Wagner Nicolas
Losser Julien
RT 331 ROM

TP – Automatisation réseaux : REST API

1. Introduction

Le but de ce TP est de vous familiariser avec REST API pour programmer votre architecture SDN via des commandes cette API. REST est un style d'architecture logicielle définissant un ensemble de contraintes à utiliser pour créer des services web. Les services web conformes au style d'architecture REST, aussi appelés services web RESTful, établissent une interopérabilité entre les ordinateurs sur Internet. Ryu est un système d'exploitation réseau open source sous licence Apache v2. 0. Il prend en charge le protocole openflow.

2. Premiers pas avec REST API

Lancez « ryu-manager » avec les scripts « REST » et « simple_switch_openflow13 »

```
(ryu-python3.9-venv) root@ryu:/home/toto# ryu-manager ryu.app.simple_switch_13 ryu.app.ofctl_rest
loading app ryu.app.simple_switch_13
loading app ryu.app.ofctl_rest
loading app ryu.controller.ofp_handler
instantiating app None of DPSet
creating context dpset
creating context wsgi
instantiating app ryu.app.simple_switch_13 of SimpleSwitch13
instantiating app ryu.app.ofctl_rest of RestStatsApi
instantiating app ryu.controller.ofp_handler of OFPHandler
(5432) wsgi starting up on http://0.0.0.0:8080
```

3.Topologie

Voici notre topologie composée de 5 switchs avec une adresse réseau 192.168.1.0/24.

Création de la topologie avec le contrôleur ryu et l'adresse réseau 192.168.1.0/24 avec 5 switch.

```
root@ryu:/home/toto# mn --controller=remote,ip=127.0.0.1 --mac -i 192.168.1.0/24 --switch=ovsk,protocols=OpenFlow13
 --topo=linear,5
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3 h4 h5
*** Adding switches:
s1 s2 s3 s4 s5
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (h4, s4) (h5, s5) (s2, s1) (s3, s2) (s4, s3) (s5, s4)
*** Configuring hosts
h1 h2 h3 h4 h5
*** Starting controller
c0
*** Starting 5 switches
s1 s2 s3 s4 s5 ...
```

Wagner Nicolas
Losser Julien
RT 331 ROM

Vérification de la topologie avec la commande dump et pingall.

```
mininet> dump
<Host h1: h1-eth0:192.168.1.1 pid=5447>
<Host h2: h2-eth0:192.168.1.2 pid=5449>
<Host h3: h3-eth0:192.168.1.3 pid=5451>
<Host h4: h4-eth0:192.168.1.4 pid=5453>
<Host h5: h5-eth0:192.168.1.5 pid=5455>
<OVSSwitch{'protocols': 'OpenFlow13'} s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=5460>
<OVSSwitch{'protocols': 'OpenFlow13'} s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid=5463>
<OVSSwitch{'protocols': 'OpenFlow13'} s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None,s3-eth3:None pid=5466>
<OVSSwitch{'protocols': 'OpenFlow13'} s4: lo:127.0.0.1,s4-eth1:None,s4-eth2:None,s4-eth3:None pid=5469>
<OVSSwitch{'protocols': 'OpenFlow13'} s5: lo:127.0.0.1,s5-eth1:None,s5-eth2:None pid=5472>
<RemoteController{'ip': '127.0.0.1'} c0: 127.0.0.1:6653 pid=5441>
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5
h2 -> h1 h3 h4 h5
h3 -> h1 h2 h4 h5
h4 -> h1 h2 h3 h5
h5 -> h1 h2 h3 h4
*** Results: 0% dropped (20/20 received)
```

- **Information sur le nombre de OpenVswitchs**

```
curl -X GET http://localhost:8080/stats/switches
```

identifiez le nombre de switchs et

- **Information sur les caractéristiques du switch 4**

```
curl -X GET http://localhost:8080/stats/desc/4
```

testez avec les switchs 1 et 3

- **Affichez le contenu de la « Flow Table » du siwtch 1**

```
curl -X GET http://localhost:8080/stats/flow/1
```

Identification des 5 switchs

```
root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/switches
[1, 5, 4, 2, 3]root@ryu:/home/toto#
```

Information du switch 4, du switch 1 et du switch 3

```
[1, 5, 4, 2, 3]root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/desc/4
{"4": {"mfr_desc": "Nicira, Inc.", "hw_desc": "Open vSwitch", "sw_desc": "2.17.8", "serial_num": "None", "dp_desc":
 "s4"}}root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/desc/1
{"1": {"mfr_desc": "Nicira, Inc.", "hw_desc": "Open vSwitch", "sw_desc": "2.17.8", "serial_num": "None", "dp_desc":
 "s1"}}root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/desc/3
{"3": {"mfr_desc": "Nicira, Inc.", "hw_desc": "Open vSwitch", "sw_desc": "2.17.8", "serial_num": "None", "dp_desc":
 "s3"}}root@ryu:/home/toto#
```

Wagner Nicolas
Losser Julien
RT 331 ROM

Le contenu de la FLOW TABLE du switch 1:

```
to# curl -X GET http://localhost:8080/stats/flow/1
{"1": [{"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 238, "duration_sec": 1270, "duration_nsec": 917000000, "packet_count": 3, "length": 104, "flags": 0, "actions": ["OUTPUT:1"], "match": {"in_port": 2, "dl_src": "00:00:00:00:00:02", "dl_dst": "00:00:00:00:00:01"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 140, "duration_sec": 1270, "duration_nsec": 914000000, "packet_count": 2, "length": 104, "flags": 0, "actions": ["OUTPUT:2"], "match": {"in_port": 1, "dl_src": "00:00:00:00:00:01", "dl_dst": "00:00:00:00:00:02"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 238, "duration_sec": 1270, "duration_nsec": 781000000, "packet_count": 3, "length": 104, "flags": 0, "actions": ["OUTPUT:1"], "match": {"in_port": 2, "dl_src": "00:00:00:00:00:03", "dl_dst": "00:00:00:00:00:01"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 140, "duration_sec": 1270, "duration_nsec": 778000000, "packet_count": 2, "length": 104, "flags": 0, "actions": ["OUTPUT:2"], "match": {"in_port": 1, "dl_src": "00:00:00:00:00:01", "dl_dst": "00:00:00:00:00:03"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 238, "duration_sec": 1270, "duration_nsec": 588000000, "packet_count": 3, "length": 104, "flags": 0, "actions": ["OUTPUT:1"], "match": {"in_port": 2, "dl_src": "00:00:00:00:00:04", "dl_dst": "00:00:00:00:00:01"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 140, "duration_sec": 1270, "duration_nsec": 557000000, "packet_count": 2, "length": 104, "flags": 0, "actions": ["OUTPUT:2"], "match": {"in_port": 1, "dl_src": "00:00:00:00:00:01", "dl_dst": "00:00:00:00:00:04"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 238, "duration_sec": 1270, "duration_nsec": 462000000, "packet_count": 3, "length": 104, "flags": 0, "actions": ["OUTPUT:1"], "match": {"in_port": 2, "dl_src": "00:00:00:00:00:05", "dl_dst": "00:00:00:00:00:01"}, "table_id": 0}, {"priority": 1, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "byte_count": 140, "duration_sec": 1270, "duration_nsec": 461000000,root@ryu:/home/toroot@ryu:/home/toroot@ryu:/home/toroot@ryu:/home/toto#
```

La même commande avec l'argument “| jq”.

```
root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/flow/1 | jq
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100  2735  100  2735    0     0   229k      0 --:--:-- --:--:-- --:--:--  242k
{
  "1": [
    {
      "priority": 1,
      "cookie": 0,
      "idle_timeout": 0,
      "hard_timeout": 0,
      "byte_count": 238,
      "duration_sec": 1398,
      "duration_nsec": 952000000,
      "packet_count": 3,
      "length": 104,
      "flags": 0,
      "actions": [
        "OUTPUT:1"
      ],
      "match": {
        "in_port": 2,
        "dl_src": "00:00:00:00:00:02",
        "dl_dst": "00:00:00:00:00:01"
      },
      "table_id": 0
    },
    {
      "priority": 1,
      "cookie": 0,
```

Le contenu de la FLOW TABLE du switch 1 avec la commande ovs-ofctl:

```
root@ryu:/home/toto# ovs-ofctl -O OpenFlow13 dump-flows s1
 cookie=0x0, duration=1580.444s, table=0, n_packets=3, n_bytes=238, priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:02,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=1580.441s, table=0, n_packets=2, n_bytes=140, priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2"
 cookie=0x0, duration=1580.308s, table=0, n_packets=3, n_bytes=238, priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:03,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=1580.305s, table=0, n_packets=2, n_bytes=140, priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:03 actions=output:"s1-eth2"
 cookie=0x0, duration=1580.115s, table=0, n_packets=3, n_bytes=238, priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:04,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=1580.084s, table=0, n_packets=2, n_bytes=140, priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:04 actions=output:"s1-eth2"
 cookie=0x0, duration=1579.989s, table=0, n_packets=3, n_bytes=238, priority=1,in_port="s1-eth2",dl_src=00:00:00:00:00:05,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x0, duration=1579.988s, table=0, n_packets=2, n_bytes=140, priority=1,in_port="s1-eth1",dl_src=00:00:00:00:00:01,dl_dst=00:00:00:00:00:05 actions=output:"s1-eth2"
 cookie=0x0, duration=1587.309s, table=0, n_packets=203, n_bytes=20378, priority=0 actions=CONTROLLER:65535
root@ryu:/home/toto#
```

Wagner Nicolas
Losser Julien
RT 331 ROM

- Openflow switch as a Hub

A présent on relance « ryu-manager » avec « ryu.app.ofctl_rest » et mininet avec les mêmes options:

```
^C(ryu-python3.9-venv) root@ryu:/home/toto# ryu-manager ryu.app.ofctl_rest
loading app ryu.app.ofctl_rest
loading app ryu.controller.ofp_handler
instantiating app None of DPSet
creating context dpset
creating context wsgi
instantiating app ryu.app.ofctl_rest of RestStatsApi
instantiating app ryu.controller.ofp_handler of OFPHandler
(5733) wsgi starting up on http://0.0.0.0:8080
```

On se rend compte que les pings entre h1 et h2 ne passent plus comme demandé.

```
mininet> h1 ping h2
PING 192.168.1.2 (192.168.1.2) 56(84) bytes of data.
From 192.168.1.1 icmp_seq=1 Destination Host Unreachable
From 192.168.1.1 icmp_seq=2 Destination Host Unreachable
From 192.168.1.1 icmp_seq=3 Destination Host Unreachable
^C
--- 192.168.1.2 ping statistics ---
6 packets transmitted, 0 received, +3 errors, 100% packet loss, time 5106ms
pipe 4
mininet>
```

On s'aperçoit que les tables des switches sont bien vides, c'est pour cela que les pings ne fonctionnent pas.

```
root@ryu:/home/toto# ovs-ofctl -O OpenFlow13 dump-flows s1
root@ryu:/home/toto# ovs-ofctl -O OpenFlow13 dump-flows s2
root@ryu:/home/toto#
```

```
 GNU nano 6.2
{
        "dpid"= 1,
        "table_id": 0,
        "idle_timeout": 0,
        "hard_timeout": 0,
        "priority": 100,
        "match":{
        },
        "actions":[
            {
                "types":"OUTPUT"
                "port": 4294967291
            }
        ]
}
```

hub.json
« dpid » : l'identité du switch
« table_id » l'identité de la Flow table, par défaut c'est « 0 »
« port » : c'est le numéro de port. Dans notre cas ce numéro représente l'action de Flooding (hub)

Wagner Nicolas
Losser Julien
RT 331 ROM

Ici on ajoute la table flow du switch 1:

```
root@ryu:/home/toto# curl -X POST http://localhost:8080/stats/flowentry/add -d '@hub_flow.json'
root@ryu:/home/toto#
```

```
(2695) accepted ('127.0.0.1', 48170)
127.0.0.1 - - [25/Jan/2024 11:07:52] "POST /stats/flowentry/add HTTP/1.1" 200 115 0.016403
```

```
root@ryu:/home/toto# curl -X GET http://localhost:8080/stats/flow/1
{"1": [{"priority": 100, "cookie": 0, "idle_timeout": 0, "hard_timeout": 0, "actions": ["OUTPUT:FLOOD"
], "match": {}, "byte_count": 247, "duration_sec": 101, "duration_nsec": 721000000, "packet_count": 3,
 "table_id": 0, "length": 80}]}root@ryu:/home/toto#
```

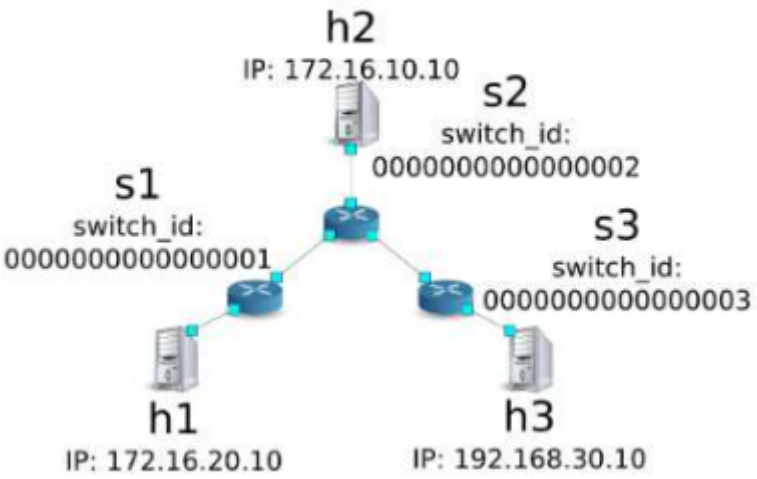
Maintenant que nous avons rempli les tables des switchs, tous les pings fonctionnent.

```
mininet> pingall
*** Ping: testing ping reachability
h1 -> h2 h3 h4 h5
h2 -> h1 h3 h4 h5
h3 -> h1 h2 h4 h5
h4 -> h1 h2 h3 h5
h5 -> h1 h2 h3 h4
*** Results: 0% dropped (20/20 received)
mininet>
```

Maintenant nous allons administrer un réseau IP dont la configuration est la suivante :

- Routage entre réseaux

Pour appréhender le routage statique avec REST API, on utilise la topologie suivante :



Wagner Nicolas
Losser Julien
RT 331 ROM

Lancez la topologie avec la commande suivante :

```
root@ryu:/home/toto# mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13
 --topo=linear,3
*** Creating network
*** Adding controller
Connecting to remote controller at 127.0.0.1:6653
*** Adding hosts:
h1 h2 h3
*** Adding switches:
s1 s2 s3
*** Adding links:
(h1, s1) (h2, s2) (h3, s3) (s2, s1) (s3, s2)
*** Configuring hosts
h1 h2 h3
*** Starting controller
c0
*** Starting 3 switches
s1 s2 s3 ...
*** Starting CLI:
mininet>
```

```
mininet> dump
<Host h1: h1-eth0:10.0.0.1 pid=3139>
<Host h2: h2-eth0:10.0.0.2 pid=3141>
<Host h3: h3-eth0:10.0.0.3 pid=3143>
<OVSSwitch{'protocols': 'OpenFlow13'} s1: lo:127.0.0.1,s1-eth1:None,s1-eth2:None pid=3148>
<OVSSwitch{'protocols': 'OpenFlow13'} s2: lo:127.0.0.1,s2-eth1:None,s2-eth2:None,s2-eth3:None pid
=3151>
<OVSSwitch{'protocols': 'OpenFlow13'} s3: lo:127.0.0.1,s3-eth1:None,s3-eth2:None pid=3154>
<RemoteController{'ip': '127.0.0.1'} c0: 127.0.0.1:6653 pid=3133>
mininet>
```

Changement d'adresse ip sur H1

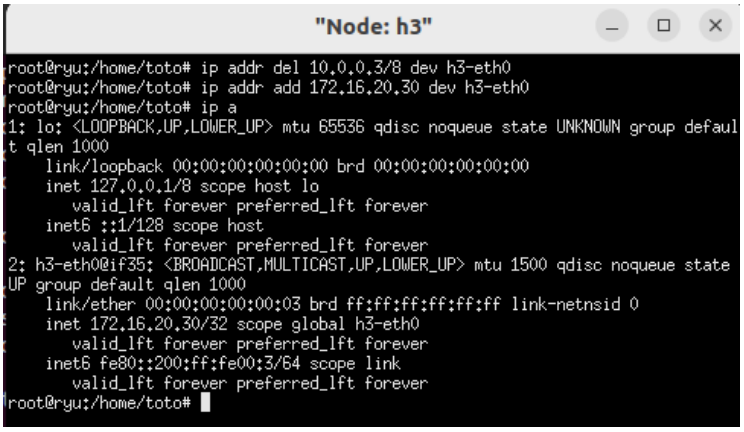
```
"Node: h1"
root@ryu:/home/toto# ip addr del 10.0.0.1/8 dev h1-eth0
root@ryu:/home/toto# ip addr add 172.16.20.10/24 dev h1-eth0
root@ryu:/home/toto# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
len 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: h1-eth0@if33: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP group default qlen 1000
    link/ether 00:00:00:00:00:01 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.16.20.10/24 scope global h1-eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::200:ff:fe00:1/64 scope link
       valid_lft forever preferred_lft forever
root@ryu:/home/toto#
```

Changement d'adresse ip sur H2

```
"Node: h2"
root@ryu:/home/toto# ip addr del 10.0.0.2/8 dev h2-eth0
root@ryu:/home/toto# ip addr add 172.16.20.20/24 dev h2-eth0
root@ryu:/home/toto# ip a
1: lo: <LOOPBACK,UP,LOWER_UP> mtu 65536 qdisc noqueue state UNKNOWN group defaul
t qlen 1000
    link/loopback 00:00:00:00:00:00 brd 00:00:00:00:00:00
    inet 127.0.0.1/8 scope host lo
       valid_lft forever preferred_lft forever
    inet6 ::1/128 scope host
       valid_lft forever preferred_lft forever
2: h2-eth0@if34: <BROADCAST,MULTICAST,UP,LOWER_UP> mtu 1500 qdisc noqueue state
UP group default qlen 1000
    link/ether 00:00:00:00:00:02 brd ff:ff:ff:ff:ff:ff link-netnsid 0
    inet 172.16.20.20/24 scope global h2-eth0
       valid_lft forever preferred_lft forever
    inet6 fe80::200:ff:fe00:2/64 scope link
       valid_lft forever preferred_lft forever
root@ryu:/home/toto#
```

Wagner Nicolas
Losser Julien
RT 331 ROM

Changement d'adresse ip sur H3



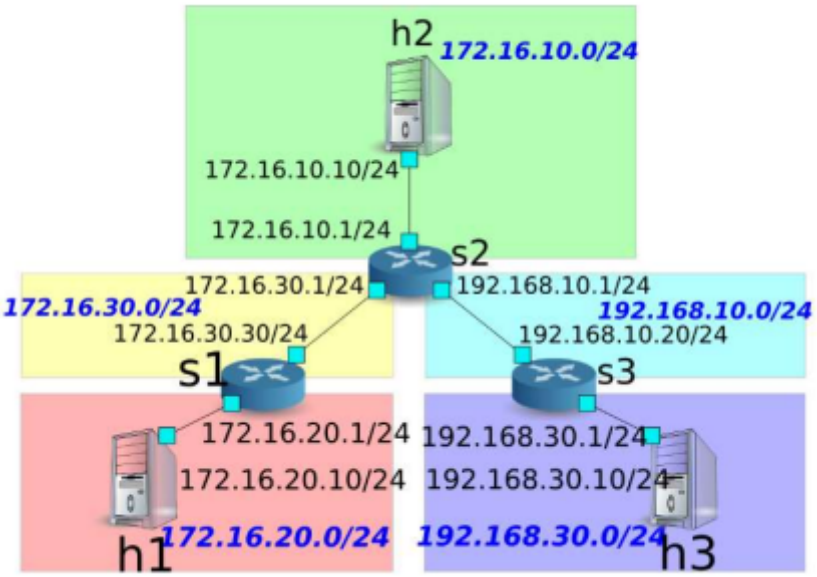
RYU avec API REST

```
^C(ryu-python3.9-venv) root@ryu:/home/toto# ryu-manager ryu.app.rest_router
loading app ryu.app.rest_router
loading app ryu.controller.ofp_handler
instantiating app None of DPSet
creating context dpset
creating context wsgi
instantiating app ryu.app.rest_router of RestRouterAPI
instantiating app ryu.controller.ofp_handler of OFPHandler
(3337) wsgi starting up on http://0.0.0.0:8080
[RT][INFO] switch_id=0000000000000001: Set SW config for TTL error packet in.
[RT][INFO] switch_id=0000000000000001: Set ARP handling (packet in) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000001: Set L2 switching (normal) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000001: Set default route (drop) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000001: Start cyclic routing table update.
[RT][INFO] switch_id=0000000000000001: Join as router.
[RT][INFO] switch_id=0000000000000003: Set SW config for TTL error packet in.
[RT][INFO] switch_id=0000000000000003: Set ARP handling (packet in) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000003: Set L2 switching (normal) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000003: Set default route (drop) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000003: Start cyclic routing table update.
[RT][INFO] switch_id=0000000000000003: Join as router.
[RT][INFO] switch_id=0000000000000002: Set SW config for TTL error packet in.
[RT][INFO] switch_id=0000000000000002: Set ARP handling (packet in) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000002: Set L2 switching (normal) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000002: Set default route (drop) flow [cookie=0x0]
[RT][INFO] switch_id=0000000000000002: Start cyclic routing table update.
[RT][INFO] switch_id=0000000000000002: Join as router.
```

Wagner Nicolas
Losser Julien
RT 331 ROM

- Configuration des interfaces routeurs

Notre but est de mettre en œuvre le plan d'adressage suivant



Configurations des interfaces des 3 routeurs.

```
root@ryu:/home/toto# curl -X POST -d '{"address":"172.16.20.1/24"}' http://localhost:8080/router/0000000000000001
[{"switch_id": "0000000000000001", "command_result": [{"result": "success", "details": "Add addresrrrrrrrorrrrrrr
root@ryu:/home/toto#
```

```
root@ryu:/home/toto# curl -X POST -d '{"address":"172.16.20.1/24"}' http://localhost:8080/router/0000000000000001
[{"switch_id": "0000000000000001", "command_result": [{"result": "success", "details": "Add addresrrrrrrrorrrrrrr
oot@ryu:/home/toto# curl -X POST -d '{"address":"172.16.30.30/24"}' http://localhost:8080/router/0000000000000001
[{"switch_id": "0000000000000001", "command_result": [{"result": "success", "details": "Add address [address_id=2r
oot@ryu:/home/toto# curl -X POST -d '{"address":"172.16.10.1/24"}' http://localhost:8080/router/0000000000000002[{
"switch_id": "0000000000000002", "command_result": [{"result": "success", "details": "Add address [address_id=1roo
t@ryu:/home/toto# curl -X POST -d '{"address":"172.16.30.1/24"}' http://localhost:8080/router/0000000000000002[{"s
witch_id": "0000000000000002", "command_result": [{"result": "success", "details": "Add address [address_id=2root@
ryu:/home/toto# curl -X POST -d '{"address":"192.168.10.1/24"}' http://localhost:8080/router/0000000000000002
root@ryu:/home/toto# curl -X POST -d '{"address":"192.168.30.1/24"}' http://localhost:8080/router/0000000000000003
[{"switch_id": "0000000000000003", "command_result": [{"result": "success", "details": "Add address [address_id=1]
root@ryu:/home/toto# curl -X POST -d '{"address":"192.168.10.20/24"}' http://localhost:8080/router/000000000000000
3
[{"switch_id": "0000000000000003", "command_result": [{"result": "success", "details": "Add address [address_id=2]
"}]}]root@ryu:/home/toto#
```

Wagner Nicolas
Losser Julien
RT 331 ROM

- Configuration des machines

Dans cette section, il faut indiquer à chaque machines sa passerelle pour lui permettre de communiquer avec les autres réseaux IP

```
ip route add default via @IP_gw_du_réseau_en_question
```

"Node: h1"

```
root@ryu:/home/toto# ip route add default via 172.16.20.1
```

```
mininet> h1 ping 172.16.20.1
PING 172.16.20.1 (172.16.20.1) 56(84) bytes of data.
64 bytes from 172.16.20.1: icmp_seq=1 ttl=64 time=7.84 ms
64 bytes from 172.16.20.1: icmp_seq=2 ttl=64 time=3.80 ms
^C
--- 172.16.20.1 ping statistics ---
2 packets transmitted, 2 received, 0% packet loss, time 1004ms
rtt min/avg/max/mdev = 3.802/5.820/7.839/2.018 ms
mininet>
```

H1 réussit à ping son routeur après la configuration de la gateway.

- Gestion des tables de routages des routeurs

Pour permettre une communication complète de notre réseau ; pour cela on configure des route par défaut pour chaque routeur

Définir le routeur 2 comme route par défaut du routeur 1

```
curl -X POST -d '{"gateway": "172.16.30.1"}'
http://localhost:8080/router/0000000000000001
```

Définir le routeur 1 comme route par défaut du routeur 2

```
curl -X POST -d '{"gateway": "172.16.30.30"}'
http://localhost:8080/router/0000000000000002
```

Définir le routeur 2 comme route par défaut du routeur 3

```
curl -X POST -d '{"gateway": "192.168.10.1"}'
http://localhost:8080/router/0000000000000003
```

```
curl -X POST -d '{"destination": "192.168.30.0/24", "gateway": "192.168.10.20"}'
http://localhost:8080/router/0000000000000002
```

```
curl http://localhost:8080/router/0000000000000001
```

Après la configuration des routeurs et des machines hx, nous pouvons pings les gateway et les autres machines.

Wagner Nicolas
Losser Julien
RT 331 ROM