Mawlana Bhashani Science and Technology University

Lab-Report

Report No:07

Report Name:Controller Rest API

Course title: Computer Network Lab

Date of Performance:31-03-21

Date of Submission:18-06-21

Submitted by

Name: Sabikun Nahar Piya

ID:IT-18020

Name:Nusrat Jahan Jui

ID:IT-18039

3rd year 2nd semester

Session: 2017-2018

Submitted To

Nazrul Islam

Assistant Professor

Dept. of ICT

MBSTU.

1.Objectives:

The objective of the lab 7 is to:

The objective of the lab 7 is to:

Understand the working principles of Controller Rest API.

Understand the difference between proactive and reactive installation flows

2.Theory:

2.1 OpenFlow: Reactive versus Proactive

OpenFlow is still the only one wire protocol that has a reasonably good chance at becoming the de-facto open SDN southbound messaging standard. When using OpenFlow to populate tables in switches there are essentially three modes of operation:

• **Reactive Flow Instantiation:** When a new flow comes into the switch, the OpenFlow agent software on the switch does a lookup in the flow tables. If no match for the flow is found, the switch creates an OFP packet-in packet and sends it off to the controller for instructions. Reactive mode reacts to traffic, consults the OpenFlow controller and creates a rule in the flow table based on the instruction. This behavior was tested on previous lab.

• **Proactive Flow Instantiation:** Rather than reacting to a packet, an OpenFlow controller could populate the flow tables ahead of time for all traffic matches that could come into the switch. By pre-defining all of the flows and actions ahead of time in the switches flow tables, the packet-in event never occurs. The result is all packets are forwarded at line rate. Proactive OpenFlow flow tables eliminate any latency induced by consulting a controller on every flow. This behavior will be tested on this lab.

• **Hybrid flow instantiation:** A combination of both would allow for flexibility of reactive for particular sets a granular traffic control that while still preserving low-latency forwarding for the rest of the traffic.

2.2 Controller: REST API

• Application program interface (API) is an interface presented by software (such as a network operating system) that provides the capability to collect information from or make a change to an underlying set of resources.

• APIs in the context of SDN: In an open SDN model, a common interface discussed is the northbound interface (NBI). The NBI is the interface between software applications, such as operational support systems, and a centralized SDN controller. One of the common API technologies used at the northbound interface is the Representational State Transfer (REST) API. REST APIs use the HTTP/HTTPS protocol to execute common operations on resources represented by Uniform Resource Identifier (URI) strings. An application may use REST APIs to send an HTTP/HTTPS GET message via an SDN controller's IP address. That message would contain a URI string referencing the relevant network device and comprising an HTTP payload with a JSON header that has the proper parameters for a particular interface and statistic.

• Datapath Identifier of Openflow Switch: Each OpenFlow instance on a switch is identified by a Datapath Identifier. This is a 64 bit number determined as follows according to the OpenFlow specification: “The datapath_id fleld uniquely identifies a datapath. The lower 48 bits are intended for the switch MAC address, while the top 16 bits are up to the implementer. An example use of the top 16 bits would be a VLAN ID to distinguish multiple virtual switch instances on a single physical switch.”

3.Methodology:

In this activity we will learn how to create flows using the Controller REST API.

Using REST APIs: REST API can be used in different ways:

1. A tool to generate REST API calls:

   The Chrome browser, for example, has multiple plug-ins to generate REST API messages. These include Postman and the Advanced REST Client.

   Firefox has the RESTClient add-on for the same functionality.

2. Command-line interface, the curl utility may also be used. Although the formatting of the REST API varies form one controller to another, the following items are common: URI string for the requested, HTTP method (e.g., GET, POST, PUT, and DELETE) and JSON/XML payload and/or

parameters. The Ryu documentation provides examples illustrating how to send a valid REST API message

RYU.APP.OFCTL_REST

ryu.app.ofctl_rest provides REST APIs for retrieving the switch stats and updating the switch stats. This application helps to debug application and get various statistics. Valid actions are:

1. **Retrieve the switch stats**
2. Update the switch stats
3. Support for experimenter multipart
4. Reference: Description of Match and Actions

Installing curl:

1. Open the Synaptic Package Manager (Navigator ->System-> Synaptic Package Manager)
2. Setup the proxy:
   Click on settings-> Preference -> Network .
   Click on manual proxy configuration
   HTT and FTP Proxy: proxy.rmit.edu.au Port: 8080
3. Search for Quick filter `curl`
4. Click on Mark for installation
5. Then click on Apply and wait until the package is installed

4.Exercise:

Section 4.2: Adding and removing files

Section 4.2.1: Add a flow entry to the switch using the following command line:

curl -X POST -d '{

"dpid": ZODIAC_ dpid_number,

"cookie": 1,

"cookie_mask": 1,

"table_id": 0,

"idle_timeout": 30,

"hard_timeout": 30,

"priority": 11111,

"flags": 1,

"match":{

"in_port":1

},

"actions":[

{

"type":"OUTPUT",

"port": 2

}

]

}'http://localhost:8080/stats/flowentry/add

Ans:

```
$ curl -X POST -d '{
    "dpid": 1,
    "cookie": 1,
    "cookie_mask": 1,
    "table_id": 0,
    "idle_timeout": 30,
    "hard_timeout": 30,
    "priority": 11111,
    "flags": 1,
    "match":{
        "in_port":1
    },
    "actions":[
        {
            "type":"OUTPUT",
            "port": 2
        }
    ]
 }' http://localhost:8080/stats/flowentry/add
```

Exercise 4.2.2: Create a command line for adding a flow entry that allows to link port 2 with port 1. Which is the command line?

Provide the screenshot of the ZODIAC flow table.

Ans:

```
$ curl -X POST -d '{
    "dpid": 1,
    "cookie": 1,
    "cookie_mask": 1,
    "table_id": 0,
    "idle_timeout": 30,
    "hard_timeout": 30,
    "priority": 11111,
    "flags": 1,
    "match":{
        "in_port":1
    },
    "actions":[
        {
            "type":"OUTPUT",
            "port": 2
        }
    ]
 }' http://localhost:8080/stats/flowentry/add
```

Exercise 4.2.3: Delete all flow entries of the switch which specified with Datapath ID in URI using the following command line: curl -X DELETE http://0.0.0.0:8080/stats/flowentry/clear/ ZODIAC_ dpid_number

```
$ curl -X DELETE http://localhost:8080/stats/flowentry/clear/1
```

Exercise 4.2.4: Add a group entry to the switch using the following command line:

curl -X POST -d '{

"dpid": ZODIAC_ dpid_number,

"type": "ALL",

"group_id": 1,

"buckets": [

{

"actions": [

{

"type": "OUTPUT",

"port": 1

}

]

}

]

}' http://0.0.0.0:8080/stats/groupentry/add

Ans:

```
$ curl -X POST -d '{
    "dpid": 1,
    "type": "ALL",
    "group_id": 1,
    "buckets": [
        {
            "actions": [
                {
                    "type": "OUTPUT",
                    "port": 1
                }
            ]
        }
    ]
 }' http://localhost:8080/stats/groupentry/add
```

5.Conclusion: From this lab we can understand the working principles of Controller Rest API.One of the key **advantages of REST APIs** is that they provide a great deal of flexibility. Data is not tied to resources or methods, so **REST** can handle multiple types of calls, return different data formats and even change structurally with the correct implementation of hypermedia.In this lab,we have performed some exercises like adding and removing flows