TP Software Defined Networks

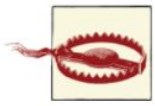
A. Mininet

1. Découverte (20”)

Mininet est un logiciel vous permettant de d'obtenir un réseau virtuel directement sur votre poste. Pour ce faire, il utilise la virtualisation par OS tout comme docker.
Pour commencer, lancer mininet avec la configuration par défaut constituée de 2 hosts (h1 et h2) connecté par un switch openflow (s1).

% sudo mn

Une fois que vous avez tapé cette commande, vous vous trouvez à l'intérieur du shell mininet.



Pour quitter mininet, toujours faire ***mininet> exit*** *(si vous avez oublié, faites sudo mn -c pour réinitialiser le réseau)*

Celui-ci vous permet de lancer des commandes arbitraires sur chacun des hosts par exemple la commande suivante
mininet> h1 ping h2
montre comment faire un ping depuis h1 vers h2.
mininet> help
vous affiche les commandes supportées par l'émulateur.

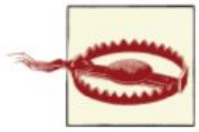
A1.1 à quoi servent les commandes pingall, iperf et xterm de mininet?

A1.2 Donez l'adresse MAC de l'interface eth0 la machine h2

Les switch de votre instance mininet sont configurés en “learning switch” par défaut.

A1.3 qu'est-ce qu'un “learning switch”? Quel est le résultat de pingall lorsque que le learning switch est utilisé?

A1.4 Quittez mininet et redémarrer en désactivant le contrôleur (ie. None). Quel est le résultat du pingall? En vous aidant du cours, quel est le donc le rôle du contrôleur dans une approche SDN?



A présent, sauf mention contraire nous utiliserons un contrôleur externe (ie. remote) appelé ryu. Celui-ci configurera les (openv)switches en utilisant openflow. Toutes les commandes mininet utiliseront donc "*--controller=remote*".
Par ailleurs, le contrôleur doit toujours être lancé *avant* mininet pour qu'il puisse fonctionner.

2. Topologies (60")

Plusieurs types de topologies sont disponibles par défaut avec mininet. La topologie par défaut que nous avons utilisé jusqu'à présent est
sudo mn --topo=tree,depth=1,fanout=2

A.2.1 Avec une topologie d'arbre d'une profondeur de 3 et un fanout de 4, combien y a-t-il de switches? Combien d'hôtes? Combien de liens? Combien de contrôleurs?

En plus d'être un contrôleur SDN, ryu propose une application permettant de visualiser la topologie. Une fois la commande suivante lancée, ouvrez le navigateur puis rendez-vous à l'adresse *http://localhost:8080*
**% > ryu run --observe-links **
/usr/local/lib/python2.7/dist-packages/ryu/app/gui_topology/gui_topology.py

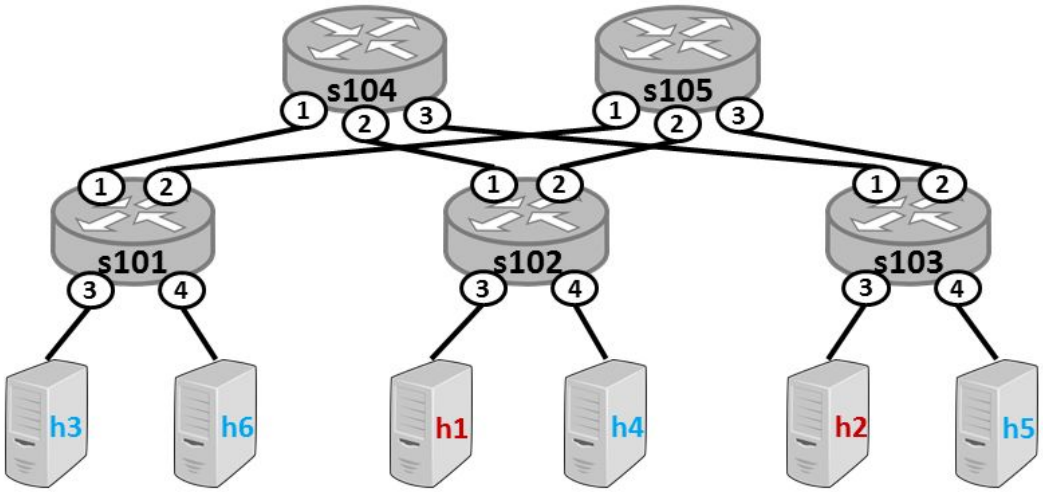
A.2.2 Avec la topologie précédente, décrivez le chemin de l'hôte h3 à l'hôte h48. Vous préciserez tous les switches et les ports traversés. (hint: utiliser la topologie de ryu et les commandes de mininet)

Mininet permet également de charger des topologies personnalisées implémentées en Python.

%>sudo mn --link=tc --custom /home/re350/re350/td01/topo/custom-topo.py --topo mytopo

l'option --link=tc permet de spécifier des options sur les liens (bandwidth, delay, loss)
A l'aide de l'exemple présent dans le fichier custom-topo.py, créez la topologie correspondante au schéma suivant (classe RE350Topo). Les switches s101,

s102 et s103 sont appellés “aggregation switch”. Les switches de plus haut niveau s104 et s105 sont les “core switch”



A.2.3 Donnez le code python permettant de générer cette topologie

A2.4 Afficher cette topologie à l'aide de ryu.

A2.5 Pour cette question utilisez le contrôleur par défaut. Quel est le résultat de pingall?

Supprimez s105 de votre topologie (ainsi que tous ses liens). Quel changement observez vous?

A.2.6. Qu'est-ce que le Spanning tree protocol?

Le Spanning tree protocol est implémenté par une application de ryu. Voici la commande pour lancer le contrôler implémentant le STP avec OpenFlow.

%> cd /usr/local/lib/python2.7/dist-packages/ryu
%> ryu-manager --observe-links ./app/gui_topology/gui_topology.py ./app/simple_switch_stp.py

Cette ligne lance également la visualisation de topologie à l'adresse http://localhost:8080
Revenez à la topologie de la question A2.3. Lancez le contrôleur, affichez la topologie sur votre navigateur puis lancez mininet.

A2.7 Est-ce que cette implémentation du contrôleur permet de résoudre le problème rencontré au A.2.5? Quelle est la topologie résultante? Quelle est l'inconvénient de cette topologie par rapport à la topologie initiale?

B. Openflow (40”)

Utilisez wireshark pour visualiser les messages openflow entre les switches et le contrôleur par défaut de mininet avec la topologie par défaut sans lancer de commande à l'intérieur de mininet pour l'instant (hint: pensez à utiliser les filtres wireshark) .

B.1.1 Faites un tableau avec tous les types des messages openflow échangés

B.1.2 Dans le tableau, écrivez la signification de chacun des types de messages. Pour cela, vous vous baserez sur le document OpenFlow Switch Specification Ver 1.5.1.

Lancez la commande pingall

B.1.3 Ajoutez les nouvelles commandes openflow capturées par wireshark et leur signification.

Lancez la commande pingall encore plusieurs autres fois.

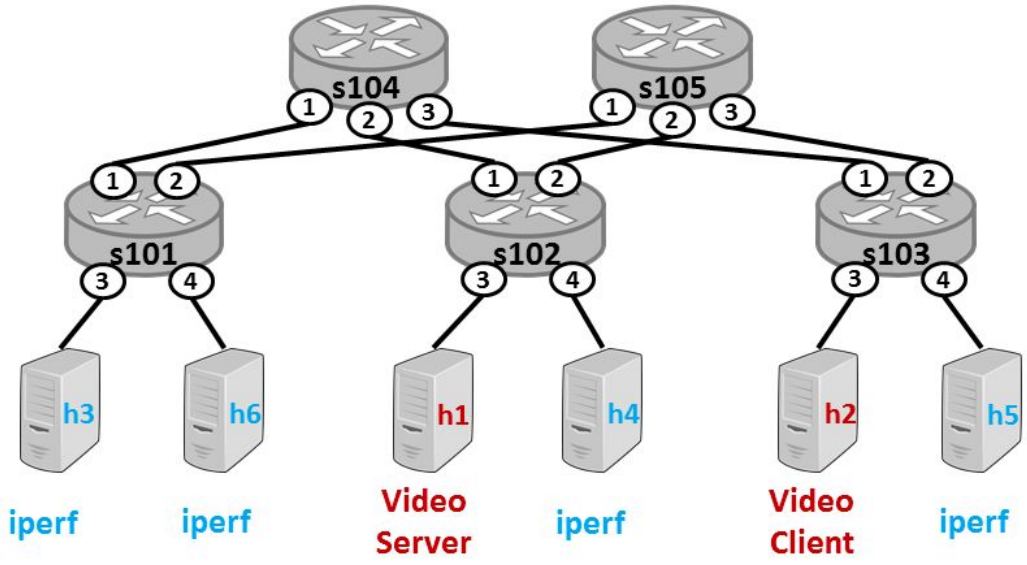
B.1.4 est-ce que les nouvelles commandes openflow du B.1.3 sont à nouveau présentes dans la capture? Si ce n'est pas le cas expliquez pourquoi.

B.1.5. Le message packet_in renferme des données appartenant au dataplane. Celles-ci sont transmises au controller lorsque le switch ne sait pas les traiter.

- Quel est dans ce cas le message retourné en packet_out?
- De quel niveau au sens OSI dépendent les données du datapath présentes dans le packet_in?

C. Contrôle de la qualité avec SDN

Dans le restant du TP, la topologie suivantes est mise en place.



Le serveur h1 est un serveur php fournissant l'accès à une video dans un format de streaming adaptatif (http dash).
Le client h2 est un navigateur Chrome qui va lire la vidéo.
Les autres serveurs sont des applications qui envoient sur le réseau des datagrammes UDP, jusqu'à saturation des liens.

1 Théorie (15")

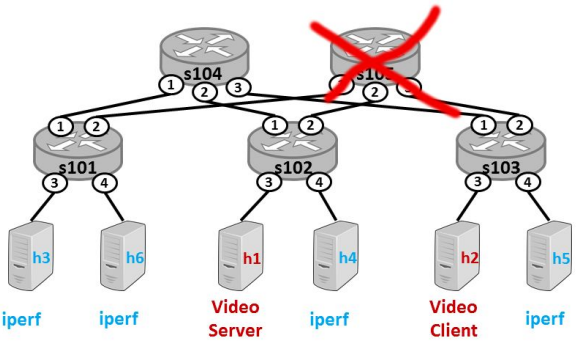
C.1.1 Quel protocole de niveau 4 est utilisé par Dash?

C.1.2 Rappellez brièvement quel est le mécanisme de contrôle de la congestion utilisé par ce protocole?

C.1.3 D'après vous, comment va se partager la bande passante entre les serveurs iperf et le serveur http dash? Justifiez.

2 Pratique pour la récupération des statistiques sans s105 (60”)

Pour l'instant, nous avons désactivé le s105 de la topologie. Nous allons utiliser mininet pour lancer la topologie sans s105, ainsi que tous les serveurs correspondant.



Pour lancer mininet, tapez les commandes suivantes
%> cd ~/re350/td01/dash
%> sudo ./mdc --vid
le programme devrait afficher la ligne suivante:
start ryu<ENTER>
Dans un autre terminal, lancez ryu avec le controlleur fourni
%> cd ~/re350/td01/dash/minidc/controller
%> ryu-manager ./controller.py
revenez ensuite sur le premier terminal et faites <ENTER>.
Une fenêtre de Chrome doit se lancer, en affichant une vidéo.

C2.1 Que pensez vous de la qualité de la vidéo? En vous aidant des éléments théoriques de la section précédente, expliquez pourquoi la qualité est ainsi.

> *Vous allez maintenant récupérer les statistiques réseaux des hôtes h1..h6 afin de mieux visualiser la bande passante consommée. Pour visualiser les statistiques ouvrez un nouveau navigateur à l'adresse http://127.0.0.1 Initialement les statistiques sont vides, votre objectif est de coder la remontée de statistiques des switches vers le controlleur SDN.*
> *Pour ce faire, ouvrez le fichier*
> ***~/re350/td01/dash/minidc/controller/bwmon.py:137***
> *Les statistiques sont remontée dans des messages openflow par les switches. Dans le cas où le switch est connecté à un hôte, envoyez le nombre de byte transférés et le nombre de byte reçus vers l'hôte.*

C2.2 listez les types des messages openflow utilisés pour remonter les statistiques au niveau du contrôleur
.

C2.3 implémenter l'algorithme suivant:

Si le nom du switch actuel est celui d'un switch de bordure:
—— Pour chaque "stat" (type: OFPPortStats) dans body faire:
——— Si le port de "stat" est un port du switch actuel
———— Si le nom du noeud branché par le port de "stat" correspond à un hôte
————— self.bwstats.addHostBwStat(hostname, transmitted bytes, received bytes)

C2.4 en utilisant le Dashboard, http://127.0.0.1/ Donnez, pour chaque hôte sa part dans la bande passante globale. Ces résultats confirment il ce que vous aviez prévu pour la théorie?

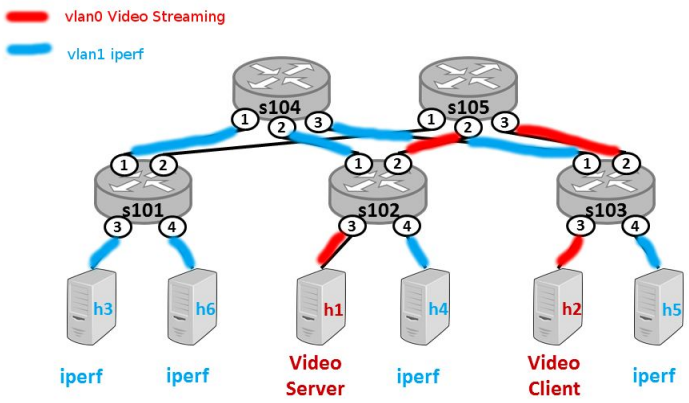
3. Routage par VLAN (45")

Nous allons maintenant modifier le contrôleur SDN afin que le traffic de h1 et h2 (vlan1) passe par s105 et que le reste (vlan0) passe par s104.

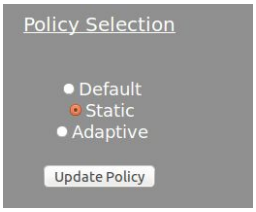
C3.1 Quelle va être l'impact de cette modification sur le réseau? Sur la saturation des liens? Sur la bande passante disponible pour le streaming video?

Ouvrez le fichier minidc/controller/policy.py. Etudiez la class DefaultPolicy.

L'objectif est de modifier la classe StaticPolicy afin que le traffic du vlan 0 transite par s104 et que celui du vlan 1 transite par s105. Le trafic entre deux hôtes connectés à deux aggregation switch ne doit pas remonter jusqu'aux core switches.



Pour vérifier le fonctionnement, utilisez la politique "Static" sur le dashboard



C3.2 Implémentez cette politique et donnez le code.

C3.4 Quel est l'impact sur la bande passante consommée par le vlan0? vlan1? Quel est l'impact sur la qualité de la video?

C3.5 Expliquez comment SDN peut être utilisé pour proposer une Qualité de service garantie à à un utilisateur du réseau.