下世代Network Slicing模組設計

實驗單元：Mininet搭配Controller 使用OpenFlow模擬SDN網路環境

國立中山大學 資訊工程系
授課教師：李宗南教授
教材編撰：曾國維

目錄
CONTENTS

01 實驗目標

實驗目標

- 實驗目標1：修課學生得以了解 SDN 的基本觀念及架構
- 實驗目標2：修課學生得以理解 Mininet的網路環境模擬以及SDN控制器Ryu的使用以及SDN應用程式Postman的使用。
- 實驗目標3：修課學生得以完成軟體定義網路環境進行實驗，及驗證OpenFlow執行。

02 實驗設備

實驗設備

- 硬體:
  - 電腦 : Ubuntu作業系統16.04

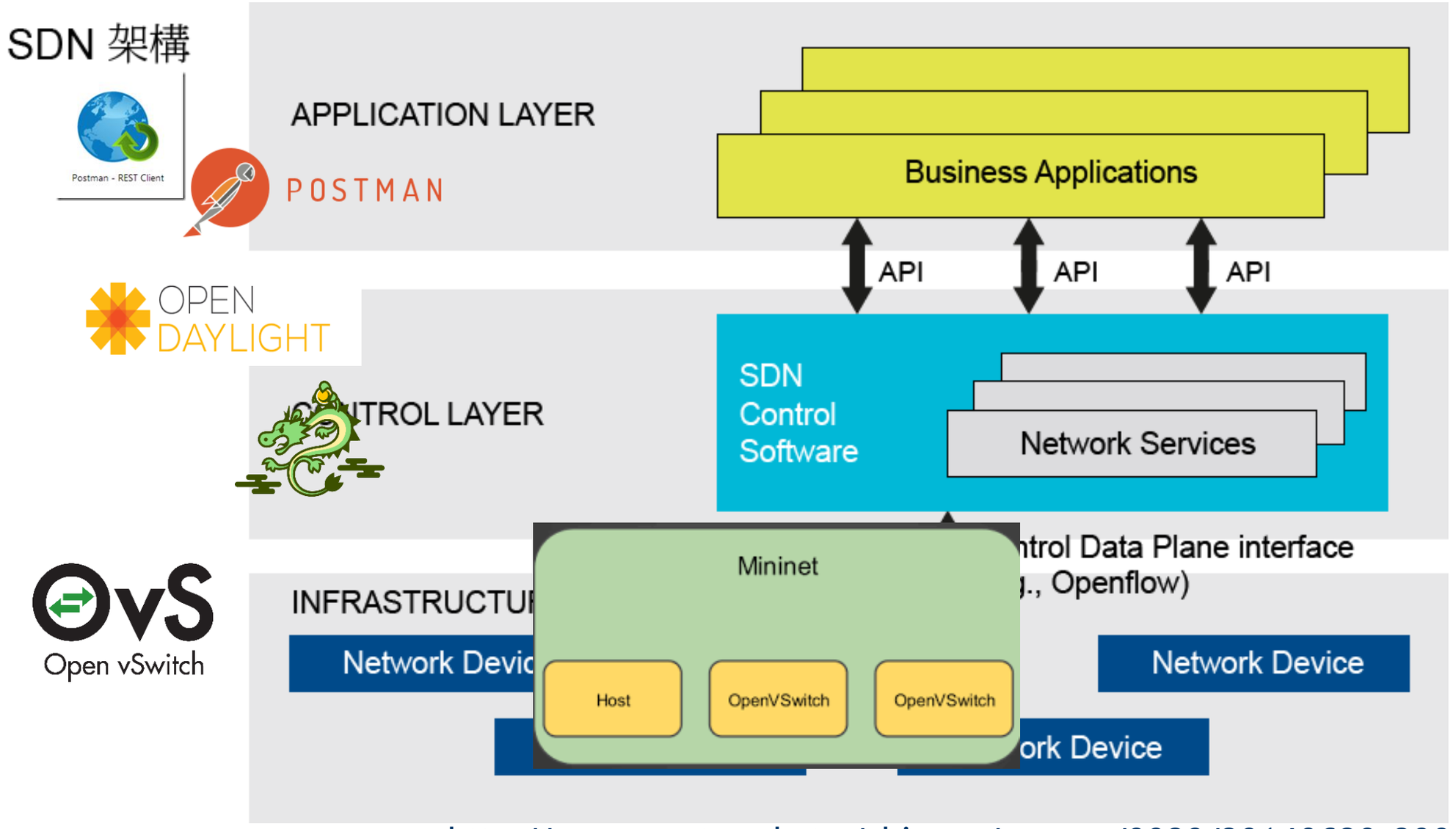
- 軟體
  - Mininet：2.3.0d4
  - Ryu： ryu-manager 4.30
  - Postman REST Client
  - gcc編譯器
  - vi/vim文字編輯器

03 SDN簡介與基本架構

SDN簡介與基本架構

- 一種新的網路架構。利用OpenFlow協定，將路由器及交換器傳輸封包中的控制與內容分離，以軟體方式實作。
  - 控制平面（Control plane）
  - 資料平面（Data plane)
- 這個架構可以讓網路管理員，在不更動硬體裝置的前提下，以中央控制方式，用程式重新規劃網路，為控制網路流量提供了新的方法，也提供了核心網路及應用創新的良好平台。

SDN簡介與基本架構

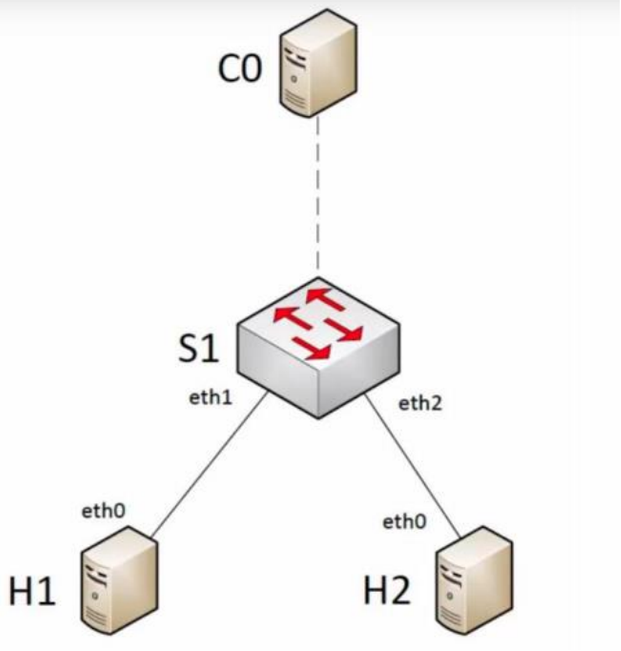


source: http://www.cc.ntu.edu.tw/chinese/epaper/0029/20140620_2908.html

04 軟體介紹

軟體介紹-Mininet(1)

- Mininet是一個可以透過一些虛擬終端機、路由器、交換器等連接創建虛擬網路拓樸的平台
- 兼容運行不同系統上（windows\linux\Mac OS）可以輕易的在自己的個人電腦中創作支援SDN的區域網路，造出的虛擬host以真實電腦般發送封包，以驗證實驗方法
- Mininet自帶switch、host、controller，同時，在mininet上可以安裝OpenvSwitch以及多種控制器
  （NOX\POX\RYU\Floodlight\OpenDaylight等）



軟體介紹-Mininet(2)

建立拓譜主要分成2種方式:

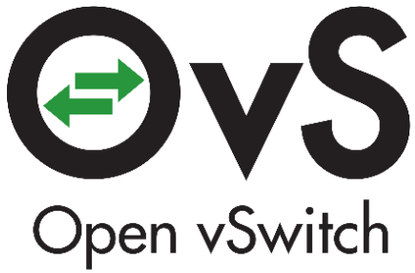
1.利用終端機指令下參數，建立拓譜。

2.撰寫python檔直接利用.py檔撰寫程式指令以及建立拓譜。

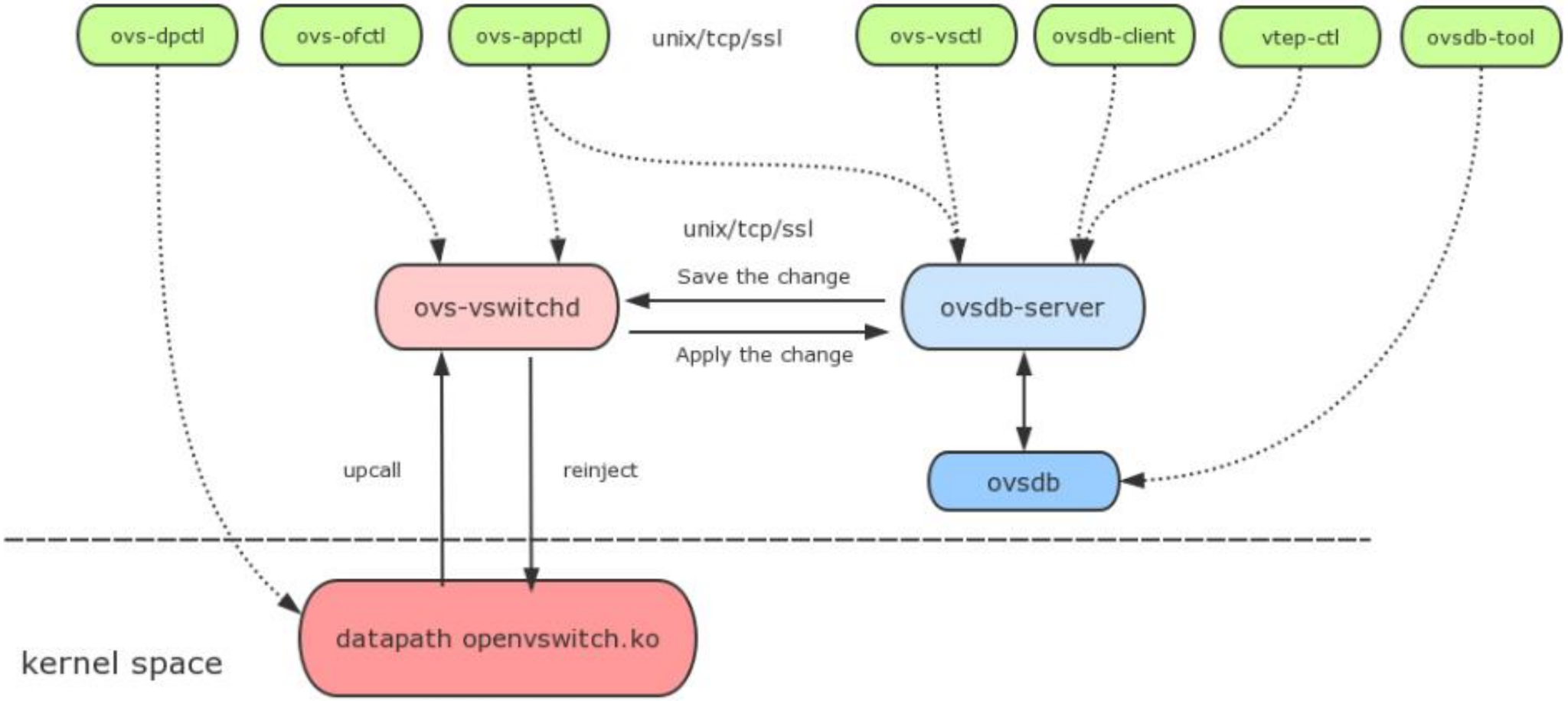
MININET

軟體介紹-OpenvSwitch(1)

- Open Networking Foundation(ONF)的開源計畫，顧名思義是一個virtual switch，它的目的是讓大規模網路透過可編程化來進行擴展及管理
- 可用於定義路徑, 切割網域，QoS或是流量監控，同時支持標準的管理接口服務和各項協議(sFlow,NetFlow,OpenFlow等)。
- Mininet中以OVS作為作為實現虛擬Switch與虛擬Host間連線



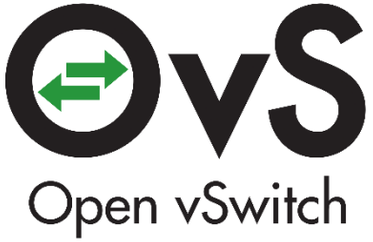
軟體介紹-OpenvSwitch(2)



參考文獻[4]

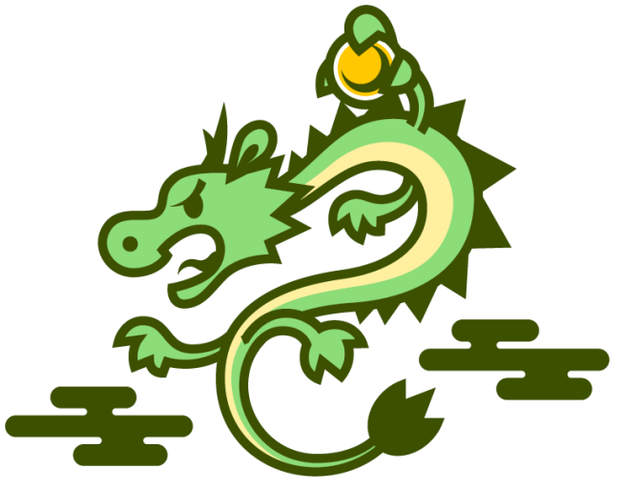
軟體介紹-OpenvSwitch(3)

1. ovs-dpctl： 管理ovs datapath，大部分資訊都是透過 netlink 反應。
2. ovs-vsctl : 對 ovsdb 操作，bridge,interface 新增，刪除，查詢，操作指令。
3. ovs-ofctl : openflow switch 管理工具，可以操作與 openflow 相關的設定。
4. ovs-appctl : ovs-vswithd 的管理工具，可以跟 ovs-vswitchd 程序溝通，ex:ofproto/trace 可用來追蹤封包flow。
5. ovs-vswitchd：主要模組，實現內核datapath upcall 處理以及ofproto查表，同时是dpdk datapath處理程序。



軟體介紹-Ryu

- Ryu is a component-based software defined networking framework.
- Ryu 是來自於日本 NTT 所開發以及設計，針對 SDN 的控制器開發框架（Framework）
- 包含了 OpenFlow（以及其他部分協定） Controller 的功能
- 使用Python進行開發



05 安裝及執行

安裝及執行

- Mininet
- Postman REST
- Mininet + Ryu - Task1, Task2, Task3
- Mininet + OpenDaylight – Task4

安裝及執行-Mininet

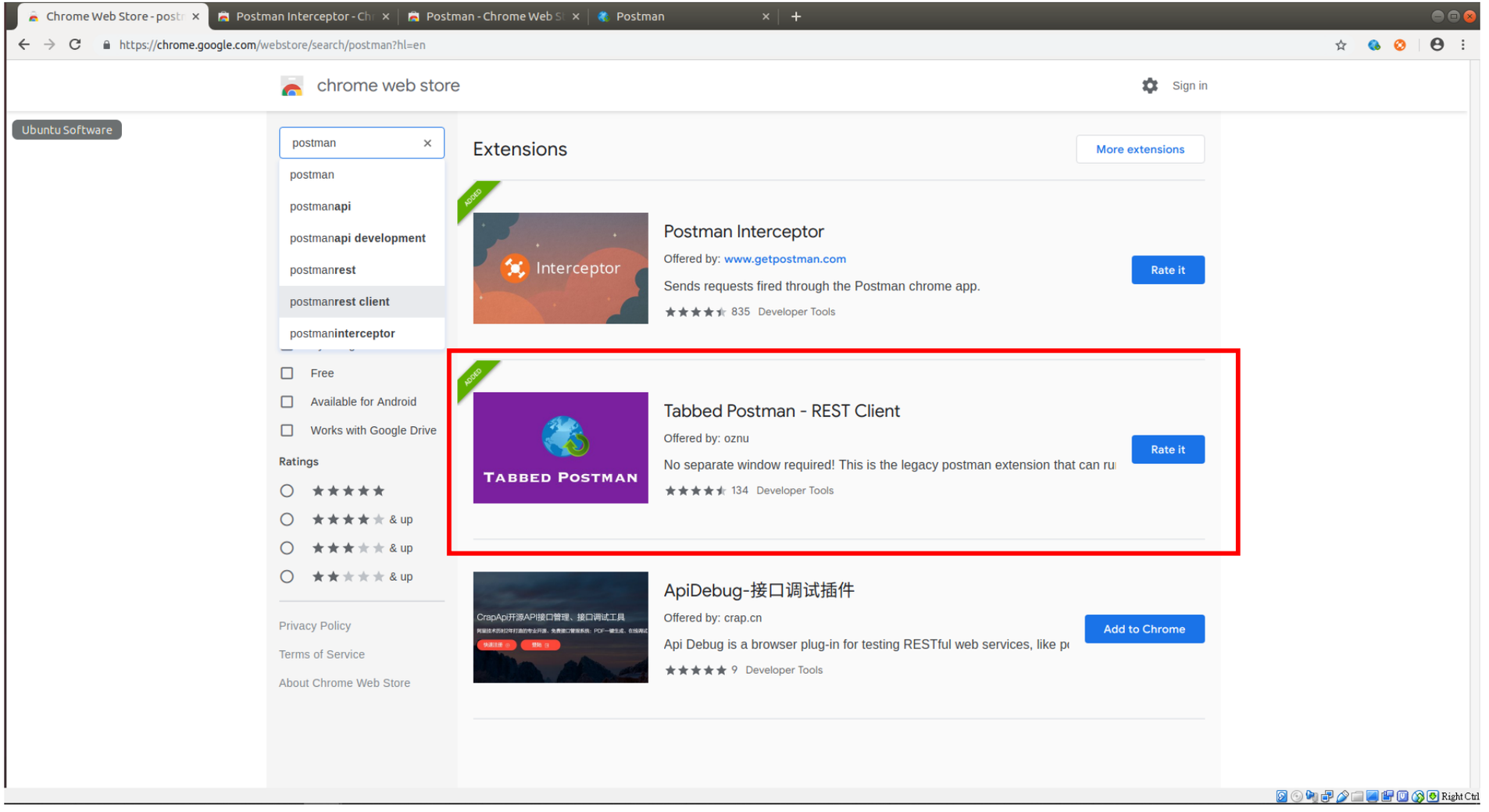
- $ git clone git://github.com/mininet/mininet
- $ cd mininet
- $ util/install.sh –a
- $ sudo mn

安裝及執行-Mininet Dump OVS Flows

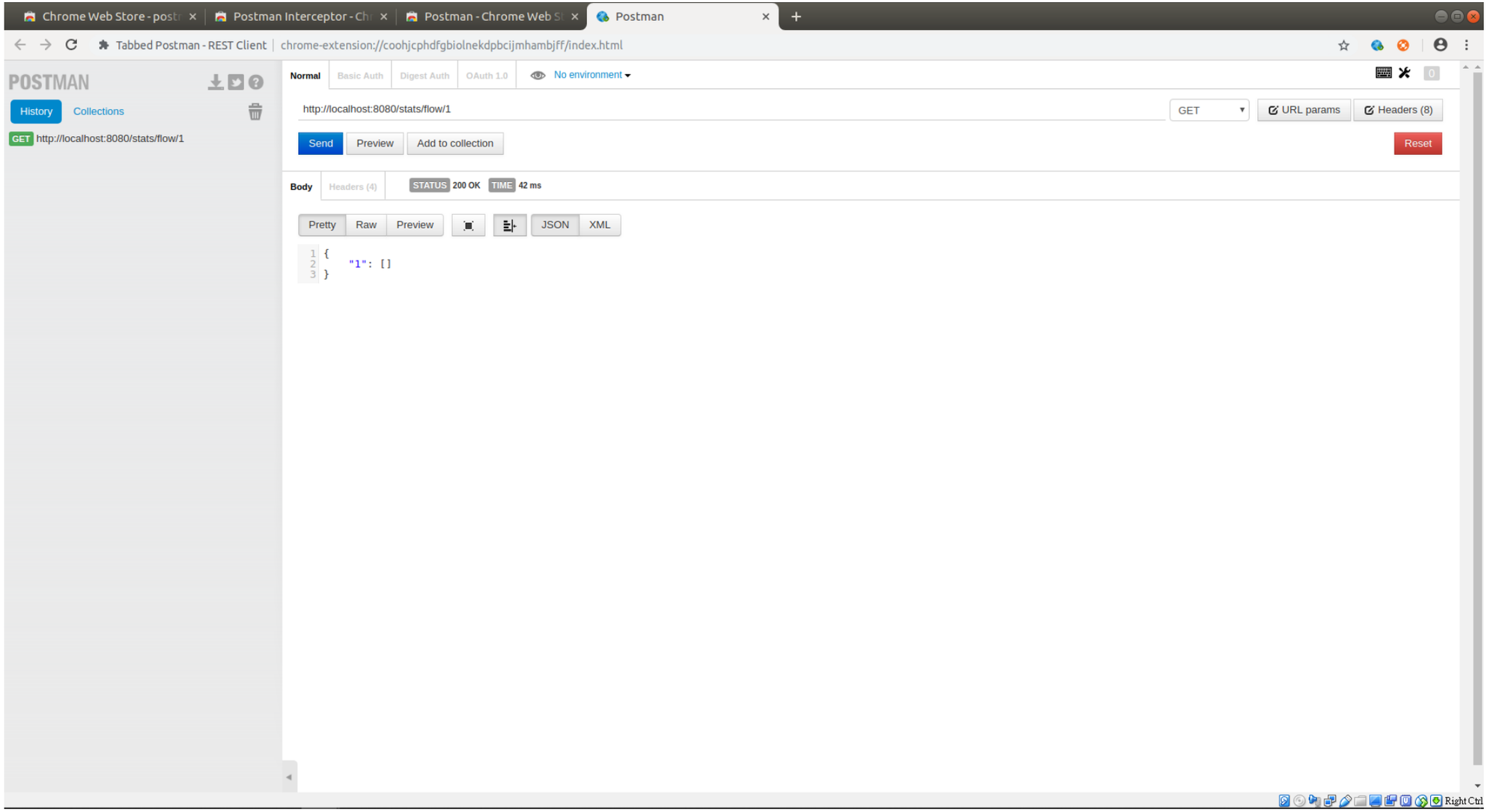
- Dump Flows
  - $ sh ovs-ofctl -O openflow13 dump-flows s1
- Dump Hidden Flows
  - $ sh ovs-appctl bridge/dump-flows s1

```
mininet> sh ovs-ofctl -O openflow13 dump-flows s1
 cookie=0x1, duration=30.901s, table=0, n_packets=5, n_bytes=378, hard_timeout=2000, send_flow_rem priority=11114,dl_dst=00:00:00:00:00:01 actions=output:"s1-eth1"
 cookie=0x1, duration=25.683s, table=0, n_packets=4, n_bytes=336, hard_timeout=2000, send_flow_rem priority=11113,dl_dst=00:00:00:00:00:02 actions=output:"s1-eth2"
 cookie=0x1, duration=23.584s, table=0, n_packets=1, n_bytes=42, hard_timeout=2000, send_flow_rem priority=11111,arp,arp_tpa=10.0.0.2 actions=output:"s1-eth2"
mininet> sh ovs-appctl bridge/dump-flows s1
duration=47s, n_packets=5, n_bytes=378, priority=11114,dl_dst=00:00:00:00:00:01,actions=output:1
duration=42s, n_packets=4, n_bytes=336, priority=11113,dl_dst=00:00:00:00:00:02,actions=output:2
duration=40s, n_packets=1, n_bytes=42, priority=11111,arp,arp_tpa=10.0.0.2,actions=output:2
table_id=254, duration=77s, n_packets=0, n_bytes=0, priority=2,recirc_id=0,actions=drop
table_id=254, duration=77s, n_packets=0, n_bytes=0, priority=0,reg0=0x1,actions=controller(reason=)
table_id=254, duration=77s, n_packets=18, n_bytes=1452, priority=0,reg0=0x2,actions=drop
table_id=254, duration=77s, n_packets=0, n_bytes=0, priority=0,reg0=0x3,actions=drop
mininet>
```

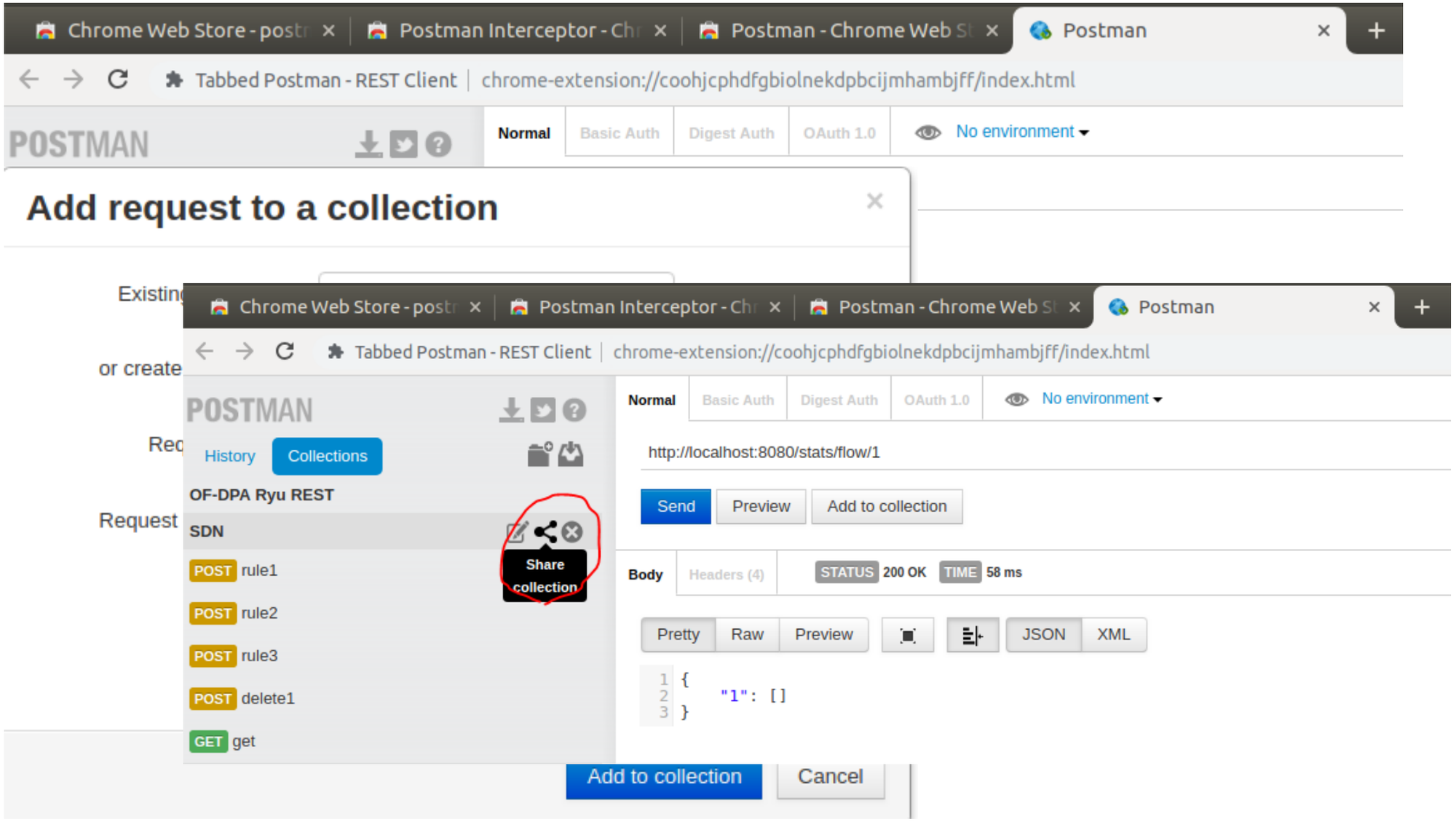
安裝及執行-Postman REST



安裝及執行-Postman REST



安裝及執行-Postman Get Collections

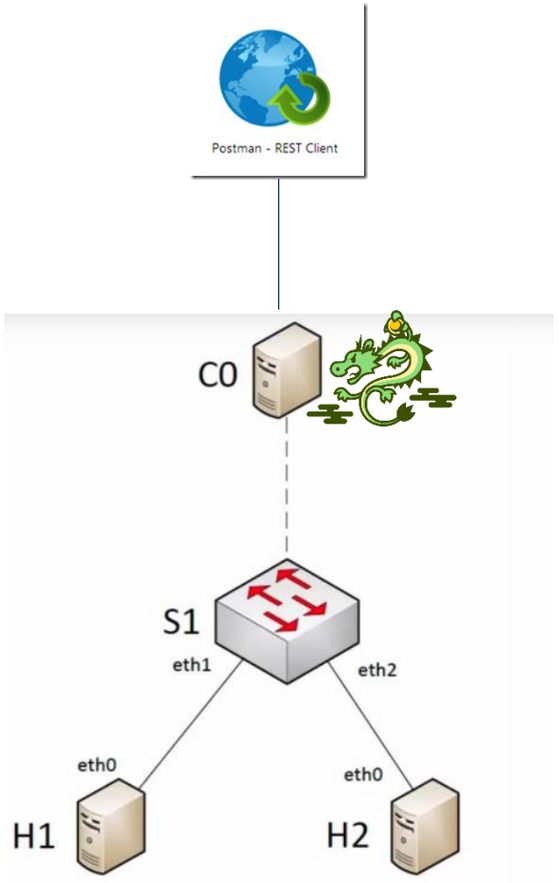


安裝及執行-Ryu

- $ sudo apt-get update
- $ sudo apt-get install libxml2-dev libxslt1-dev python-pip python-eventlet python-routes python-webob python-paramiko
- $ sudo pip install msgpack-python eventlet==0.15.2
- $ sudo pip install six --upgrade
- $ sudo pip install oslo.config q --upgrade
- $ sudo pip install ryu

Task1 – Mininet + Ryu

- Let h1 and h2 be able to access each other via Postman REST. (pingall)



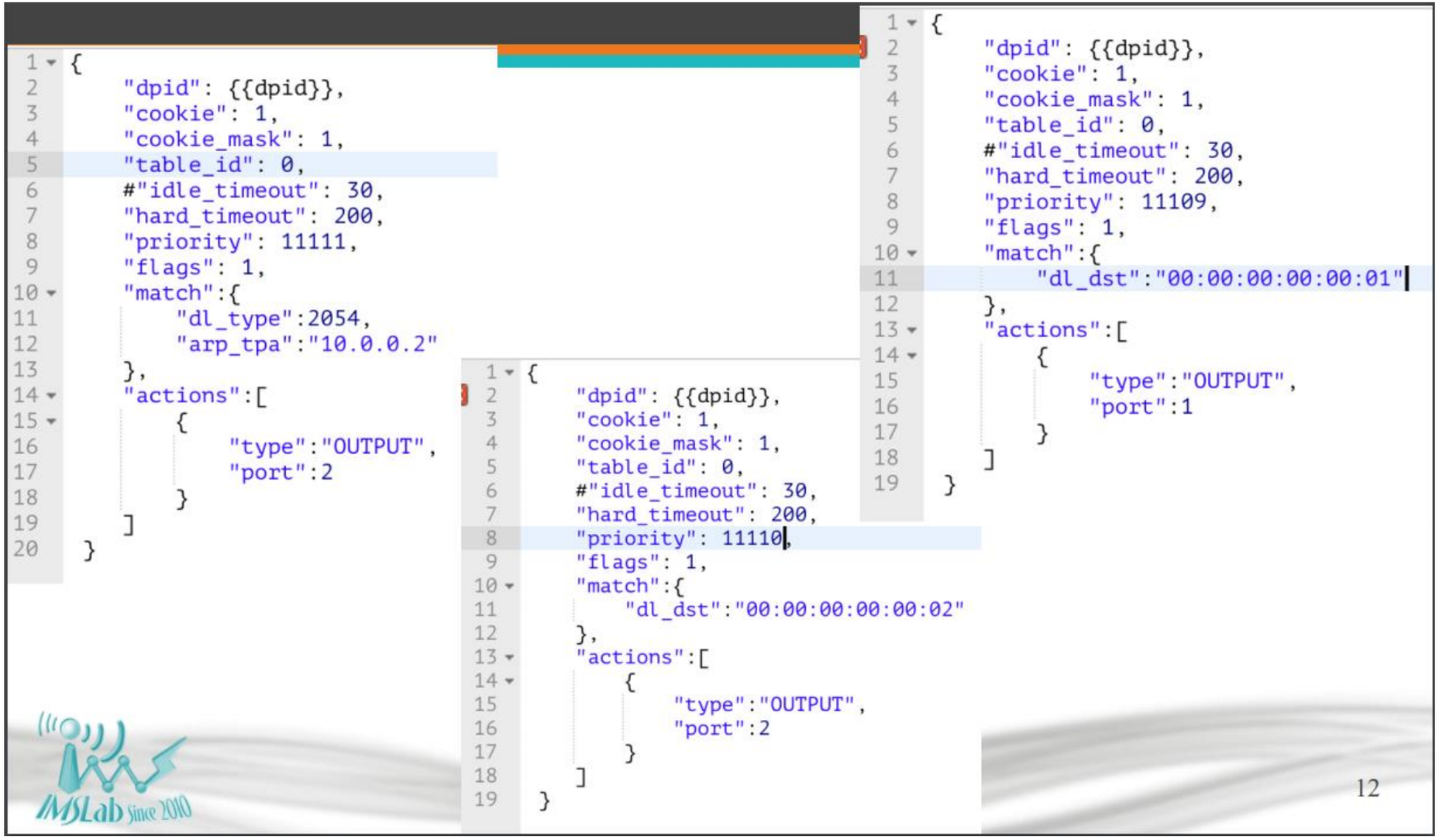
Task1 – Mininet + Ryu

- Ryu
- $ ryu-manager --verbose ryu.app.ofctl_rest
- Mininet
- $ sudo mn --mac --switch ovsk,protocols=OpenFlow13 --controller remote,ip=127.0.0.1,port=6633

Task1 – Mininet + Ryu

- Add 3 Flows
- Flow 1 : Arp Requery Query:h1(10.0.0.2) to h2(10.0.0.2)
- Flow 2 : h2 mac addr(00:00:00: 00:00:02) ->output to port 2
- Flow 3 : h1 mac addr(00:00:00: 00:00:01) ->output to port 1

Task1 – Mininet + Ryu

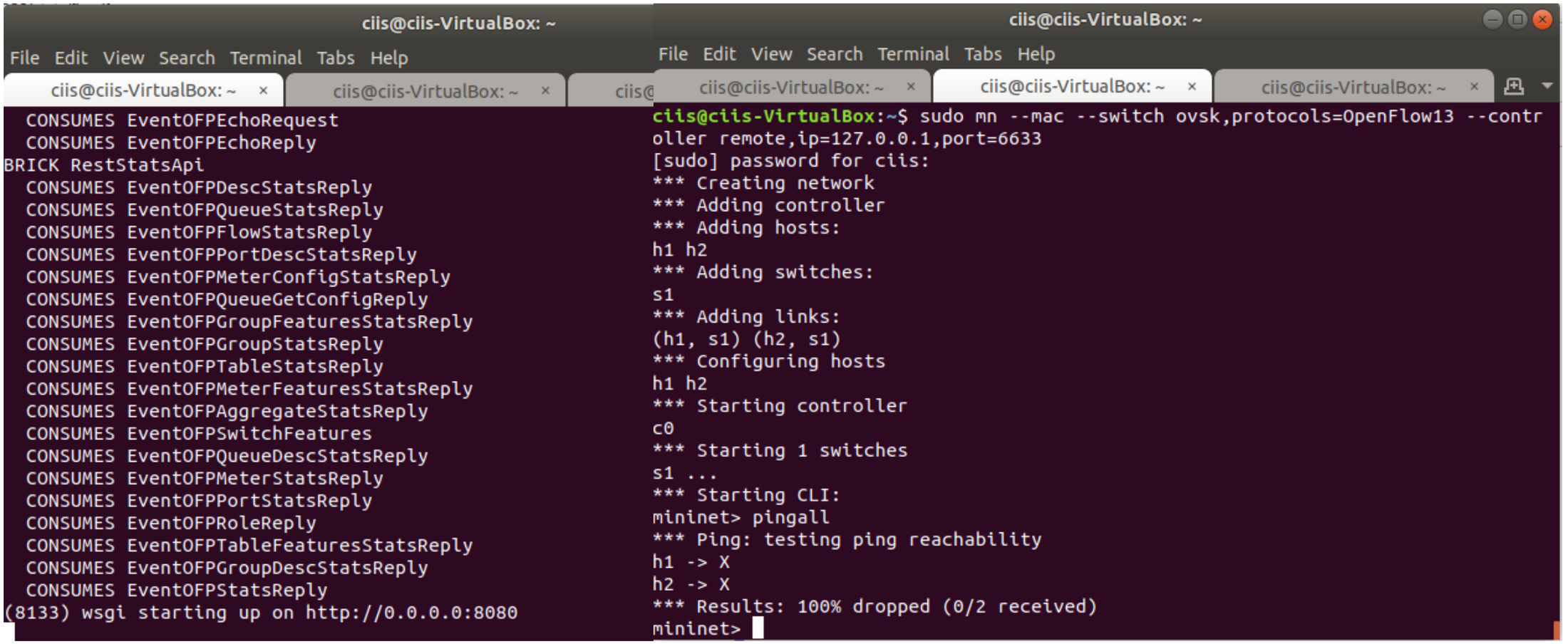


```
{
    "dpid": {{dpid}},
    "cookie": 1,
    "cookie_mask": 1,
    "table_id": 0,
    #"idle_timeout": 30,
    "hard_timeout": 200,
    "priority": 11111,
    "flags": 1,
    "match":{
        "dl_type":2054,
        "arp_tpa":"10.0.0.2"
    },
    "actions":[
        {
            "type":"OUTPUT",
            "port":2
        }
    ]
}
```

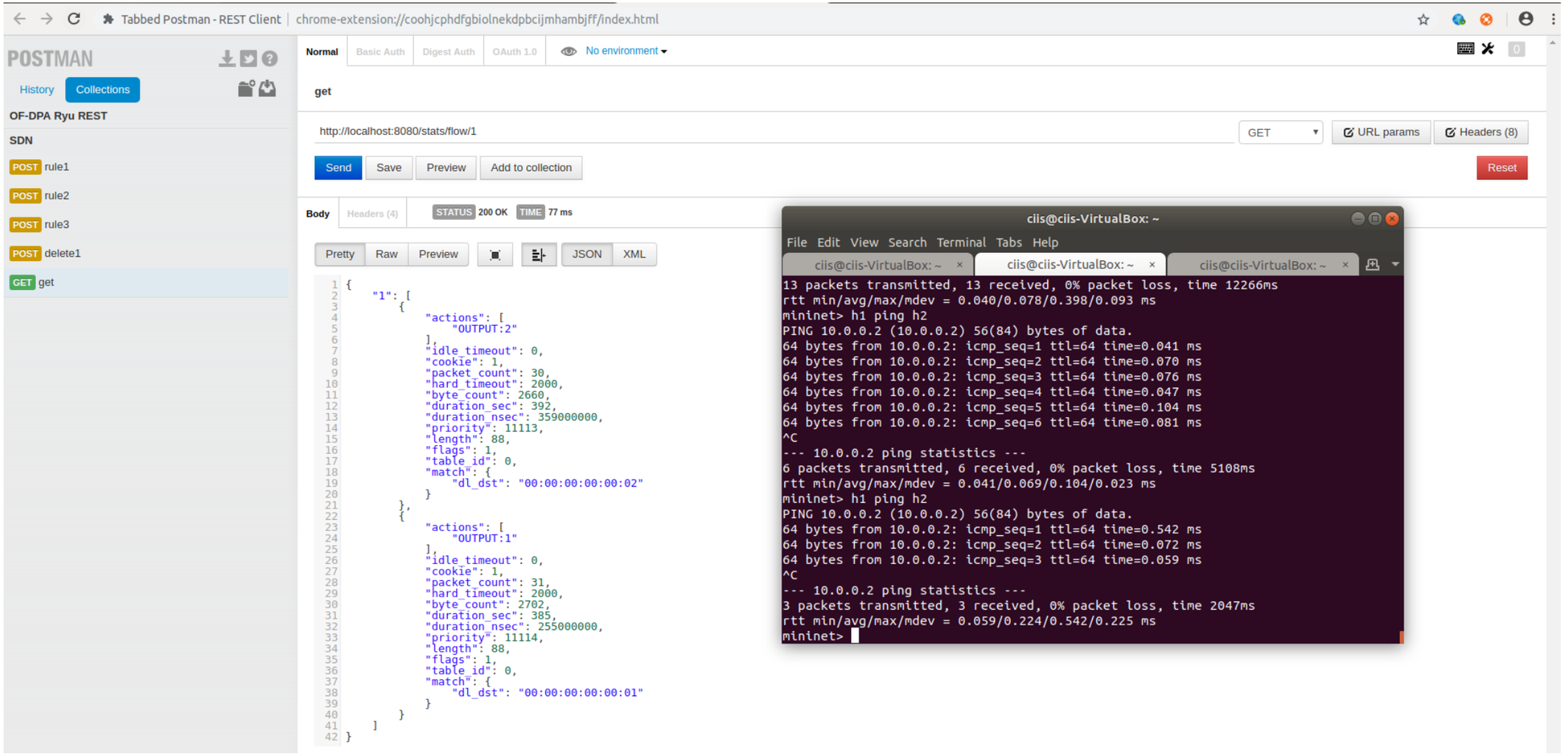
```
{
    "dpid": {{dpid}},
    "cookie": 1,
    "cookie_mask": 1,
    "table_id": 0,
    #"idle_timeout": 30,
    "hard_timeout": 200,
    "priority": 11110,
    "flags": 1,
    "match":{
        "dl_dst":"00:00:00:00:00:02"
    },
    "actions":[
        {
            "type":"OUTPUT",
            "port":2
        }
    ]
}
```

```
{
    "dpid": {{dpid}},
    "cookie": 1,
    "cookie_mask": 1,
    "table_id": 0,
    #"idle_timeout": 30,
    "hard_timeout": 200,
    "priority": 11109,
    "flags": 1,
    "match":{
        "dl_dst":"00:00:00:00:00:01"
    },
    "actions":[
        {
            "type":"OUTPUT",
            "port":1
        }
    ]
}
```

Task1 – Mininet + Ryu



Task1 – Mininet + Ryu



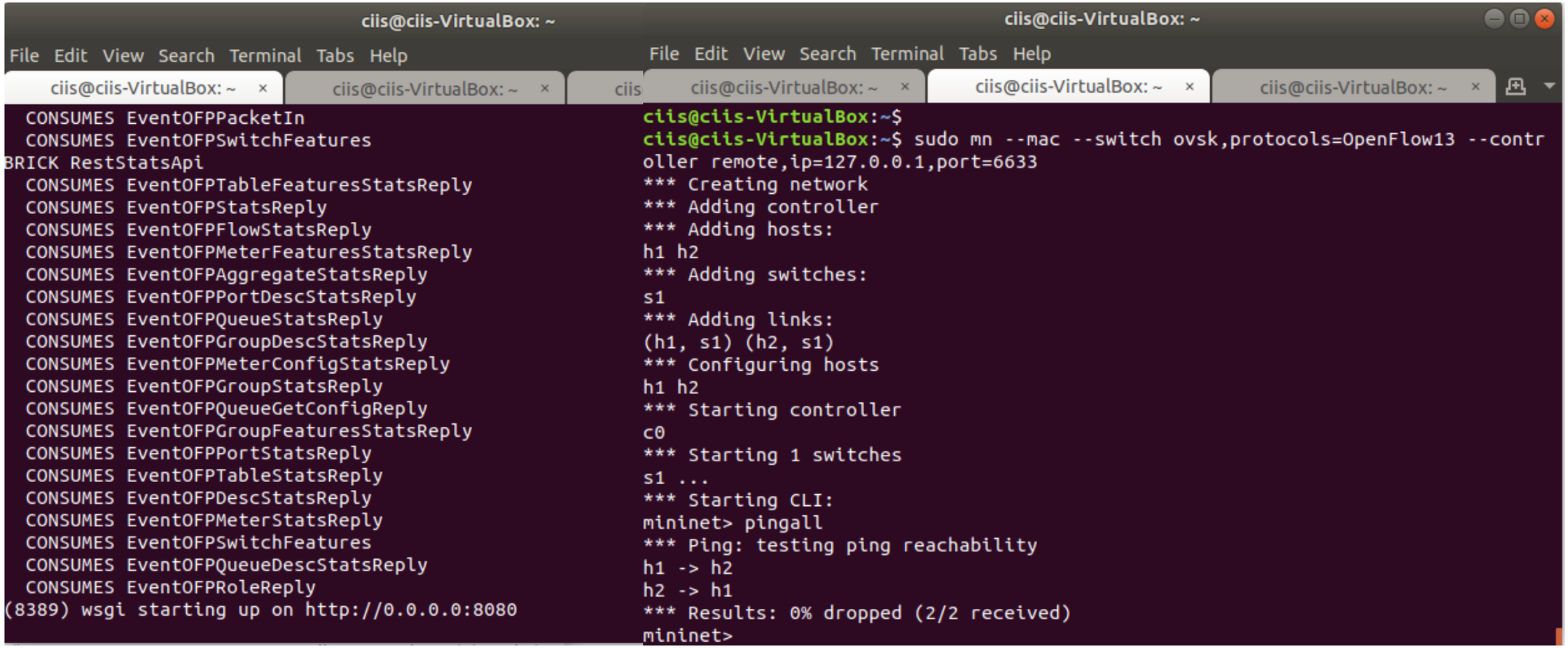
Task2 – Mininet + Ryu

- Develop an app which allows controller to flood every packet that comes to it.
- Ryu
- $ ryu-manager --verbose ryu.app.ofctl_rest ryu.app.mysw_basic
- Path
- $ /usr/local/lib/python2.7/dist-packages/ryu/app/
- Mininet
- $ sudo mn --mac --switch ovsk,protocols=OpenFlow13 --controller remote,ip=127.0.0.1,port=6633

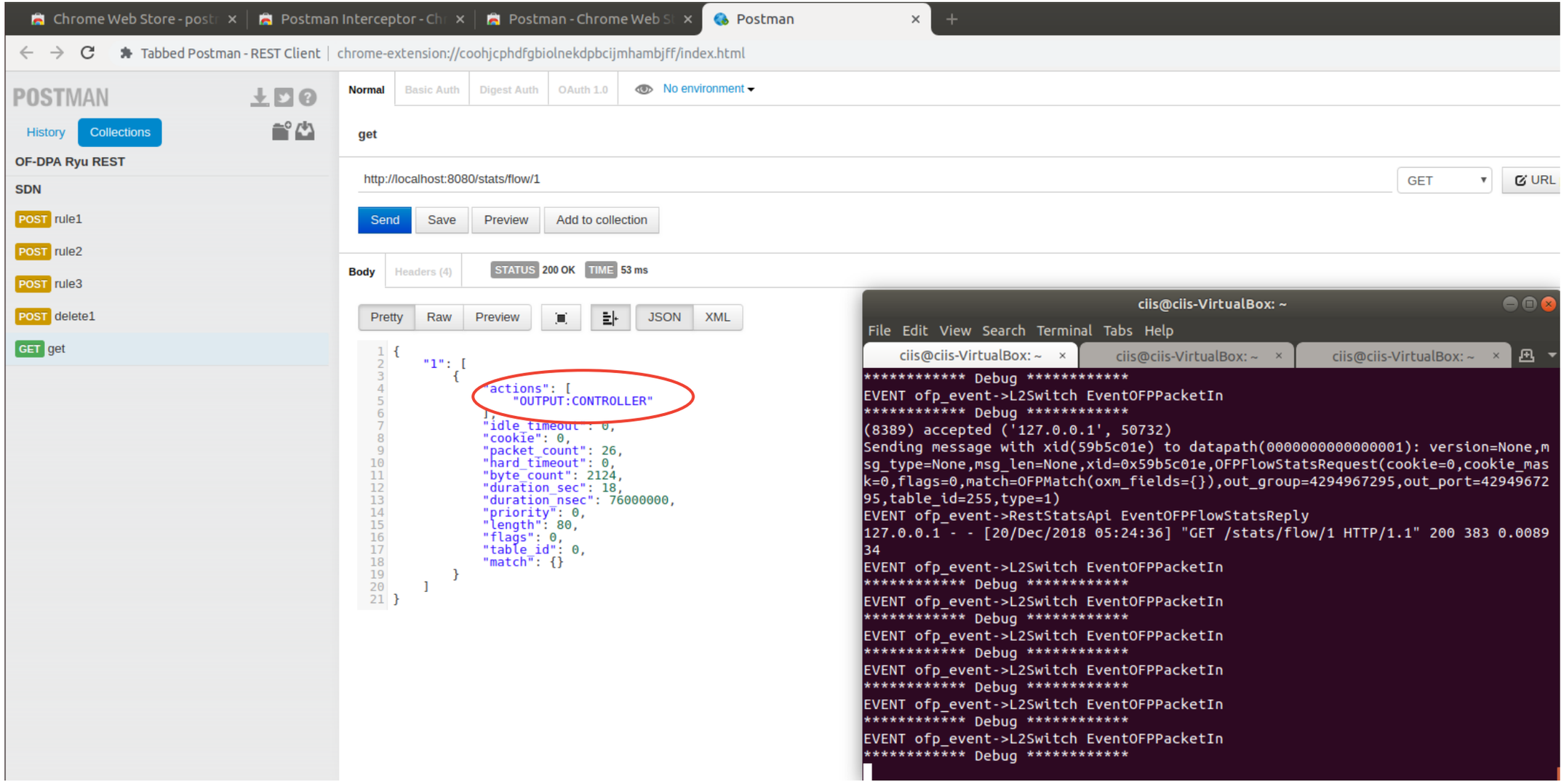
Task2 – Mininet + Ryu

```python
from ryu.base import app_manager
from ryu.controller import ofp_event
from ryu.controller.handler import CONFIG_DISPATCHER, MAIN_DISPATCHER
from ryu.controller.handler import set_ev_cls
from ryu.ofproto import ofproto_v1_3
class L2Switch(app_manager.RyuApp):
    OFP_VERSIONS = [ofproto_v1_3.OFP_VERSION]
    def __init__(self, *args, **kwargs):
        super(L2Switch, self).__init__(*args, **kwargs)
    @set_ev_cls(ofp_event.EventOFPSwitchFeatures, CONFIG_DISPATCHER)
    def switch_features_handler(self, ev):
        self.logger.info("****** Add Defualt Flow *******")
        dp = ev.msg.datapath
        ofp = dp.ofproto
        parser = dp.ofproto_parser
        match = parser.OFPMatch()
        actions = [parser.OFPActionOutput(ofp.OFPP_CONTROLLER,ofp.OFPCML_NO_BUFFER)]
        inst = [parser.OFPInstructionActions(ofp.OFPIT_APPLY_ACTIONS,actions)]
        mod = parser.OFPFlowMod(datapath=dp, priority=0,match=match, instructions=inst)
        dp.send_msg(mod)
    @set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
    def packet_in_handler(self, ev):
        msg = ev.msg
        dp = msg.datapath
        ofp = dp.ofproto
        parser = dp.ofproto_parser
        self.logger.debug("************ Debug ************")
        # output port
        data = None
        if msg.buffer_id == ofp.OFP_NO_BUFFER:
            data = msg.data
        actions = [parser.OFPActionOutput(ofp.OFPP_FLOOD)]
        out = parser.OFPPacketOut(datapath=dp, buffer_id=msg.buffer_id,in_port=msg.match['in_port'], actions=actions, data=data)
        dp.send_msg(out)
```

Task2 – Mininet + Ryu



Task2 – Mininet + Ryu



Task3 – Mininet + Ryu

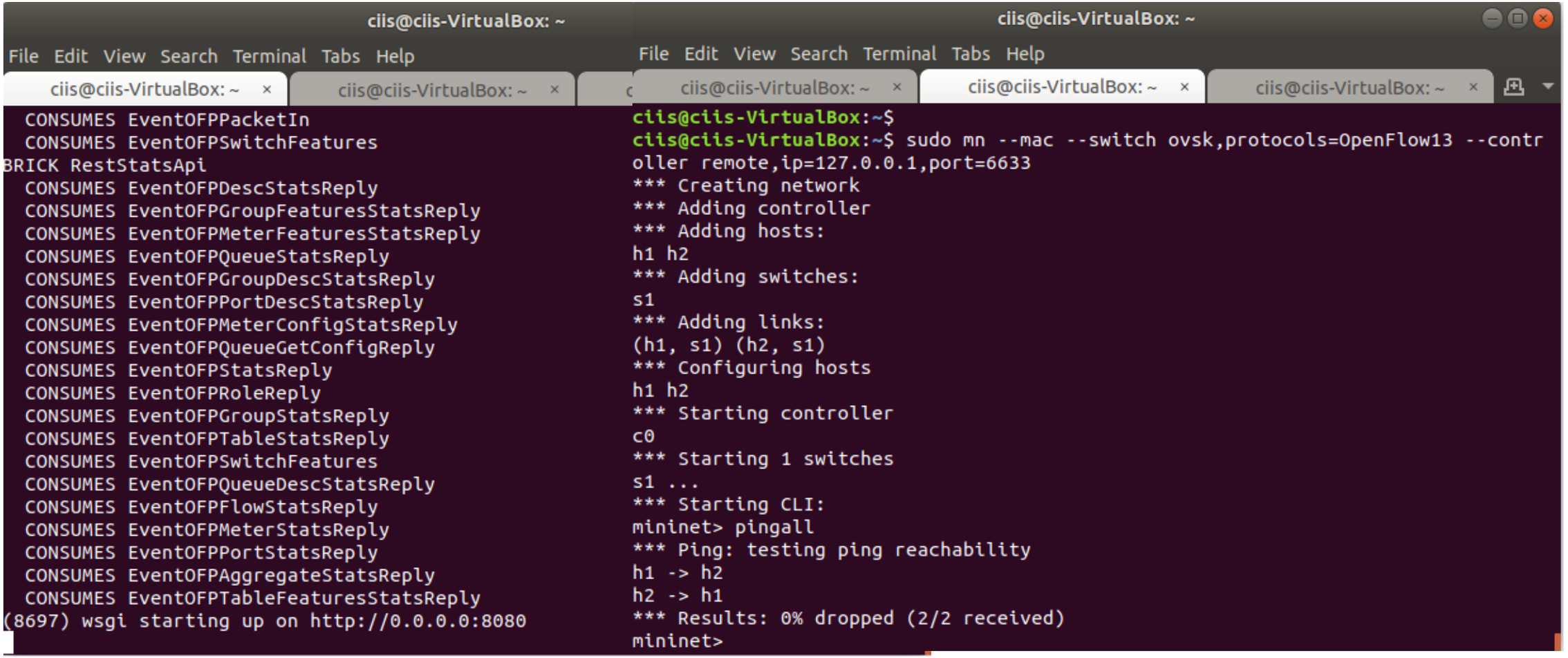
- Develop an app which allows controller to add a flow.
  - Match : everything
  - Action : Flood
- Ryu
- $ ryu-manager --verbose ryu.app.ofctl_rest ryu.app.mysw_flow
- Path
- $ /usr/local/lib/python2.7/dist-packages/ryu/app/
- Mininet
- $ sudo mn --mac --switch ovsk,protocols=OpenFlow13 --controller remote,ip=127.0.0.1,port=6633

Task3 – Mininet + Ryu

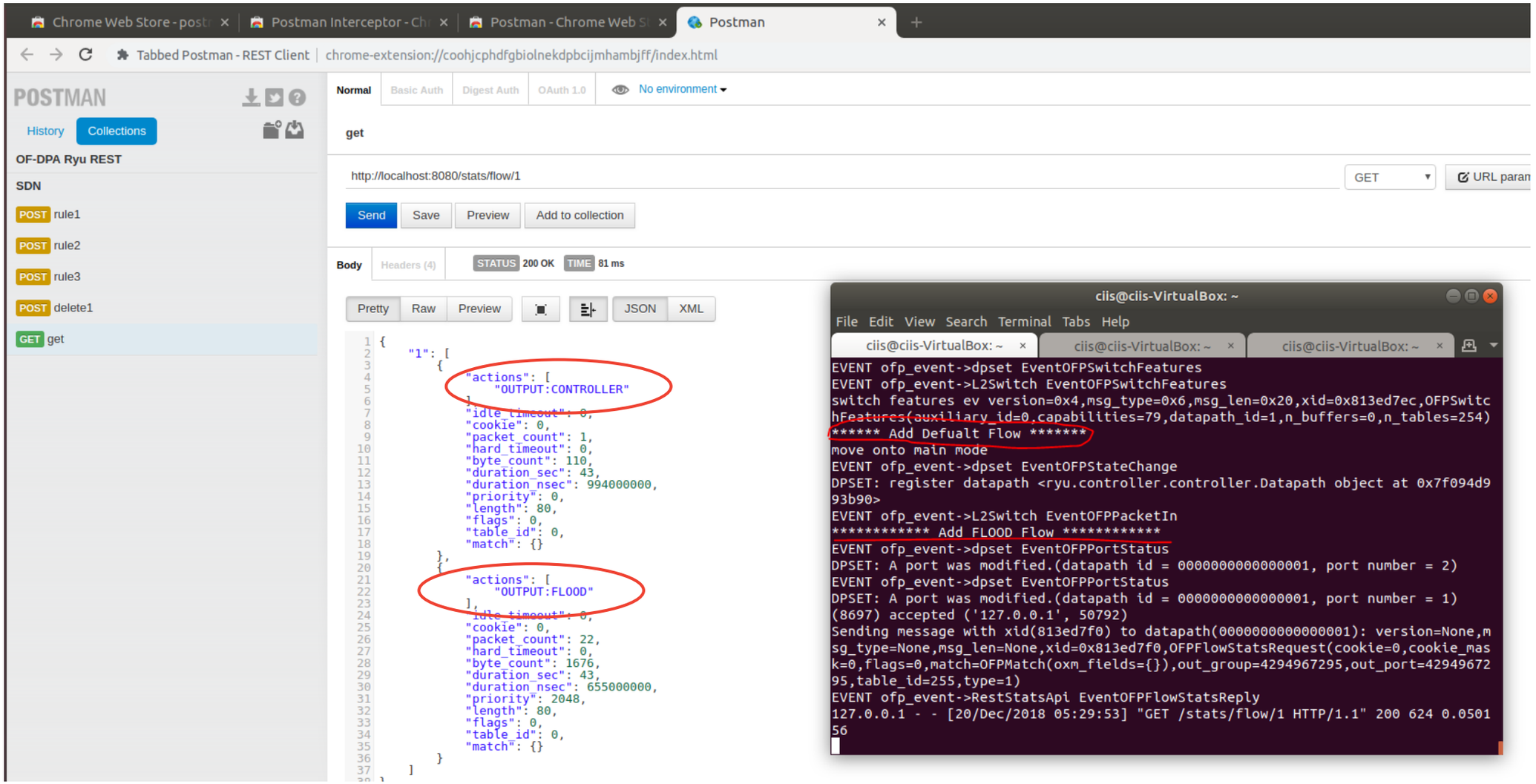
```python
from ryu.base import app_manager
from ryu.controller import ofp_event
from ryu.controller.handler import CONFIG_DISPATCHER, MAIN_DISPATCHER
from ryu.controller.handler import set_ev_cls
from ryu.ofproto import ofproto_v1_3
class L2Switch(app_manager.RyuApp):
    OFP_VERSIONS = [ofproto_v1_3.OFP_VERSION]
    def __init__(self, *args, **kwargs):
        super(L2Switch, self).__init__(*args, **kwargs)
    @set_ev_cls(ofp_event.EventOFPSwitchFeatures, CONFIG_DISPATCHER)
    def switch_features_handler(self, ev):
        self.logger.info("****** Add Defualt Flow *******")
        dp = ev.msg.datapath
        ofp = dp.ofproto
        parser = dp.ofproto_parser
        match = parser.OFPMatch()
        actions = [parser.OFPActionOutput(ofp.OFPP_CONTROLLER,ofp.OFPCML_NO_BUFFER)]
        inst = [parser.OFPInstructionActions(ofp.OFPIT_APPLY_ACTIONS,actions)]
        mod = parser.OFPFlowMod(datapath=dp, priority=0,match=match, instructions=inst)
        dp.send_msg(mod)
    @set_ev_cls(ofp_event.EventOFPPacketIn, MAIN_DISPATCHER)
    def packet_in_handler(self, ev):
        msg = ev.msg
        dp = msg.datapath
        ofp = dp.ofproto
        parser = dp.ofproto_parser

        self.logger.debug("************ Add FLOOD Flow ************")
        match = parser.OFPMatch()
        actions = [parser.OFPActionOutput(ofp.OFPP_FLOOD)]
        inst = [parser.OFPInstructionActions(ofp.OFPIT_APPLY_ACTIONS,actions)]
        mod = parser.OFPFlowMod(datapath=dp, priority=2048,match=match, instructions=inst)
        dp.send_msg(mod)
        ## output port
        #data = None
        #if msg.buffer_id == ofp.OFP_NO_BUFFER:
        #    data = msg.data
        #actions = [parser.OFPActionOutput(ofp.OFPP_FLOOD)]
        #out = parser.OFPPacketOut(datapath=dp, buffer_id=msg.buffer_id,in_port=msg.match['in_port'], actions=actions, data=data)
        #dp.send_msg(out)
```

Task3 – Mininet + Ryu



Task3 – Mininet + Ryu



06 實驗要求

標題文字

- Task1
  - 請將Postman下達的flow集成一Collection並儲存，並也將實驗結果一併截圖繳交
- Task2
  - 請在程式碼中加入學號並在controller中印出，並將實驗結果截圖繳交
- Task3
  - 請在程式碼中加入學號並在controller中印出，並將實驗結果截圖繳交

參考資料

[1]軟體定義網路（SDN）

https://www.xinguard.com/content.aspx?id=34

[2] Mininet+OpenDaylight
http://140.117.164.12/data/SDN_NFV_class(v2_2017)/SDN_Lab1.pdf

[3] Mininet+Ryu
http://140.117.164.12/data/SDN_NFV_class(v2_2017)/SDN_Lab4.pdf

[4] OpenFlow 通訊協定

https://osrg.github.io/ryu-book/zh_tw/html/openflow_protocol.html