**CCGC-5002 Lab 4-2- RYU OFCTL REST API**

Compile solutionsinto a single word/pdf file and upload in the Lab 4 folder under Assignments tab of Blackboard. Lab 4 has got 2 parts – Lab 4-1 Introduction to Ryu and Lab 4-2 OFCTL REST

## 1. **Introduction**

OFCTL - Openflow control using REST Interface.

ryu.app.ofctl\_rest application provides REST APIs for configure/update/retrieve the switch flows /stats etc. This application helps you debug your application and get various statistics.

This application supports OpenFlow version 1.0, 1.2, 1.3, 1.4 and 1.5.

This application is not switch application.

## 2.**Example - Get Statistics**

### Objective

Retrive the Switch Details/Stats,

### Steps

1. Run the mininet simple topology

*sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=linear,4*

1. Run the RYU OFCTL REST & Simple Switch application

*ryu-manager ryu.app.simple\_switch\_13 ryu.app.ofctl\_rest*

1. do pingall from mininet. and check the ovs flows
2. Get All switches

tanvir@tanvir-vm:~$  curl -X GET http://localhost:8080/stats/switches

[1, 3, 2, 4] The switches may be discovered out of order.

1. Get the desc stats of the switch which specified with Datapath ID in URI.

tanvir@tanvir-vm:~$ curl -X GET http://localhost:8080/stats/desc/1

{"1": {"dp\_desc": "None", "sw\_desc": "2.9.2", "hw\_desc": "Open vSwitch", "serial\_num": "None", "mfr\_desc": "Nicira, Inc."}}

1. Get all flows stats of the switch which specified with Datapath ID in URI.

tanvir@tanvir-vm:~$ curl -X GET http://localhost:8080/stats/flow/1

{"1": [{"actions": ["OUTPUT:1"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 3, "hard\_timeout": 0, "byte\_count": 238, "duration\_sec": 213, "duration\_nsec": 654000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:01", "dl\_src": "00:00:00:00:00:02", "in\_port": 2}}, {"actions": ["OUTPUT:2"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 2, "hard\_timeout": 0, "byte\_count": 140, "duration\_sec": 213, "duration\_nsec": 653000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:02", "dl\_src": "00:00:00:00:00:01", "in\_port": 1}}, {"actions": ["OUTPUT:1"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 3, "hard\_timeout": 0, "byte\_count": 238, "duration\_sec": 213, "duration\_nsec": 634000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:01", "dl\_src": "00:00:00:00:00:03", "in\_port": 2}}, {"actions": ["OUTPUT:2"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 2, "hard\_timeout": 0, "byte\_count": 140, "duration\_sec": 213, "duration\_nsec": 632000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:03", "dl\_src": "00:00:00:00:00:01", "in\_port": 1}}, {"actions": ["OUTPUT:1"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 3, "hard\_timeout": 0, "byte\_count": 238, "duration\_sec": 213, "duration\_nsec": 614000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:01", "dl\_src": "00:00:00:00:00:04", "in\_port": 2}}, {"actions": ["OUTPUT:2"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 2, "hard\_timeout": 0, "byte\_count": 140, "duration\_sec": 213, "duration\_nsec": 612000000, "priority": 1, "length": 104, "flags": 0, "table\_id": 0, "match": {"dl\_dst": "00:00:00:00:00:04", "dl\_src": "00:00:00:00:00:01", "in\_port": 1}}, {"actions": ["OUTPUT:CONTROLLER"], "idle\_timeout": 0, "cookie": 0, "packet\_count": 129, "hard\_timeout": 0, "byte\_count": 15332, "duration\_sec": 215, "duration\_nsec": 993000000, "priority": 0, "length": 80, "flags": 0, "table\_id": 0, "match": {}}]}

1. Get Port stats

tanvir@tanvir-vm:~$ curl -X GET http://localhost:8080/stats/port/1

{"1": [{"tx\_dropped": 0, "rx\_packets": 0, "rx\_crc\_err": 0, "tx\_bytes": 0, "rx\_dropped": 131, "port\_no": "LOCAL", "rx\_over\_err": 0, "rx\_frame\_err": 0, "rx\_bytes": 0, "tx\_errors": 0, "duration\_nsec": 436000000, "collisions": 0, "duration\_sec": 287, "rx\_errors": 0, "tx\_packets": 0}, {"tx\_dropped": 0, "rx\_packets": 25, "rx\_crc\_err": 0, "tx\_bytes": 20026, "rx\_dropped": 0, "port\_no": 1, "rx\_over\_err": 0, "rx\_frame\_err": 0, "rx\_bytes": 1906, "tx\_errors": 0, "duration\_nsec": 449000000, "collisions": 0, "duration\_sec": 287, "rx\_errors": 0, "tx\_packets": 159}, {"tx\_dropped": 0, "rx\_packets": 131, "rx\_crc\_err": 0, "tx\_bytes": 5923, "rx\_dropped": 0, "port\_no": 2, "rx\_over\_err": 0, "rx\_frame\_err": 0, "rx\_bytes": 15895, "tx\_errors": 0, "duration\_nsec": 449000000, "collisions": 0, "duration\_sec": 287, "rx\_errors": 0, "tx\_packets": 52}]}

## 

## \*\*\*\*\*\*\*\*\*\*\*\*\*\*take screenshot of the above screen.

## 3. **Example - Hub**

### Objective

Enable the Openflow switch will act as HUB.

### Steps

1. Run the mininet simple topology

*sudo mn --controller=remote,ip=127.0.0.1 --mac --switch=ovsk,protocols=OpenFlow13 --topo=single,4*

1. Run the RYU OFCTL REST Api application

*ryu-manager ryu.app.ofctl\_rest*

1. do pingall from mininet. and check the ovs flows

Flow table will be empty and ping fails.

1. Add a "Flood" flow using REST API.

cat hub\_flow.json

{

   "dpid": 1,

   "table\_id": 0,

   "idle\_timeout": 0,

   "hard\_timeout": 0,

   "priority": 100,

   "match":{

  },

   "actions":[

      {

           "type":"OUTPUT",

           "port": 4294967291

      }

  ]

}

Here Port Number 4294967291 (0xFFFFFFFB) means OFP\_FLOOD Port.

**API command:**

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@hub\_flow.json'*

5.check the ovs flows

tanvir@tanvir-vm:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1

cookie=0x0, duration=3.536s, table=0, n\_packets=0, n\_bytes=0, idle\_timeout=0, hard\_timeout=0, priority=100 actions=FLOOD

tanvir@tanvir-vm:~$

6.Now, perform ping between hosts in mininet

mininet> h1 ping h2

PING 10.0.0.2 (10.0.0.2) 56(84) bytes of data.

64 bytes from 10.0.0.2: icmp\_seq=1 ttl=64 time=1.11 ms

64 bytes from 10.0.0.2: icmp\_seq=2 ttl=64 time=0.189 ms

64 bytes from 10.0.0.2: icmp\_seq=3 ttl=64 time=0.153 ms

^C

--- 10.0.0.2 ping statistics ---

3 packets transmitted, 3 received, 0% packet loss, time 2003ms

rtt min/avg/max/mdev = 0.153/0.485/1.113/0.444 ms

mininet>

​

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*take screenshot of the above screen.

## 4. **Example - Switch**

### Objective

Enable the Openflow switch will act as Layer2 Mac based Switch.

### Steps

1. Use the same procedure as Example1 for running mininet and ryu ofctl application.
2. Add the Broadcast flow(for L2 Broadcast/ARP packet)

* flow1: dst-mac(ff:ff:ff:ff:ff:ff) - output FLOOD

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@switch\_arp.json'*

1. Add the destination mac specific flows

* flow1: dst-mac(00:00:00:00:00:01) - output port 1
* flow2: dst-mac(00:00:00:00:00:02) - output port 2
* flow3: dst-mac(00:00:00:00:00:03) - output port 3
* flow4: dst-mac(00:00:00:00:00:04) - output port 4

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@switch\_flow1.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@switch\_flow2.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@switch\_flow3.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@switch\_flow4.json'*

1. Check the flows and do pingall from mininet

tanvir@tanvir-vm:~$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1

cookie=0x0, duration=307.210s, table=0, n\_packets=26, n\_bytes=2212, priority=100,dl\_dst=00:00:00:00:00:02 actions=output:"s1-eth2"

cookie=0x0, duration=304.663s, table=0, n\_packets=10, n\_bytes=756, priority=100,dl\_dst=00:00:00:00:00:03 actions=output:"s1-eth3"

cookie=0x0, duration=300.185s, table=0, n\_packets=27, n\_bytes=2254, priority=100,dl\_dst=00:00:00:00:00:01 actions=output:"s1-eth1"

cookie=0x0, duration=260.081s, table=0, n\_packets=9, n\_bytes=714, priority=100,dl\_dst=00:00:00:00:00:04 actions=output:"s1-eth4"

cookie=0x0, duration=39.302s, table=0, n\_packets=6, n\_bytes=252, priority=100,dl\_dst=ff:ff:ff:ff:ff:ff actions=FLOOD

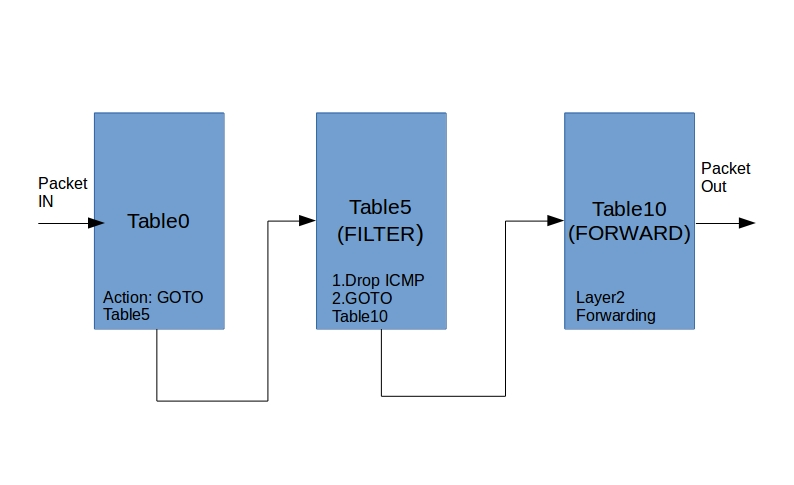
tanvir@tanvir-vm:~$

\*\*\*\*\*\*\*\*\*\*\*\*\*\*take screenshot of the above screen.

## 5. **Example - Multitable**

### Objective

Demonstrate the Multitables (pipeline processing) concepts, and ACL Rules(Drop).



We will create two tables

* FILTER TABLE(Table 5) and FORWARD TABLE(Table 10).
* FILTER TABLE : will add packet filter rules (ACL rules). Block ICMP traffic for h4.
* FORWARD TABLE: our forwarding rules (L2 switch)

High level Steps:

1. In Table 0(default), Add the Match all Flow with action GO TO - Filter Table.
2. In Table 5(Filter Table), Add the filter rules(icmp block for h4) in higher priority, Add the Match All Flow with the Action GO TO - Forward Table
3. In Table 10(Forward Table), Add the L2 forwarding rules.

### Steps

1. Use the same procedure as Example1 for running mininet and ryu ofctl application.
2. Add the flows as below,

* Add flow in table0, forward all traffic in it to table 5
* Add the flow in table5, to go to table 10
* Add the forwarding flows in table10

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table0\_flow1.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table5\_flow1.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table10\_flow1.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table10\_flow2.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table10\_flow3.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table10\_flow4.json'*

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table10\_arp.json'*

1. Check the flows and do pingall in mininet.

tanvir@tanvir-vm:~/1$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1

cookie=0x0, duration=163.683s, table=0, n\_packets=99, n\_bytes=8274, priority=0 actions=goto\_table:5

cookie=0x0, duration=86.453s, table=5, n\_packets=97, n\_bytes=8134, priority=0 actions=goto\_table:10

cookie=0x0, duration=72.661s, table=10, n\_packets=24, n\_bytes=2016, priority=0,dl\_dst=00:00:00:00:00:01 actions=output:"s1-eth1"

cookie=0x0, duration=69.423s, table=10, n\_packets=23, n\_bytes=1974, priority=0,dl\_dst=00:00:00:00:00:02 actions=output:"s1-eth2"

cookie=0x0, duration=66.424s, table=10, n\_packets=22, n\_bytes=1932, priority=0,dl\_dst=00:00:00:00:00:03 actions=output:"s1-eth3"

cookie=0x0, duration=62.997s, table=10, n\_packets=21, n\_bytes=1890, priority=0,dl\_dst=00:00:00:00:00:04 actions=output:"s1-eth4"

cookie=0x0, duration=33.654s, table=10, n\_packets=6, n\_bytes=252, priority=0,dl\_dst=ff:ff:ff:ff:ff:ff actions=FLOOD

tanvir@tanvir-vm:~/1$

1. Add a ICMP Drop flow for h4

*curl -X POST*[*http://localhost:8080/stats/flowentry/add*](http://localhost:8080/stats/flowentry/add)*-d '@table5\_flow2.json'*

tanvir@tanvir-vm:~/1$ sudo ovs-ofctl -O OpenFlow13 dump-flows s1

cookie=0x0, duration=472.976s, table=0, n\_packets=99, n\_bytes=8274, priority=0 actions=goto\_table:5

cookie=0x0, duration=3.663s, table=5, n\_packets=0, n\_bytes=0, priority=100,icmp,nw\_dst=10.0.0.4 actions=drop

cookie=0x0, duration=395.746s, table=5, n\_packets=97, n\_bytes=8134, priority=0 actions=goto\_table:10

cookie=0x0, duration=381.954s, table=10, n\_packets=24, n\_bytes=2016, priority=0,dl\_dst=00:00:00:00:00:01 actions=output:"s1-eth1"

cookie=0x0, duration=378.716s, table=10, n\_packets=23, n\_bytes=1974, priority=0,dl\_dst=00:00:00:00:00:02 actions=output:"s1-eth2"

cookie=0x0, duration=375.717s, table=10, n\_packets=22, n\_bytes=1932, priority=0,dl\_dst=00:00:00:00:00:03 actions=output:"s1-eth3"

cookie=0x0, duration=372.290s, table=10, n\_packets=21, n\_bytes=1890, priority=0,dl\_dst=00:00:00:00:00:04 actions=output:"s1-eth4"

cookie=0x0, duration=342.947s, table=10, n\_packets=6, n\_bytes=252, priority=0,dl\_dst=ff:ff:ff:ff:ff:ff actions=FLOOD

tanvir@tanvir-vm:~/1$

Now h4 ping traffic will be blocked in FILTER table.

Try pingall from mininet prompt.

 \*\*\*\*\*\*\*\*\*\*\*\*\*\*take screenshot of the mininet screen showing h4 is blocked.

## 6. **References**:

* <https://ryu.readthedocs.io/en/latest/app/ofctl_rest.html#add-a-flow-entry>
* <https://github.com/osrg/ryu/blob/master/ryu/ofproto/ofproto_v1_3.py>
* <https://ryu.readthedocs.io/en/latest/app/ofctl_rest.html#description-of-match-and-actions>