# Floodlight: Lab 1 redirection

| | |
|---|---|
| **Objet** | contrôleur SDN avec floodlight |
| **Niveau requis** | débutant, avisé |
| **Débutant, à savoir** | floodlight-overview |
| **Suivi** | :DRAFT: |

## Objectif

Ce didacticiel, permet d'apprendre à utiliser Floodlight conjointement avec Open vSwitch et OpenFlow pour effectuer une redirection transparente des paquets..

À la fin de ce didacticiel, on doitz être familiarisé avec les éléments suivants:

- Openflow
- floodlight
- Création de flux OpenFlow
  - Matchs OpenFlow
  - Actions OpenFlow

## Description de la topologie

La topologie de test nécessite la mise en oeuvre:

- de trois hôtes, tous connectés l'un à l'autre via un seul commutateur activé par OpenFlow. Ceci est le plan de données.
- d'un commutateur connecté à un contrôleur. Ceci est le plan de contrôle.

**A propos du contrôleur**: Dans ce didacticiel, on utilise un contrôleur Floodlight. C'est un contrôleur basé sur Java, qui va avoir une vue globale du réseau! Cela signifie qu'il saura tout sur le commutateur et tous les hôtes qui y sont connectés. Floodlight va décider de la manière dont les paquets doivent voyager dans le plan de données. En fait, on indiquera à Floodlight comment il doit gérer certains paquets en lui faisant insérer des flux bien conçus sur le commutateur logiciel.

### Mise en oeuvre de la topologie

La topologie GNS3 est composée de 5 noeuds:

- **Noeud 1**: host 1
  - ifconfig eth0 10.0.0.1 netmask 255.255.255.0
  - route default gateway 10.0.0.7
- **Noeud 2**: host 2
  - ifconfig eth0 10.0.0.2 netmask 255.255.255.0
  - route default gateway 10.0.0.7
- **Noeud 3**: host 3
  - ifconfig eth0 10.0.0.1 netmask 255.255.255.0

    - route default gateway 10.0.0.7
- **Noeud 4**: interface NAT pour accés au réseau via openvswitch et au contrôleur floodlight (réseau 192.168.122.0/24)
- **Noeud 5**: commutateur
    - vlan 10 : ip address 10.0.0.7 24
    - vlan 212 : ip address 192.168.122.7 24
    - ge1/0 : por access vlan 10 (host1)
    - ge2/0 : por access vlan 10 (host2)
    - ge3/0 : por access vlan 10 (host3)
    - ge4/0 : por access vlan 212 (NAT)

## Configuration du contrôleur

Une fois la topologie installée et que toutes les ressources sont en place, lancer le contrôleur floodlight à partir de la ressource contrôleur.

**floodlight** a intégré ce qu'on appelle une API REST. Grâce à Curl et à ces commandes REST, on peut donc communiquer avec **Floodlight**.

On doit maintenant voir un journal stdout de Floodlight. Rien d'intéressant pour le moment, mais si vous attend assez longtemps, on peut Floodlight envoyer des paquets LLDP pour essayer de découvrir la topologie. Cependant, il ne verra pas grand-chose car aucun commutateur n'a été connecté à Floodlight.

## Configuration du commutateur

Maintenant que le contrôleur est configuré et fonctionne, il faut configurer le commutateur.

### Configurer une instance

Pour configurer OpenFlow, dans la vue système, utiliser la commande `openflow` et spécifier une instance. Ceci amène aux options de configuration globales OpenFlow:

```
system-view
System View: return to User View with Ctrl+Z.
[HPE]openflow instance 1
```

Une instance OpenFlow doit être configurée. Sur les commutateurs ProVision, il existe un mappage un à un entre les instances et les VLAN. En d'autres termes, chaque VLAN nécessite une instance OpenFlow distincte. Ceci n'est pas vrai pour les commutateurs Comware (plusieurs VLAN peuvent être mappés sur une seule instance OpenFlow).

Contrairement aux commutateurs 5900 prenant en charge 4094 instances d'OpenFlow, les commutateurs de la série 5500 ne prennent en charge que 8 instances.

#### Configurer le contrôleur

Un ID de contrôleur et une adresse IP doivent être spécifiés. On peut configurer plusieurs contrôleurs pour la redondance et l'équilibrage de la charge. Dans ce cas, un seul contrôleur est configuré.

```
[HPE-of-inst-1]controller 1 address ip 192.168.56.7
```

## Associer les vlans

Les commutateurs hybrides OpenFlow prennent en charge le fonctionnement OpenFlow et le fonctionnement de commutation Ethernet normal. Certains VLAN utilisent la commutation Ethernet L2 traditionnelle, l'isolation de VLAN, le routage L3, le traitement des listes de contrôle d'accès et de qualité de service, etc. Ce type de commutateur doit fournir un mécanisme de classification qui achemine le trafic vers le pipeline OpenFlow ou le pipeline normal. Les commutateurs HP utilisent des VLAN (un ou plusieurs) à cette fin.

```
[5500-1-of-inst-1]classification vlan 10
This command isn't effective until the active instance command is issued.
[5500-1-of-inst-1]
```

#### Activer l'instance

La dernière étape consiste à activer l'instance:

```
[5500-1-of-inst-1] active instance
```

C'est tout. Il n’est pas trop difficile de configurer une seule instance de base d’OpenFlow sur les commutateurs HP ProVision

On doit voir le commutateur se connecter au contrôleur dans le journal Floodlight:

Pour s'assurer de pouvoir communiquer avec tout le monde, sur l'hôte 1 :

```
ping 10.0.0.2
ping 10.0.0.3
```

Les deux devraient fonctionner.

# Contrôle de l'existant

dans l'hôte 2 lancer la commande suivante:

```
sudo tcpdump -i eth1 -eenn
```

dans l'hôte 3 lancer la même commande:

```
sudo tcpdump -i eth1 -eenn
```

**tcpdump** est juste un outil qui permet de voir quels paquets apparaissent sur une interface spécifiée. Le «-eenn» indique simplement de donner plus d'informations et de ne pas essayer de distinguer les noms des périphériques qu'il voit.

À ce stade, vous ne devriez plus rien voir sauf un paquet ARP ou LLDP occasionnel. C'est sur le point de changer.

Sur l'hôte 1, lancer une requête ping à l'hôte 2 à l'aide de la commande suivante:

```
ping 10.0.0.2
```

Maintenant, on doit enfin voir du trafic réseau! Les paquets ne doivent être vus que sur l'hôte 2, ce qui est logique car c'est à lui qu'ils sont adressés!

# Recherche des informations sur les périphériques connectés

La première chose à faire est de demander au contrôleur quels périphériques sont connectés. Cela donnera toutes les informations dont on aura besoin, par exemple. ports, adresses MAC, adresses IP.

Pour cela exécuter cette commande dans un nouvel onglet sur la ressource contrôleur:

```
curl http://localhost:8080/wm/device/ | python -m json.tool
```

Dans la sortie, on obtiendra une liste des périphériques que Floodlight a appris.

# Fabrication des flux statiques

Un flux statique est simplement un flux OpenFlow inséré par Floodlight via l'API REST. Fondamentalement, on crée un message JSON à envoyer à Floodlight pour lui expliquer le flux que

l'on veut qu'il insère en indiquant s le commutateur pour mettre le flux, le nom du flux, les matches et les actions.

<WRAP info>OpenFlow, un flux est un outil assez puissant et peut faire beaucoup de choses. Il peut correspondre à beaucoup de champs différents et même réécrire ces champs en utilisant des actions!<WRAP info>

les flux statisques insérés redirigeront de manière transparente tous les paquets IP de l'hôte 1 (destiné à l'hôte 2) vers l'hôte 3. Toute cette redirection se passe sans que l'hôte 1 ne le sache. Cela signifie que l'hôte 1 pense toujours qu'il parle à l'hôte 2 tout le temps.

Il faut créer deux flux, l'un pour une direction et l'autre pour l'autre direction.

# Fabrication du Flux 1: hôte 1 vers hôte 3

Pour commencer, on va expliquer ce qu'on essaye exactement d'accomplir avec ce premier flux. On à déjà vu que lorsqu'on envoie un ping à l'hôte 2 (10.0.0.2) à partir de l'hôte 1 (10.0.0.1), l'hôte 2 reçoit les paquets. Cela a du sens cependant. Tout dans cet en-tête de paquet indique à celui-ci d'aller sur l'hôte 2 (l'adresse MAC, l'adresse IP), et même par quel port l'envoyer!

Mais on veut changer les choses …

On veut rediriger ces paquets vers l'hôte 3.

Alors comment le fait-on? Il faut créer un flux qui correspond aux paquets IP destinés à l'hôte 2 à partir de l'hôte 1, et en utilisant des actions, on va réécrire les adresses MAC et IP pour inciter l'hôte 3 à penser que le paquet lui est destiné. Il faut également indiquer au commutateur sur quel port envoyer ce paquet.

## Etape 1: Indiquer l'id du switch

La première chose à faire est de dire à Floodlight le commutateur sur lequel on veut insérer ce flux. Pour ce faire, il faut indiquer le DatapathID (DPID) du commutateur dans le champ "commutateur" de l'objet JSON qui sera envoyé à Floodlight.

On peut obtenir ce DPID en entrant la commande suivante sur le commutateur:

```
show openflow instance 1
```

Dans la sortie du commutateur ci-dessous on remarquera que le DPIDest retourné au format hexa, pour l'insérer dans l'objet JSON, il faut le saisir au format suivant: 'xx:xx:xx:xx:xx:xx:xx:xx'

## Etape 2: Indiquer un nom

La prochaine chose à faire est de donner un nom au flux. Ce nom doit être unique, Il ne peut y avoir d'autre flux dans la topologie avec le même nom. On va simplement nommer ce premier flux «flux-1» dans le champ "name" de notre objet JSON.

## Etape 3: Indiqué une priorité et activer

Le champ suivant est le champ “priorité”. Lorsqu'un paquet atteint un commutateur, il parcourt chaque flux en ordre de priorité jusqu'à ce qu'il corresponde à l'un d'entre eux. Dans cet esprit, il faut donner à nos flux une très haute priorité afin de s'assurer qu'ils sont toujours en première ligne! on indiquera '32768' dans le champ “priority”.

Le prochain paramètre est assez facile. Il faut décider si on veut que le flux soit actif ou non. Pour cela simplement définir le champ “actif” sur “true”. Cela garantira que le flux fait réellement son travail.

## Etape 4: Fabriquer les correspondances (Match Maker)

Il faut maintenant définir les correspondances ou, en d'autres termes, le type de paquets sur lequel on veut effectuer les actions. C'est une bonne pratique d'être aussi précis que possible. Nous allons match par match ici expliquer ce que chaque champ correspond.

Dans la sortie constiuée à l'étape “Recherche des informations sur les périphériques connectés”, rechercher l'hôte 1 et l'hôte 2 (10.0.0.1 et 10.0.0.2).

Les champs matchs à faire correspondre sont :

- **in_port**: doit correspondre au port sur lequel le paquet arrive. et la valeur qu'on utilise est le numéro de port que l'on viens de trouver pour l'hôte 1 (10.0.0.1).
- **eth_type**: doit correspondre à un EtherType spécifique, qui dans notre cas est IPv4. Pour ce faire, utiliser la valeur hexadécimale '0x0800'.
- **eth_src**: doit correspondre à l'adresse MAC de la source (l’hôte 1: 10.0.0.1).
- **ipv4_src**: doit correspondre à l'adresse IP l’adresse IP de la source (l’hôte 1: 10.0.0.1).
- **eth_dst**: doit correspondre à l'adresse MAC de la destination (l’hôte 2: 10.0.0.2)
- **ipv4_dst**: doit correspondre à l'adresse IP de la destination (l’hôte 2: 10.0.0.2)

Voilà pour les matchs! Maintenant qu'on a déterminé la partie matchs des paquets IPv4 dans la direction de l'hôte 1 vers l'hôte 2, il fautcréer des actions pour manipuler ces paquets ...

## Etape 5: Fabriquer les actions

L'objectif du tuto est de rediriger les paquets, de l'hôte 1 destiné à l'hôte 2, vers l'hôte 3. Pour ce faire, il faut réécrire les adresses de destination de l'hôte 2 vers l'hôte 3. Il faut également spécifier le port sur lequel les paquets doivent sortir. Récupérer dans la liste des périphériques les informations (adresse MAC, l'adresse IP et port) relatives à l’hôte 3 (10.0.0.3).

Pour cela on a juste besoin de définir des actions pour:

- réécrire les adresses de destination en utilisant les champs **eth_dst** et **ipv4_dst**. Les valeurs de ces champs vont être l'adresse MAC et l'adresse IP de l'hôte 3, respectivement.
- indiquer au paquet sur quel port sortir, (le port de l'hôte 3) dans champ **output**, qui devra toujours être la dernière entrée de la liste d'actions.

## Etape 6: Assembler l'objet json

Tout cela doit être assemblé maintenant. C'est la commande complète qu'il faut entrer sur la ressource de contrôleur. Il faut renseigner les champs **matchs** et **actions** avec les informations propres à votre projet.

```
{
"switch":"<DPID_OF_SWITCH>",
"name":"flow-1",
"priority":"32768",
"in_port":"<PORT_OF_10.0.0.1>",
"active":"true",
"eth_type":"0x0800","eth_src":"<MAC_OF_10.0.0.1>",
"eth_dst":"<MAC_OF_10.0.0.2>",
"ipv4_src":"10.0.0.1",
"ipv4_dst":"10.0.0.2",
"actions":"set_field=eth_dst-><MAC_OF_10.0.0.3>,set_field=ipv4_dst->10.0.0.3
,output=<PORT_OF_10.0.0.3>"
}
```

## Etape 7: Soumettre l'objet JSON

| **Méthode** | POST |
|---|---|
| **URI** | /wm/staticflowpusher/json |

Par exemple avec CURL :

```
$ curl -X POST -d
'{"switch":"<DPID_OF_SWITCH>","name":"flow-1","priority":"32768","in_port":"
<PORT_OF_10.0.0.1>","active":"true",
"eth_type":"0x0800","eth_src":"<MAC_OF_10.0.0.1>",
"eth_dst":"<MAC_OF_10.0.0.2>", "ipv4_src":"10.0.0.1", "ipv4_dst":"10.0.0.2",
"actions":"set_field=eth_dst-><MAC_OF_10.0.0.3>,set_field=ipv4_dst->10.0.0.3
,output=<PORT_OF_10.0.0.3>"}' http://localhost:8080/wm/staticflowpusher/json
```

Après avoir inséré ce flux, effectuer les test suivants:

- exécuter tcpdump sur les hôtes 2 et 3 à l'aide de la commande suivante:

```
sudo tcpdump -i eth1 -eenn
```

- envoyer une requête ping à l'hôte 2 à partir de l'hôte 1 en exécutant la commande suivante sur l'hôte 1:

```
ping 10.0.0.2
```

- Si le flux a été inséré correctement, on doit voir tous less paquets ping sur l'hôte 3, même s'ils

étaient initialement destinés à l'hôte 2!

**Note**: On peut confirmer que le flux est mis en correspondance en exécutant la commande suivante sur la ressource switch:

```
sudo ovs-ofctl dump-flow br0
```

Si on continue à faire un ping, on peut exécuter cette commande plusieurs fois et observer le compteur de paquets augmenter en fonction du flux inséré!.

# Fabrication du Flux 2: hôte 3 à hôte 1

On a donc créé un flux pour rediriger les paquets vers l'hôte 3, mais lorsqu'on envoi un requête ping de l'hôte 1 à l'hôte 2, on remarquera que l'hôte 1 réalise que même s'il envoie à l'hôte 2, il reçoit les réponses de l'hôte 3. Ce n'est pas transparent! On ne veut pas que l'hôte 1 sache qu'il ne parle pas à l'hôte 2. Pour que cela fonctionne, il faut également réécrire tous les paquets IP de l'hôte 3 vers l'hôte 1, en réécrivant les adresses source pour qu'ils semblent vennir de l'hôte 2. C'est essentiellement la même chose que notre premier flux, mais en sens inverse.

À ce stade, on dispose déjà de toutes les informations pour l'hôte 1, l'hôte 2 et l'hôte 3.

## Etape 1: Indiquer l'id du switch

Le commutateur DPID est identique au premier flux (il n'y a qu’un commutateur)

## Etape 2: Indiquer un nom

On lui donnera le nom “flux-2”

## Etape 3: Indiquer unepriorité et activer

On lui donnera la priorité sera “32768” et on mettra active à “true”.

## Etape 4: Fabriquer les correspondances (Match Maker)

Dans la sortie constiuée à l'étape “Recherche des informations sur les périphériques connectés”, rechercher l'hôte 1 et l'hôte 3 (10.0.0.1 et 10.0.0.3).

Pour les matchs, il faut faire correspondre les paquets IPv4 (eth_type=0x0800), les adresses MAC et IP (eth_src, eth_dst, etc. .) pour l’hôte 3 (la source) et l’hôte 1 (la destination) et le port d'entrée (“in_port”) sera le port de l’hôte 3.

### Etape 5: Fabriquer les actions

Pour les actions, il faute réécrire la source depuis l'adresse et le port de l'hôte 3 vers l'adresse et le port de l'hôte 2. Pour ce faire, définir les champs “eth_src”, “ipv4_src” et “output” dans la liste d'actions.

### Etape 6: Assembler l'objet json

Exécuter la commande sur la ressource de contrôleur pour insérer le second flux en renseignant les champs indiqués avec les informations propres à la topologie du test.

```
curl -X POST -d '{"switch": "<DPID_OF_SWITCH>","name":"flow-2",
"priority":"32768", "in_port":"<PORT_OF_10.0.0.3>", "active":"true",
"eth_type":"0x0800", "eth_src":"<MAC_OF_10.0.0.3>",
"eth_dst":"<MAC_OF_10.0.0.1>", "ipv4_src":"10.0.0.3", "ipv4_dst":"10.0.0.1",
"actions":"set_field=eth_src-><MAC_OF_10.0.0.2>,set_field=ipv4_src->10.0.0.2
,output=<PORT_OF_10.0.0.1>"}' http://localhost:8080/wm/staticflowpusher/json
```

### Etape 7: Soumettre l'objet JSON

| **Méthode** | POST |
|---|---|
| **URI** | /wm/staticflowpusher/json |

Tester de la même manière que nous lors du test du premier flux:

- exécuter tcpdump sur l'hôte 2 et l'hôte 3, puis sur l'hôte 1
- envoyer une requête ping à l'hôte 2 (10.0.0.2)
- on doit voir tous les paquets arriver sur l'hôte 3, mais l'hôte 1 pense qu'il parle à l'hôte 2

# Conclusion

Pendant ce tutoriel on a vu:

- Utilisation de l'API Floodlight
- Configuration du commutateur
- Fabrication des flux OpenFlow (pour rediriger de manière transparente le trafic IPv4!)