MAWLANA BHASHANI SCIENCE AND TECHNOLOY UNIVERSITY

Santosh,Tangail – 1902

Course Title : Introduction to Telecommunication System

Assignment No : 2

Submitted by,

Name : Mahbuba Zaman Mitu

ID: IT-16044

Session: 2015-2016

Dept. of ICT,MBSTU.

Submitted to,

NAZRUL ISLAM

Assistant Professor

Dept. of ICT,MBSTU.

Assignment Name: OpenFlow Protocol

Objectives:

1. Understand the working principles of OpenFlow protocol.
2. Configure a basic Software Defined Network for end-to-end communications.
3. Understand the difference between interacting with real and virtual networks.

Theory:

This speciation covers the components and the basic functions of the switch, and the OpenFlow protocol to manage an OpenFlow switch from a remote controller.

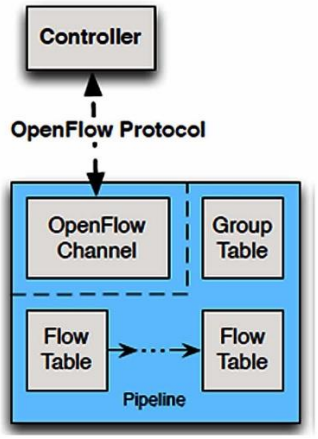


Figure 2-1. Main components of an OpenFlow switch.

Switch Components

An OpenFlow Switch consists of one or more flow tables and a group table, which perform packet lookups and forwarding, and an OpenFlow channel to an external controller (Figure 2-1). The switch communicates with the controller and the controller manages the switch via the OpenFlow protocol.

Using the OpenFlow protocol, the controller can add, update, and delete flow entries in flow tables, both reactively (in response to packets) and proactively. Each flow table in the switch contains a set of flow entries; each flow entry consists of match fields, counters, and a set of instructions to apply to matching packets.

OpenFlow specification terms

This section describes key OpenFlow specification terms:

1. **Byte**: an 8-bit octet.
2. **Packet**: an Ethernet frame, including header and payload.
3. **Port:** where packets enter and exit the OpenFlow pipeline. May be a physical port, a logical port defined by the switch, or a reserved port defined by the OpenFlow protocol.

4. **Pipeline**: the set of linked flow tables that provide matching, forwarding, and packet modifications in an OpenFlow switch.

5**. Flow Table**: A stage of the pipeline, contains flow entries.

6. **Flow Entry**: an element in a flow table used to match and process packets. It contains a set of match fields for matching packets, a priority for matching precedence, a set of counters to track packets, and a set of instructions to apply.

7. **Match Field**: a field against which a packet is matched, including packet headers, the ingress port, and the metadata value. A match field may be wildcarded (match any value) and in some cases bitmasked.

8. **Metadata**: a maskable register value that is used to carry information from one table to the next.

9**. Instruction**: Instructions are attached to a flow entry and describe the OpenFlow processing that happen when a packet matches the flow entry. An instruction either modifies pipeline processing, such as direct the packet to another flow table, or contains a set of actions to add to the action set, or contains a list of actions to apply immediately to the packet.

10**. Action**: an operation that forwards the packet to a port or modifies the packet, such as decrementing the TTL field. Actions may be specified as part of the instruction set associated with a flow entry or in an action bucket associated with a group entry. Actions may be accumulated in the Action Set of the packet or applied immediately to the packet.

11.**Action Set:** a set of actions associated with the packet that are accumulated while the packet is processed by each table and that are executed when the instruction set instructs the packet to exit the processing pipeline.

12**. Group:** a list of action buckets and some means of choosing one or more of those buckets to apply on a per-packet basis.

13.**Action Bucket**: a set of actions and associated parameters, defined for groups.

14**. Tag**: a header that can be inserted or removed from a packet via push and pop actions.

15. **Outermost Tag**: the tag that appears closest to the beginning of a packet.

16.**Controller:** An entity interacting with the OpenFlow switches using the OpenFlow protocol.

17. **Meter:** a switch element that can measure and control the rate of packets. The meter trigger a meter band if the packet rate or byte rate passing through the meter exceed a predefined threshold. If the meter band drops the packet, it is called a Rate Limiter.

Methodology

This lab is designed for working on group of three students. In this activity students will learn how a switch hub works and how it is implemented using OpenFlow.

Traditional Switching Hub

Switching hubs have a variety of functions. Here, we take a look at a switching hub having the following simple functions.

• Learns the MAC address of the host connected to a port and retains it in the MAC address table.

• When receiving packets addressed to a host already learned, transfers them to the port connected to the host.

• When receiving packets addressed to an unknown host, performs flooding.

Switching Hub by OpenFlow

OpenFlow switches can perform the following by receiving instructions from OpenFlow controllers such as Ryu:

• Rewrites the address of received packets or transfers the packets from the specified port. • Transfers the received packets to the controller (Packet-In).

• Transfers the packets forwarded by the controller from the specified port (Packet-Out). It is possible to achieve a switching hub having those functions combined.

After learning, the switch transfers the received packets. The switch investigates whether the destination MAC address of the packets belong to the learned host. Depending on the investigation results, the switch performs the following processing.

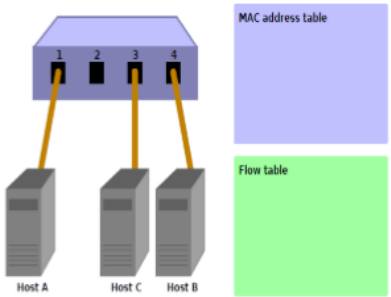
• If the host is already a learned host ... Uses the Packet-Out function to transfer the packets from the connected port.

• If the host is unknown host ... Use the Packet-Out function to perform flooding.

The following explains the above operation in a step-by-step way using figures.

1. Initial status

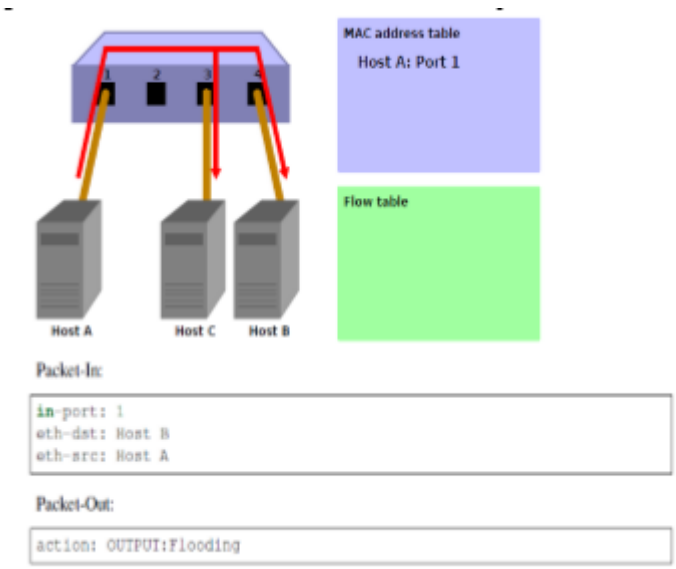
• This is the initial status where the flow table is empty. • Assuming host A is connected to port 1, host B to part 4, and host C to port 3.



2. Host A -> Host B

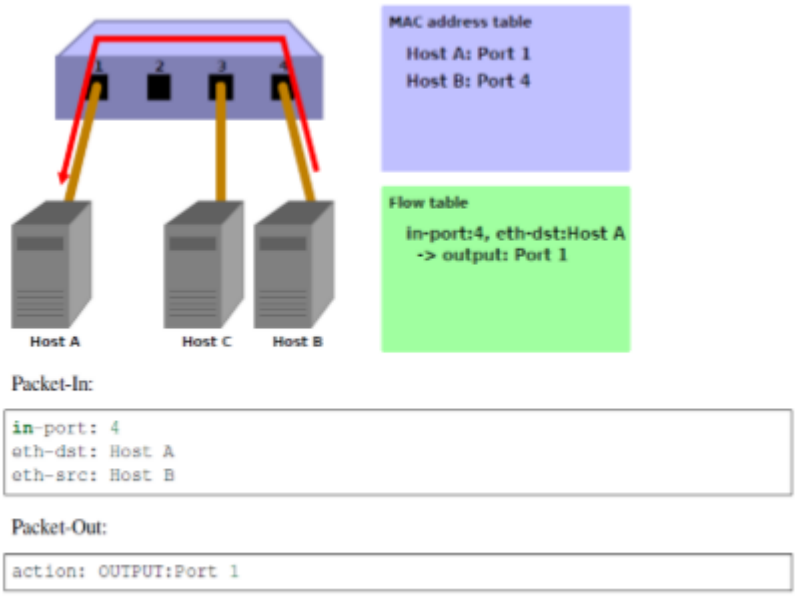
• When packets are sent from host A to host B, a Packet-In message is sent and the MAC

address of host A is learned by port 1. Because the port for host B has not been found, the packets are flooded and are received by host B and host C.



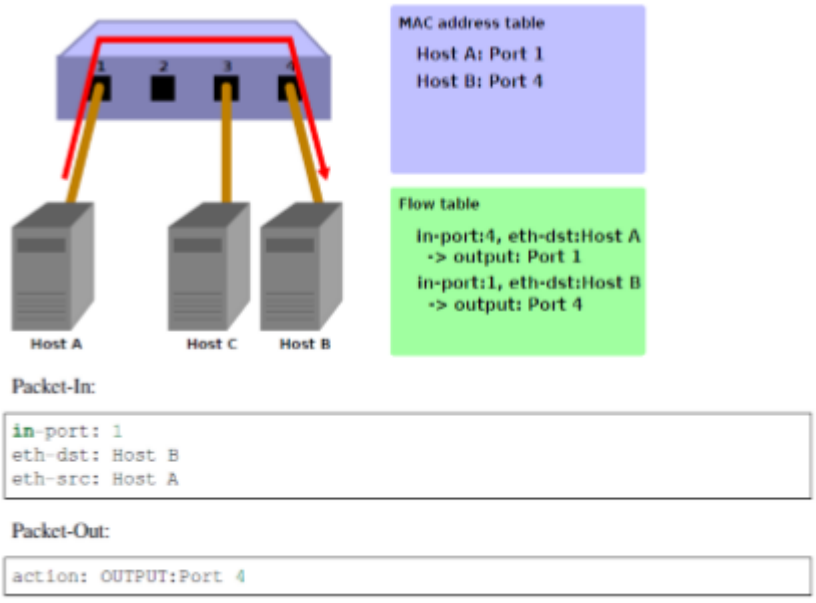
3. Host B -> Host A

• When the packets are returned from host B to host A, an entry is added to the flow table and also the packets are transferred to port 1. For that reason, the packets are not received by host C.



4. Host A -> Host B

• Again, when packets are sent from host A to host B, an entry is added to the flow table and also the packets are transferred to port 4.



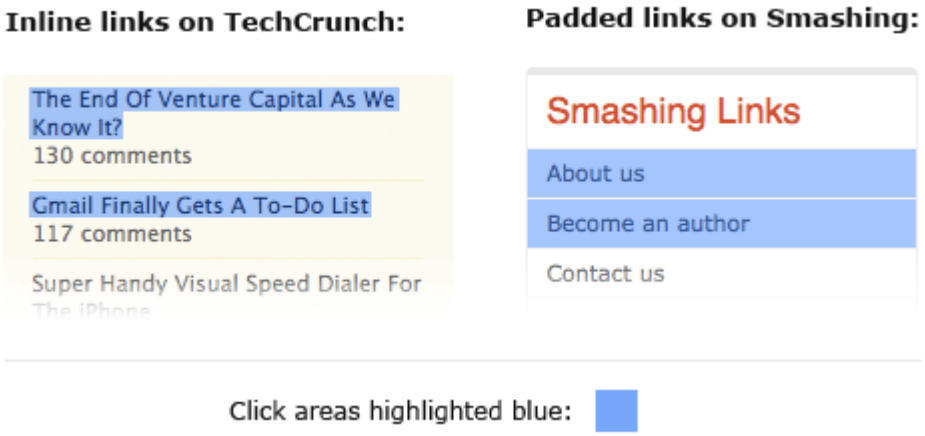
Question 5.1: How the RYU GUI interface can be improved? Provide some ideas.

Ans:

There are many techniques involved in crafting beautiful and functional interfaces. Here's my collection of 10 that I think you'll find useful in your work. They're not related to any particular theme, but are rather a collection of techniques I use in my own projects. Without further ado, let's get started.

1. Padded Block Links

Links (or anchors) are inline elements by default, which means that their clickable area spans only the height and width of the text. This clickable area, or the space where you can click to go to that link's destination, can be increased for greater usability. We can do this by **adding padding** and, in some cases, also **converting the link into a block element**. Here's an example of inline and padded links, with their clickable areas highlighted to show the difference:



Obviously, the larger the clickable area is, the easier it is to click on the link because there is less of a chance of missing it.

2. Typesetting Buttons

Attention to every detail is what separates a great product from a mediocre one. Interface elements such as buttons and tabs are clicked on many times a day by your users, so it pays to typeset them properly; and by typesetting I mean positioning the label. Here's a couple of examples of the kind of misplaced labels I sometimes notice:

At first glance they look okay, but notice that the text is placed too high because the lowercase letters have been used as a guide to align the text vertically in the center, like so:

However, if we use uppercase letters as well as lowercase letters with ascenders ("t," "d," "f," "h," "k" and "l"), **the balance shifts upwards**, making the label appear too high on the button. In such cases, we should set the type using the uppercase height as a guide – or set it a little bit higher if most of the letters are lowercase. Here's what I mean:

3. Using Contrast To Manage Focus

Similarly, you can also manage the focus of your visitors' attention with contrast between elements. Here's an example of a post headline and some meta information underneath regarding who posted the article and its date:

All the text is set in black. Let's decrease the contrast between the meta information (the date and author's name) and the background by putting the text in a light shade of gray:

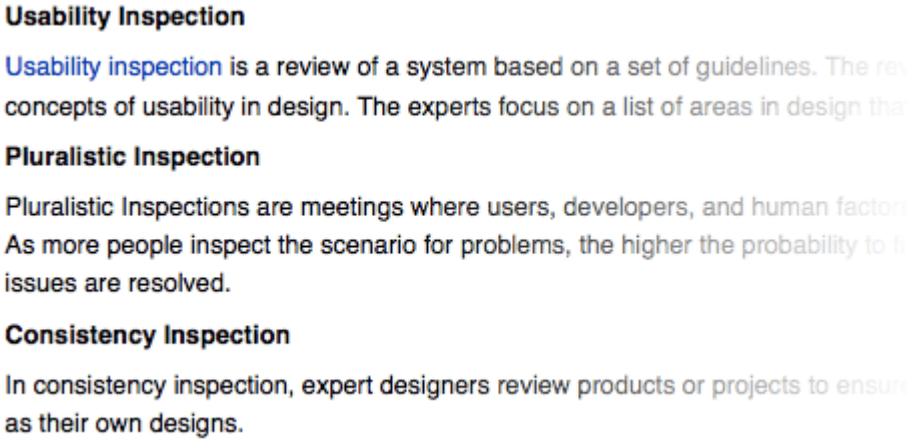
The highest contrast element here is the headline, so it literally pops out at us.

4. Using Color To Manage Attention

Color can also be used to effectively **focus your visitors' attention on important or actionable elements**. For example, during the US presidential election, pretty much all of the candidates' websites had the donation button colored red. Red is a very bright and powerful color so it attracts attention, and it stands out even more when the rest of the website is blue or another colder color.

5. White Space Indicates Relationships

One of the most crucial elements in an interface is the white space between elements. If you're not familiar with the term white space, it means just that: space between one interface element and another, be it a button, a navigation bar, article text, a headline and so on. By manipulating white space, we can indicate relationships between certain elements or groups of elements.



6. Letter Spacing

Web design is pretty limiting for typographers. But while there are only a few safe Web fonts and not a great many things you can do to style them, it's worth remembering that we do still have some level of control. "Tracking" is a term used in the field of typography to describe the adjustment of **spacing between letters in words**. We've got the ability to do this with CSS using the letter-spacing property.

h1. Lorem ipsum dolor

h2. Lorem ipsum dolor

H3. LOREM IPSUM DOLOR

Question 5.2: Explain at least two the advantage and disadvantage of using mininet or Zodiac FX?

Ans: The advantages of mininet:

Boots faster: seconds instead of minutes

Scales larger: hundreds of hosts and switches vs. single digits

Provides more bandwidth: typically 2Gbps total bandwidth on modest hardware

Installs easily: a prepackaged VM is available that runs on VMware or VirtualBox for Mac/Win/Linux with OpenFlow v1.0 tools already installed.

The disadvantages are:

Mininet-based networks cannot (currently) exceed the CPU or bandwidth available on a single server.

Mininet cannot (currently) run non-Linux-compatible OpenFlow switches or applications; this has not been a major issue in practice.

Question 5.3: Explain how the open flow tables are created?

Ans:

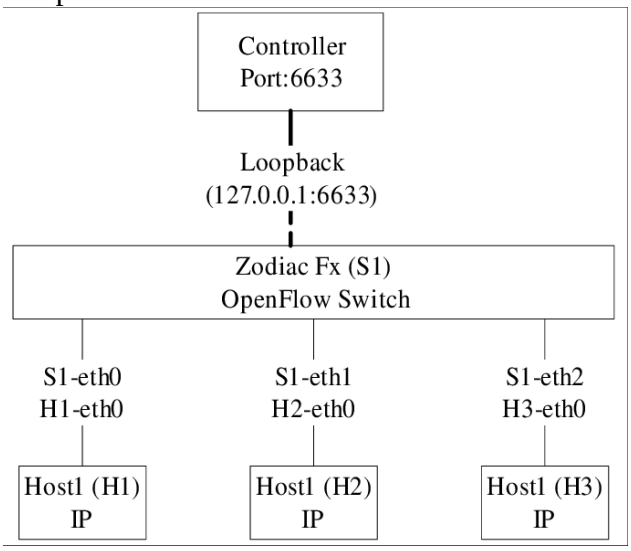
The OpenFlow protocol allows a server to instruct network switches where to send data packets. In a non-OpenFlow or legacy switch, packet forwarding (the data path) and route determi- nation (the control path) occur on the same device. A switch using the OpenFlow protocol separates the data path from the control path

Question 5.4: When Zodiac FX receive a packet, which functions an OpenFlow Switch performs:

Ans:

Software-defined network (SDN) is a new programmable networking designed to perform tasks easier by enabling network administrators to add network control via a centralized control platform and open interfaces. Common network use procedures such as traffic shifts, troubleshooting and various types of device configuration. Thus devices are needed to reconfigure with the network, in order to create a reaction with events. In this paper, we have developed an SDN testbed using Zodiac FX a hardware switch for OpenFlow experiment. This research is utilized Zodiac FX a hardware switch to test the usage and discuss about the SDN controllers. Ryu controller is configured for testbed development. The main contribution of this paper in threefold as follows: first it provides the configuration of Ryu

controller, second, it provides the configuration of Zodiac FX switch and lastly it develops a simple SDN testbed for OpenFlow.



Question 5.6: Provide your personal opinion about OpenFlow protocol? Focus on the functionalities and flexibility.

Ans:

The OpenFlow functionality provided by the OpenFlow packages in the Floodlight, Beacon, and OpenDaylight controllers is essentially the same. Applications that run on Floodlight run with little modification on Beacon and OpenDaylight.

OpenDaylight provides an additional abstraction called the *Service Abstraction Layer* (SAL). SAL purportedly allows network programmers to write applications similarly to the way they would be written calling the OpenFlow Java APIs, with the advantage that SAL can translate the API requests into either OpenFlow or another non-OpenFlow API mechanism that may be available on the target switch. It should thus be possible to write applications that work with both OpenFlow and non-OpenFlow switches.