type d'utilisation montre bien l'importance dans le Vpn d'avoir une authentification forte des utilisateurs. Cette authentification peut se faire par une vérification "login / mot de passe", par un algorithme dit "Tokens sécurisés" (utilisation de mots de passe aléatoires) ou par certificats numériques.[13]

III.2.2.2 - L'intranet VPN

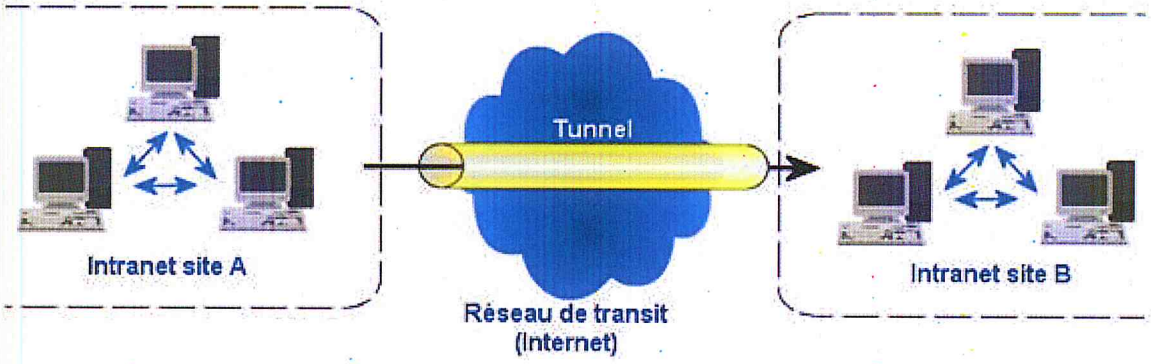


Figure 19.Intranet VPN

L'intranet VPN est utilisé pour relier au moins deux intranets entre eux. Ce type de réseau est particulièrement utile au sein d'une entreprise possédant plusieurs sites distants. Le plus important dans Ce type de réseau est de garantir la sécurité et l'intégrité des données. Certaines données très sensibles peuvent être amenées à transiter sur le VPN (base de données clients, informations financières...). Des techniques de cryptographie sont mises en œuvre pour vérifier que les données n'ont pas été altérées. Il s'agit d'une authentification au niveau paquet pour assurer la validité des données, de l'identification de leur source ainsi que leur non-répudiation. La plupart des algorithmes utilisés font appel à des signatures numériques qui sont ajoutées aux paquets. La confidentialité des données est, elle aussi, basée sur des algorithmes de cryptographie. La technologie en la matière est suffisamment avancée pour permettre une sécurité quasi parfaite. Le coût matériel des équipements de cryptage et décryptage ainsi que les limites légales interdisent l'utilisation d'un codage " infaillible ". Généralement pour la confidentialité, le codage en lui-même pourra être moyen à faible, mais sera combiné avec d'autres techniques comme l'encapsulation l'adresse IP pour assurer une sécurité raisonnable.

III.2.2.3 - L'extranet VPN

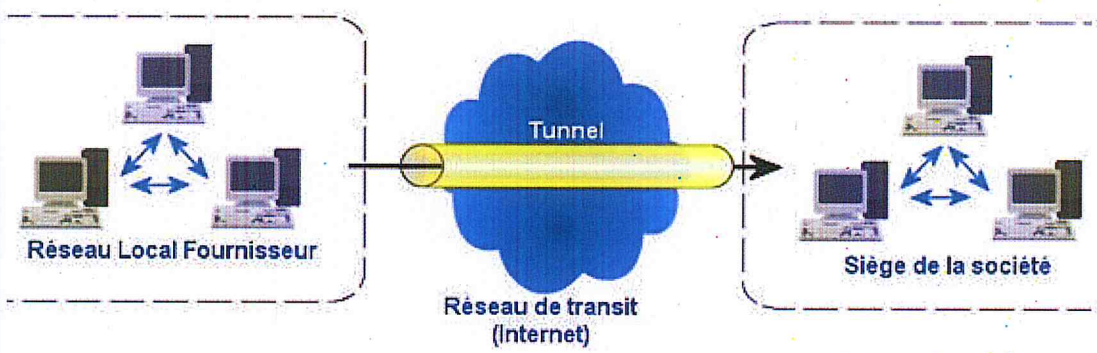


Figure 20.Extranet VPN

Une entreprise peut utiliser le VPN pour communiquer avec ses clients et ses partenaires. Elle ouvre

alors son réseau local à ces derniers. Dans Ce cadre, il est fondamental que l'administrateur du VPN puisse tracer les clients sur le réseau et gérer les droits de chacun sur celui-ci.

III.2.2.4 - Bilan des caractéristiques fondamentales d'un VPN

Un système de VPN doit pouvoir mettre en œuvre les fonctionnalités suivantes :

Authentification d'utilisateur : Seuls les utilisateurs autorisés doivent pouvoir s'identifier sur le réseau virtuel. De plus, un historique des connexions et des actions effectuées sur le réseau doit être conservé.

Gestion d'adresses : Chaque client sur le réseau doit avoir une adresse privée. Cette adresse privée doit rester confidentielle. Un nouveau client doit pouvoir se connecter facilement au réseau et recevoir une adresse.

Cryptage des données : Lors de leurs transports sur le réseau public les données doivent être protégées par un cryptage efficace.

Gestion de clés : Les clés de cryptage pour le client et le serveur doivent pouvoir être générées et régénérées.

Prise en charge multi protocole : La solution VPN doit supporter les protocoles les plus utilisés sur les réseaux publics en particulier l'IP.

Le VPN est un principe : il ne décrit pas l'implémentation effective de ces caractéristiques. C'est pourquoi il existe plusieurs produits différents sur le marché dont certains sont devenus standard, et même considérés comme des normes.[9]

III.3 - Protocoles utilisés pour réaliser une connexion VPN

Nous pouvons classer les protocoles que nous allons étudier en deux catégories:

- ✓ Les protocoles de niveau 2 comme PPTP et L2TP.
- ✓ Les protocoles de niveau 3 comme IPSEC ou MPLS.

Il existe en réalité trois protocoles de niveau deux (02) permettant de réaliser des VPN : PPTP (de Microsoft), L2F (développé par CISCO) et enfin L2TP. Nous n'évoquerons dans cette étude que PPTP et L2TP : le protocole L2F ayant aujourd'hui quasiment disparut. Le protocole PPTP aurait sans doute lui aussi disparut sans le soutien de Microsoft qui continue à l'intégrer à ses systèmes d'exploitation Windows.

L2tp est une évolution de PPTP et de L2F, reprenant les avantages des deux protocoles, les protocoles de couche 2 dépendent des fonctionnalités spécifiées pour Ppp (Point to Point Protocol), c'est pourquoi nous allons tout d'abord rappeler le fonctionnement de Ce protocole.

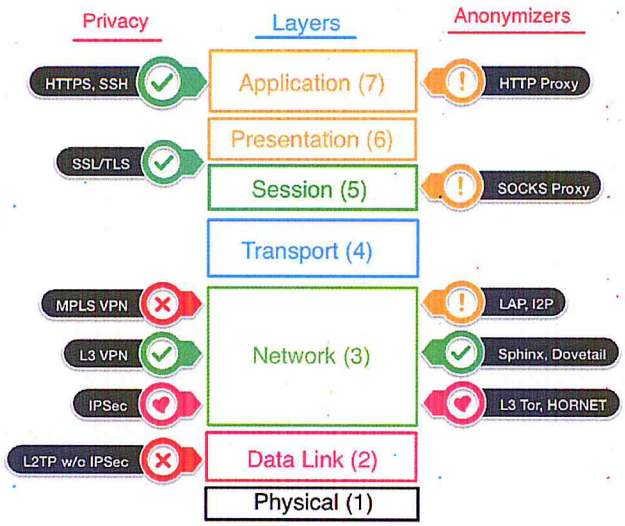


Figure 21.Les protocoles utilisés dans les VPN dans le Modèle OSI

III.3.1 - Le protocole IPSEC

Ipsec, est un protocole qui vise à sécuriser l'échange de données au niveau de la couche réseau(Figure 23). Le réseau IPv4 étant largement déployé et la migration vers IPV6 étant inévitable, mais néanmoins longue, il est apparu intéressant de développer des techniques de protection des données communes à IPv4 et IPv6. Ces mécanismes sont couramment désignés par le terme IPSEC pour IP Security Protocols. IPSEC est basé sur deux mécanismes. Le premier, AH, pour Authentification Header vise à assurer l'intégrité et l'authenticité des datagrammes IP. Il ne fournit par contre aucune confidentialité : les données fournies et transmises par Ce "protocole" ne sont pas cryptées. Le second, ESP, pour Encapsulating Security Payload permet en plus de cryptage des informations. Bien qu'indépendants ces deux mécanismes sont presque toujours utilisés conjointement. Enfin, le protocole IKE permet de gérer les échanges ou les associations entre protocoles de sécurité. Avant de décrire ces différents protocoles, nous allons exposer les différents éléments utilisés dans IPSEC.[9]

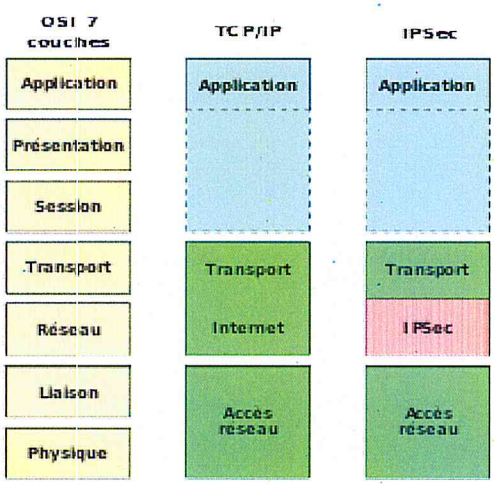


Figure 22.IPSEC dans le modèle OSI

III.3.1.1 - Vue d'ensemble

Les mécanismes mentionnés ci-dessus font bien sûr appel à la cryptographie et utilisent donc un certain nombre de paramètres (algorithmes de chiffrement utilisés, clefs, mécanismes sélectionnés...) sur lesquels les tiers communicants doivent se mettre d'accord. Afin de gérer ces paramètres, IPSEC a recours à la notion d'association de sécurité (Security Association, SA).

Une association de sécurité IPSEC est une "connexion" uni-directionnelle qui fournit des services de sécurité au trafic qu'elle transporte. On peut aussi la considérer comme une structure de données servant à stocker l'ensemble des paramètres associés à une communication donnée.

Une SA est unidirectionnelle; en conséquence, protéger les deux sens d'une communication classique requiert deux associations, une pour chaque sens. Les services de sécurité sont fournis par l'utilisation soit de AH soit de ESP. Si AH et ESP sont tous deux appliqués au trafic en question, deux SA sont créées; on parle alors de paquet (bundle) de SA.

Chaque association est identifiée de manière unique à l'aide d'un triplé composé de:

- ✓ L'adresse de destination des paquets,
- ✓ L'identifiant du protocole de sécurité utilisé (AH ou ESP),
- ✓ Un index des paramètres de sécurité (Security Parameter Index, SPI). Un SPI est un bloc de 32 bits inscrit en clair dans l'en-tête de chaque paquet échangé ; il est choisi par le récepteur.

Pour gérer les associations de sécurités actives, on utilise une "base de données des associations de sécurité" (Security Association Database, SAD). Elle contient tous les paramètres relatifs à chaque SA et sera consultée pour savoir comment traiter chaque paquet reçu ou à émettre.

Les protections offertes par IPSEC sont basées sur des choix définis dans une "base de données de politique de sécurité" (Security Policy Database, SPD). Cette base de données est établie et maintenue par un utilisateur, un administrateur système ou une application mise en place par ceux-ci. Elle permet de décider, pour chaque paquet, s'il se verra apporter des services de sécurité, s'il sera autorisé à passer ou rejeté.

III.3.1.2 - Principe de fonctionnement

Le schéma ci-dessous représente tous les éléments présentés ci-dessus (en bleu),leursp ositions et leurs interactions.

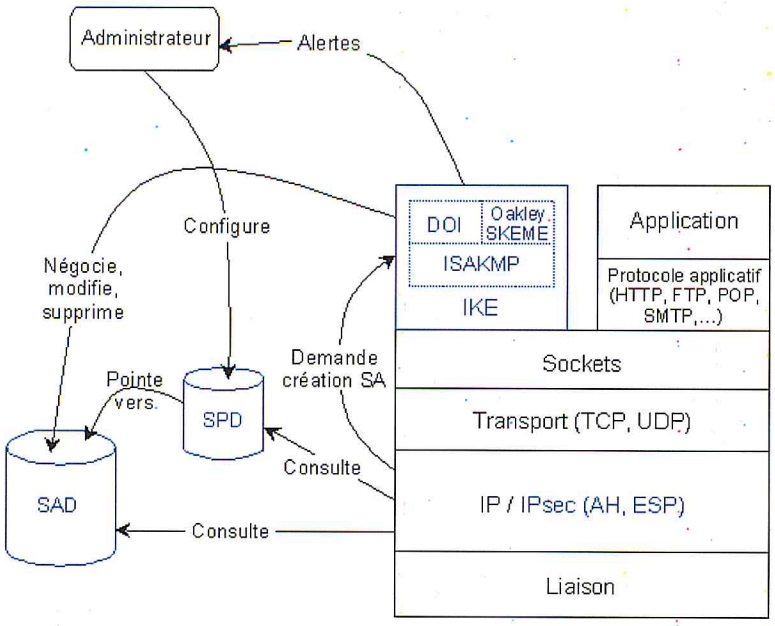


Figure 23.Processus d'un échange IPSEC

On distingue deux situations :

Trafic sortant : Lorsque la "couche" IPSEC reçoit des données à envoyer, elle commence par consulter la base de données des politiques de sécurité (SPD) pour savoir comment traiter ces données. Si cette base lui indique que le trafic doit se voir appliquer des mécanismes de sécurité, elle récupère les caractéristiques requises pour la SA correspondante et va consulter la base des SA (SAD). Si la SA nécessaire existe déjà, elle est utilisée pour traiter le trafic en question. Dans le cas contraire, IPSEC fait appel à IKE pour établir une nouvelle SA avec les caractéristiques requises.

Trafic entrant : Lorsque la couche IPSEC reçoit un paquet en provenance du réseau, elle examine l'en-tête pour savoir si Ce paquet s'est vu appliquer un ou plusieurs services IPSEC et si oui, quelles sont les références de la SA. Elle consulte alors la SAD pour connaître les paramètres à utiliser pour la vérification et/ou le déchiffrement du paquet. Une fois le paquet vérifié et/ou déchiffré, la SPD est consultée pour savoir si l'association de sécurité appliquée au paquet correspondait bien à celle requise par les politiques de sécurité.

Dans le cas où le paquet reçu est un paquet IP classique, la SPD permet de savoir s'il a néanmoins le droit de passer. Par exemple, les paquets IKE sont une exception. Ils sont traités par Ike, qui peut envoyer des alertes administratives en cas de tentative de connexion infructueuse.

III.3.1.3 - Le protocole AH (Authentication Header)

L'absence de confidentialité permet de s'assurer que Ce standard pourra être largement répandu sur Internet, y compris dans les endroits où l'exportation, l'importation ou l'utilisation du chiffrement dans des buts de confidentialité est restreint par la loi.

Son principe est d'adjoindre au datagramme IP classique un champ supplémentaire permettant à la réception de vérifier l'authenticité des données incluses dans le datagramme. Ce bloc de données est appelé "valeur de vérification d'intégrité" (Intégrity Check Value, ICV). La protection contre le rejet se fait grâce à un numéro de séquence.

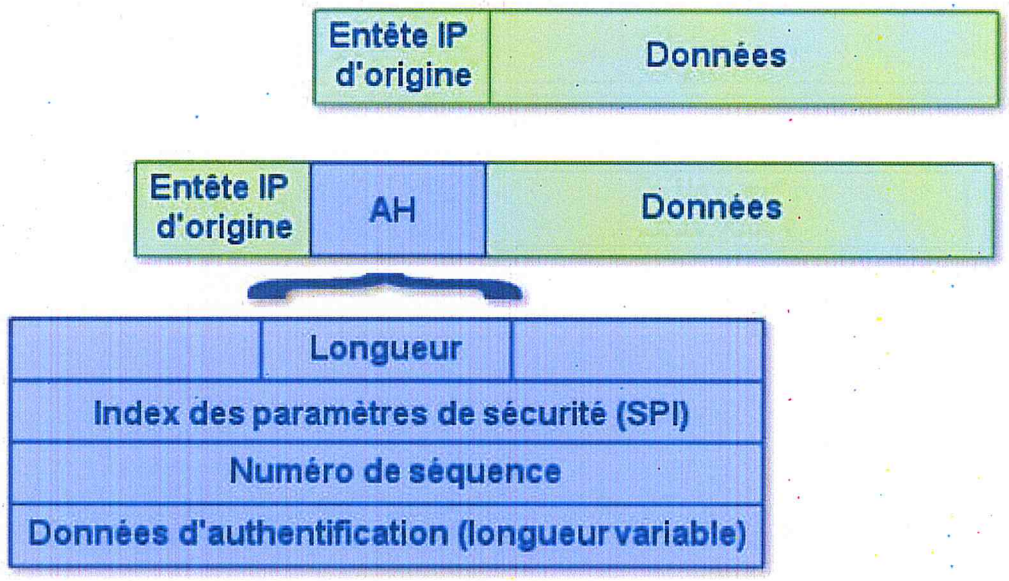


Figure 24.composition du datagramme AH

III.3.1.4 – Protocole ESP (Encapsulating Security Payload)

ESP peut assurer au choix, un ou plusieurs des services suivants :

- ✓ Confidentialité (confidentialité des données et protection partielle contre l'analyse du trafic si l'on utilise le mode tunnel).
- ✓ Intégrité des données en mode non connecté et authentification de l'origine des données, protection contre le rejoue.
- ✓ La confidentialité peut être sélectionnée indépendamment des autres services, mais son utilisation sans intégrité/authenticité (directement dans ESP ou avec AH) rend le trafic vulnérable à certains types d'attaques actives qui pourraient affaiblir le service de confidentialité.

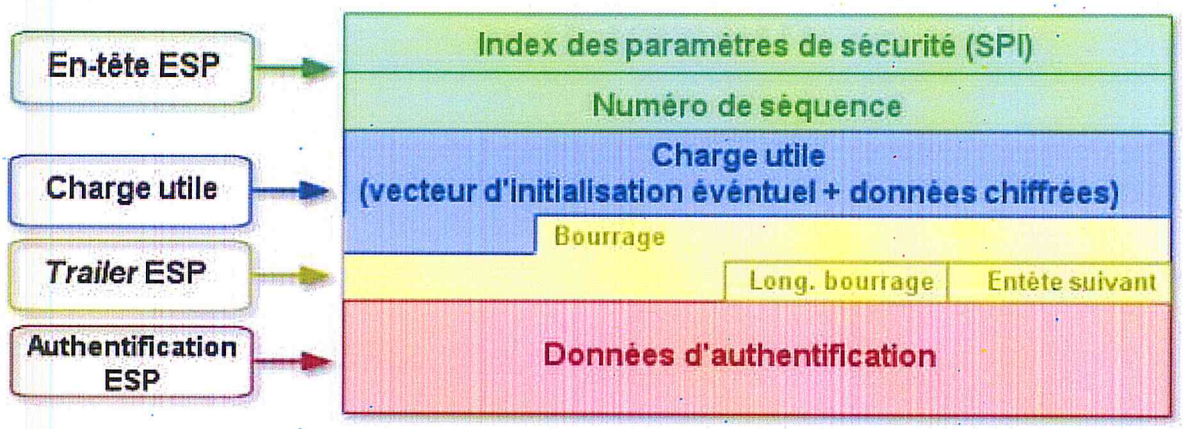


Figure 25.Le protocole ESP

Le champ bourrage peut être nécessaire pour les algorithmes de chiffrement par blocs ou pour aligner le texte chiffré sur une limite de 4 octets.

Les données d'authentification ne sont présentes que si Ce service a été sélectionné. Voyons maintenant comment est appliquée la confidentialité dans Esp. L'expéditeur :

- ✓ Encapsule, dans le champ "charge utile" de ESP, les données transportées par le datagramme original et éventuellement (mode tunnel).
- ✓ Ajoute si nécessaire un bourrage.
- ✓ Chiffre le résultat (données, bourrage, champs longueur et en-tête suivant).
- ✓ Ajoute éventuellement des données de synchronisation cryptographiques (vecteur d'initialisation) au début du champ "charge utile".

III.3.1.5 - La gestion des clefs pour IPSEC : ISAKMP et IKE

Les protocoles sécurisés présentés dans les paragraphes précédents ont recours à des algorithmes cryptographiques et ont donc besoin de clefs. Un des problèmes fondamentaux d'utilisation de la cryptographie est la gestion de ces clefs. Le terme "gestion" recouvre la génération, la distribution, le stockage et la suppression des clefs.

IKE (Internet Key Exchange) est un système développé spécifiquement pour IPSEC qui vise à fournir des mécanismes d'authentification et d'échange de clef adaptés à l'ensemble des situations qui peuvent se présenter sur l'Internet. Il est composé de plusieurs éléments : le cadre générique ISAKMP et une partie des protocoles OAKLEY et SKEME. Lorsqu'il est utilisé pour IPSEC, IKE est de plus complété par un "domaine d'interprétation" pour IPSEC.

III.3.1.5.1 - ISAKMP (Internet Security Association and Key Management Protocol)

ISAKMP a pour rôle la négociation, l'établissement, la modification et la suppression des associations de sécurité et de leurs attributs. Il pose les bases permettant de construire divers protocoles de gestion des clefs (et plus généralement des associations de sécurité). Il comporte trois aspects principaux :

Il définit une façon de procéder, en deux étapes appelées phase 1 et phase 2 : dans la première, un certain nombre de paramètres de sécurité propres à Isakmp sont mis en place, afin d'établir entre les deux tiers un canal protégé ; dans un second temps, Ce canal est utilisé pour négocier les associations de sécurité pour les mécanismes de sécurité que l'on souhaite utiliser (AH et Esp par exemple).

Il définit des formats de messages, par l'intermédiaire de blocs ayant chacun un rôle précis et permettant de former des messages clairs.

Il présente un certain nombre d'échanges types, composés de tels messages, qui permettant des négociations présentant des propriétés différentes : protection ou non de l'identité, perfectforwardsecrecy. Isakmp est décrit dans la .

III.3.1.5.2 IKE (Internet Key Exchange)

IKE utilise ISAKMP pour construire un protocole pratique, il comprend quatre modes :

- ✓ Le mode principal (Main mode)
- ✓ Le mode agressif (Aggressive Mode)
- ✓ Le mode rapide (Quick Mode)
- ✓ Le mode nouveau groupe (New Groupe Mode)

Main Mode et Aggressive Mode sont utilisés durant la phase 1, Quick Mode est un échange de phase 2. New Group Mode est un peu à part : Ce n'est ni un échange de phase 1, ni un échange de phase 2, mais il ne peut avoir lieu qu'une fois qu'une SA Isakmp est établie ; il sert à se mettre d'accord sur un nouveau groupe pour de futurs échanges Diffie-Hellman.

Phase 1 : Main Mode et Aggressive Mode les attributs suivants sont utilisés par Ike et négociés durant la phase 1 : un algorithme de chiffrement, une fonction de hachage, une méthode d'authentification et un groupe pour Diffie-Hellman, trois clefs sont générées à l'issue de la phase 1, une pour le chiffrement, une pour l'authentification et une pour la dérivation d'autres clefs. Ces clefs dépendent des cookies, des aléas échangés et des valeurs publiques Diffie-Hellman ou du secret partagé préalable. Leur calcul fait intervenir la fonction de hachage choisie pour la SA ISAKMP et dépend du mode d'authentification choisi. Les formules exactes sont décrites dans la .

Phase 2 : Quick Mode les messages échangés durant la phase 2 sont protégés en authenticité et en confidentialité grâce aux éléments négociés durant la phase 1. L'authenticité des messages est assurée par l'ajout d'un bloc Hash après l'en-tête ISAKMP et la confidentialité est assurée par le chiffrement de l'ensemble des blocs du message.

Quick Mode est utilisé pour la négociation de SA pour des protocoles de sécurité donnés comme IPSEC. Chaque négociation aboutit en fait à deux SA, une dans chaque sens de la communication.

Plus précisément, les échanges composant Ce mode ont le rôle suivant :

Négocier un ensemble de paramètres IPSEC (paquets de SA) échanger des nombres aléatoires, utilisés pour générer une nouvelle clef qui dérive du secret généré en phase 1 avec le protocole Diffie-Hellman. De façon optionnelle, il est possible d'avoir recours à un nouvel échange Diffie-Hellman, afin d'accéder à la propriété de PerfectForwardSecrecy, qui n'est pas fournie si on se contente de générer une nouvelle clef à partir de l'ancienne et des aléas.

Optionnellement, identifier le trafic que Ce paquet de SA protégera, au moyen de sélecteurs (blocs optionnels IDI et IDR ; en leur absence, les adresses IP des interlocuteurs sont utilisées).

Les groupes : New Groupe Mode, le groupe à utiliser pour Diffie-Hellman peut être négocié, par le biais du bloc SA, soit au cours du Main Mode, soit ultérieurement par le biais du New Group Mode. Dans les deux cas, il existe deux façons de désigner le groupe à utiliser : donner la référence d'un groupe prédéfini : il en existe actuellement quatre, les quatre groupes OAKLEY (deux groupes MODP et deux groupes EC2N), donner les caractéristiques du groupe souhaité : type de groupe (MODP, ECP, EC2N), nombre premier ou polynôme irréductible, générateurs.

Phases et modes au final, le déroulement d'une négociation IKE suit le diagramme suivant :

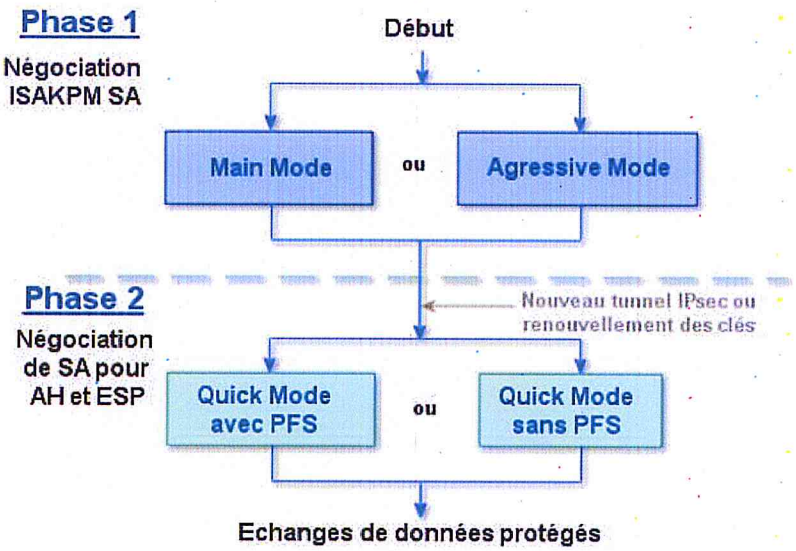


Figure 26.Déroulement d'une négociation IKE

III.3.1.6 - Les deux modes de fonctionnement d'IPSEC

Le mode transport prend un flux de niveau transport () et réalise les mécanismes de signature et de chiffrement puis transmet les données à la couche IP. Dans Ce mode, l'insertion de la couche IPSEC est transparente entre TCP et IP. TCP envoie ses données vers IPSEC comme il les enverrait vers IPv4. L'inconvénient de Ce mode réside dans le fait que l'en-tête extérieur est produit par la couche IP c'est-à-dire sans masquage d'adresse. De plus, le fait de terminer les traitements par la couche IP ne permet pas de garantir la non-utilisation des options IP potentiellement dangereuses. L'intérêt de Ce mode réside dans une relative facilité de mise en œuvre.

Dans le mode tunnel, les données envoyées par l'application traversent la pile de protocole jusqu'à la couche IP incluse, puis sont envoyées vers le module IPSEC. L'encapsulation IPSEC en mode tunnel permet le masquage d'adresses. Le mode tunnel est utilisé entre deux passerelles de sécurité (routeur, , ...) alors que le mode transport se situe entre deux hôtes.[9]

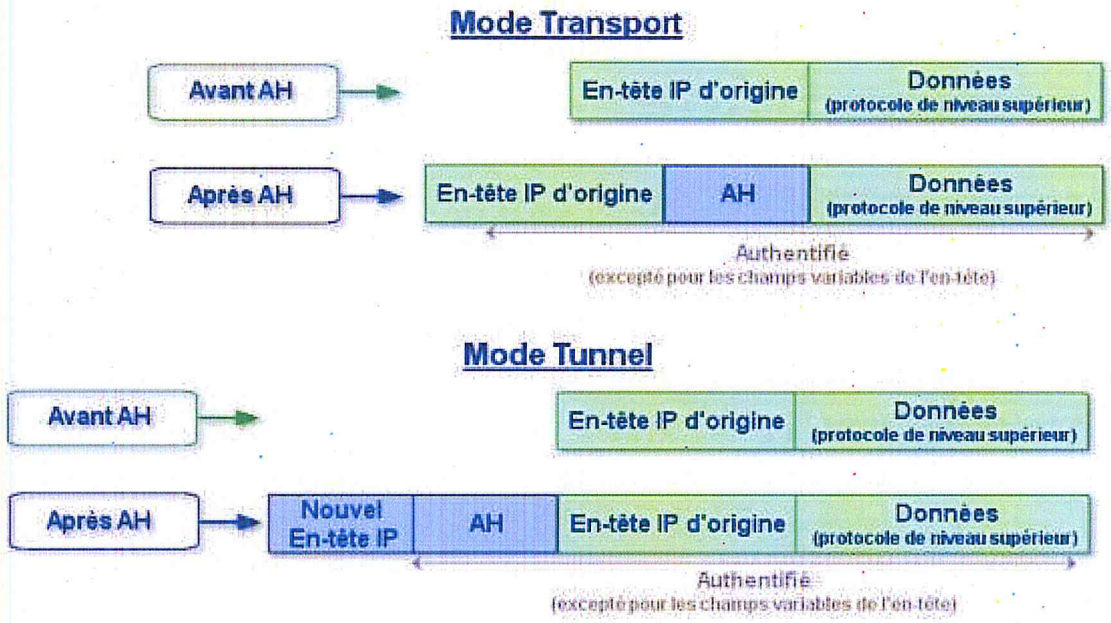


Figure 27.Encapsulation IPSEC

III.4. Conclusion

Cette étude met en évidence les différentes solutions disponibles pour l'établissement de VPN au service des utilisateurs nomades et des sites décentralisés d'une organisation. Le choix d'une solution dépend de beaucoup de paramètres à la fois stratégiques et techniques. Il dépend aussi de la capacité de l'opérateur télécoms ou l'ISP à fournir des services de VPN déchargeant l'organisation des frais de mise en œuvre et d'exploitation induits. Dans ce dernier cas le protocole MPLS est le plus approprié car déployé par la plupart des opérateurs et permet d'allier sécurité et qualité de service. Lorsque la confidentialité des données devient critique, les entreprises ont recours généralement à des solutions mettant en œuvre les techniques de cryptage et de signature numérique. IPSEC est le protocole indiqué pour les réseaux IP avec un ensemble de protocoles assurant l'intégrité (AH) et la confidentialité (ESP) Enfin le VPN-SSL est une solution qui assure les mêmes avantages d'IPSEC avec moins de contraintes, puisqu'il s'agit d'un protocole applicatif ne nécessitant aucune intervention au niveau des équipements réseau. Dans la suite de ce rapport nous présentons notre approche pour l'intégration de la sécurité dans un réseau SDN à travers des nœuds spécialisés mettant en œuvre le protocole IPSEC.

Chapitre IV: Conception

IV.1. Introduction :

Dans le monde de la virtualisation et du CLOUD Open Source, le logiciel OVS (Open vSwitch) représente la brique de base pour l'offre de réseaux virtuels associé aux différents types de services, en particulier l'IaaS. Il s'agit d'un système d'exploitation qui transforme une machine physique ou virtuelle en un Switch de niveau 3 avec toutes les fonctionnalités traditionnelles (gestion des VLAN, de VXLAN, Trunking, Tunelling, Stacking etc.)

Les principaux points forts de ce système sont: sa souplesse, sa performance, sa base de données qui permet de rendre les configurations persistantes et enfin sa compatibilité totale avec le protocole OpenFlow.

IV.1.1.Rappel des Objectifs :

Dans le cadre de ce projet, nous nous intéressons à l'utilisation du système OVS et du protocole OpenFlow afin de sécuriser le trafic réseau, en fonction de critères variés englobant et dépassant les VPNs classiques.

Notre contribution est d'étudier les différentes architectures de sécurisation de flux dans un SDN, basé sur OpenFlow (via OpenvSwitch) tels que les VLANs et les VPNs et de concevoir et de développer une application dédiée à la gestion de la sécurité des communications en réseau en utilisant le protocole OpenFlow et le système OVS.

L'utilisateur ou l'administrateur, aura la possibilité, via cette application de déterminer le ou les flux du trafic réseau à sécuriser à travers le plus grand nombre de critères possibles, de choisir les moyens de sécurisation et enfin d'appliquer et de superviser le trafic en question.

IV.2.Architecture générale du système

Notre contribution dans ce projet consiste à étudier, concevoir et implémenter un mini-contrôleur qui permet de déployer des circuits sécurisés utilisant diverses technologies (VLAN et VPNs) afin d'acheminer des flux de données sélectionnés à travers divers critères supportés par OpenFlow (adresses IP source et destination, numéros de ports source et destination etc.) ;

Nous présentons dans ce chapitre la conception de notre mini-contrôleur afin qu'il puisse permettre aux administrateurs de sélectionner des flux de données devant traverser des tunnels sécurisés via un ou plusieurs technologies combinés.

La figure 28 illustre l'architecture générale de notre système composée d'un ensemble de switch Openflow (deux ou plus) renforcée par des nœuds de traitement (les extrémités VPNs) et contrôlée par notre application dédiée à la gestion de tunnels sécurisés.

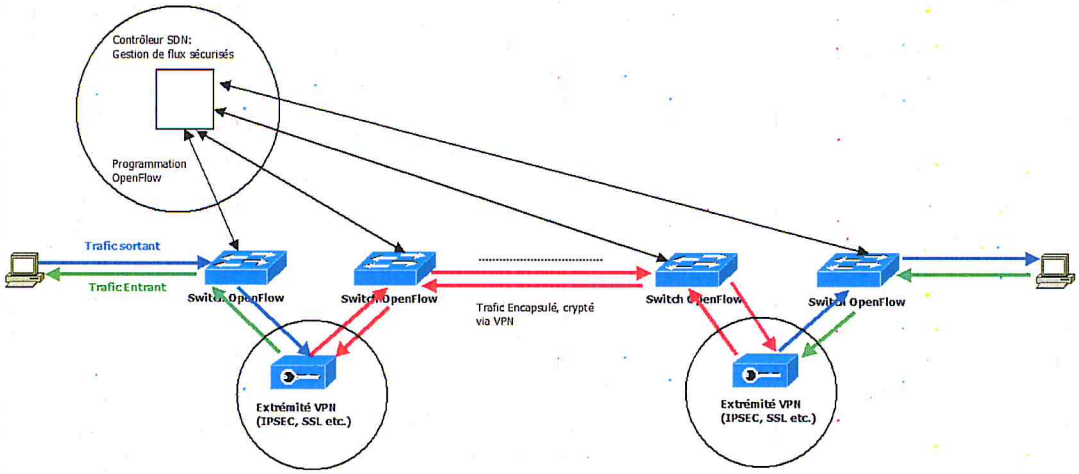


Figure 28.Architecture générale et fonctionnement du système

Cette application se positionne en tant que Contrôleur de flux utilisant le protocole Openflow afin de sécuriser des flux de données sélectionnés via la technologie des VLANs. Mais elle compte également sur l'existence de nœuds particuliers de traitement disposés dans le réseau SDN permettant de configurer des tunnels sécurisés via diverses technologies connus (principalement IPSEC et VPN-SSL) qui peuvent être combinés ou non à la technologie de base supportés par OpenFlow ,les VLANs.

IV.2.1. Communication entre contrôleur et OpenVswitch :

Openflow définit différents messages pour permettre la communication entre le switch et le contrôleur, y compris les messages d'établissement de connexion, les messages de configuration, les messages de statistiques de changement, les messages continus, les messages d'événements asynchrones, les messages d'erreur, les messages expérimentateurs et plus encore.

messages Openflow:

Une fois la connexion TCP effectuée, le message Openflow HELLO est échangé entre les deux entités pour négocier la version Openflow sur laquelle une autre communication aura lieu. Il est nécessaire que l'Openflow switch et le contrôleur puissent fonctionner sur la même version Openflow.

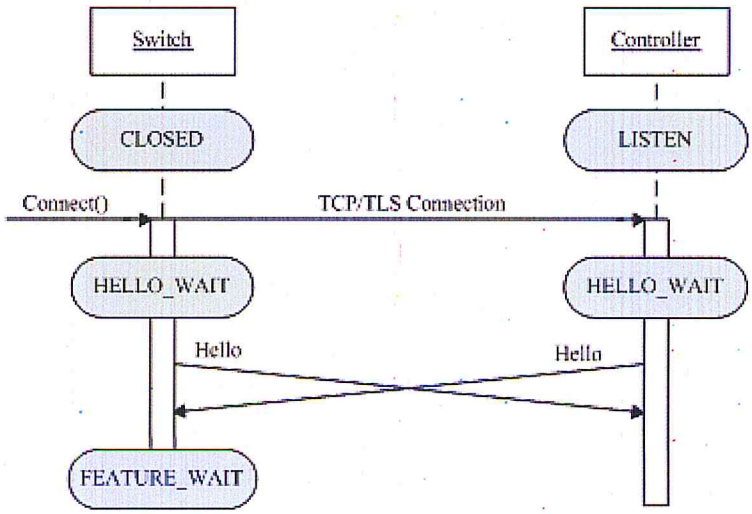


Figure 29.Etablissement de connexion Contrôleur-OpenVswitch

Une fois la négociation de la version terminée, le contrôleur envoi d'abord le message de demande de fonctionnalité Openflow pour obtenir principalement l'ID de transfert de données du message de réponse et pour déterminer le type de prise de fonctions compatible.

Pour configurer le switch Openflow, afin de gérer le trafic réseau, les messages Openflow comme les entrées de flux peuvent être envoyés par le contrôleur. Ceux-ci sont maintenus dans les tables de flux à l'intérieur des switchs.

Pour regrouper les entrées de flux, les groupes peuvent être configurés par le contrôleur via des messages de groupe qui peuvent être stockés dans des tables de groupe à l'intérieur des switchs.

Pour obtenir les détails des statistiques à partir du switch, les messages Openflow comme les statistiques de flux, les statistiques de port, les statistiques de file d'attente, les statistiques de groupe, les statistiques de table, etc. peuvent être envoyés par le contrôleur.

Jusqu'à maintenant, nous avons passé par l'architecture SDN, ses couches et le rôle d'Openflow pour réaliser le principe de base SDN pour séparer le plan de contrôle du plan de données. Maintenant, nous devons voir comment le contrôleur pourrait utiliser Openflow pour configurer et gérer le réseau sous-jacent.

Rappelons que le rôle du contrôleur consiste à injecter les règles de flux dans les tables de switchs Openflow en fonction desquelles ce dernier peut gérer le trafic réseau en les appliquant. Le message Openflow pour les entrées de flux comporte un large éventail de champs de tuple pour les critères de correspondance (L2, L3, L4, etc.) des paquets provenant du réseau, ce qui aiderait à configurer les règles ACL, les règles de politique de sécurité, les règles de limitation de débit QoS, Les règles de routage, les règles de mise en miroir des ports et les règles de modification des paquets.

Dans ce qui suit nous présentons les principaux scénarios possible avec notre système en fonction de la (ou les) technologies utilisée(s) afin d'établir des tunnels sécurisés pour des flux ciblés.

IV.2.2. Tunnels VLAN

C'est le moyen le plus simple pour déployer un tunnel sécurisé de niveau 2 en utilisant les commandes OpenFlow. En effet les switchs OpenFlow permettent de créer des VLANs à la volée, créant un réseau isolé traversant plusieurs switchs (suivant le même principe que la technologie du trunking dans les switchs traditionnels). Un trafic sécurisé par cette méthode se voit marqué (taggé) par un numéro de VLAN crée dynamiquement afin de pouvoir l'isoler des autres flux de données (figure 30). Cette technologie va permettre de protéger le trafic cible contre la capture de donnée au sein du réseau (Sniffing).

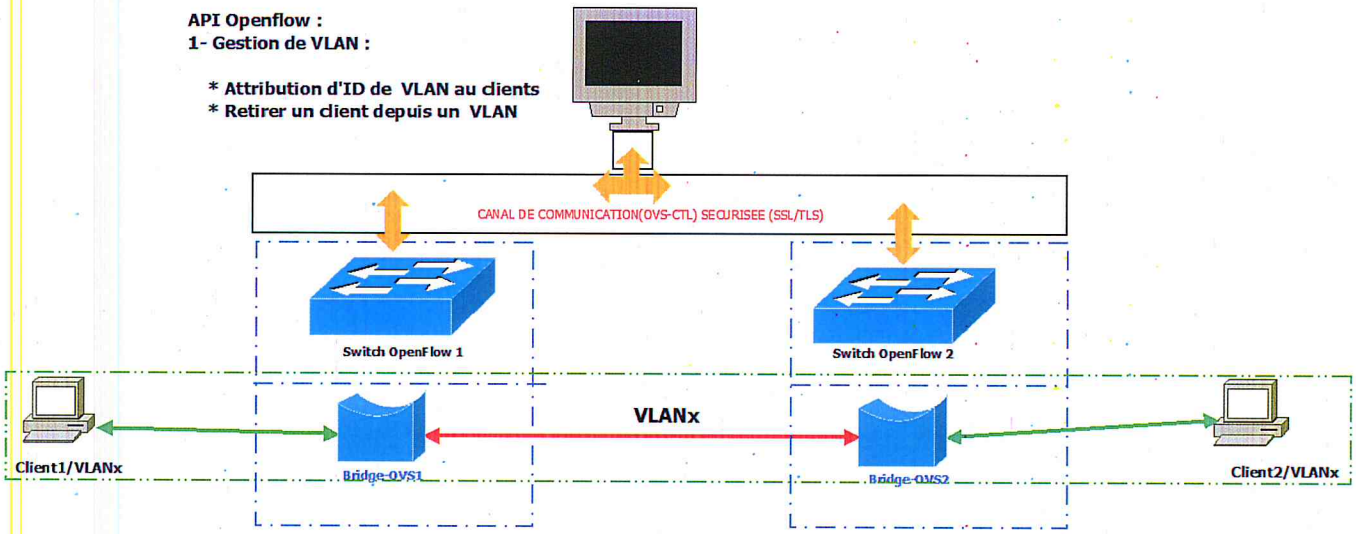


Figure 30.Architecture VLAN dans un SDN

Le contrôleur Attend l'arrivée des paquets, une fois le paquet arrive, avec un ensemble de paramètres de configuration : MAC(source , destination),IP(Source ,destination), vlan(ID , priorité), port(source, destination) et l' Action(ajouter au vlan, suppression du vlan), le contrôleur Charge les en-têtes des paquets dans la structure de données et effectue les tests de correspondance avec les entrées déjà introduit par l'administrateur, l'adresse IP source et l'Adresse IP de destination, Si 'Action' est ''ajouter au Vlan'' alors une entrée est créer avec attribution d'un identificateur de Vlan par une nouvelle valeur, cette valeur est soit créer au moment du traitement(valeur comprise entre 1 et 4096), soit c'est un identifiant d'un VLAN existant, le paquet est modifié est distribué à l'ensemble des switchs openflow.

La protection par l'utilisation de VLAN distincts par trafic, ne peut être efficace que sous la condition que tout le réseau soit sécurisé et n'est administré que par les administrateurs autorisés. En effet cette technologie peut être compromise lorsque, par exemple, les switch OpenFlow tombent sous l'emprise d'un hacker malintentionné qui pourra par la suite accéder à tout le trafic qui traverse les switchs. Afin d'augmenter le niveau de protection des données, il serait nécessaire de faire appel aux technologies utilisant la cryptographie comme moyen de sécuriser les données circulant dans le réseau. Nous avons choisi d'intégrer la technologie IPSEC pour atteindre cet objectif. Mais ceci n'exclut pas d'autres technologies comme VPN-SSL qui peuvent être intégrée de la même manière..

IV.2.3 Tunnels IPSEC

Afin d'intégrer IPSEC à notre environnement SDN, nous avons placé deux nœuds de contrôle aux extrémités de notre réseau, offrant un service de tunneling IPSEC. Les deux extrémités établissent entre elles un tunnel IPSEC offrant les fonctionnalités d'intégrité (sous-protocole IPSEC AH) et de cryptage (sous-protocole IPSEC-ESP) de sorte que le trafic qui passe entre les deux ne peut être

consulté ou modifié par une tierce partie au milieu. En utilisant des commandes OpenFlow particulières, l'utilisateur pourrait grâce à notre système, sélectionner un trafic particulier (à travers un masque OpenFlow adéquat) et instruire les switchs OpenFlow de le diriger vers une extrémité IPSEC qui va se charger de l'encapsuler dans le bon tunnel . À sa réception, l'autre extrémité décrypte le trafic et le redirige vers le switch de sortie qui va le délivrer à sa destination (figure 31).

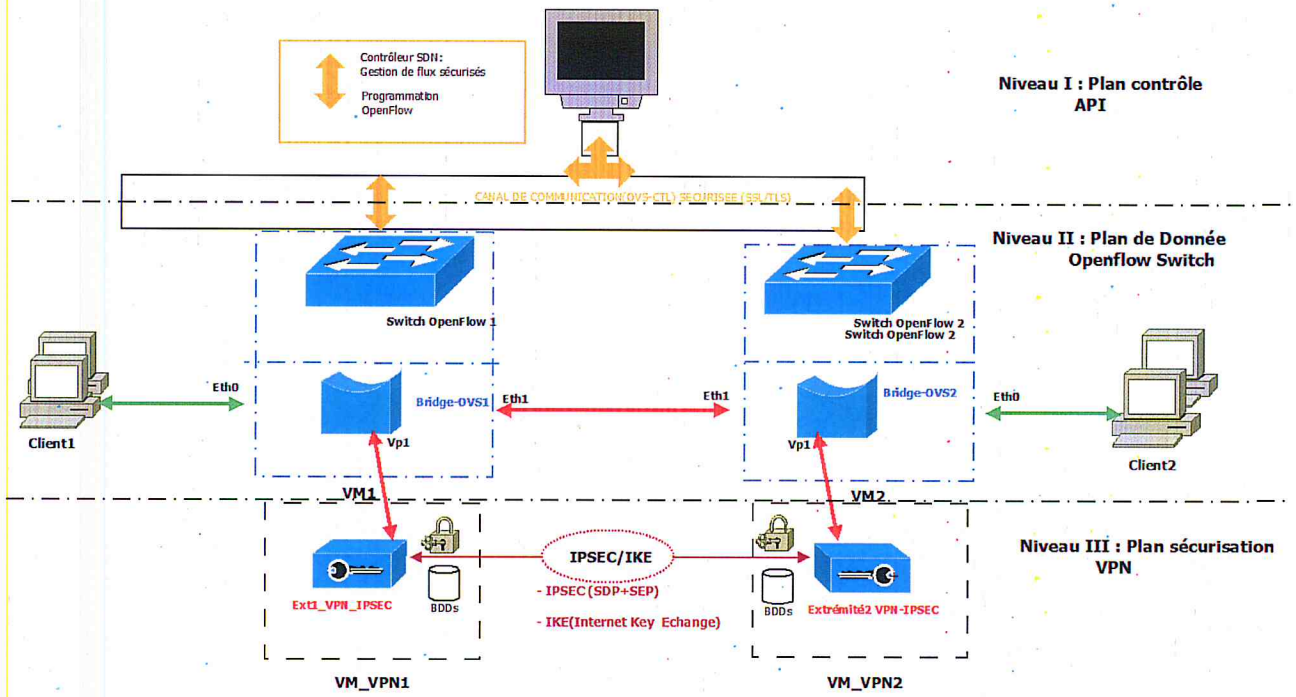


Figure 31.Architecture VPN/IPSEC dans un SDN

Notre architecture est composée de trois couches:

Couche I : Plan de contrôle : L'application développée par nos soins qui sert au traitement du flux Openflow et de gestion des tunnels de sécurisation.

Couche II : Plan de donnée : Les switchs Openflow formant l'infrastructure réseau d'un environnement SDN ou plus.

Couche III : Plan de sécurisation: c'est un ensemble de nœuds spécialisés dédiés à la mise en place de tunnels VPNs IPSEC. Concrètement il s'agit de deux machines situées aux extrémités du réseau.

Le Workflow des opérations qui permettent d'atteindre notre objectif : faire passer un trafic particulier par un tunnel IPSEC, peut être divisé en trois principales étapes :

- ✓ la configuration IPSEC entre les deux nœuds dédiés à cette tâche.
- ✓ La définition des critères de sélection du trafic à sécuriser (adresse MAC source/destination, adresse(s) IP source/destination et/ou port(s) source/destination, protocole de transport (UDP/TCP, etc.).

- ✓ L'élaboration et l'exécution des commandes OpenFlow par notre contrôleur et leurs redirection à l'ensemble des switchs OpenFlow.

Traitement de flux Openflow :

Dans le cas où le paquet est envoyé au contrôleur , Le contrôleur reçoit le paquet, Lecture des informations de l'entête (adresse IP source, adresse IP destination, adresse MAC source, adresse MAC destination, VLAN id, VLAN priority,…..), si les informations introduites par l'administrateur (les adresses IP source et destination par exemple) correspondent aux critères alors le contrôleur génère une nouvelle entrée(règle) qui définit l'adresse de destination correspondante au paquet provenant de l'entité qui demande une connexion, par l'adresse de l'extrimité du VPN-IPSEC, alors tous les paquets provenant de cette entité doivent passer par le tunnel IPSEC, ensuite envoyé cette règle dans un nouveau paquet (modifié) au switch OpenFlow.

Ensuite Générer une autre entrée correspondant au trafic devant être reçu de la destination qui permet de rediriger ce dernier vers l'autre extrémité VPN-IPSEC, par l'attribution d'une autre action qui définit l'adresse de destination correspondante au paquet provenant de l'autre extrimité VPN-IPSEC par l'adresse de l'entité destination.

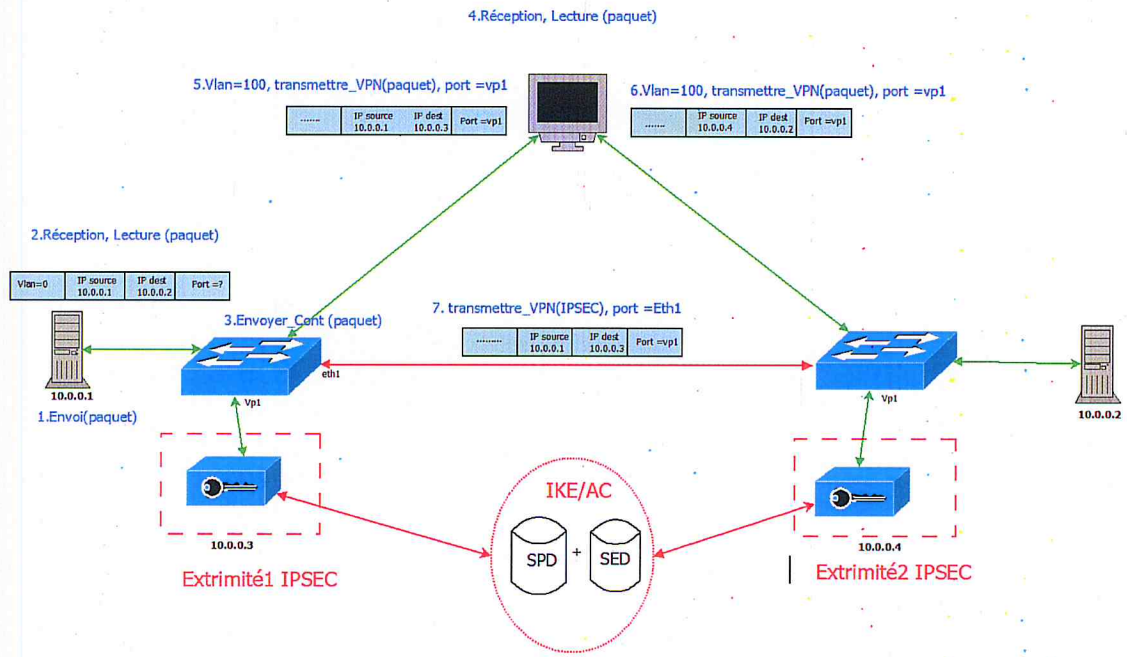


Figure 32.Exemple de déroulement du processus de tunnel par IPSEC

Finalement ces deux entrées définissent deux règles de redirection de trafic entre les deux partenaires de la communication à travers le tunnel IPSEC.

IV.2.4 Tunnel combinant IPSEC et VLAN

Afin d'augmenter encore plus le niveau de sécurité, le trafic IPSEC peut être isolé en utilisant la technique des VLANs expliquée en premier plus haut. Ceci est possible à travers la combinaison de commandes OpenFlow permettant de marquer le trafic sélectionné avec un VLAN distinct avec les commandes OpenFlow permettant de forcer ce même trafic à passer par le tunnel IPSEC établi entre les nodes IPSEC dédiés.

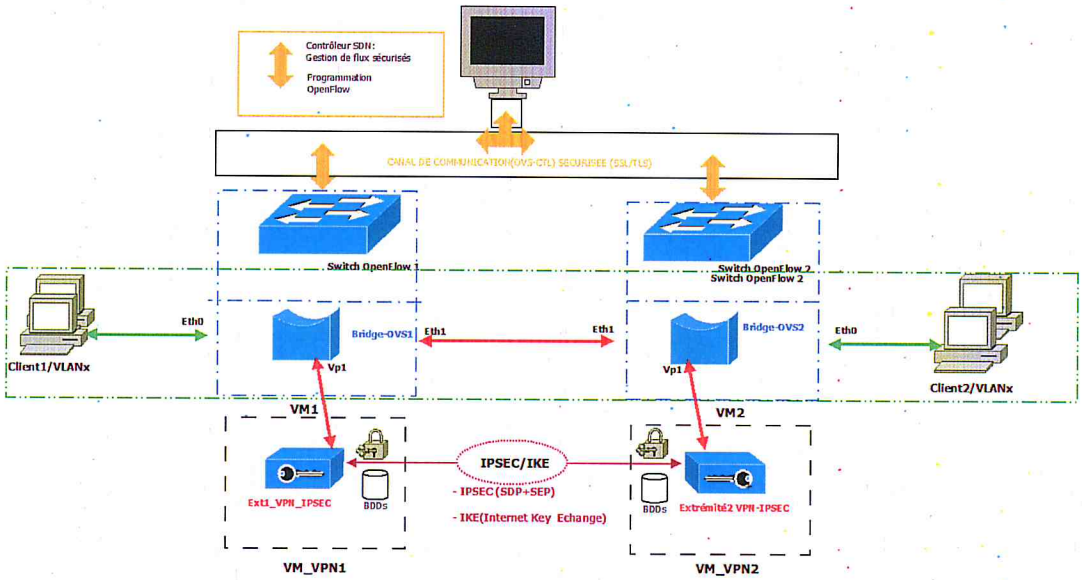


Figure 33.Architecture IPSEC+VLAN

IV.2.5 Tunnel VPN-SSL

Cette méthode est fondée sur le même principe sur lequel se fonde la méthode IPSEC expliqué plus haut, sauf qu'elle utilise la technique du VPN-SSL. À part les moyens utilisés pour établir ce genre de tunnels, le résultat est à peu près le même que celui obtenu avec IPSEC. L'intérêt d'utiliser cette technologie réside dans le fait qu'elle est beaucoup plus simple à installer et configurer et aussi elle est plus souple, en particulier lorsqu'il s'agit de traverser les passerelles NAT (SSL a déjà été présentée dans un chapitre dédié).

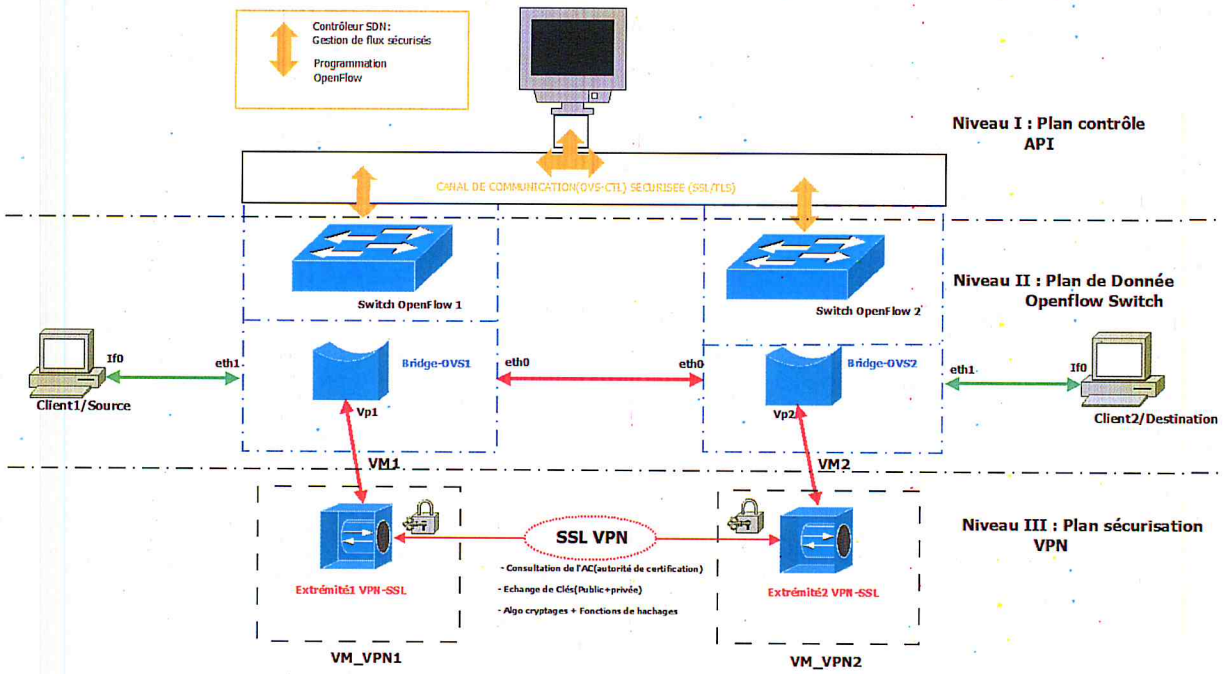


Figure 34.Architecture VPN/SSL dans un SDN

Notre système est composé de trois couches (niveaux) :

Couche I : Plan de contrôle : L'application développée par nos soins qui sert au traitement du flux Openflow et de gestion des tunnels de sécurisation.

Couche II : Plan de donnée : Les switchs Openflow formant l'infrastructure réseau d'un environnement SDN.

Couche III : Plan de sécurisation: c'est un ensemble de nœuds spécialisés dédiés à la mise en place de tunnels VPNs SSL.

Le Workflow des opérations qui permettent d'atteindre notre objectif : faire passer un trafic particulier par un tunnel SSL, peut être divisé en trois principales étapes :

la configuration SSL entre les deux nœuds dédiés à cette tâche.

La définition des critères de sélection du trafic à sécuriser (adresse MAC source/destination, adresse(s) IP source/destination et/ou port(s) source/destination, protocole de transport (UDP/TCP, etc.).

L'élaboration et l'exécution des commandes OpenFlow par notre contrôleur et leurs redirection à l'ensemble des switchs OpenFlow.

Elaboration de tunnel SSL :

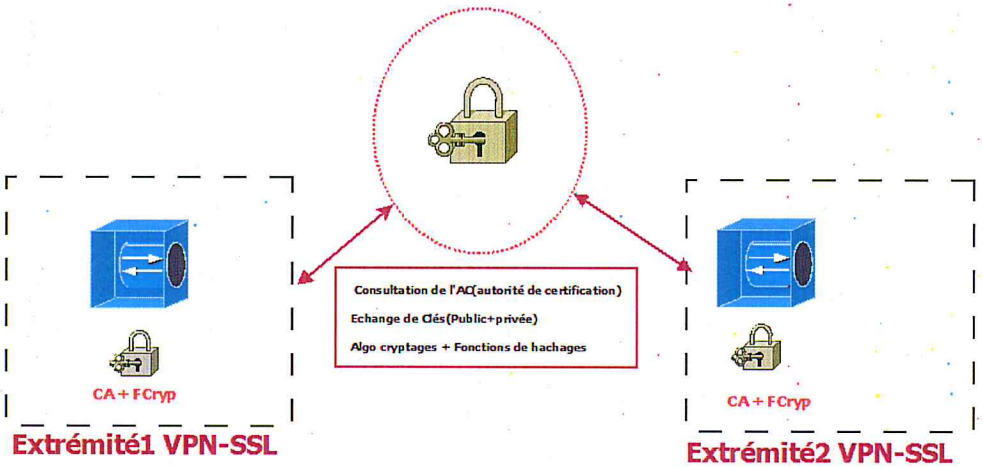


Figure 35.Elaboration de tunnel VPN/SSL

Dans cette partie et dis la réception du flux, le paquet est crypté à la première extrémité VPN-SSL et transmis vers la deuxième extrémité VPN-SSL, qui a son tour décrypte le paquet et l'envoi vers la destination.

La première extrémité VPN-SSL reçoit un paquet provenant du premier Openflow switch, ce paquet est crypté (SSL) et transféré à la deuxième extrémité VPN-SSL ;

La deuxième extrémitéVPN-SSL reçoit un paquet provenant de la première extrémité VPN-SSL, le décrypte (SSL) et l'envoi au deuxième Openflow switch, qui a son tour le redirige vers sa destination finale;

Traitement de flux Openflow :

Comme déjà expliqué pour le traitement de tunnel IPSEC, le principe reste le même, Dans le cas où le paquet est envoyé au contrôleur, Le contrôleur reçoit le paquet, Lecture des informations de l'entête il génère une nouvelle entrée qui définit la modification ou la redirection du flux qui doit passer de la source vers la destination, vers le tunnel VPN-SSL et transmettre cette règle dans un nouveau paquet (modifié) au switch OpenFlow.

Ensuite Générer une autre entrée correspondante au trafic devant être reçu de la destination qui permet de rediriger ce dernier vers l'autre extrémité VPN-SSL, par l'attribution d'une autre action qui définit l'adresse de destination correspondante au paquet provenant de l'autre extrémité VPN-SSL par l'adresse de destination.

IV.2.6 Tunnel combinant VPN-SSL et VLAN

Dans ces scénarios, la technologie des VLANs est combinée avec celle du VPN-SSL pour offrir une sécurité renforcée.

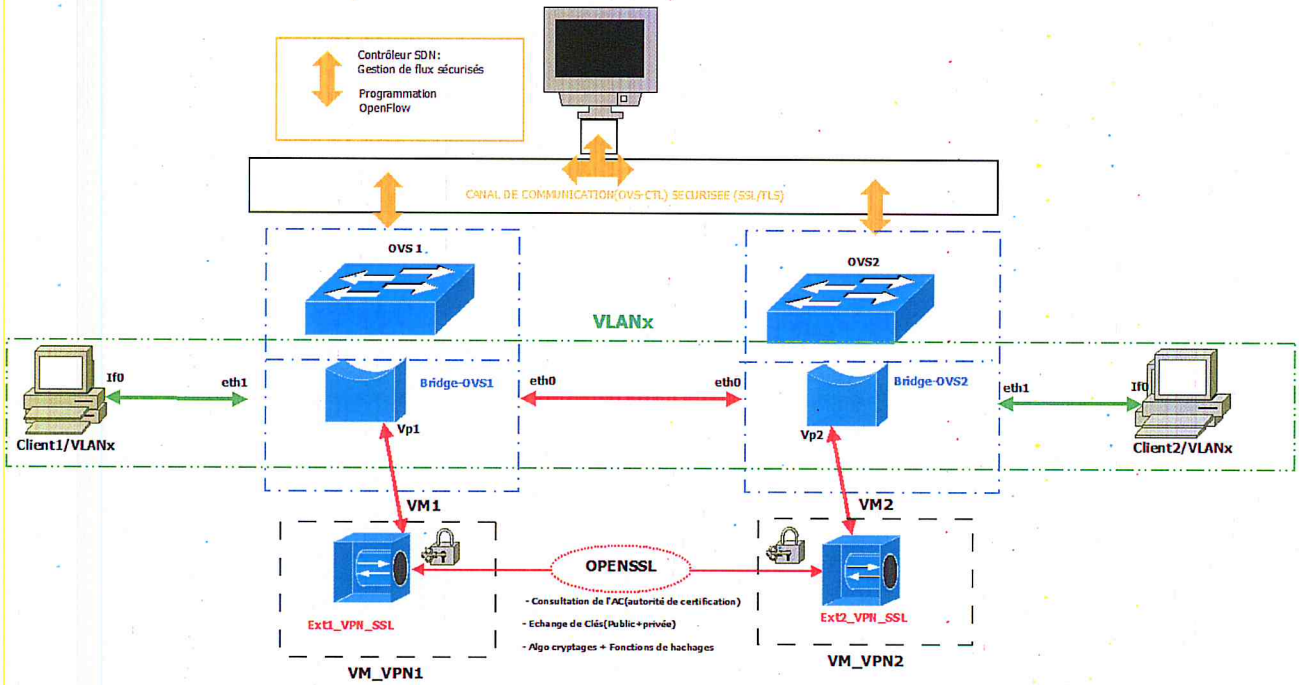


Figure 36.Architecture Mixte VLAN + VPN/SSL

L'action pourrait indiquer qu'un paquet provenant d'une source IP x et de destination IP y ne devrait pas appartenir à VLAN z. Ou l'Action peut indiquer qu'un paquet provenant d'une source IP x et une destination IP y devraient appartenir à VLAN z. Si le paquet arrive et qu'il n'a pas de correspondance, il passe tout simplement par le traitement normal d'OpenVSwitch.

OFPP NORMAL est une spécification OpenFlow du port OpenFlow prédéfinit. Cela signifie que le paquet sera envoyé par le port VP1(Bridge-OVS1) pour atteindre sa destination (Extrimité1 VPN/SSL), le paquet est crypté est envoyé vers l'Extrimité2 VPN/SSL, puis sera décrypté est envoyé vers OVS2 via le port VP2(Bridge-OVS2). Selon cette entrée, toutes les entrées OpenFlow sont écrites sur tous les OpenVSwitch du réseau et le paquet suivant correspondant à l'entrée n'ira plus au contrôleur.

VI.3. Conclusion :

Dans ce chapitre nous avons présenté la conception d'une architecture permettant d'appliquer des mesures de sécurité à des flux définis par l'administrateur. L'architecture SDN est augmentée avec des nœuds spécialisés et un mini-contrôleur dédié à cet objectif.

Deux types de mesures sont mis à contribution :

- Une fonctionnalité offerte nativement par le protocole et les switch OpenFlow : la segmentation en VLAN
- Une fonctionnalité externe à OpenFlow nécessitant l'intégration d'un tunnel sécurisé (IPSEC a été choisi, mais peut être étendu à VPN-SSL)

Dans le premier cas, le mini-contrôleur instruit les switchs OpenFlow d'affecter un numéro de VLAN différent, unique et distinguable au flux de données dont on souhaite sécuriser

Dans le second cas, le mini-contrôleur instruit les switches d'extrémité à rediriger le trafic ayant des caractéristiques définis, vers le bout d'un tunnel IPSEC préalablement établi.

Enfin les deux techniques peuvent être combinées pour accroître le niveau de sécurité.

CHAPITRE V: Implémentation

V.1.Introduction :

Notre contribution est d'implémenter une application qui jouera le rôle de contrôleur dans notre architecture SDN, afin de pouvoir donner à l'administrateur réseau la possibilité d'appliquer des configurations de virtualisation et segmentation de réseau pour permettre à deux entités voulant communiquer de passer en mode VPN(SSL/IPSEC) ou d'appartenir à un VLAN donné.

Afin d'aboutir à notre objectif, les étapes de réalisation de cet application de simulation d'un VPN dans environnement SDN se résument comme suit :

- ✓ La génération de l'environnement simule un réseau SDN, dans un environnement virtuel à base de VMware.
- ✓ La configuration d'un Tunnel IPSEC.
- ✓ L'application qui permet de configurer l'ensemble en fonction des besoins/objectifs des utilisateurs.

V.2.Description de l'environnement de développement et de tests

Nous avons choisi l'émulateur Mininet (émulateur SDN) qui offre les même possibilités de création d'un environnement SDN(Openvswitch, VMs)) avec des possibilités d'administration et de développements et de configuration plus avancées .. Notre environnement de test est composée de deux extrémités (hosts) nommées Client1 et Client2 connectées chacun à un switch Openflow (OVS1 et OVS2), ces derniers connectent également deux extrémités VPN(VPN1 et VPN2).

La figure 37 illustre cet environnement de travail.

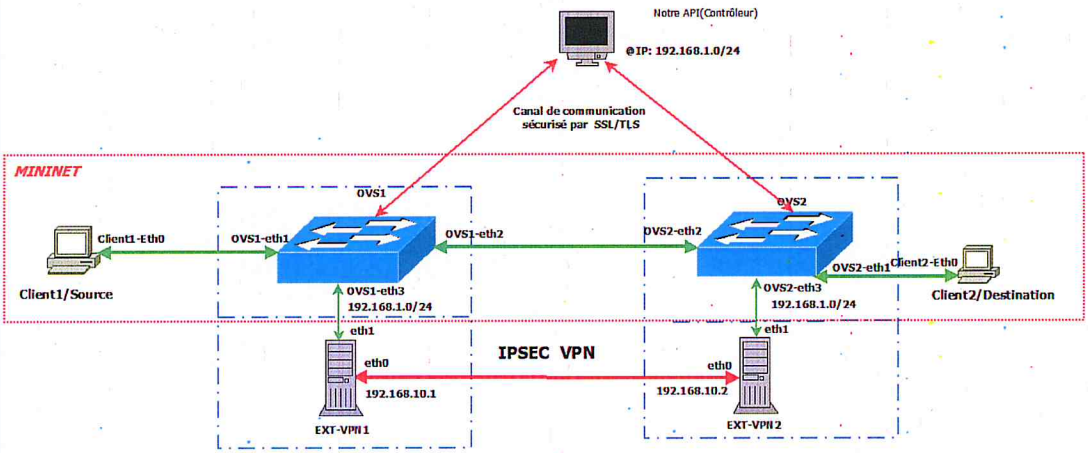


Figure 37.Architecture Proposée

Notons que les machihne virtuelles ont été crées à l'aide de Oracle-VirtualBox.

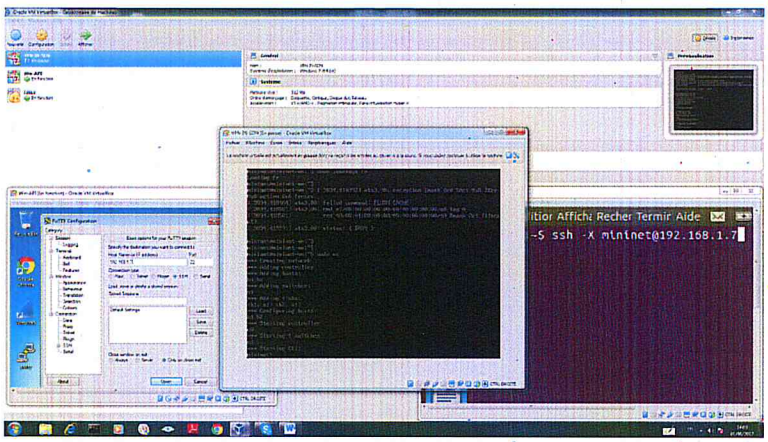


Figure 38.Lancement des VM

- ✓ Deux machines virtuelles : installées à base de Linux Ubuntu server : C'est les machines qui vont servir de tunnel VPN (VPN1 et VPN2).
- ✓ Deux machines virtuelles : installées à base de Linux Ubuntu :
  - La première : C'est la machine principale qui va héberger notre plateforme « SDN», qui va ensuite héberger l'émulateur MININET (Switchs Openflow Switch OVS et des hosts);
  - La deuxième : à base de système d'exploitation Linux Ubuntu: héberge l'application qui va administrer, sécuriser et visualiser un réseau SDN ;

V.2.Configuration :

V.2.1.Configuration du scénario Mininet :

Mininet est un émulateur de réseau, , il gère une collection d'hôtes d'extrémités, de switchs, de routeurs et de liens, le tout sur un seul noyau Linux. Il utilise une virtualisation légère pour créer un réseau complet dans un système unique , en utilisant le même noyau, le système et le code utilisateur. Un hôte Mininet émulé se comporte comme une véritable machine; Vous pouvez accéder (à travers un tunnel ssh) et exécuter des programmes arbitraires (y compris tout ce qui est installé sur le système Linux sous-jacent). Les programmes que vous exécutez peuvent envoyer des paquets à travers ce qui semble être une interface Ethernet réelle, Avec une vitesse et un retard de liaison donnés. Les paquets sont traités par ce qui ressemble à un véritable commutateur Ethernet ou routeur, Avec une quantité donnée de files d'attente. Iperf Lorsque deux programmes, comme un client et un serveur, communiquent via Mininet, est la performance mesurée correspond à celle de deux machines réelles (mais plus lentes).

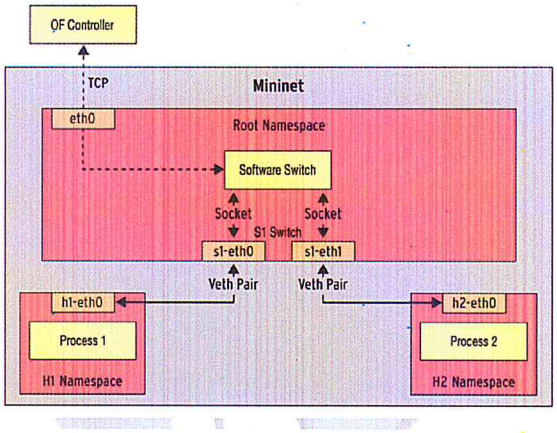


Figure 39.Architecture de Mininet

Notre plateforme SDN émulé (à base de Mininet) est générée à partir d'un script python dont les nœuds, les interfaces et les liens sont configurés automatiquement dans notre application.

La figure 40 illustre les résultats de la génération de la plateforme Mininet :

Génération de l'environnement SDN

Etablissement des liens entre nœuds (Switchs et Hosts et le tunnel VPN)

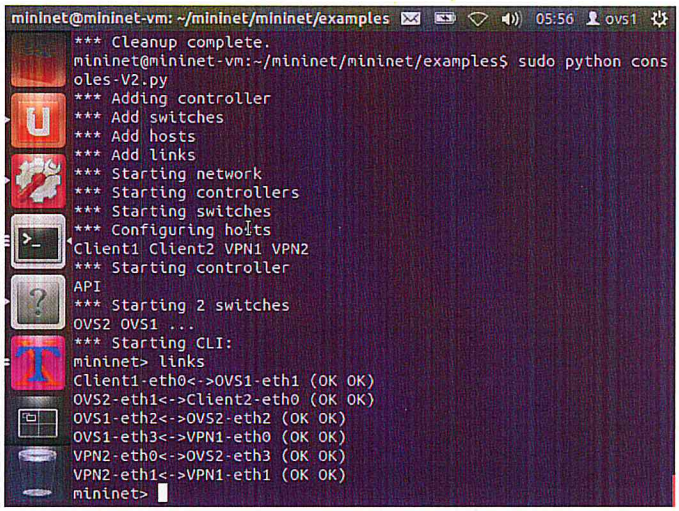


Figure 40.Résultats de la génération de la plateforme Mininet

V.2.2.Choix du langage de programmation :

Pour le développement des différentes parties de notre contribution, nous avons opté pour le langage Python. C'est un langage de script très puissant et populaire. bien documenté et supporté par une lartge communauté, parmi ces avantages citons :

- ✓ conçu pour produire du code de qualité, portable et facile à intégrer,
- ✓ de haut niveau, orienté objet et totalement libre,
- ✓ hautement productif,
- ✓ dynamique.
- ✓ la rapidité de développement. Un programme Python de 50 lignes peut représenter dans d'autres langages, des programmes de plusieurs centaines de lignes. Ce qui fait qu'en fin de

compte, même avec un programmeur Python pas assez rapide, on peut gagner beaucoup de temps au niveau du développement ;

- ✓ Python tourne sur presque toutes les plateformes ;
- ✓ Python est bien comme langage de script ;

V.3.Installation et configuration du tunnel VPN(IPsec):

L'implémentation d'un tunnel VPN(IPSEC) dans un environnement SDN, s'est réalisée comme suit :

- ✓ Configuration des deux extrémités VPN(VPN1 et VPN2)
- ✓ installation du noyau « Ipsec » sur les deux serveurs(VMs)
- ✓ Configuration du tunnel

La sécurité de la couche réseau est assurée en utilisant le protocole IPsec qui consiste à suivre deux composants.

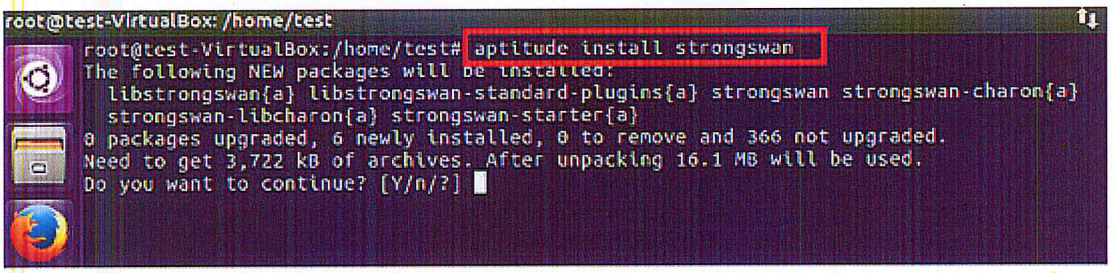
- ✓ En-tête d'authentification (AH)
- ✓ Encapsulating Security Payload (ESP)

L'intégrité des paquets et l'authentification sont assurées en utilisant AH, le composant ESP fournit des fonctionnalités de confidentialité. La mise en œuvre open source d'IPsec, StrongSwan(Strong Secure WAN), est un outil bien connu qui prend en charge les deux versions de l'échange de clés Internet (IKE v1 / 2) /. Le partage de clés ou l'échange de clés Internet fait partie du VPN IPSec (réseau privé virtuel). Le mécanisme IKE est utilisé pour partager la clé entre deux parties pour le cryptage des données dans le protocole ESP. Les algorithmes de cryptage et d'intégrité (tels que AES, SHA etc.) des bibliothèques OpenSSL et crypto sont utilisés pendant l'étape IKE. Cependant, l'implémentation du noyau Linux de l'algorithme de sécurité est utilisée dans la partie principale d'IPSec (ESP & AH).

V.3.1- Installation de StrongSwan

Les paquets binaires (deb / rpm) de strongswan sont disponibles dans presque toutes les distributions Linux largement utilisées. Le package binaire de strongswan peut être installé en utilisant la commande suivante sur Ubuntu 16.04 LTS.

Aptitude install strongswan



Après l'installation sur la plate-forme Ubuntu, les fichiers et dossiers de configuration (ipsec.conf, ipsec.secrets, ipsec.d, strongswan.conf, strongswan.d) sont stockés dans le répertoire / etc.

V.3.2- Configuration d'un tunnel basé sur une clé pré-partagée :

- Fichiers de configuration :

```
 nano 2.6.3                                File: ipsec.secrets

# This file holds shared secrets or RSA private keys for authentication.

# RSA private key for this host, authenticating it to any other host
# which knows the public part.
192.168.10.1  192.168.10.2 : PSK 'sharedsecret'
```

Fichier **ipsec.secrets** pour le partage du secret "Clé" entre les deux extrémités du VPN

Fichier **ipsec.conf** : c'est le fichier de configuration principal.

```
 nano 2.6.3                                File: ipsec.conf

# ipsec.conf - strongSwan IPsec configuration file

# basic configuration

config setup
        strictcrlpolicy=no
        uniqueids = yes
        charondebug="all"
# Add connections here.
conn %default
# Sample VPN connections

conn sample-self-signed
      leftsubnet=192.168.1.0/24
      right=192.168.1.2
      rightsubnet=192.168.10.0/24
      ike=aes256-sha2_256-modp1024!
      esp=aes256-sha2_256!
      keyingtries=0
      ikelifetime=1h
      lifetime=9999h
      keyexchange=ikev2
      type=tunnel
      auto=start

conn sample-with-ca-cert
      leftsubnet=192.168.1.0/24
      right=192.168.10.2
      rightsubnet=192.168.10.0/24
      rightid="C=CH, O=Linux strongSwan CN=peer name"
      ike=aes256-sha2_256-modp1024!
      esp=aes256-sha2_256!
```

La figure 41 suivante montre le placement du tunnel VPN basé sur IPsec dans un réseau. Un canal de communication sécurisé sera établi entre les réseaux privés A : 192.168.20.0/24 et B : 192.168.30.0/24 de la plateforme.

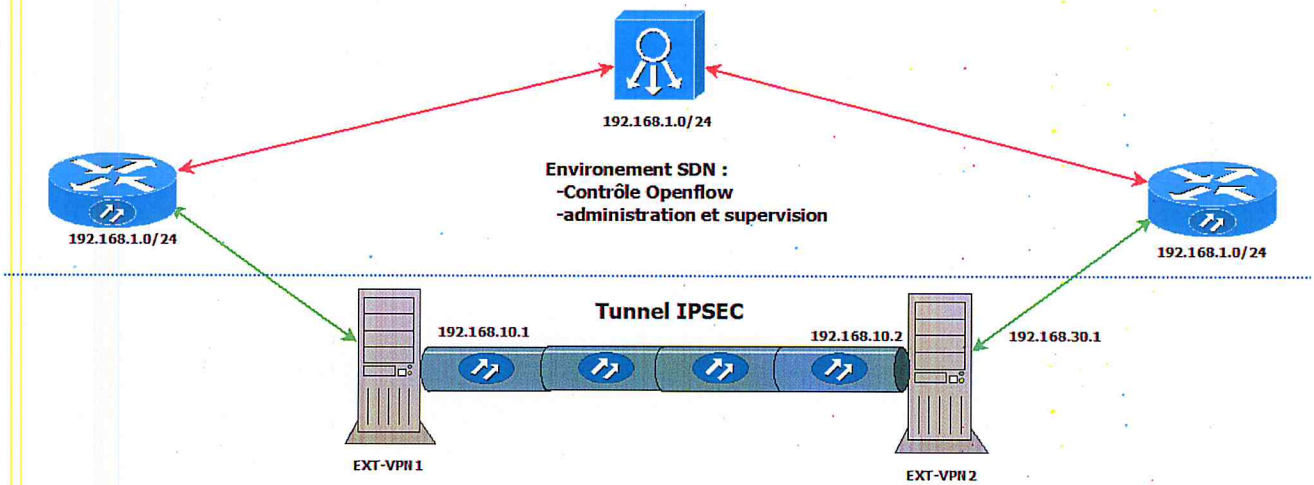


Figure 41.Tunnel IPSEC dans une Architecture SDN

Avant d'utiliser IPsec entre le réseau privé A et B, il faut s'assurer que le routage entre les extrémités du VPN fonctionne afin que les machines de du réseau A puisse communiquer avec les machines du réseau B, ce qui garantit que la connectivité réseau est correcte.

Démarrer le service IPSEC :

Démarrer le démon de IPSEC en utilisant la commande suivante après avoir configuré le fichier de configuration dans les deux côtés.

$ ipsec restart

La commande suivante montre l'état du VPN créé sur les périphériques.

$ ipsec statusall

```
Starting strongSwan 5.3.5 IPsec [starter]...
vpn1@vpn1:/etc$ sudo ipsec statefull
/usr/sbin/ipsec: unknown IPsec command `statefull' (`ipsec --help' for list)
vpn1@vpn1:/etc$ sudo ipsec statusall
Status of IKE charon daemon (strongSwan 5.3.5, Linux 4.8.0-22-generic, x86_64):
  uptime: 35 seconds, since Sep 16 12:22:19 2017
  malloc: sbrk 1462272, mmap 0, used 351728, free 1110544
  worker threads: 11 of 16 idle, 5/0/0/0 working, job queue: 0/0/0/0, scheduled: 2
  loaded plugins: charon test-vectors aes rc2 sha1 sha2 md4 md5 random nonce x509 revocation constra
ints pubkey pkcs1 pkcs7 pkcs8 pkcs12 pgp dnskey sshkey pem openssl fips-prf gmp agent xcbc hmac gcm
attr kernel-netlink resolve socket-default connmark stroke updown
Listening IP addresses:
  192.168.10.1
  192.168.20.4
  192.168.10.4
  192.168.20.2
  192.168.122.1
Connections:
sample-self-signed:  %any...192.168.1.2  IKEv2
sample-self-signed:   local:  uses public key authentication
sample-self-signed:   remote: [192.168.1.2] uses public key authentication
sample-self-signed:   child:  192.168.1.0/24 === 192.168.10.0/24 TUNNEL
sample-with-ca-cert:  %any...192.168.10.2  IKEv2
sample-with-ca-cert:   local:  uses public key authentication
sample-with-ca-cert:   remote: [C=CH, O=Linux strongSwan CN=peer name] uses public key authenticatio
n
sample-with-ca-cert:   child:  192.168.1.0/24 === 192.168.10.0/24 TUNNEL
Security Associations (0 up, 2 connecting):
sample-with-ca-cert[2]: CONNECTING, 192.168.10.1[%any]...192.168.10.2[%any]
sample-with-ca-cert[2]: IKEv2 SPIs: fe110bea7726039a_i* 0000000000000000_r
sample-with-ca-cert[2]: Tasks active: IKE_VENDOR IKE_INIT IKE_NATD IKE_CERT_PRE IKE_AUTH IKE_CERT_PO
ST IKE_CONFIG CHILD_CREATE IKE_AUTH_LIFETIME IKE_MOBIKE
sample-self-signed[1]: CONNECTING, 192.168.10.4[%any]...192.168.1.2[%any]
sample-self-signed[1]: IKEv2 SPIs: 7339b1adbb5c7c9a_i* 0000000000000000_r
sample-self-signed[1]: Tasks active: IKE_VENDOR IKE_INIT IKE_NATD IKE_CERT_PRE IKE_AUTH IKE_CERT_POS
T IKE_CONFIG CHILD_CREATE IKE_AUTH_LIFETIME IKE_MOBIKE
vpn1@vpn1:/etc$
```

V.3.3-Tunnel basé sur un certificat X.509

Dans le tunnel basé sur les certificats X.509 (authentification par clé publique), il est nécessaire d'utiliser des certificats générés par une autorité de certification (CA) compétente (local ou externe) pour les deux extrémités du tunnel IPSEC

Lorsque les certificats sont gérés en local, il est nécessaire d'avoir un certificat auto-signé dont la clé privée va être utilisé pour signer les certificats des partenaires de la communication.

Pour cela il est possible d'utiliser les commandes suivantes :

$cd /usr/local/etc/ipsec.d

$ipsec pki --gen --type rsa --size 4096 --outform pem > private/PUB1Key.pem

$ipsec pki --self --ca --lifetime 3650 --in private/strongswanKey.pem --type rsa --dn "C=CH, O=strongSwan, CN=Root CA" --outform pem > cacerts/PUBCert.pem

La génération des certificats pour le client A est indiquée ci-dessous.

$ipsec pki --gen --type rsa --size 2048 --outform pem > private/client1Key.pem

$chmod 600 private/client1Key.pem

$ipsec pki --pub --in private/client1Key.pem --type rsa | ipsec pki --issue --lifetime 730 --cacert cacerts/strongswanCert.pem --cakey private/strongswanKey.pem --dn "C=CH, O=strongSwan, CN=device1" --san device1 --flag serverAuth --flag ikeIntermediate --outform pem > certs/client1Cert.pem

Après une génération réussie des certificats du CA et des clients, la prochaine étape consiste à modifier la configuration ipsec.conf et ipsec.secrets.

Le contenu de ipsec.conf et de ipsec.secrets pour le côté A est donné ci-dessous.

```
config setup
    charondebug="all"
    uniqueids=yes
    strictcrlpolicy=no
conn %default
conn tunnel #
    left=192.168.20.2
    leftsubnet=192.168.20.0/24
    right=192.168.30.2
    rightsubnet=192.168.30.0/24
    ike=aes256-sha2_256-modp1024!
    esp=aes256-sha2_256!
    keyingtries=0
    ikelifetime=1h
    lifetime=8h
    dpddelay=30
    dpdtimeout=120
    dpdaction=restart
    #authby=secret
    auto=start
    keyexchange=ikev2
    type=tunnel
    leftcert=client1Cert.pem
    leftid="C=CH, O=strongSwan, CN=device1"
    rightid="C=CH, O=strongSwan, CN=device2"
```

Démarrer le démon de IPSEC en utilisant la commande suivante après avoir configuré le fichier de configuration dans les deux côtés.

$ipsec restart

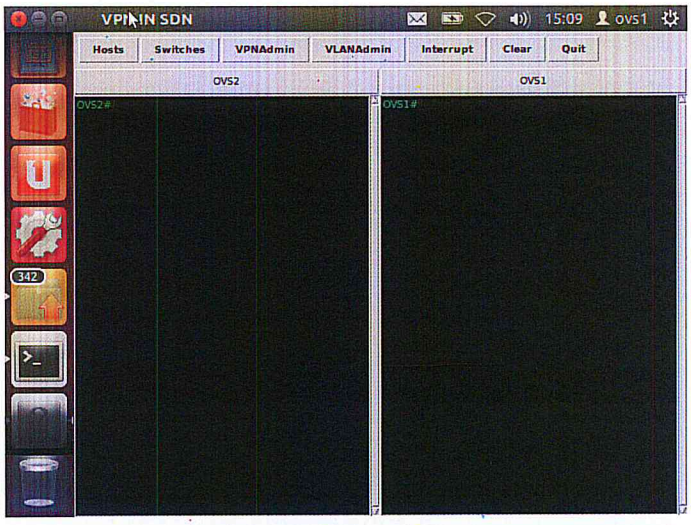
La commande suivante montre l'état du VPN créé sur les périphériques.

$ipsec statusall

V.4.Présentation de l'application

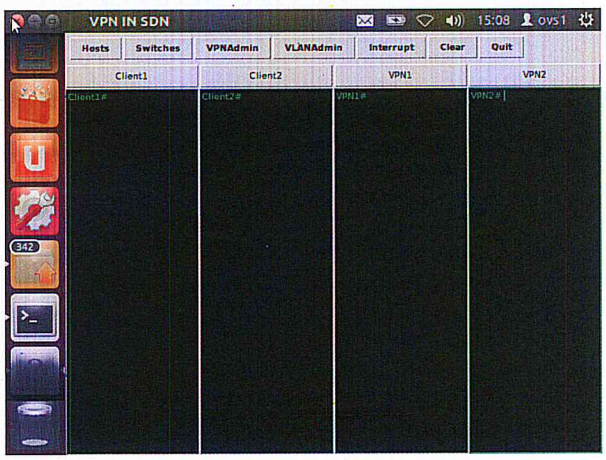
Après avoir préparé notre environnement de développement et de tests, nous présentons notre contribution dans ce projet, il s'agit d'une application jouant le rôle d'un mini-contrôleur pouvant être inséré dans un environnement SDN, elle sert comme moyen de configuration et de gestion du flux Openflow pour garantir un transfert de flux sécurisé par son acheminement vers un tunnel configuré auparavant.

L'administrateur fait introduire les paramètres nécessaires pour configurer le VPN(IPSEC), telle que l'adresse IP(source et destination).



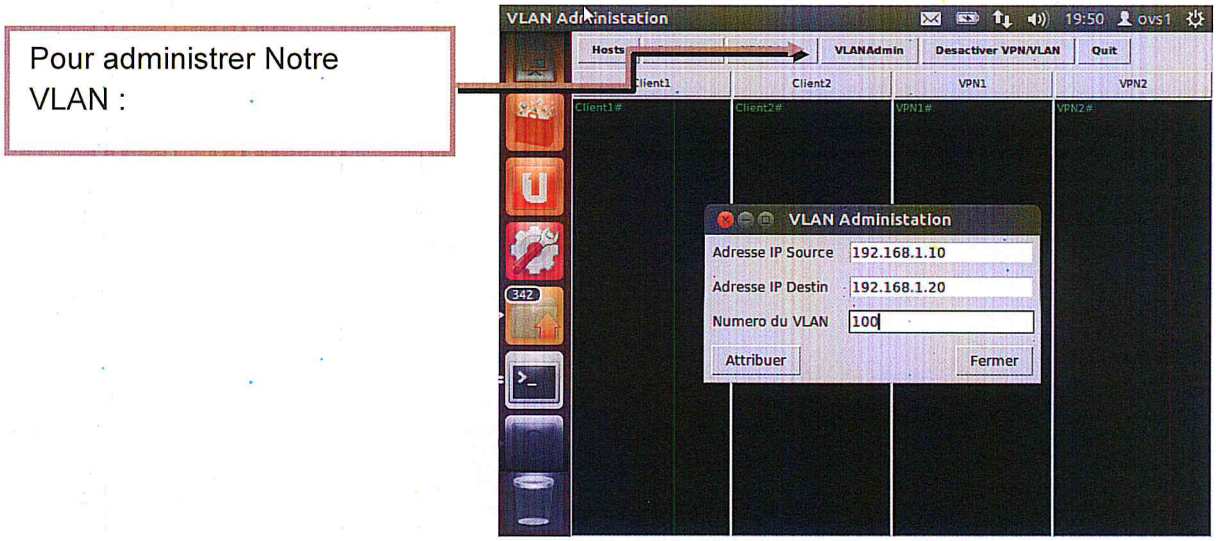
L'administrateur aura trois choix, soit d'administrer son VPN(IPSEC), soit d'administrer son VLAN soit de déployer une combinaison VPN/VLAN.

L'administrateur aura la possibilité aussi de visualiser les activités ou le comportement des Switchs/Clients et même des extrémités VPN, à l'aide de consoles et un accès, immédiat sur les switch (Openvswitch), sur les clients et sur les extrémités VPN, afin de confirmer des configurations.

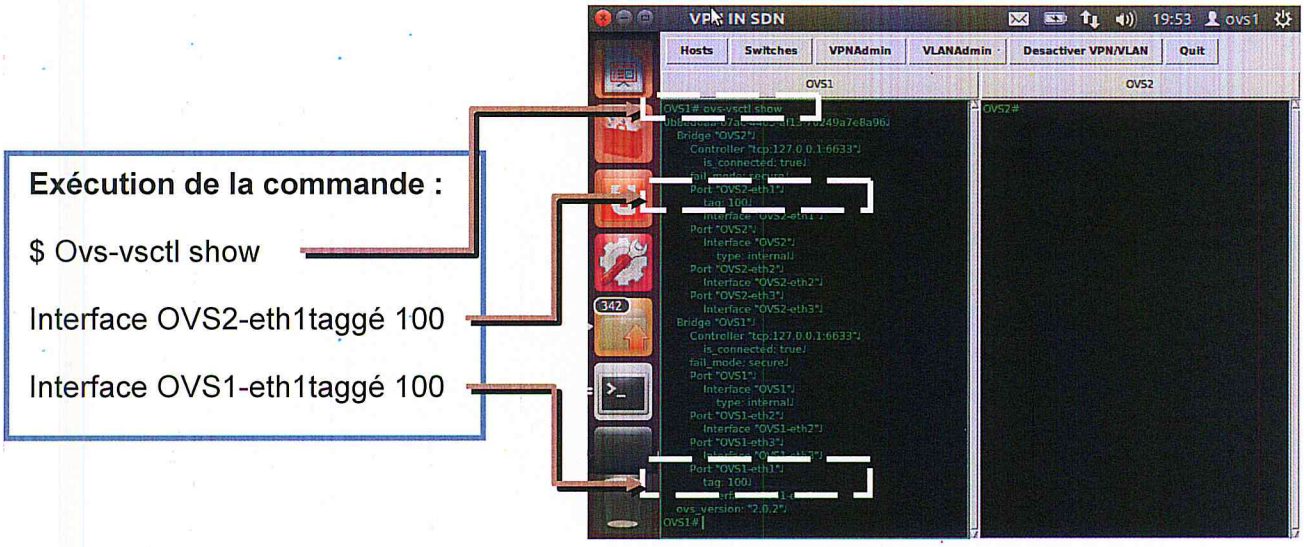


V.4.1.Configuration d'un Tunnel en mode VLAN :

Administration VLAN : L'administrateur renseigne les champs Adresse IP source et destination, ensuite il choisit soit d'activer ou de désactiver le VPN selon la situation demandée.



Voici le résultat après exécution d'une commande de visualisation du Switch OVS1 et OVS2 :



Voici un pseudo code des principales instructions exécutées par le Contrôleur.

Entrée: Packet, Adresse IP , vlanID, Action

Sortie: Action modifiée (Mod_Action) selon la demande, appliquée au Switch Openflow.

Étape 1: le contrôleur attend l'arrivée des paquets. Si le paquet arrive, passez à l'étape 2

Étape 2: Charger les en-têtes de paquets dans la structure de données «match»

Étape 3: si l'adresse IP source du réseau dans 'match' = 'Adresse IP'

Étape 3.1: si 'Action' est AddToVLAN

Étape 3.1.1: Créer une entrée 'FEntry' avec rules = match, actions = OFAction Modifiez Virtual LAN Indentifier à 'vlanID' + OF Output to portOFPP_NORMAL

Étape 3.2: si 'Action' est DeleteFromVAN

Étape 3.2.1: Si VLAN ID dans 'match' = 'vlanID'

Étape 3.2.1.1: Créez une entrée 'FEntry' avec rules = match, actions= OF action Virtual LAN Indentifier=0, OF Action Sortie vers le port OFPP_NORMAL

Étape 4: Else

Étape 4.1: Créez une entrée 'FEntry' avec rules = match, actions= OF Action Output to port OFPP_NORMAL

Étape 5: Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch

Étape 6: For all n Switch : Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch n

Étape 7: Passez à l'étape 1

V.4.2.Configuration d'un Tunnel en mode VPN-IPSEC :

Administration du VPN :

Nous utilisons une configuration pré-établie du tunnel IPSEC entre deux hôtes spécialisés, le lancement de quelques commandes sur le Switch OVS nous permet de visualiser les configurations actuelles du switch ainsi que l'état des tables de flux.

Exécution de la commande :

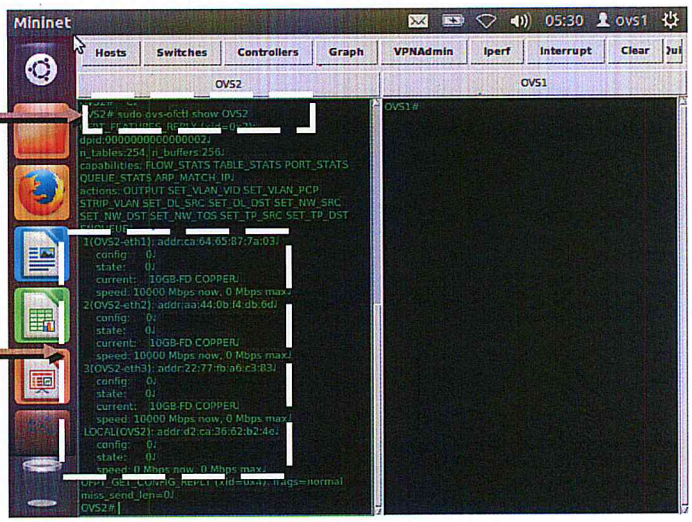
$Ovs-ofctl show OVS2

Résultats : la liste des interfaces(ports) sur le Switch OVS2:

-OVS2-eth1 le port :1

-OVS2-eth2 le port :2

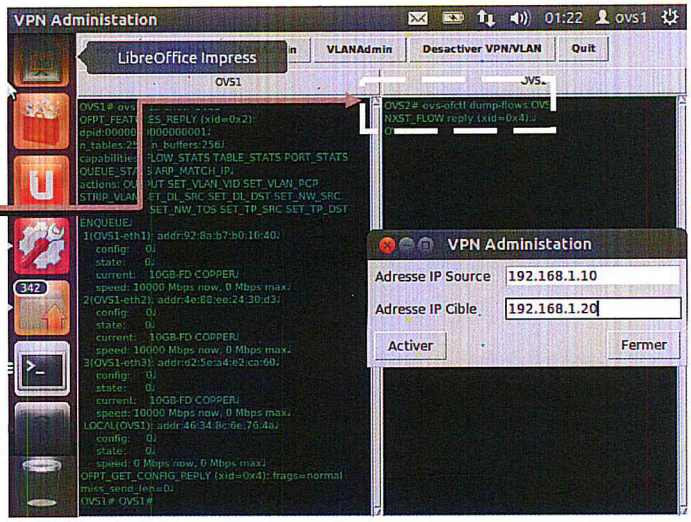
-OVS2-eth3 le port :3



Exécution de la commande :

$Ovs-ofctl show OVS1

Résultats : Pas de flux dans les tables du Switch Openflow OVS1

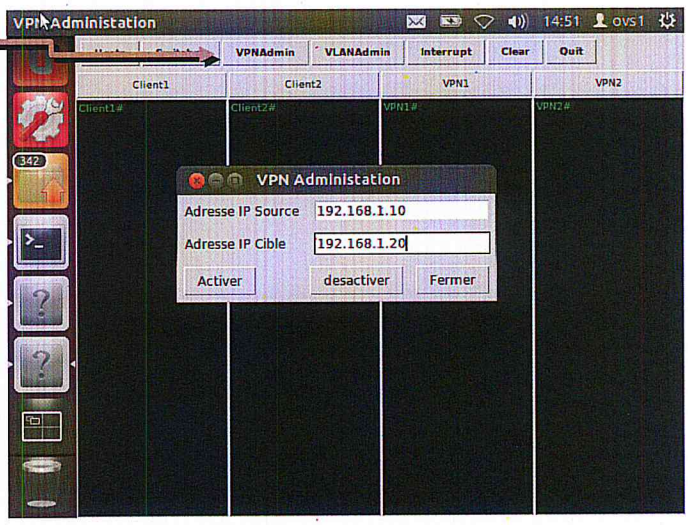


Lancement du module d'activation du VPN:

L'administrateur renseigne les champs Adresse IP source et destination, ensuite il choisit soit d'activer ou de désactiver le VPN selon la situation demandée.

Pour administrer Notre VPN :

Aller sur VPNAdmin

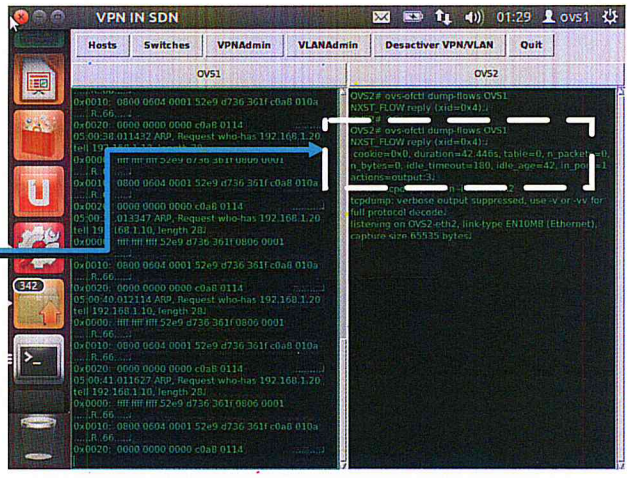


Après exécution de la commande d'activation de VPN entre Client 1 avec l'adresse ip 192.168.1.10 et le Client2 avec l'adresse ip 192.168.1.20 :

Exécution de la commande :

$sudo ovs-ofctl dump-flows OVS1

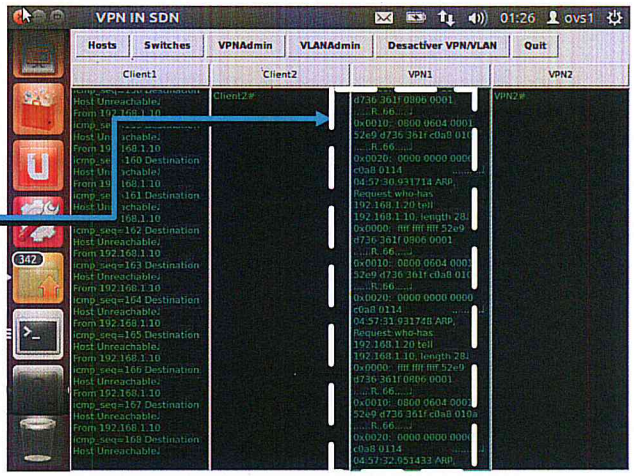
Résultats : présence de redirection vers le VPN dans les tables de flux du Switch Openflow OVS1, avec Action : redirection vers port 3(connecté au VPN).



– Exécution de la commande :

$sudo tcpdump –XX –n –i VPN1-eth0

Résultat : présence de flux dans le VPN



Voici un pseudo code des principales instructions exécutées au niveau du Contrôleur :

1) Activation du VPN

Entrée: Packet, Adresse IP source,Adresse IP destination, Action

Sortie: Action modifiée (Mod_Action) selon la demande, appliquée au Switch Openflow.

Étape 1: le contrôleur attend l'arrivée des paquets. Si le paquet arrive, passez à l'étape 2

Étape 2: Charger les en-têtes de paquets dans la structure de données «match»

Étape 3: si l'adresse IP source du réseau dans 'match' = 'Adresse IP source' et l'Adresse IP de destination du réseau dans 'match' = 'Adresse IP destination'

Étape 3.1: si 'Action' est Transférer vers Client2 à travers le Port VPN

Étape 3.1.1: Créer une entrée 'FEntry1' avec rules = match, actions = OF_Action1 transfert vers port-VPN1.

Étape 3.1.2: Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch1

Étape 3.1.3: Sortie du paquet via le port VPN1

Étape 3.1.4: Créer une entrée 'FEntry2' avec rules = match, actions = OF_Action2 transfert vers port-VPN2.

Étape 3.1.5: Insérez l'entrée 'FEntry2' dans la table OpenFlow de l'OpenVswitch2

Étape 3.1.6: Sortie du paquet via le port VPN2

Étape 3.2: Sinon, passez à l'étape 5.1

Étape 5: Else

Étape 5.1: Créez une entrée 'FEntry' avec rules = match, actions= OF Action Output to port OFPP_NORMAL

Étape 6: Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch2

Étape 7: Sortie du paquet via le port VP1

Étape 8: For all n Switch : Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch n

Étape 9: Passez à l'étape 1

2) Désactiver le VPN :

Étape 1: Créer une entrée 'FEntry' avec rules = match, actions = OF_Action transfert vers all_ports ;

Étape 2: For all n Switch : Insérez l'entrée 'FEntry' dans la table OpenFlow de l'OpenVswitch n.

V.4.3.Configuration d'un Tunnel en mode combiné VLAN et IPSEC :

Cette architecture comme le montre la figure 42 est une combinaison des deux dernière architecture traités au-dessus; l'architecture de base pour le déploiement de cette architecture est la même que celle utilisée pour déployé un VPN/IPSEC(section précédente) ; si un flux du Client1 se présente au niveau du Openvswitch(OVS1 ou OVS2) et si aucune table ne correspond au traitement de ce flux, il est transmis au contrôleur, qui a son tour vérifie ces paramètres , si ce flux demande un accès à notre extrémité Client2 alors il (flux)est redirigé ensuite à travers le déploiement d'un VLAN vers les Ports connectés au VPN(interfaces eth3 des deux Switchs qui correspond au ports N°3).

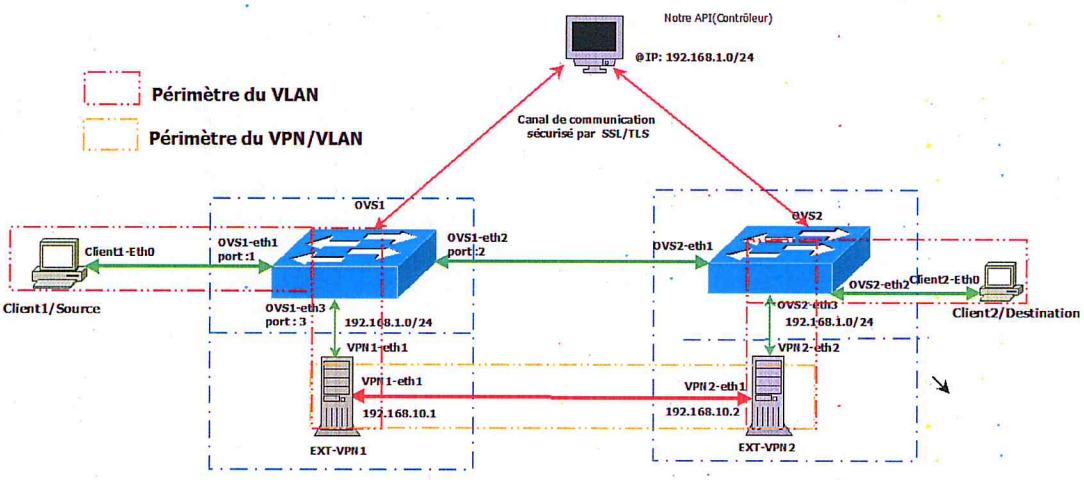


Figure 42.Un VPN/VLAN dans une Architecture SDN

CONCLUSION GENERALE

La technologie SDN devrait révolutionner à terme les architectures des réseaux des opérateurs, et permettre de déployer des nouveaux services de manière beaucoup plus rapide et avec des coûts significativement réduits, elle changerait complètement l'écosystème des infrastructures Télécoms dans les années à venir.

OpenFlow permet une programmation simplifiée via une interface standard pour les périphériques réseau, cette facilité de programmation permet de concevoir une couche de contrôle robuste, afin de centraliser l'intelligence dans le réseau et de fournir la programmabilité promise par SDN, bien que ce protocole soit fonctionnel, il reste néanmoins plusieurs anomalies à améliorer, surtout au niveau sécurité.

La perméabilité de la liaison entre le contrôleur et le switch, pourrait permettre permet à un attaquant ayant accès au réseau d'administration d'agir sur les le paramétrage du switch.

L'écoute passive des messages non chiffrés envoyés par le contrôleur permet à l'attaquant d'obtenir plein de renseignements sur le réseau qui peuvent compromettre la sécurité de celui-ci.

Notre projet a pour objectif de réaliser une implémentation pour assurer la gestion centralisée des réseaux en permettant aux administrateurs réseau d'avoir un outil simple pour la gestion du VPN, tout en utilisant l'architecture SDN. Cet objectif a été atteint via la mise en œuvre d'un plan de contrôle basé à la fois sur OpenFlow et sur une politique de cryptage IPsec.

En effet, nous sommes partis de la conception et de la mise sur pied d'une architecture réseau en respectant le principe SDN. Nous avons utilisé les protocoles OpenFlow standard et Mininet comme environnement d'émulation. Nous avons fait un déploiement du protocole OpenFlow standard à l'aide de trois éléments (activation automatique du protocole OpenFlow, création automatique des interfaces IPsec et démarrage automatique des composants VPN/IPsec via une architecture SDN) que nous avons conçus et implémentés. Afin de garantir un échange des données sécurisé, nous avons conçu et implémenté une passerelle permettant aux paquets OpenFlow une encapsulation et un transport via un tunnel IPsec et de pouvoir garantir une sécurité des flux transporté.

D'autres outils de virtualisation et de segmentation et de monitoring ont été testés et implémentés, tels que le déploiement des VLANs, afin de mieux renforcer cette politique de sécurisation.

Comme perspective à ce projet, il serait intéressant d'intégrer les fonctions de sécurité au sein même des switch OpenFlow et proposer une interface Sud permettant au contrôleurs d'activer dynamiquement des tunnels VPN (basés sur IPSEC ou d'autres technologies en option) et envoyer les instructions OpenFlow nécessaires à la redirection du trafic critique via ces tunnels.

Annexe 1 : REFERENCES BIBLIOGRAPHIQUES

[1] Brandon Heller, (OpenFlow Specification Version 1.1/1.2/1.3/1.4/1.5 (Protocol version 0x06) consulté le 02/04/2017 sur : https://www.opennetworking.org/wp-content/uploads/.../openflow-switch-v1.5.1.pdf

[2] Benjamin Villain (09 octobre 2015), New generation of network access controller: an SDN approach, consulté le 25/05/2017 sur : https://tel.archives-ouvertes.fr

[3] Steven Wallace, Uwe Dahlmann, Ron Milford, Chris Small; Openflow in a day, consulté le 16/06/2017 sur : https://www.nanog.org

[4] Feamster, N., Rexford, J., & Zegura, E. (2013, 09 30). The Road to SDN: An Intellectual History of Programmable Networks. Consulté le 18/06/2017, sur Princeton University web site: https://www.princeton.edu

[5] Martin Casado. SDN, VMware, and Software Defined Networking. Consulté le 29/06/2017, sur http://searchsdn.techtarget.com/news/2240183487/SDN-vs-network-virtualization-QA-with-VMwares-Martin-Casado

[6]IXIA, Black Book SDN/Openflow, Edition 10, juin 2014.Consulté le 28/05/2017, sur https://support.ixiacom.com

[7] Nick Feamster, Jennifer Rexford et Ellen Zegura The Road to SDN: An Intellectual History of Programmable Networks. 30December 2013, Consulté le 22/04/2017, sur https://support.ixiacom.com

[8] JeanTourrilhes, Puneet Sharma, SujataBanerjee OpenFlow: Enabling innovation in campus networks, Consulté le 22/06/2017, sur https://dl.acm.org/citation.cfm?id=1355746

[9] Brieuc Jeunhomme IPsec Consulté le 29/05/2017

[10] Doug Lowe, Networking All-in-One For Dummies 6th Edition,Consulté le 29/05/2017, sur www.wiley.com

[11] Linux Foundation. (s.d.). Consulté le 10/05/2017, sur Open Daylight

[12] Emmett Dulaney, Linux® All-in-One For Dummies®, 4th Edition

[13]VPN (Virtual Private Network) : définition, traduction et acteurs Fiche pratique Consulté le: 15/04/2017, sur: www.awt.be/web/

ANNEXE 2 : LES ABREVIATIONS ET ACRONYMES

Ce document utilise les abréviations et acronymes suivants:

3GPP : ThirdGenerationPartnership Project

ACID : Atomicity, consistency, isolation, durability

ACL : Access control list

A-CPI : Application-controller plane interface

AIS : Alarm indication signal

API: Applications programming interface

BFD : Bidirectionalforwardingdetection

BGP: Border gatewayprotocol

BIP : Bit interleavedparity

BSS: Business support system

C2C: Controller to controller

CCM : Continuity check message

CFM : Connectivityfault management

CPI : Controller plane interface

CRUD : Create, read, update, delete

DC: Data center

D-CPI: Data-controller plane interface

DDOS: Distributeddenial of service

DNS: Domain name system

DOS: Denial of service

DPCF: Data plane control function

EMS: Element management system

ETSI: EuropeanTelecommunications Standards Institute

GMPLS: Generalized multi-protocol label ing

GRE: Genericrouting encapsulation

HAL: Hardware abstraction layer

ICMP: Internet control messaging protocol

I-CPI: Intermediate-controller plane interface

IETF: Internet Engineering Task Force

IP: Internet protocol

IRTF: Internet ResearchTask Force

ISO: International Standards Organization

ITU-T: International Telecommunications Union – TelecommunicationStandardizationSector

LAN: Local area network

LLDP: Link layerdiscoveryprotocol

MAC: Media access control

MEF: Metro Ethernet Forum

MEP: Maintenance association end point, Maintenance entity group end point

MO: Managedobject

MPLS-TP: Multi-protocol label ing, transport profile

NAT: Network address translation

NBI: Northbound interface

NCD: Network control domain

NE: Network element

NFV: Network FunctionsVirtualization

NGMN: Next-generation mobile networks

NMS: Network management system

OAM: Operations, administration, maintenance

OFC: OpenFlow-Config protocol

OFS: OpenFlow- protocol

OIF: Optical Interworking Forum

ONF: Open Networking Foundation

OSS: Operations support system

OTN: Optical transport network

OVSDB: Open VSwitch data base

PCE: Path computation element

PCEP: Path computation element communication protocol

PEP: Policy enforcement point

PM: Performance monitoring

PON: Passive optical network

QoS: Quality of service

RDB: Resource data base

SDH: Synchronous digital hierarchy

SDN: Software-defined networking

SDNC: SDN controller

SDO: Standards developmentorganization

SLA: Service level agreement

SNMP: Simple network management protocol

STP: Spanningtreeprotocol

TCA: Thresholdcrossingalert

TL1: Transaction language 1

TMF: TM Forum

TOR: Top of rack

VID: VLAN identifier

VLAN: Virtual local area network

VM: Virtual machine

VN: Virtual network

VNE: Virtual network element

WAN: Wide area network

WG [ONF]: working group