MAWLANA BHASHANI SCIENCE AND TECHNOLOGY UNIVERSITY

Santosh,Tangail – 1902

Course Title : **Telecommunication Engineering**

Assignment Name : **Controller REST API**

Assignment No. : **03**

Submitted by,

Name : ANIKA JAHIN

ID: IT-17056

Session: 2016-17

Dept. of ICT,MBSTU.

Submitted to,

NAZRUL ISLAM

Assistant Professor

Dept. of ICT,MBSTU.

Theory:

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.

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.”

Question 5.1: Explain the advantages of REST API of the Controller.

Answer: The advantages are given below:

- **Due to its scalability**. This protocol stands out due to its scalability. Thanks to the separation between client and server, the product may be scaled by a development team without much difficulty.

- **Due to its flexibility and portability.** With the indispensable requirement for data from one of the requests to be properly sent, it is possible to perform a migration from one server to another or carry out changes on the database at any time. Front and back can therefore be hosted on different servers, which is a significant management advantage.

- **Due to its independence.** Due to the separation between client and server, the protocol makes it easy for developments across the various areas of a project to take place independently. In addition, the **REST API** adapts at all times to the working syntax and platform. This offers the opportunity to try several environments while developing.

Question 5.5: what is the difference between sdn and openflow?

Answer: SDN and OpenFlow are prone to be confused and misunderstood. Take a look at SDN vs. OpenFlow, the two are indeed interconnected. First of all, as an open protocol, OpenFlow underpins the various SDN controller solutions. The complete SDN solution is taking SDN controller as the core, backed by OpenFlow switches and NFV to offer bountiful SDN app for a new smart, dynamic, open, custom network.

Conclusion

Even though REpresentational State Transfer, also known as REST, is often referred to as a protocol, it’s an architectural style. It defines how applications communicate over the Hypertext Transfer Protocol (HTTP). Applications that use REST are loosely-coupled and transfer information quickly and efficiently. While REST doesn’t define data formats, it’s usually associated with exchanging JSON or XML documents between a client and a server. This interface overcomes the disadvantages **SOAP** exhibited, such as the need for clients to know the operation semantics as a pre-requisite for its use, or the need for ports for different types

of notifications. In addition, with a few operations, **REST** can handle many resources, while **SOAP** needs many operations to accomplish that. These are its advantages:

- It is usually simple to build and adapt.
- Low use of resources.
- Process instances are created explicitly.
- With the initial URI, the client does not require routing information.
- Clients can have a generic 'listener' interface for notifications.

CRUD!

The name REpresentational State Transfer implies exchanging data. The server acts as a data store, and the client retrieves and stores data. The server transfers object states to the client. The client can update these states too.

Most REST APIs implement CRUD: Create, Retrieve, Update, and Delete.

Go back to the Swagger page and click on the blue GET box so it collapses. Here's a quick tip: at the top of the page, there is the **List Operations** option. Clicking there will collapse the operations into a list again.

Let's look at the list of operations again.

On the left-hand side of the page we see GET, POST, DELETE, GET, PATCH, and PUT. These are HTTP methods that correspond to operations.

We can map these operations into CRUD.

- POST—Create
- GET—Retrieve
- PUT / PATCH—Update
- DELETE—Delete