# Le Protocole OpenFlow dans l'Architecture SDN (Software Defined Network)

**EFORT**
**http://www.efort.com**

## 1. Introduction

Dans le fonctionnement actuel des réseaux IP, chaque équipement de réseau (routeur IP, commutateur Ethernet, Firewall, load balancer, système de détection d'intrusion, NAT, etc.) exécute ses fonctions du plan de contrôle et du plan de données. SDN (Software Defined Network) propose de créer un point central qui gère le plan de contrôle, tandis que les commutateurs/routeurs physiques n'ont plus à prendre en charge que le plan de données. Pour ce faire, l'Open Networking Foundation (ONF) a publié OpenFlow. Il s' agit d'un protocole standard utilisé par le contrôleur pour transmettre au commutateur des instructions qui permettent de programmer leur plan de données et d'obtenir des informations de ces commutateurs 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'dresse, etc).
Le contrôleur SDN fournit une API aux « applications SDN » qui inclut 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 émules les « appliances » réseau.
Le but de ce tutoriel est d'introduire la genèse d'OpenFlow, décrire un commutateur OpenFlow ainsi que sa table de flux, présenter les messages OpenFlow et décrire des call flows associés.

## 2. La genèse d'OpenFlow

L 'histoire d 'OpenFlow est intéressante et permet de mieux comprendre son rôle fondamental dans la conception de l 'architecture SDN (Software Defined Network) et la virtualisation des fonctions réseau. OpenFlow a été initié comme un projet à l 'université de Stanford lorsqu 'un groupe de chercheurs exploraient la manière de tester de nouveaux protocoles dans le monde IP (créer un réseau expérimental confondu avec le réseau de production) mais sans arrêter le trafic du réseau de production lors des tests.
C 'est dans cet environnement que les chercheurs à Stanford ont trouvé un moyen de séparer le trafic de recherche du trafic du réseau de production qui utilisent le même réseau IP. Ils ont découvert que bien que les constructeurs de matériel réseau concevaient leurs produits différemment, tous utilisaient des tables de flux (flow table) afin d 'implanter les services réseau tels que les NATs, la QoS, les firewalls, les systèmes de deep packet inspection, etc. Par ailleurs, bien que l 'implantation des tables de flux différait entre ces constructeurs, les chercheurs ont découvert qu 'ils pouvaient exploiter un ensemble de fonctions communes.
Le résultat de l 'équipe de recherche à Stanford a été OpenFlow, qui fournit un protocole ouvert (open protocol) qui permet aux administrateurs de réseau de programmer les tables de flux (flow tables) dans leurs différents routeurs IP et commutateurs Ethernet dans un but (dans le cas de Stanford) de séparer le trafic de recherche du trafic de production, chacun avec son ensemble de fonctionnalités et caractéristiques de flux.
La figure 1 présente l'architecture SDN et la place d'Openflow.

- **La couche infrastructure** contient les éléments de réseau responsable de l'acheminement du trafic et qui supportent le protocole OpenFlow qu'ils partagent avec le contrôleur.
- **La couche de contrôle** est logicielle, basée sur le contrôleur SDN qui offre une visibilité globale du réseau et des équipements d'infrastructure.
- **La couche applicative** apporte l'automatisation des applications au travers du réseau à l'aide d'interfaces programmables ouvertes.

 

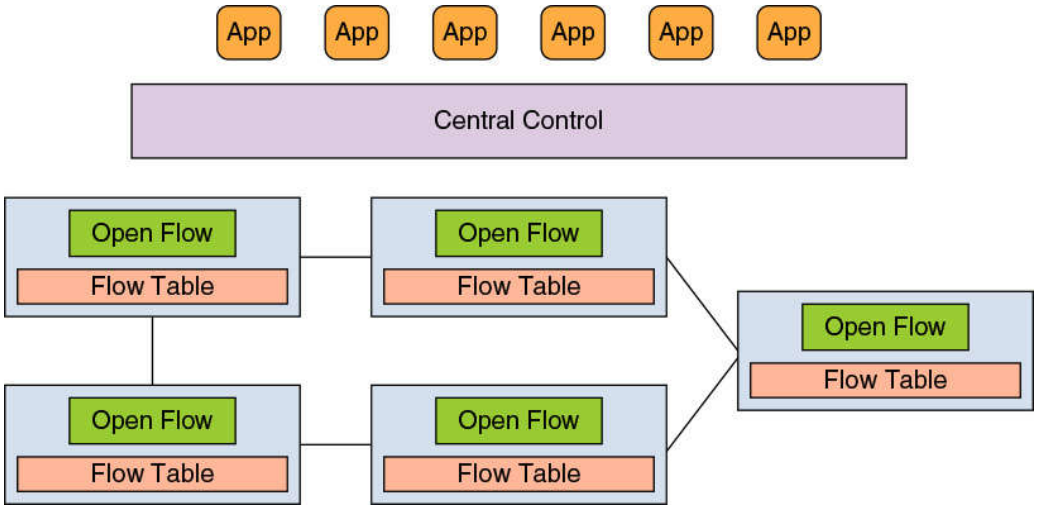


Figure 1 : Architecture SDN

## 3. Structure d'un commutateur OpenFlow

Les commutateurs 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, indiquées dans les en-têtes de paquets. Bien que chaque fournisseur ait des tables de flux différentes, il existe un ensemble commun de fonctions pour une large gamme de commutateurs et de routeurs. Cet ensemble commun de fonctions est mis à profit par OpenFlow, protocole entre un contrôleur central OpenFlow et un commutateur OpenFlow et qui, comme indiqué, peut être utilisé pour programmer la logique de transfert ou d'acheminement du commutateur.
La figure ci-dessus décrit les fonctions réalisées par les équipements de réseau du plan de données (data plane) aussi appelés commutateurs au sens générique. Les principales fonctions des commutateurs sont les suivantes :

- Fonction de support du contrôle (Control support function) : Interagit avec la couche contrôle SDN afin de supporter la programmabilité via les interfaces ressource-contrôle. Le commutateur communique avec le contrôleur et le contrôleur gère le commutateur avec le protocole OpenFlow. OpenFlow 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 et redescendues au commutateur..

Ces règles d'acheminement des données (data forarding rules) sont présentes dans les tables d'acheminement (forwarding tables). Ces règles indiquent pour des catégories de paquet données quel doit être le prochain saut sur la route. Le commutateur peut par ailleurs modifier l'en-tête du paquet avant son acheminement, ou rejeter le paquet. Comme montré à la figure 2, les paquets arrivant sont placés dans une file d'attente en entrée, attendant leur traitement par le commutateur; les paquets acheminés sont placés dans une file d'attente en sortie, avant d'être transmis.
Le commutateur à la figure 2 dispose de trois ports d'entrée/sortie : Un port fournissant la communication de contrôle avec un contrôleur SDN, et deux autres ports pour les entrées et sorties de paquets de données. Il s'agit d 'un exemple simple. Le commutateur peut disposer de plusieurs ports pour communiquer avec plusieurs contrôleurs SDN, et de plus de 2 ports pour les paquets de données entrant et sortant du commutateur.

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 'en-tête de protocole de couche supérieure, tel que TCP, UDP, SCTP ou protocole d 'application. Le commutateur 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..

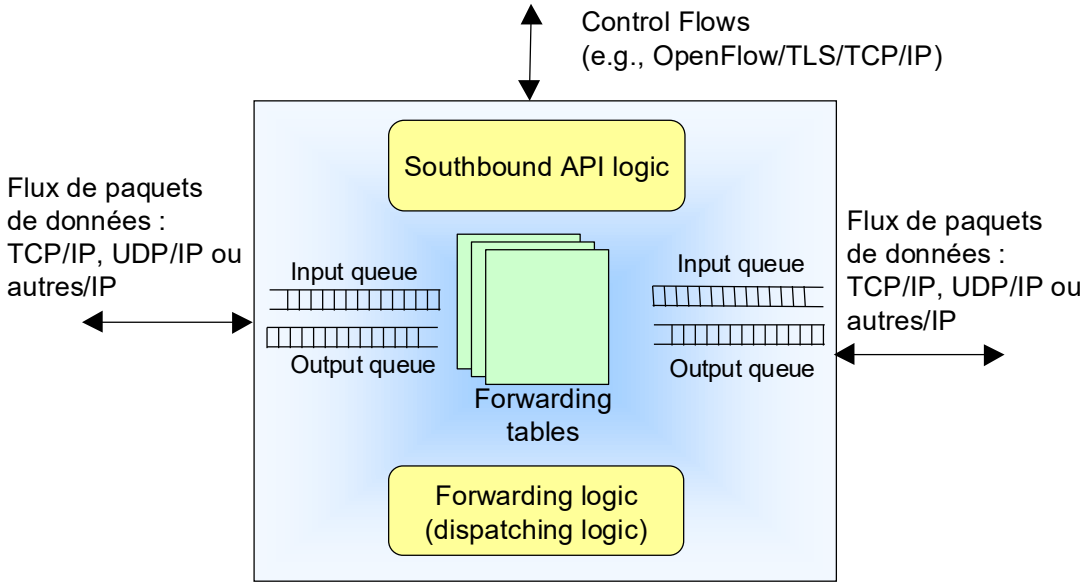

Figure 2 : Commutateur OpenFlow

La figure 3 décrit les fonctions du commutateur OpenFlow V1.0 et sa relation au contrôleur. Comme attendu dans un commutateur de paquet, la fonction de base est de recevoir les paquets qui arrivent sur un port (Chemin X sur port 2 sur la figure 3) et les relayer via un autre port (Port N sur la figure 3) en réalisant toute modification nécessaire sur les paquets sur le chemin. La fonction de correspondance de paquet (packet-matching function) est très importante dans le commutateur 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.
Cette 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 paquest 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. Si le contrôleur a soit un message de contrôle (e.g. mettre hors service un port du commutateur) ou un paquet de données à fournir au commutateur, le contrôleur utilise ce même canal sécurisé dans le sens inverse. Lorsque le contrôleur a un paquet de données à relayer via le commutateur, il utilise le message OpenFlow PACKET-OUT. Sur la figure 3, 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 commutateur OpenFlow peut être uniquement OpenFlow ou hybride. Dans ce dernier cas, le commutateur peut aussi commuter les paquet selon son mode traditionnel comme commutateur Ethernet ou routeur IP. Le cas hybride peut être vu comme un commutateur OpenFlow qui réside à côté d ’un commutateur traditionnel et indépendant. Ce type de commutateur hybrique requiert un mécanisme de classification qui soit dirige les paquets au contrôleur OpenFlow, soit les traite de manière traditionnelle.

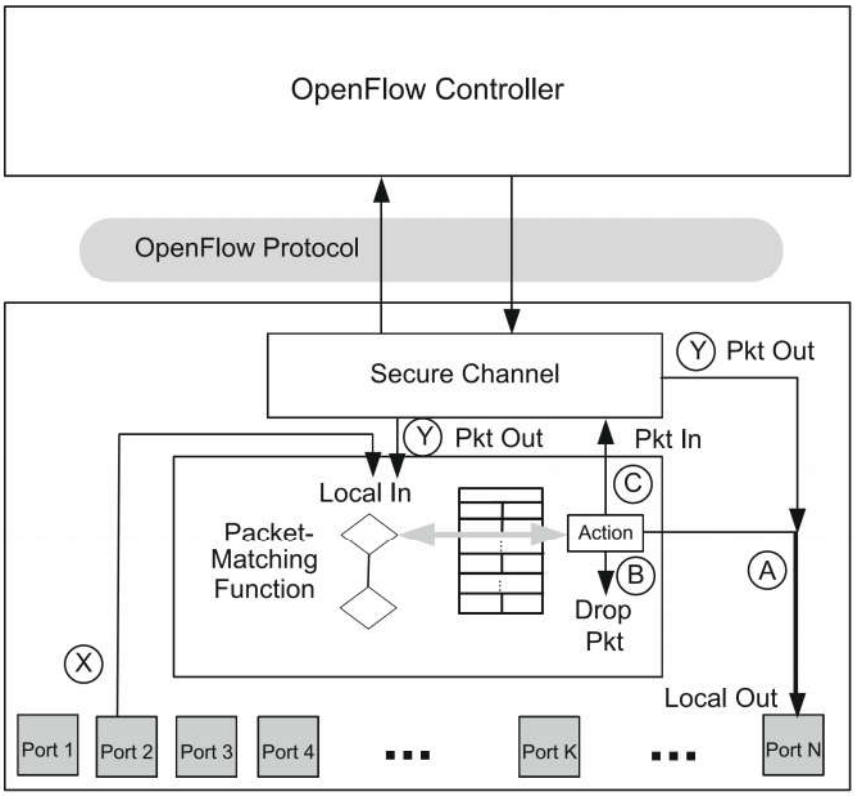

Figure 3 : Fonctions du commutateur OpenFlow 1.0

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 commutateurs 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.

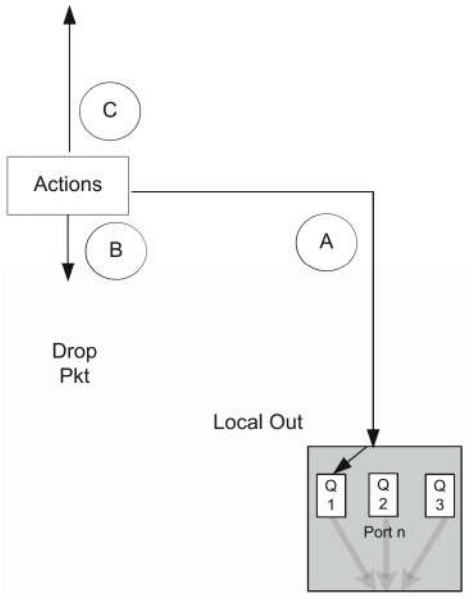

Figure 4 : Port Physique et fie d'attente OpenFlow

## 3.1 Table de flux

Chaque table de flux du commutateur contient un ensemble d'entrées de flux qui présentent les règles d'acheminement des paquets.
Une entrée de flux est composée de (Figure 5):

- Match fields : champs de correspondance qui définissent le modèle du flux de paquets à travers l'instanciation des champs d'en-tête allant de la couche Ethernet à la couche Transport
- Counters : des compteurs sur les paquets
- Actions : Actions à appliquer aux paquets qui correspondent à l'entrée de flux.

| Match Fields | Counters | Actions |
|---|---|---|

Figure 5 : Structure d'une entrée de flux

Chaque entrée de flux est associée à zéro ou plusieurs actions qui dictent la façon dont le commutateur 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 du paquet garanti dans un port. 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 de paquets doivent correspondre à ceux générés par une exécution séquentielle des actions.
Un commutateur 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 commutateur n'a pas à supporter tous les types d'action - seulement celles marquées " Actions obligatoires ". Lors de la connexion au contrôleur, un commutateur indique lesquelles des actions optionnelles il supporte.

### 3.1.1. Champs de correspondance

Les Match fields pouvant être utilisés 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 avec OpenFlow 1.0.
- 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

### 3.1.2. 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.
- Packet Lookups : 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**

- Received Packets : 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**

- Received Packets : Nombre de paquets reçus sur ce port
- Transmitted Packets : Nombre de paquets transmis par ce port
- Received Bytes : 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
- Receive Errors : Nombre de paquets reçus en erreur
- Transmit Errors : Nombre de paquet transmis en erreur
- Receive Frame Alignment Errors : Nombre de paquet reçus avec erreur d'alignement
- Receive Overrun Errors : 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 Overrun Errors : Nombre de paquet transmis sur cette file d'attente et rejetés 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.

### 3.1.3. Actions

Il existe des actions obligatoires que doit supporter tout commutateur OpenFlow et des actions OpenFlow que peut supporter un commutateur OpenFlow.
Action (Obligatoire): Forward. Les commutateurs OpenFlow doivent supporter l'acheminement de packet aux ports physiques ainsi qu'aux ports virtuels suivants :

- ALL: Est utilisé pour inonder un paquet sur tous les ports du commutateur à l'exception du port d'entrée ; ceci permet d'offrir une capacité broadcast rudimentaire au commutateur OpenFlow.
- CONTROLLER: Encapsuler le paquet et l'envoyer au contrôleur (Message PACKET_IN du commutateur au contrôleur). Cette action peut être indiquée dans l'entrée de flux. Aussi, si aucune entrée ne correspond au paquet à acheminer, le commutateur émet par défaut ce paquet au contrôleur.
- LOCAL: Envoyer le paquet à la CPU locale (contrôleur local dans le commutateur). Un commutateur dispose d'un contrôleur local vers lequel le paquet peut être envoyé pour la décision de son acheminement.
- TABLE: S'applique uniquement aux paquets que le contrôleur émet au commutateur. 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 commutateur (e.g., émetteur et récepteur présents sur le même hotspot WiFi).

Action (Optionnelle): Forward. Le commutateur 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 commutateur. 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 commutateur est hybride (commutateur 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.

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 Ethernet avec la nouvelle valeur.
- Modify Ethernet destination MAC address (48 bits) : Remplace l 'adresse MAC destination Ethernet 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 champs 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 6 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.

**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 6 : Exemples d'entrées de table de flux

# 4. Messages OpenFlow

Les messages OpenFlow entre le contrôleur et le commutateur sont transmis via un canal sécurisé, implanté via une connexion TLS sur TCP. Le commutateur initie la connexion TLS lorsqu'il connaît l'adresse IP du contrôleur. Chaque message entre le commutateur 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.

## 4.1. Messages symétriques

Les messages symétriques peuvent être émis indifféremment par le contrôleur ou le commutateur 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 commutateur et le contrôleur; Le protocole spécifie que la plus petite des deux versions doit être utilisée pour la communication contrôleur -commutateur sur le canal sécurisé
Les messages **ECHO** sont utilisés par n'importe quel entité (contrôleur, commutateur) 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**.
Les messages **VENDOR** sont disponibles pour des améliorations et pour une expérimentation spécifique au vendeur.

## 4.2. Messages asynchrones

Les messages asynchrones sont émis par le commutateur au contrôleur sans que le commutateur ait été sollicité par le contrôleur. Le message **PACKET_IN** est utilisé par le commutateur 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 commutateur 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 commutateurs ne supportant pas de mémorisation interne (ou ne disposant plus de mémoire) émettent le paquet entier au contrôleur dans le message PACKET-IN.
Le commutateur 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 commutateur.
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 commutateur utilise le message **ERROR** pour notifier des erreurs au contrôleur, e.g., ajout d'une entrée de flux par le contrôleur contenant des actions non supportées par le commutateur.

## 4.3. Messages contrôleur-commutateur

Les messages contrôleur-commutateur représentent la catégorie la plus importante de messages OpenFlow. Ils peuvent être représentés 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 commutateur. Le contrôleur utilise la paire de message **FEATURES_REQUEST** et **FEATURES_REPLY** afin d'interroger le commutateur 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 commutateur.
Les messages *command from controller* sont au nombre de 3. **PACKET-OUT** est analogue à PACKET_IN mentionné précédemment. Le contrôleur utilise PACKET_OUT afin d'émettre des paquets de données au commutateur pour leur acheminement via le plan usager (Plan de données).
Le contrôleur modifie les entrées de flux existantes dans le commutateur via le message **FLOW_MOD**.
**PORT_MOD** est utilisé pour modifier l'état d 'un port OpenFlow.
Des statistiques sont obtenues du commutateur 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 commutateur 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 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 commutateur. La confirmation est retournée par le commutateur via la réponse **BARRRIER_REPLY**.
La figure 7 montre des échanges OpenFlow entre contrôleur et commutateurs après l'établissement de la connexion TCP et du tunnel TLS, initiés par le commutateur.

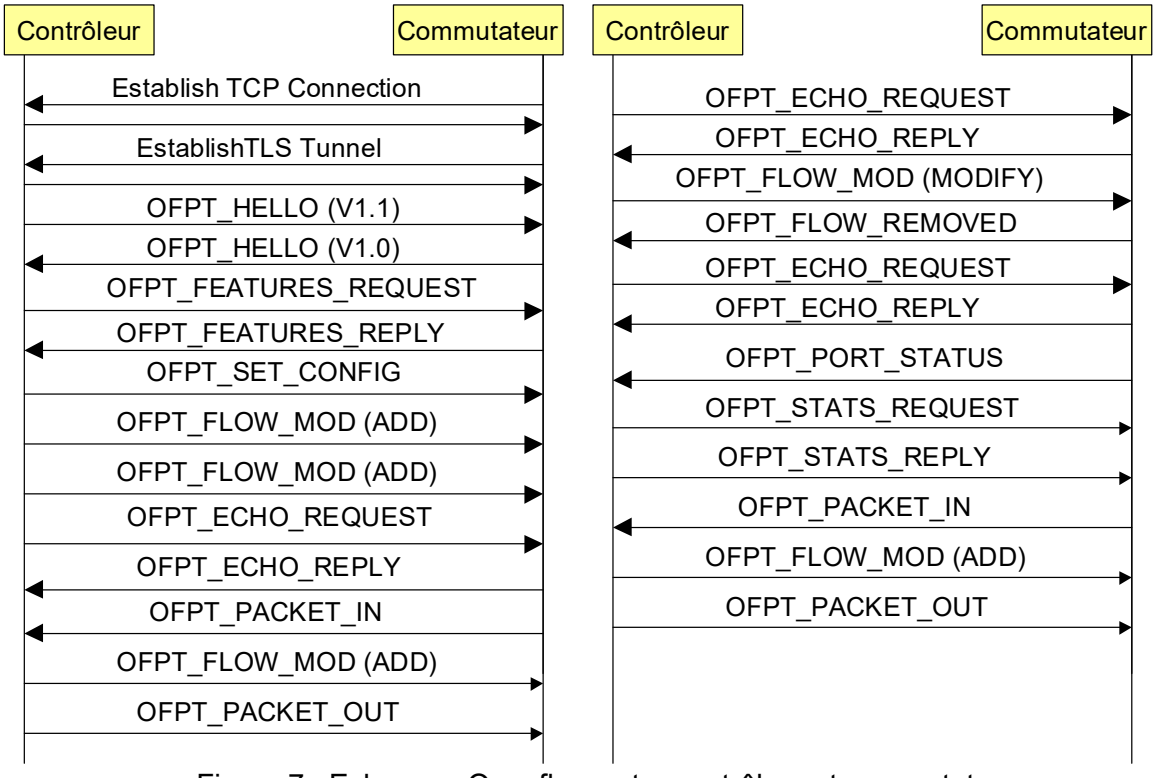

Figure 7 : Echanges Openflow entre contrôleur et commutateur

La formation EFORT « Virtualisation de Réseau et de Service, SDN et NFV » décrit la virtualisation de réseau et de service, les nouveaux concepts pour y parvenir tels que SDN et NFV avec leurs architectures et composants clés ainsi que les stratégies des principaux acteurs du monde des réseaux de télécommunication dans ce domaine.
http://efort.com/index.php?PageID=21&l=fr&f_id=178&imageField.x=3&imageField.y=1