RSX 217
Projet n°17

IN Morgan
DRAGHI Vincent
GODEFROY Nicolas

Janvier 2020

Contexte

Notre objectif :

- Notre équipe est chargée de **développer une application ou une intent** afin de répartir le trafic entre 2 hôtes à travers un contrôleur Onos.

- La répartition se fera sur **3 chemins** avec les taux respectifs suivants : **10 %**, **30%**, **60%**

Nos recherches initiales :

- Le wiki ONOS, la chaîne YouTube officielle
- Réseaux communautaires : Google Groups, Slack

Contexte

- **Mots-clés**

- **QoS** (pour traduire la création de bande passante réservée en pourcentage pour les 3 chemins)
- **Load balancing** (pour la même raison que pour la QoS, en supposant que le load balancing pourrait être paramétrable).
- **File d'attente ou queue** (nécessaire lorsqu'on parle de QoS)
- **MPLS** (technologie simple pour créer les chemins que les flux vont emprunter)
- **Mininet** (pour dresser une topologie)
- **OVS** (pour dresser une topologie)
- **Python** (pour créer des scripts déployables dans mininet)

Contexte

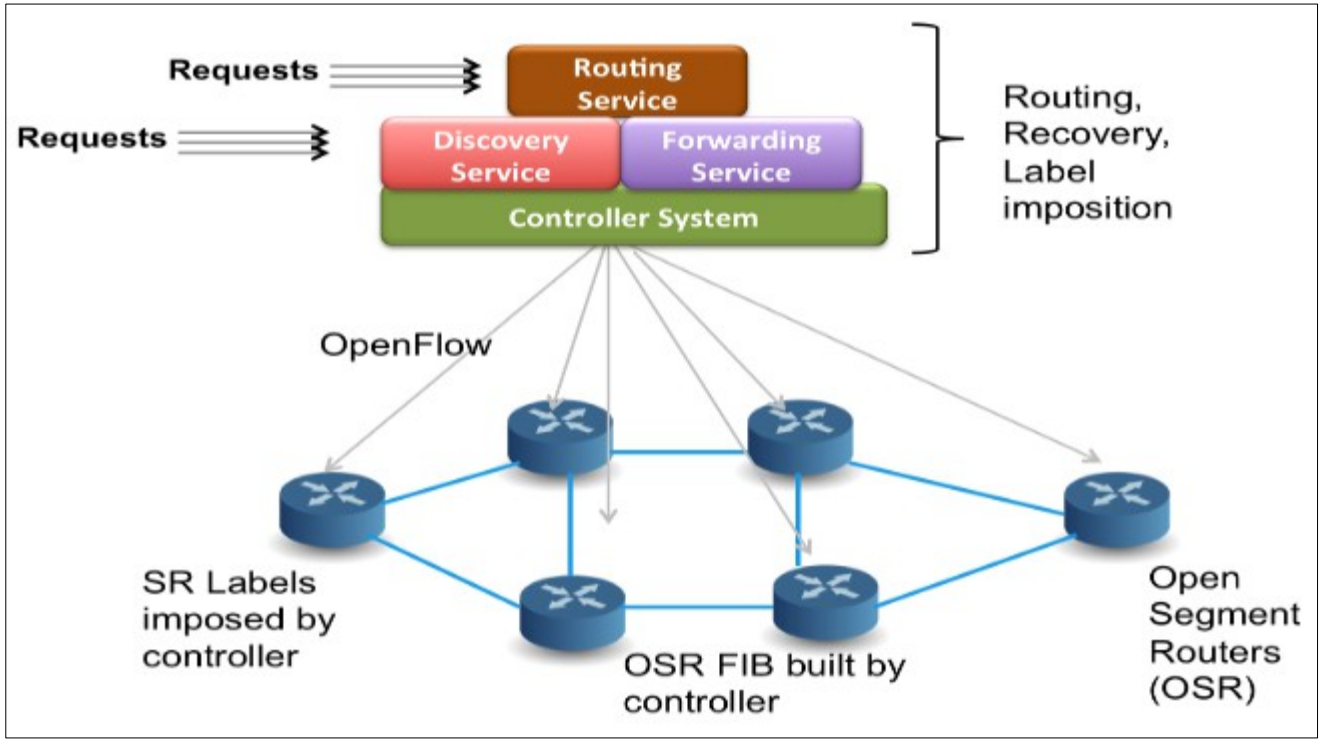
- Absence de trame d’ application existante que nous pourrions customiser.
- Choix d’ un ancien projet **Spring-Open,** qui utilise le segment routing.
- Spring-Open inclut des **packages d’ applications** permettant de créer très facilement des *tunnels*, par lesquels les flux de données peuvent passer.
- Usage de **fragments de code et namespaces** pour créer des files d’ attente et de la QoS dans un environnement créé par OVS

Le Projet Spring-Open

1ere piste

- Spring Open est un projet opérationnel mais qui a déjà **3 ans d'existence** et n'est **plus suivi ou maintenu**.

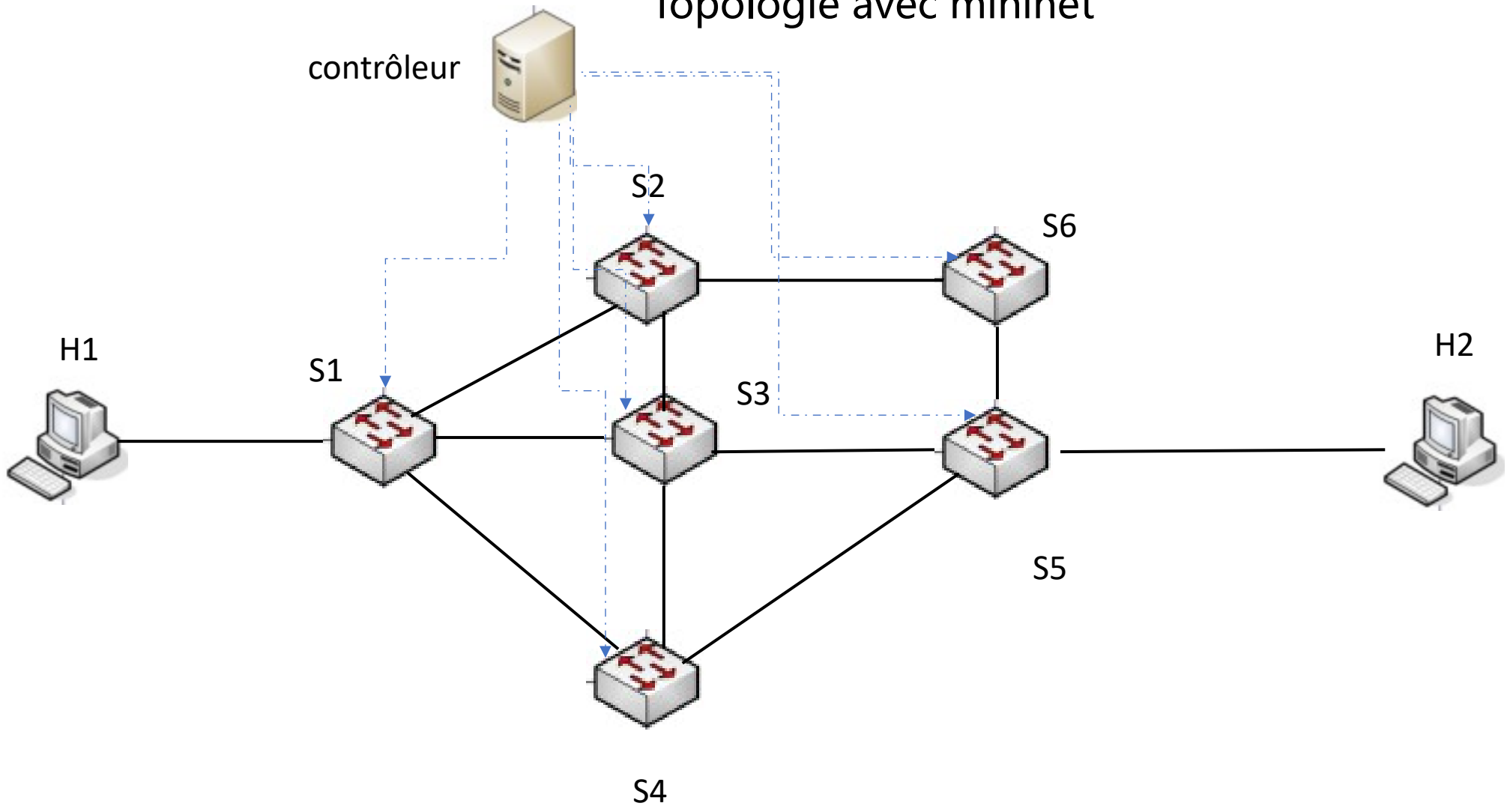
- Une fois la configuration pour les 3 tunnels effectuée, il nous a été **impossible d'arriver à mixer les lignes de code pour la QoS**



Fonctionnement de Spring-Open

Application

Topologie avec mininet



le cnam

Application

ONOS Open Network Operating System

ONOS-2 [En fonction] - Oracle VM VirtualBox

Fichier Machine Écran Entrée Périphériques Aide

```
mininet@mininet-vm:~/mininet$ ls
bin           custom  doc       LICENSE   mininet.egg-info  mnexec.1   setup.py
build         debian  examples  Makefile  mn.1              mnexec.c   util
CONTRIBUTORS  dist    INSTALL   mininet   mnexec            README.md
mininet@mininet-vm:~/mininet$ cd custom
mininet@mininet-vm:~/mininet/custom$ ls
alterableNet.py   README              srTopo8.py        testEcmp_10sw.py
alterableNet.pyc  sr_cpqd_full.py     srTopo8.py~       topo-2sw-2host.py
ecmpTopo10.py     sr_cpqd_full.py~    srTopo8.pyc       toporsx217.py
ecmpTopo10.pyc    sr-mn-script.py     srtopo.py
ecmpTopobis.py    sr_thesis_topo.py   srtopo.pyc
ecmpTopobis.pyc   sr_thesis_topo.pyc  sr-toporsx217.py
mininet@mininet-vm:~/mininet/custom$ sudo ./sr-toporsx217.py
*** Creating network
*** Adding controller
Unable to contact the remote controller at 127.0.0.1:6633
*** Adding hosts:
h1 h2
*** Adding switches:
s1 s2 s3 s4 s5 s6 s7 s8 s9
*** Adding links:
(h1, s1) (h2, s6) (s1, s2) (s1, s5) (s1, s9) (s2, s3) (s3, s4) (s4, s6) (s5, s6)
 (s6, s7) (s7, s8) (s8, s9) (s8, s9)
*** Configuring hosts
h1 h2
*** Starting controller
*** Starting 9 switches
s1 s2 s3 s4 s5 s6 s7 s8 s9
*** Starting CLI:
mininet>
```

Le ping fonctionne

```
*** Starting CLI:
mininet> h1 ping h2
PING 10.0.2.1 (10.0.2.1) 56(84) bytes of data.
64 bytes from 10.0.2.1: icmp_seq=1 ttl=59 time=167 ms
64 bytes from 10.0.2.1: icmp_seq=2 ttl=60 time=2.29 ms
64 bytes from 10.0.2.1: icmp_seq=3 ttl=60 time=2.16 ms
64 bytes from 10.0.2.1: icmp_seq=4 ttl=60 time=1.88 ms
64 bytes from 10.0.2.1: icmp_seq=5 ttl=60 time=2.94 ms
64 bytes from 10.0.2.1: icmp_seq=6 ttl=60 time=2.10 ms
^C
--- 10.0.2.1 ping statistics ---
6 packets transmitted, 6 received, 0% packet loss, time 5008ms
rtt min/avg/max/mdev = 1.884/29.863/167.781/61.679 ms
mininet>
```

le cnam

Application

Création de tunnel avec Spring-Open

```
mininet-vm(config)# tunnel TENPERCENT
mininet-vm(config-tunnel)# node
101  102  103  104  105  106  107  108  109
mininet-vm(config-tunnel)# node 10
101  102  103  104  105  106  107  108  109
mininet-vm(config-tunnel)# node 101
mininet-vm(config-tunnel)# node 102
mininet-vm(config-tunnel)# node 103
mininet-vm(config-tunnel)# node 104
mininet-vm(config-tunnel)# exit
mininet-vm(config)# show tunnel
# Id         Policies Tunnel Path [Head-->Tail] Label Stack [Outer-->Inner]
-|----------|--------|-----------------------|-------------------------
1 TENPERCENT          [101, 102, 103, 104]      [[103, 104]]
mininet-vm(config)#
```

Les routeurs sous Spring-Open

```
default controller: 127.0.0.1:8000, SDN OS 1.0 - custom version
mininet-vm> show router
# Router DPID             Router Name Router IP      Router Mac        Edge Rout
er Node SId
-|----------------------|----------|-------------|----------------|--------
--|--------
1 00:00:00:00:00:00:00:01 SFO-ER1     192.168.0.1/32 00:00:01:01:01:80 true
   101
2 00:00:00:00:00:00:00:02 SFO-CR2     192.168.0.2/32 00:00:02:02:02:80 false
   102
3 00:00:00:00:00:00:00:03 SFO-CR3     192.168.0.3/32 00:00:03:03:03:80 false
   103
4 00:00:00:00:00:00:00:04 Dallas-CR4  192.168.0.4/32 00:00:04:04:04:80 false
   104
5 00:00:00:00:00:00:00:05 Dallas-CR5  192.168.0.5/32 00:00:05:05:05:80 false
   105
6 00:00:00:00:00:00:00:06 Dallas-ER6  192.168.0.6/32 00:00:06:06:06:80 true
   106
7 00:00:00:00:00:00:00:07 NY-CR7      192.168.0.7/32 00:00:07:07:07:80 false
   107
8 00:00:00:00:00:00:00:08 NY-CR8      192.168.0.8/32 00:00:08:08:08:80 false
   108
9 00:00:00:00:00:00:00:09 NY-CR9      192.168.0.9/32 00:00:09:09:09:80 false
   109
mininet-vm>
```

Implémentation

2eme piste

Le TP SDN nous avait montré la simplicité d' utilisation **d' OpenVSwitch et des namespaces linux**

- Hypothèse de pouvoir créer la topologie avec cette technologie, afin d' intégrer les lignes de codes pour la QoS et les files d' attente.

le cnam

Implémentation

onos
Open Network Operating System

```
#!/bin/bash
set +xe

# 1. Create 2 namespaces(Simulated clients)
ip netns add h1
ip netns add h2

# 2. Create 5 openvswitches
sudo ovs-vsctl add-br s1
sudo ovs-vsctl add-br s2
sudo ovs-vsctl add-br s3
sudo ovs-vsctl add-br s4
sudo ovs-vsctl add-br s5
sudo ovs-vsctl add-br s6

# 3. creation of link and ports necessary for QOS and Queue
ip link add h1-eth1 type veth peer name s1-eth1
ip link add s1-s2 type veth peer name s2-s1
ip link add s1-s3 type veth peer name s3-s1
ip link add s1-s4 type veth peer name s4-s1
ip link set h1-eth1 netns h1
sudo ovs-vsctl add-port s1 s1-eth1
sudo ovs-vsctl add-port s1 s1-s2
sudo ovs-vsctl add-port s1 s1-s3
sudo ovs-vsctl add-port s1 s1-s4

# 4. creation of qos and queue
sudo ovs-vsctl set interface s1-s2 ofport_request=5 -- set interface s1-s3 ofport_request=6 -- set interface s1-s4 ofport_request=7 -- set port s1-eth1 qos=@newqos -- --id=@newqos create qos typ
e=linux-htb other-config:max-rate=10000000 queues:123=@s1-s2queue queues:234=@s1-s3queue queues:345=@s1-s4queue -- --id=@s1-s2queue create queue other-config:max-rate=1000000 --  --id=@s1-s3queu
e create queue other-config:max-rate=3000000 -- --id=@s1-s4queue create queue other-config:max-rate=6000000

# 5. To direct packets from the port to the queues reserved for them
sudo ovs-ofctl add-flow s1 in_port=5,actions=set_queue:123,normal
sudo ovs-ofctl add-flow s1 in_port=6,actions=set_queue:234,normal
sudo ovs-ofctl add-flow s1 in_port=7,actions=set_queue:345,normal

# 3. Create missing vethernet links
ip link add h2-eth1 type veth peer name s5-eth1
ip link add s2-s3 type veth peer name s3-s2
ip link add s3-s4 type veth peer name s4-s3
ip link add s3-s5 type veth peer name s5-s3
ip link add s4-s5 type veth peer name s5-s4
ip link add s2-s6 type veth peer name s6-s2
ip link add s6-s5 type veth peer name s5-s6
```

Implémentation

- La partie Qos fonctionne et les paquets de données empruntent bien les 3 chemins.

- L'objectif principal non atteint: nous répondons partiellement au thème du sujet, qui demande d'utiliser Onos.

3eme piste

Cette fois, les recherches ont été plus fructueuses et nous avons trouvé un **script .java** qui semble permettre la création de la QOS.

- Il reste à le **compiler et à le tester**.

3eme piste -> problème de compilation – OVSDB nécessite d'autres connaissance que nous ne sommes pas arrivés à trouver

4eme piste

Script mininet ou OVS (topologie et QOS + commandes Onos pour gérer les flux.

Voir le rapport pour le compte rendu de cette piste

Problème principal = ressources nécessaires trop élevées- Ordinateur pas assez puissant (voir vidéo)

Merci de votre attention