A Bring-Your-Own-Device Framework over Software Defined Networking

BY

KHOO WAI LUNG

A REPORT

SUBMITTED TO

University Tunku Abdul Rahman

In partial fulfilment of the requirements

For the degree of

BACHELOR OF INFORMATION TECHNILOGY (HONS)

COMMUNICATIONS AND NETWORKING

Faculty of Information and Communication Technology

(Perak Campus)

MAY 2015

DECLARATION OF ORIGINALITY

I declare that this report entitled "A Bring-Your-Own-Device Framework over Software Defined Networking" is my own work except as cited in the references. The report has not been accepted for any degree and is not being submitted concurrently in candidature for any degree or other award.

Signature : __________________________

Name : __________________________

Date : __________________________

ACKNOWLEDGMENT

I would like to take this opportunity to thank my supervisor, Dr. Liew Soung Yue who has always given me advices and guidance throughout the entire duration of my Final Year Project.

Next, I would like to thank Mr. Kong Chee Meng and Mr. Sridhar Sivanand from MIMOS Berhad, who have gave me the advices and supports throughout the entire duration of my Final Year Project.

Also, I would like to thank Dr. Gan Ming Lee who has helped me to setup a dedicated internet line for my Final Year Project. It is very helpful by giving me a lot of conveniences throughout my project development period.

Last but not least, I would like to say thank you to all the people who have ever help me throughout the entire duration of my Final Year Project.

ABSTRACT

In recent years, Software-defined networking (SDN) is emerging and intensively growth as one of the most promising technologies. The main purpose of SDN is to allow networks become programmable and automatically. To achieve the objectives of automatically and programmable, various SDN network applications are developing by network programmers based on customer's requirements.

Regarding to my Final Year Project, I have developed a Bring-Your-Own-Device framework over Software Defined Networking in FYP lab. The purpose of my project is to overcome the issues of users overload and misuse internet resources. In the SDN network, I am able to control and manage the network using my own developed application named MySDNApp. This application can allow only registered users connect to the SDN network access point and router. Moreover, in order to reduce the misuse of internet resources, MySDNApp also provided the function which restrict to access different web sites among different users.

TABLE OF CONTENTS

1.0 Introduction

1.1 Introduction of Traditional Network

Few years ago, the network environment has started a tremendous growth of using high bandwidth requirements services. These services included cloud services, server virtualization, mobile personal devices, immense data transfer, etc. In order to adapt those complicated requirements for today's network trends, we must improve the functionality of current traditional network.

According to Peter Galvin (n.d.), traditional network is designed in hierarchical with a tree structure and it is well suit for client-server network structure. However, according to Open Networking Foundation White Paper (2012, p.2), today's networks tend to be more flexible and are with dynamic architecture, transfer immense data with minimum delay time, etc. Open Network Function White Paper (2012, p.4) has noted that with the traditional network setting it is nearly impossible to meet market and user's requirements nowadays because there are a few number of limitations of traditional networks as described below.

1.1.1 Limitations of Traditional Network

- Complexity of network re-configuration

  It is very complicated if one decides to change the structure of the network in an organization, e.g. removing devices from or adding devices to a network. To add or remove devices, it has to be involved multiple devices such as routers and switches. Moreover, network administrators need to update all protocol-based mechanisms such as ACLs, QoS, VLANs, and etc. These processes require time to complete and it could interference organization's network services.

- Scalability

  Today's networks have become very complex since they involve thousands of devices such as routers, switches, etc., that must be configured by network administrators. With oversubscription, network can be scaled based on predictable traffic patterns.

However, today's network traffic patterns are difficult to predict and calculate. According to Mary Shacklett (2013), some companies such as Facebook and Google which operate with giant network services are dealing with more complex scalability challenges. Therefore, those companies need networks that can perform in high performance in a very large scale so that they can provide better services for customers.

- Security
  Network administrators have to access privilege account and configure the network manually. In this case, security is a concern because administrators have the authority to change the network configuration such as bandwidth limitations or protocol prohibit for particular users. These problems could be caused by improper configuration also.

- Hardware and vendors lockdown

  Many enterprises are unable to develop and introduce new services in order to adapt the changing business environments due to the fixed vendor's equipment structure. Different vendors have their own network defined software which is embedded in their equipments. Therefore, network administrators need extra knowledge about the equipments in order to configure them.

  All of these limitations have to be overcome in order to adapt the tremendous growth of mobile and cloud technologies in the market. Thus, new network architecture has to introduced and deployed to replace traditional network architecture.

1.2 Introduction to Software Defined Networking (SDN)

To adapt complex requirements of current market, there arises a need for programmable networking which allows network administrators to control and manage network services through software, and this is known as software defined networking (SDN). Such network architecture can perfectly match today's network requirements and allow network operators to tune the network into their individual network environment. According to Brandon Carroll (2013), SDN allows us to decouple the control planes and data planes physically from a switch or router. With this concept, control plane will function on a SDN controller and the data planes functions remain in the switch.

The controller can only communicate with a switch data plane through a specific protocol (such as Openflow). Moreover, SDN architectures provide better Application Program Interface (API) and implement into network services such as routing, quality of service, access control, bandwidth allocated management and etc. With better APIs between SDN control and application layer, business applications are easier to operate on a network. Figure 1.2.1 and 1.2.2 has shown the traditional and SDN network architecture.

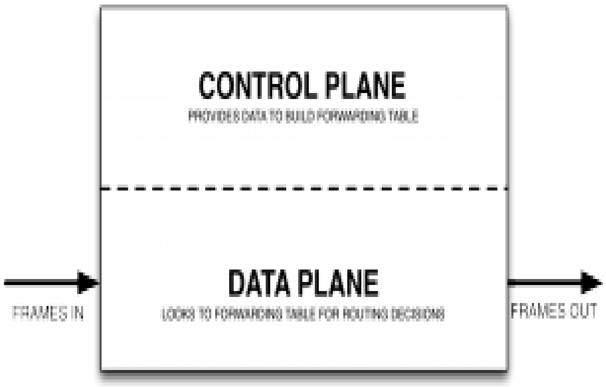

CONTROL PLANE
PROVIDES DATA TO BUILD FORWARDING TABLE
DATA PLANE
LOOKS TO FORWARDING TABLE FOR ROUTING DECISIONS
FRAMES IN
FRAMES OUT

Figure 1.2.1: Traditional network architecture, a switch or router consists of Control Plane and Data Plane. (Software Defined Networking vs. Traditional Networking 2013)

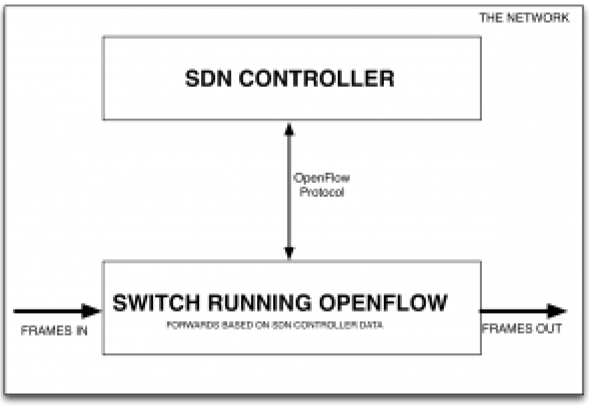


Figure 1.2.2: SDN network architecture which decoupled the Data Plane and Control Plane. (Software Defined Networking vs. Traditional Networking 2013)

According to Nick McKeown (n.d.), a specific protocol is required when a SDN controller is going to communicate with a data planes switch. OpenFlow is a new protocol which is purposely designed to implement with the SDN technology. OpenFlow follows the concept of flow table, it allowed network traffics to be identified based on the per-defined network rules which are programmed by SDN control software.

Figure 1.2.3 shows the structure of OpenFlow device and figure 1.2.4 shows the network structure of SDN.

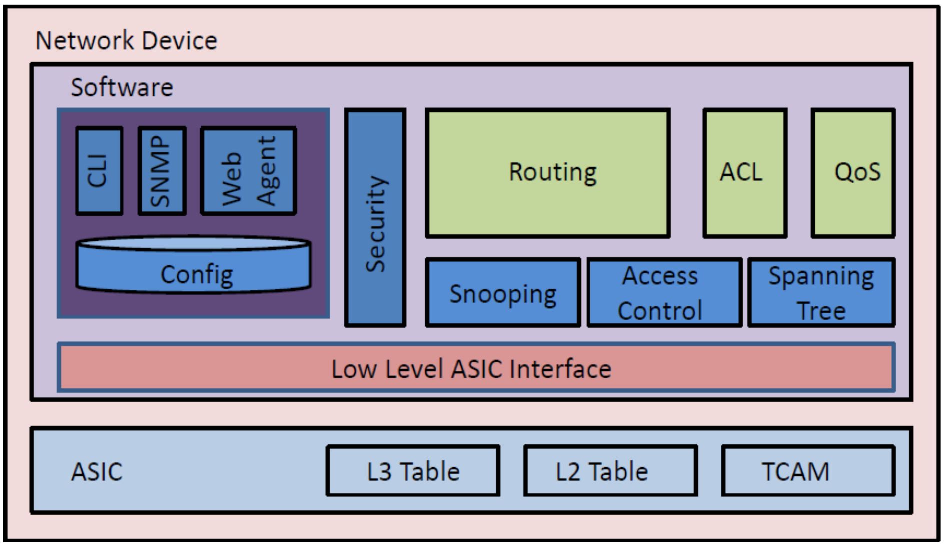


Figure 1.2.3: Network devices with OpenFlow. Note. From 'Training Slides_part 1 - Developing OpenFlow SDN Applications Based on HP ExpertOne Learning Guide' (2014, p.28)

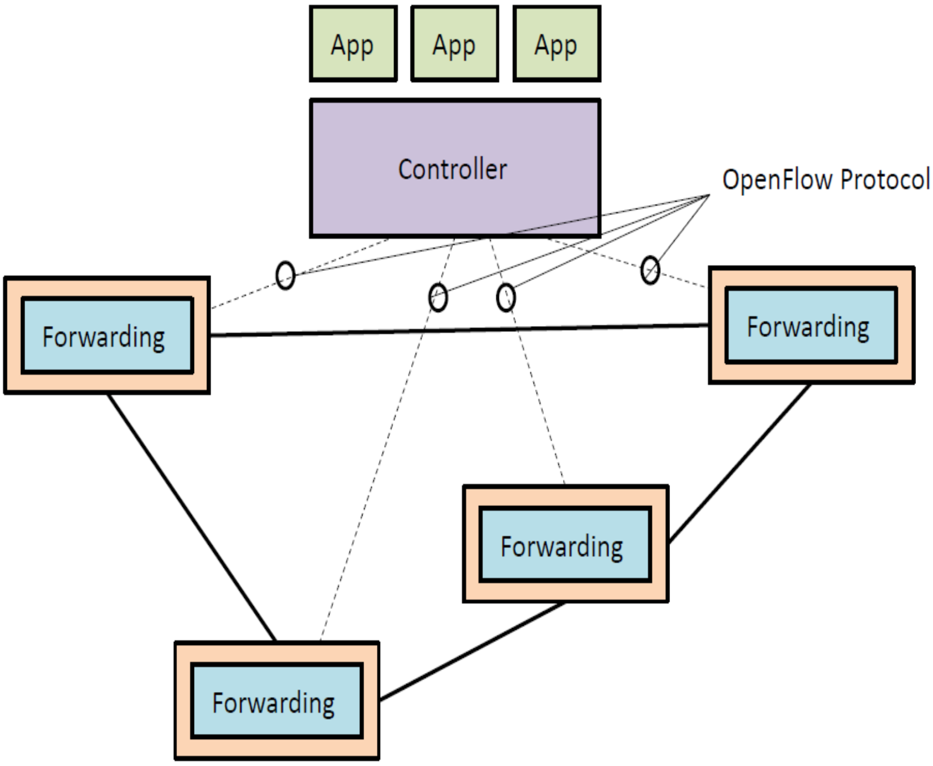
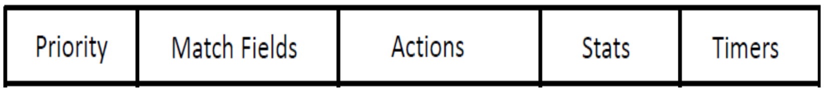


Figure 1.2.4: Network structure with OpenFlow protocol. Note. From 'Training Slides_part 1 - Developing OpenFlow SDN Applications Based on HP ExpertOne Learning Guide' (2014, p.27)

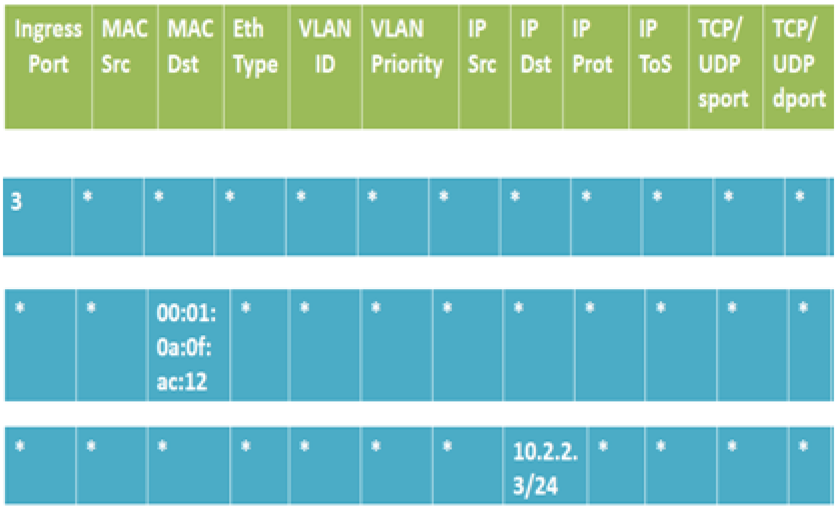
1.2.1 OpenFlow Table

In Software defined networking, switches and other network devices which support OpenFlow protocol must have at least a flow table. The number of flow tables required depends on the version of Open Flow from version 1.0 to 1.3. The function of the flow table is to manage the network packets by follow the action setting. For instance, action can be to drop the packet, forward the packet to a controller, or to forward the packet to a specific port, or other possible command. Figure 1.2.5 shows an example of flow table.



Priority	Match Fields	Actions	Stats	Timers

Figure 1.2.5: OpenFlow table attributes. Note. From 'Training Slides_part1 - Developing OpenFlow SDN Applications Based on HP ExpertOne Learning Guide' (2014, p.45)



Ingress Port	MAC Src	MAC Dst	Eth Type	VLAN ID	VLAN Priority	IP Src	IP Dst	IP Prot	IP ToS	TCP/ UDP sport	TCP/ UDP dport
3	*	*	*	*	*	*	*	*	*	*	*
*	*	00:01: 0a:0f: ac:12	*	*	*	*	*	*	*	*	*
*	*	*	*	*	*	*	10.2.2. 3/24	*	*	*	*

Figure 1.2.6: OpenFlow entries. Note. From 'Training Slides_part1 - Developing OpenFlow SDN Applications Based on HP ExpertOne Learning Guide' (2014, p.48)

The row with green colour is attributes of the flow match fields while rows with blue colour are flow entries. When a flow goes into an Openflow table, it will find and match the "Match

Field" attributes (figure 1.2.6). For instance, if the first flow entry in figure 1.2.6 match with the flow table's "Match Field" attribute, it will be forward by follow the flow table's action.

1.2.2 **Advantages of SDN** (Defined Networking vs. Traditional Networking 2013)

- Centralize control of different vendor's network devices

  Centralized control and management of multi-vendor's network devices such as routers, switches or even virtual switches are achievable by using SDN software controller, as long as these network devices can support OpenFlow protocol. Moreover, with the centralized control of network devices which are using OpenFlow-base SDN, upgrade and install new network devices in the network can become easier because the configurations for every devices are consolidate even they are from different vendors.

- Simple configurations for network services

  SDN software application has provided a better user interface. Kate Greene (2009) noted that network administrator can use SDN software in their computers to control and manage the network services such as routing, quality of service, access control, bandwidth allocated, traffic flows and etc with just a click on mouse.

- Improved management through automation

  OpenFlow-based SDN provide us a network automation management framework which can help to automate most of the management tasks which are manually done previously. Furthermore, automate network management can help to improve network stability and business agility by reducing operational overhead.

- Programmability can improved network security

  Programmability of SDN has provided the network with ability of automation management. In SDN architecture, network automation is done through programming code and the communication between controllers and switches are encrypted. With the encrypted communication, any human intervention to the network is not allowed. Additionally, it can also prevent problems that cause by configuration mistakes that always made by network administrators.

- Increase the rate of innovation

  Programmability of SDN allows network operators to reprogram their network and tune it into their individual network environment. In order to adapt changing business environment, network operators can program their own network to meet particular customer's requirements. Programmable network also provide the innovation for new services. Thus, many of the new services and ideas those are impossible to achieve in traditional network, now they can done through SDN.

  In a nutshell, SDN provide a solution to adapt the tremendous user's growth of cloud services, server virtualization, mobile personal devices and etc. Also, in order to adapt rapidly changing business environment, performance of current network must be improve to meet certain requirements. Thus, SDN will be the key solution and the future trend of network architecture.

1.3 Project Scope and Objectives

1.3.1 Project Scope

This project is to develop a framework over Software Defined Networking with a SDN application installed in the OpenDaylight Hydrogen Controller name MySDNApp. All the network equipments are setup at FYP Lab N-109. MySDNApp has the function to allow particular users who had registered their device's MAC address to connect the SDN network and access the internet. Besides, MySDNApp also provided the function to restrict different users to access different web sites. With this application, we can manage the number of users and the availability of the network connection in FYP lab.

1.3.2 Problem Statement

There are a few problems that the FYP lab is facing. One of those is the overload users on the access point and routers. There are a lot of desktop computers are connecting with the FYP lab router. Sometimes students and lecturers may bring their own devices and connect with the FYP lab's wireless access point or using Ethernet cable to connect the router. Besides that, there are also many unknown devices which will connect to the lab's wireless access point and access internet. These have reduced the network availability in the FYP lab. Furthermore, sometimes students and staffs could misuse the internet resources for their own purposes such as downloading, streaming high quality video and etc. This may reserve all the bandwidth allocation and cause the network become slowly.

1.3.3 Project Objectives

Main objective of this project is to create a better network environment for students, lecturers and staffs in FYP lab. According to Open Network Foundation White Paper (2013, p.3), SDN router is programmable and it provides automated provisioning of network connection to accommodate scheduled and tremendous data transfer. Thus, SDN network is easy to manage and maintain. With the MySDNApp we have developed, we can control users who are eligible only to connect the wireless access point and router to prevent them from users overload. Besides that, with the function of restrict different users to access different web sites can help to reduce the network misuse among users and improve the network availability. The last objective of my project is to encourage other students to extend and explore the SDN. With a well developed framework over SDN, students can explore and extends the current SDN system by create some functions or application with their innovative idea.

2.0 Literature Review

2.1 Open Source Virtual Software Router

Software Router is a lower cost option to accommodate expensive hardware router but it only provide moderate performance. The software router that is running in a virtual environment is defined as virtual software router. Because it is open source and free, users do not constraints by the hardware and vendors. However the software router provides only limited functions and its moderate performance is not capable of supporting a large network such as a campus network, cloud network, and etc. Compared to the software router, network defined networking has the advantages to support a large and big data network because network administrator can define whatever network functions they want by develop SDN application. Moreover, the performance of SDN network is surely better than software router because the SDN network requires more expensive devices such as switch which can support SDN and more powerful server. Thus, we decided to choose SDN as our project title because it has better performance and capable to support campus network. Below are 2 examples of software router.

2.1.1 **Vyatta** (vyatta.org 2008)

Vyatta is a Linux based routing software which is open source and developed by the Vyatta Company in year 2005. Vyatta support of large panel of normalized network protocols and it can improve the network security. Because of it is free and open source, it can be audited by other external companies in case of the security issues and almost everyone can solve these issues. However, Vyatta is less functionality such as VPN compared to the commercial router products.

2.1.2 **VyOS** (vyos.net 2013)

VyOS is an open source Linux based software router which is running in virtual environment. Besides, a free and open source routing platform that competes directly with other commercially solutions are provided by VyOS. Because VyOS is running on any x86 based hardware in 32 bit or 64 bit, so it is able to be used for cloud deployments.

2.2 Network Function Virtualization (NFV)

According to Open Network Foundation White Paper by Don, Peter and Andy (2012, p.5), Network Functions Virtualization (NFV) is based on the concept of SDN and they are complementary. However, they do not depend upon each other. NFV provides a new method to design, implement and manage networking services.

Different from SDN, NVF decouples networking functions such as network address translation (NAT), intrusion detection service, domain name service (DNS), and etc. NVF is designed to improve the networking components which are required to support a virtualized infrastructure. These infrastructures included virtual servers, storage and other networks. It is use to virtualized network functions in high-volume service, switch and storage hardware.

According to Sterling Perrin et. al. (2013, p.3), the main advantage of NVF is reducing operating and capital expenses by consolidate the network equipment and reduce the need to purchase specific hardware. Besides, it also reduces the space and energy consumption of hardware. All these benefits are because the network function is executed by software but not hardware anymore. However, NFV is only target and benefits to the service provider networks and ignore the public sector site such as cloud, campus and etc. At the end, we decided to choose SDN over NFV as our project title because the reason of the targeted customers. Below are the comparison between NFV and SDN.

T	SDN	NFV
a Decouple b l	Control Plane and Data Plane from network switch	Network functions, eg. NAT, DNS
e Targeted Customers 2 . 2	Targeted big data network, campus network, cloud and etc.	Targeted service provider network and ignore public sector site such as cloud and campus network

Table 2.2.1: Comparison between NFV and SDN.

3.0 Method and Technologies involved

3.1 Method Involved

In this project, we will choose rapid application development methodology because it can develop product faster and with higher quality.

PPDIOO is a Cisco methodology which uses to apply in network services implementation and design. It is stands for prepare, plan, design, implement, operate and optimize which are the 6 stages to define the continuous life-cycle services in a network. In the other hand, PPDIOO has provided us some benefits. The main benefit is to lower the total cost of ownership. In the early stages, we can evaluate and determine the technology requirements and we can make a plan to change the requirements before design the network proposed. Moreover, PPDIOO can also increase the network availability and the speed access to services and applications. Below are the 6 stages of PPDIOO. The cycle of PPDIOO methodology has shown in the Figure 4.1.1.

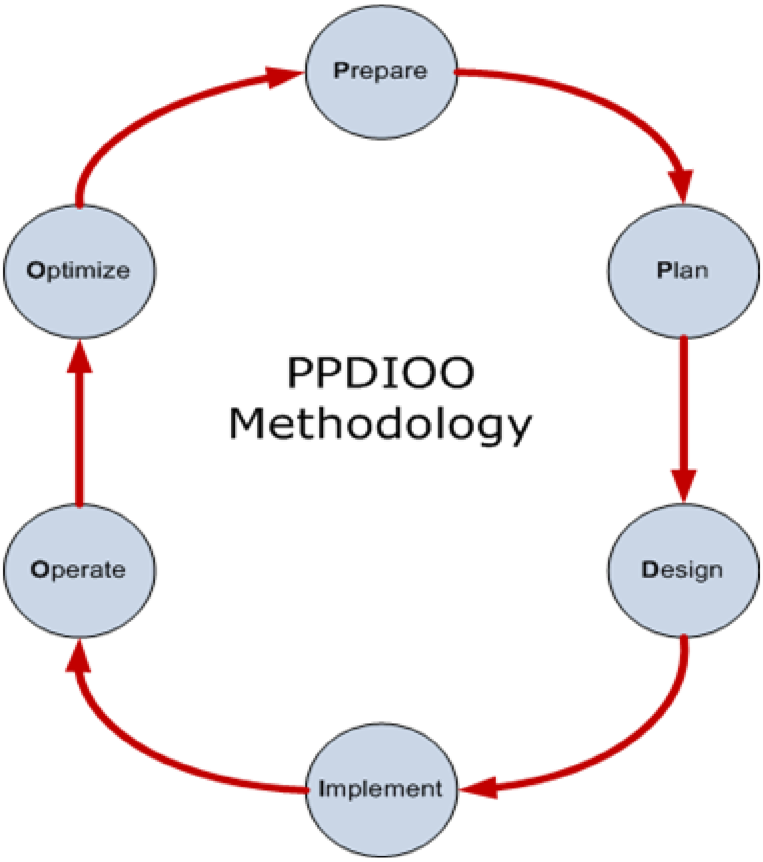


Figure 3.1.1: The life cycle of PPDIOO Methodology.

3.1.1 Preparation

Prepare stage discuss about the organizational requirements, network strategy development and technologies that is use to support the network architecture. This stage can constitute a justification of financial for the network strategy by assessing and evaluating the network strategy and organizational requirements. I am going to define the requirements of to setting up the SDN BYOD framework and application design in this stage.

3.1.2 Planning

Plan stage is use to discuss about the initial network requirements based on facilities, user needs, goals and etc. This stage also uses to determine and evaluate whether the existing SDN controller can support the MySDNApp application. Additionally, a good project plan can help to manage and execute project's tasks with more efficiently and effectively. In plan stage, I will also determine all the tasks and functions that are require in my application.

3.1.3 Design

In this stage, design specification is the foundation for implementation. Thus, design specification should meet the business and technical requirements. It is concern about reliability, availability, scalability, security, and performance. I will start to create and design the functions of the application.

3.1.4 Implementation

This stage is concern about all the operations and configurations. If there are any changes in this stage, it must be inform to all the team members first. After that we made the require modifications in plan and design stages. After the design stage, I will start to upload and configure MySDNApp in OpenDaylight SDN controller.

3.1.5 Operation

This stage included maintains the network condition and status through routine operations. These tasks included monitoring performance of difference devices and equipment, reports generating, maintenance and etc. I am going to monitoring and maintaining the MySDNApp application after the installation. I should ensure the application is in good condition and can perform routine activities without any problem.

3.1.6 Optimization

Optimize stage is a proactive network management. It is use to identify and resolve the problems encounter before it affect the whole system. If there is necessity, we can go back to previous stages such as plan and design to make modifications and solve the problems. After that we can test the changes in operate and optimize stages. Proactive network management allow us to find and solve any problems in the BYOD framework and MySDNApp application before it becomes serious.

3.2 Technology Involved

According to Lee Doyle (n.d.), software-defined network does not require any special SDN switches but the controller (a computer with Virtual Machine) need a specific protocol that uses APIs (application programming interfaces) to configure switches within the network. However, different vendors have their unique way to implement SDN. In this project, I am going to develop a Bring-Your-Own-Device framework over SDN with an application named MySDNApp.

The devices we are going to use are HP Switch 2920-24G which supports OpenFlow 1.0 and later version in an OpenFlow-only network, a TP-Link Wireless Access Point, and a Cisco router model 1841. The communication between all the network devices must use IPv4 addresses because the OpenDaylight controller does not support devices with IPv6.

Furthermore, Mininet, Eclipse LUNA, and Ubuntu 12.04 are required for me to develop and test the MySDNApp application. Mininet can used to create a virtual network and generate network traffic. I will use Eclipse Junior with Java Programming Language to develop the application. After that I will install the SDN controller software - OpenDaylight Hydrogen in a server running Ubuntu 12.04 Operating

System. In the end, we will test and run the MySDNApp application in a real network environment instead of a virtual network environment created by mininet. Below are the descriptions for all the hardware and software we will be use in the project.

3.2.1 OpenDaylight Hydrogen Controller Software

Minimum Hardware Requirements

- Processor: 3.0 GHz x86-64 quad core processor
- RAM: 8 GB
- Disk Space: 40 GB
- Ethernet NIC: One 1 Gbps Ethernet NIC

Recommended Hardware Requirements

- Processor: 3.0 GHz x86-64 16-core processor
- RAM: 32 GB
- Disk Space: 64 GB
- Ethernet NIC: One 10 Gbps Ethernet NIC

Software Requirements

- OpenFlow version supported: OpenFlow 1.3.1
- Operating System: Ubuntu 12.04 LTS 64-bits Server
- Java version: Patch 25 of Java 7 JRE or later version
- Supported Web Browser: Google Chrome 30.0.1599.101m
- Internet services must be available in order to download the controller software and other prerequisite software

3.2.2 MySDNApp

This is a SDN application that developed by us will be installed in the OpenDaylight Hydrogen controller software. Since this is an OSGI bundles java project, we are required to use the Maven to build and compile the project. This application provided the functions which allowed only registered users (register with device’s MAC Address) to connect the SDN network and restriction of web sites accessibility among different users. Moreover, it also provide the basic function to forward packets within the SDN network.

Software Requirements for MySDNApp:

- Eclipse LUNA
- Maven 3.0 or later version

3.2.3 Additional Hardware

- TP-Link Wireless Router
- Cisco Router with Model 1841
- HP SDN Switch 2920-24G

HP SDN Switch model 2920-24G is the core equipment in this project because it provided the SDN and Openflow function. The TP-Link Router will be used to connect the HP Switch and provide the wireless access point for users. The Cisco Router 1841 is use to connect the HP Switch and route all the network traffic from the SDN network in order to gain internet connection.

3.3 Gantt Chart

3.3.1 FYP 1 Gann Chart

Task	**Week**													
	1	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**
Review of other SDN Applications	🟩	🟩	🟩	🟩										
Understanding Open Flow Table			🟩	🟩	🟩	🟩								
Develop SDN Applications With Simple Functions					🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩
Identify Application Functional Modules						🟩	🟩							
Develop Project Application							🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩

3.3.2 FYP 2 Gantt Chart

Task	Week													
	1	2	3	4	5	6	7	8	9	10	11	12	13	14
Develop Main Functions for Project Application	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩					
Project Application Testing and Debug		🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩	🟩				
Completed Project Application Demo											🟩			
Project Reports and Documentations											🟩	🟩	🟩	🟩

4.0 System Design

4.1 System Architecture

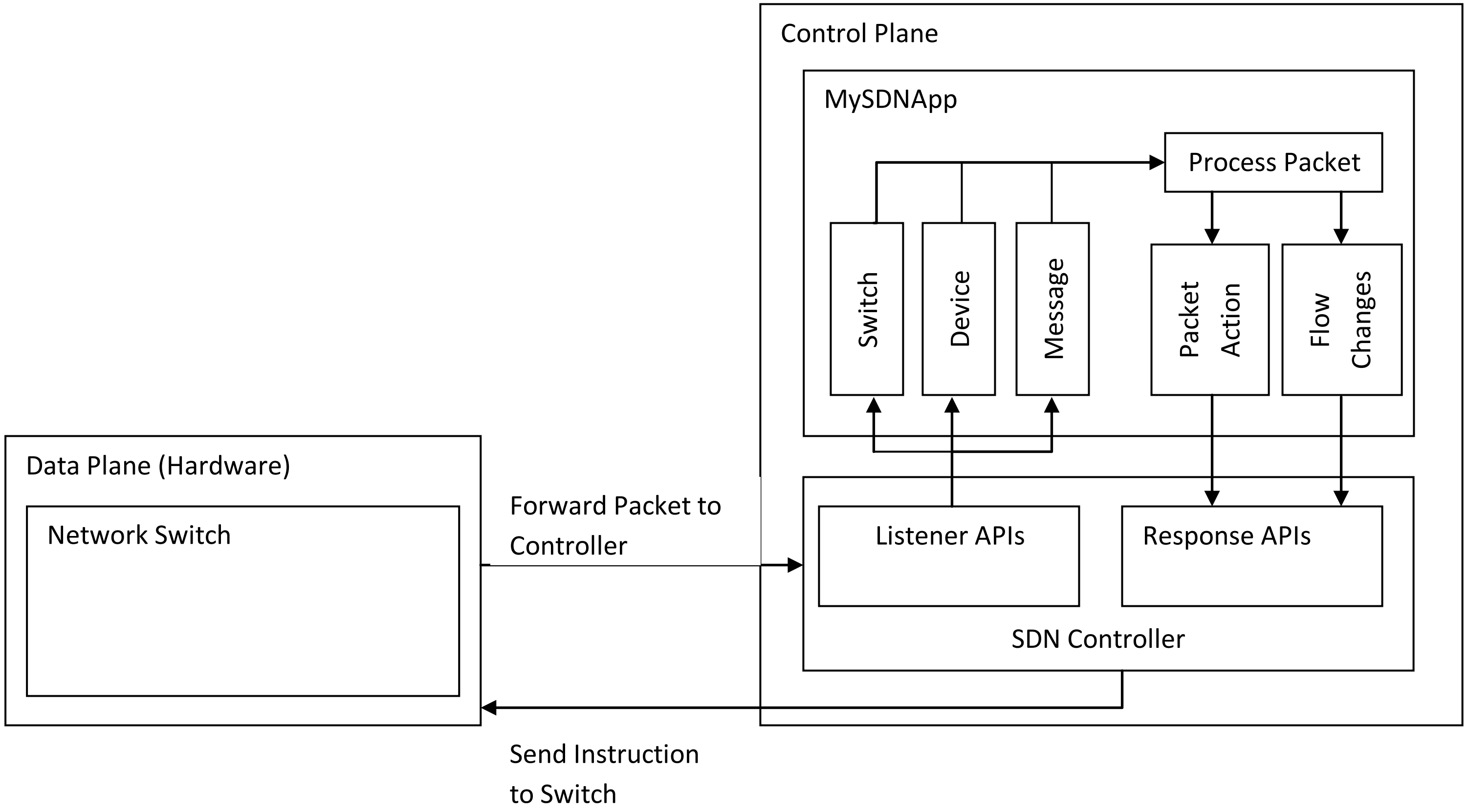


Figure 4.1.1: System Architecture of MySDNApp and SDN Controller.

Figure 4.1.1 is the architecture of the whole system which included hardware switch, SDN controller, and MySDNApp Application. From the hardware switch, every time it has received a new packet it will forward to the SDN controller. The SDN controller has the Listener APIs to handle the packet and send it to the MySDNApp Application. After that, the application will process the packet and define which action should be done such as create a flow or ignore the packet. This instruction will send back to the hardware switch through the Response APIs of SDN controller.

4.2 Functional Modules of MySDNApp

- **start()**

  A module that uses to inform network administrator MySDNApp has started.

- **stop()**

  A module that uses to inform network administrator MySDNApp has started.

- **parseMacAddress()**

  This module is use to translate the mac address from String type to byte. It will return the Mac Address in bytes.

- **intToInetAddress()**

  This module is use to translate every raw packet to an ip address object. It will return the ip address in bytes.

- **btyeToString()**

  In order to display the Mac Address on the OpenDaylight console, we need this module to translate the Mac Address to String type. I will return the Mac Address in String type.

- **receiveDataPacket()**

  When the packet has sent from the switch to controller, receiveDataPacket() is required for MySDNApp in order to capture the raw packet and decode it in the further process. This is the main module in MySDNApp that provide the core functions which give the instructions such as drop the packet, create flows, remove flows and etc.

4.3 Modules to Setup the SDN Network in FYP Lab

- **Install OpenDaylight Hydrogen in Server Computer**

  Since OpenDaylight is an open source SDN controller platform, we can easily find the download source from internet and install it into the server computer which is running Ubuntu 12.04 64 bits Linux operating system.

- **Create VLANs and Openflow Instance in HP Switch 2920-24G**

  In order to enable the openflow, an openflow instance and VLAN must be created. We have created an openflow instance named "Opendaylight" and configured the VLAN 20 as a member in the "OpenDaylight" intance. In my project, The SDN controller will be remains in the default VLAN while the VLAN 20 will be assigned to the "Opendaylight" instance. All the configurations are done through the HP switch console.

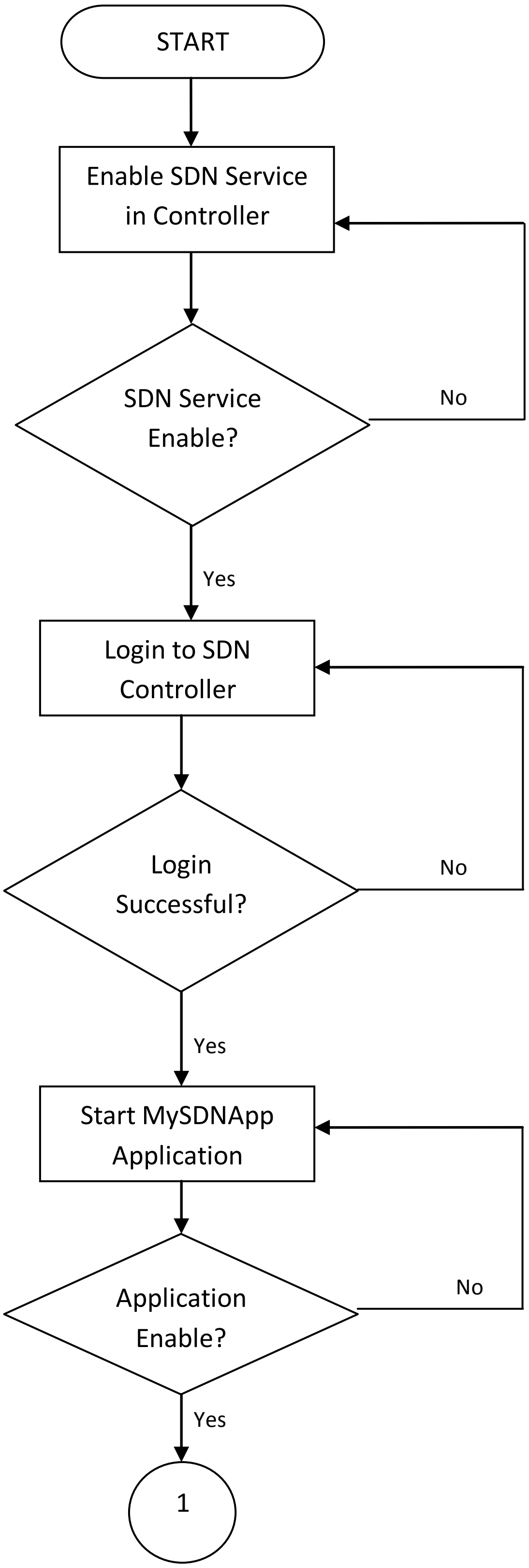
- **Configure Cisco Router model 1841**

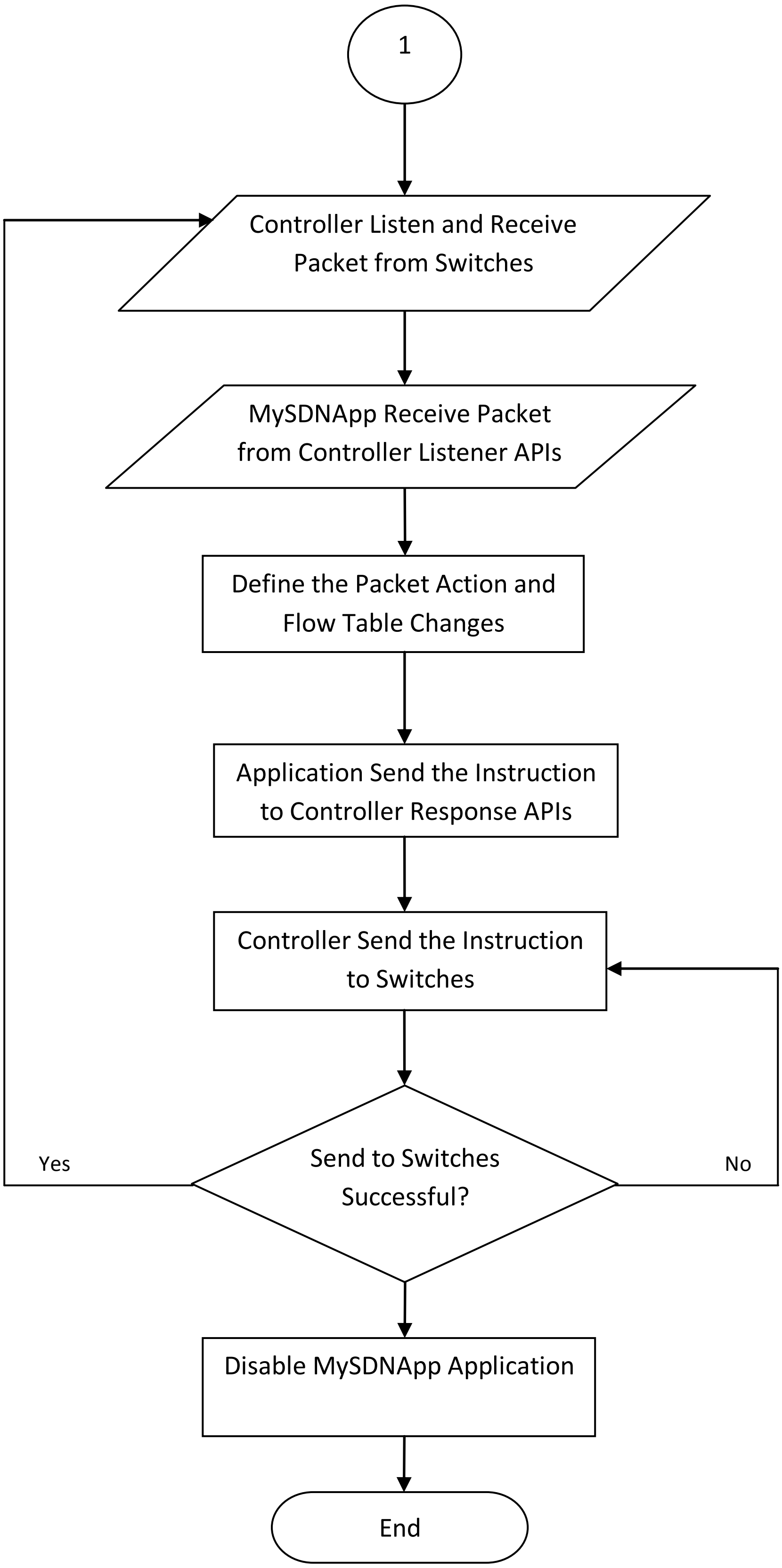
  The main purposes of the Cisco router 1841 are to create a trunk port between the HP switch and itself in order to allow different packets among VLANs route to the internet. Also, to access the internet, Network Address Translation (NAT) has been configured in the Cisco router in order to translate the IP addresses from different VLAN network into the public IP.

- **Setup and Configure TP-Link Wireless Access Point**

  Since the HP switch 2920-24G does not has the function to broadcast wireless connection, so we decided to connect a TP-Link wireless access point to the HP switch. The wireless access point will broadcast its Service Set Identifier (SSID) and enable the wireless connection to the wireless devices nearby the FYP lab.

4.4 System Flow Chart




1
Controller Listen and Receive Packet from Switches
MySDNApp Receive Packet from Controller Listener APIs
Define the Packet Action and Flow Table Changes
Application Send the Instruction to Controller Response APIs
Controller Send the Instruction to Switches
Send to Switches Successful?
Yes
No
Disable MySDNApp Application
End

5.0 System Implementation

In order to implement my project system, there are 2 essential parts I need to complete which are software and hardware setup.

5.1 Hardware Setup

5.1.1 SDN Network Topology

To develop a framework over SDN, the first part that I need to do is design and set up a SDN network with real equipment. Below is the SDN network topology.

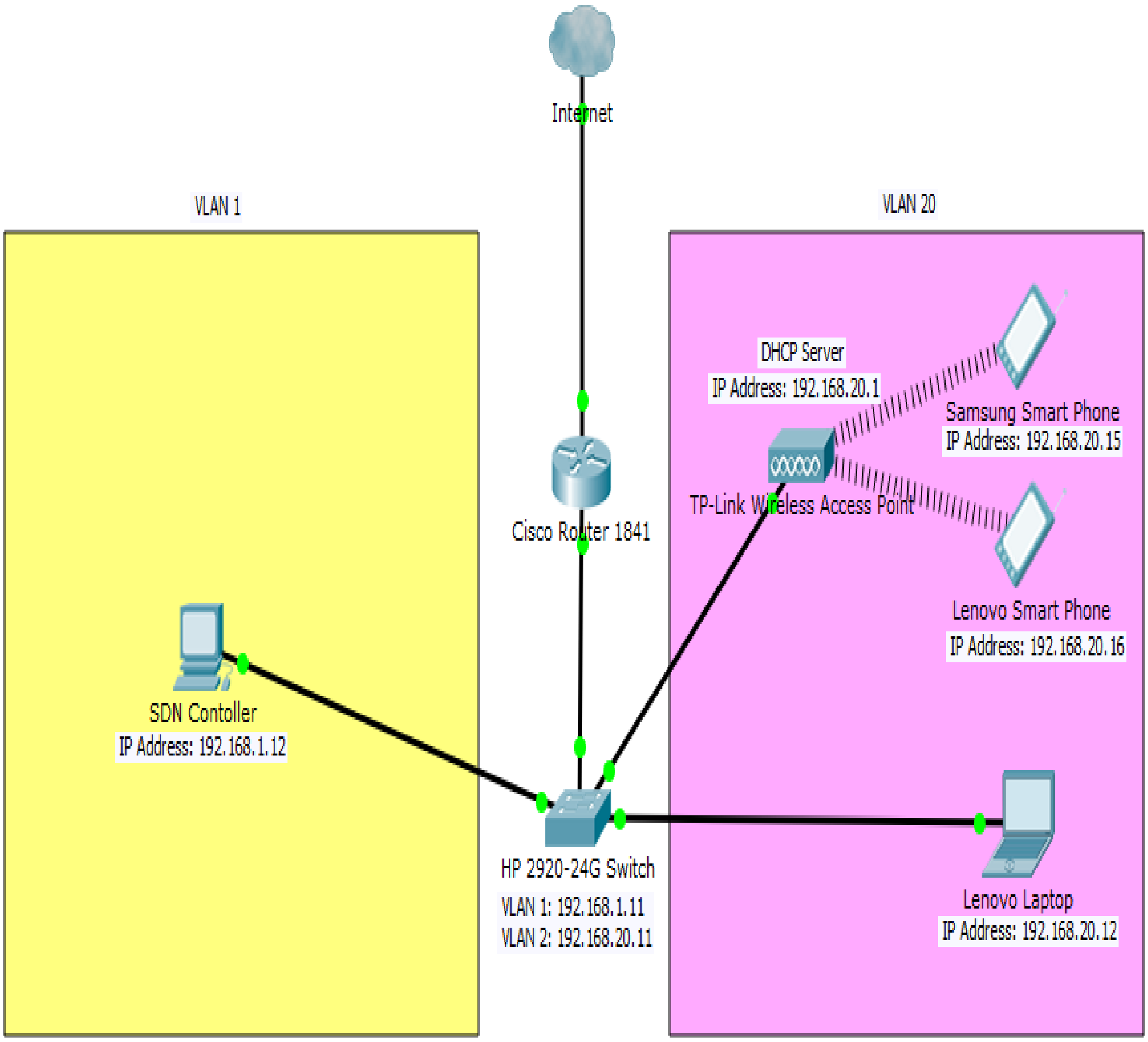


Figure 5.1.1.1: SDN Network Topology.

5.1.2 HP 2920-24G Switch

Based on the SDN topology, I have designed 2 VLANs which are VLAN 1 and 20. I have assigned port 5 to port 12 to VLAN 20 while others ports are remain with default VLAN which is VLAN 1 except the Port 23. Port 23 has been set as a trunk port that allows the traffic from both VLAN 1 and 20 to pass through. Also, HP Switch 2920-24G use a Fast Ethernet cable to connect the Cisco Router 1841 though Port 23.

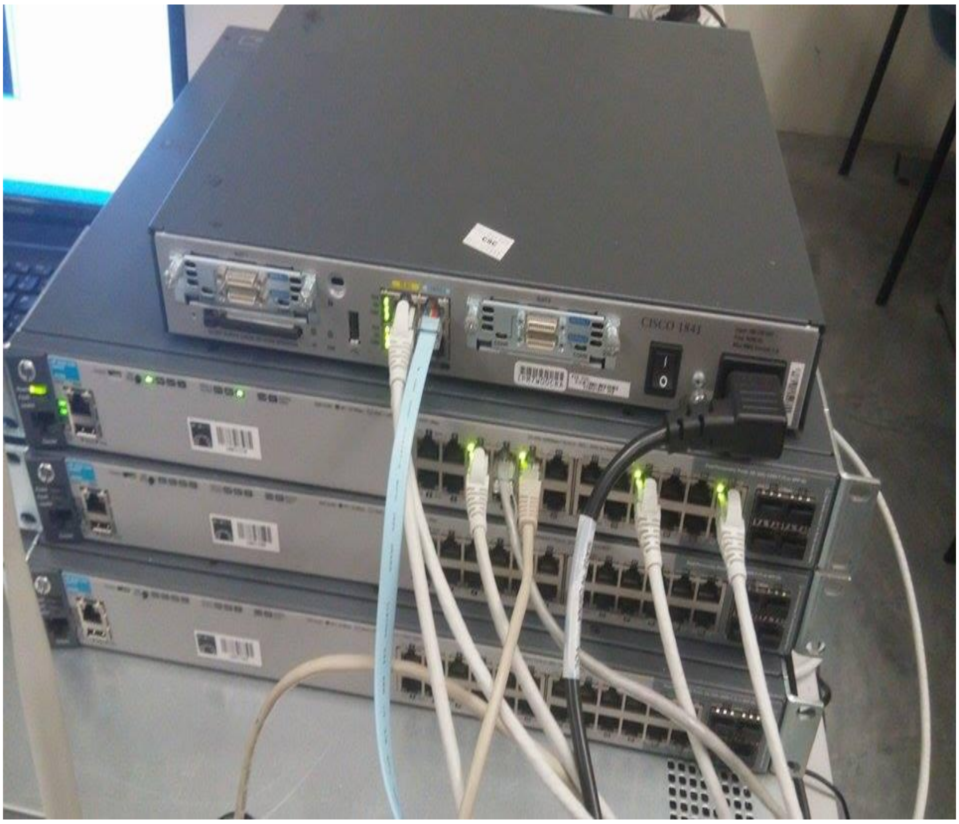
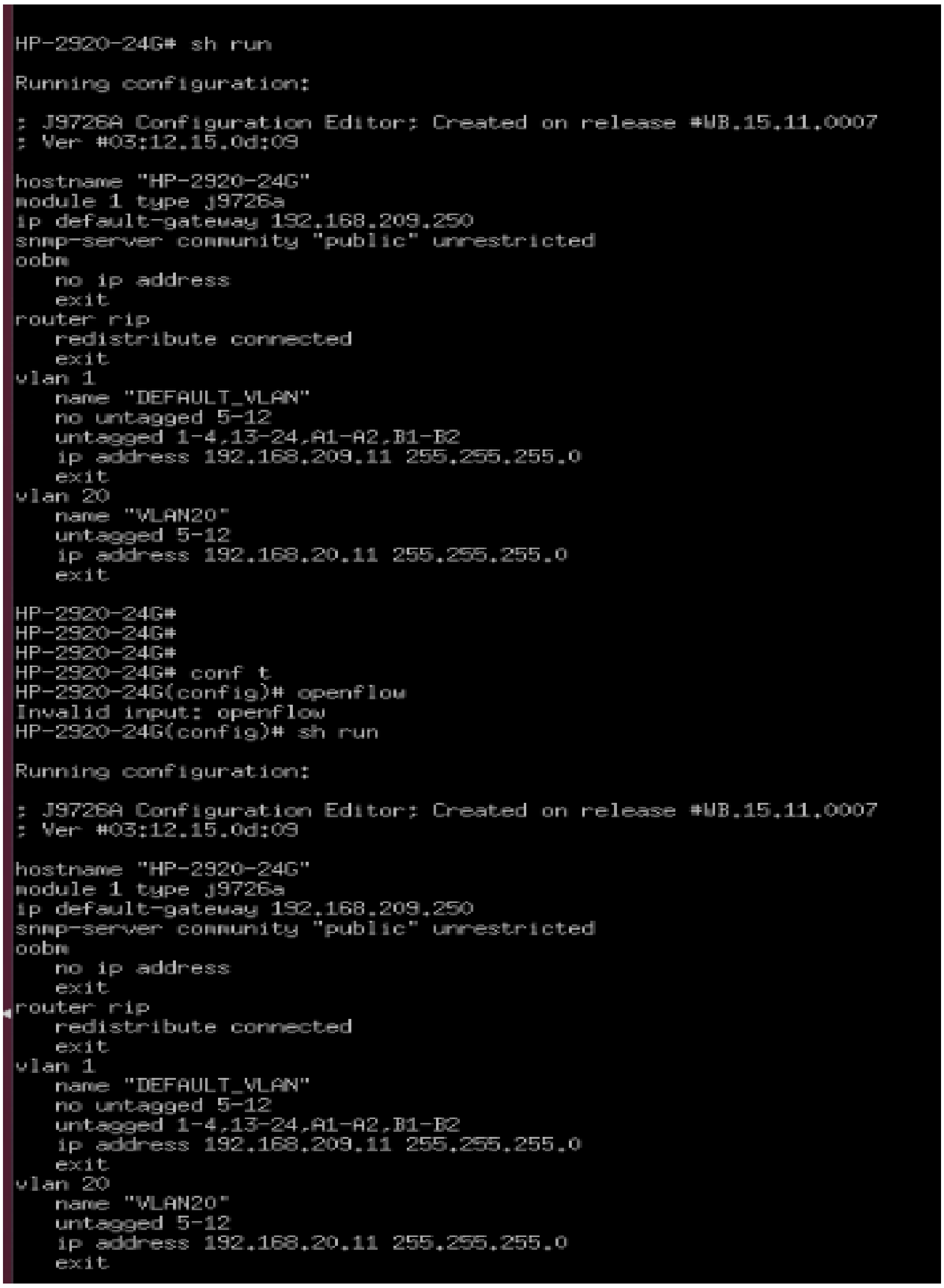


Figure 5.1.2.1: The device on the second row with light on is HP Switch 2920-24G while the device at the top is Cisco Router 1841.



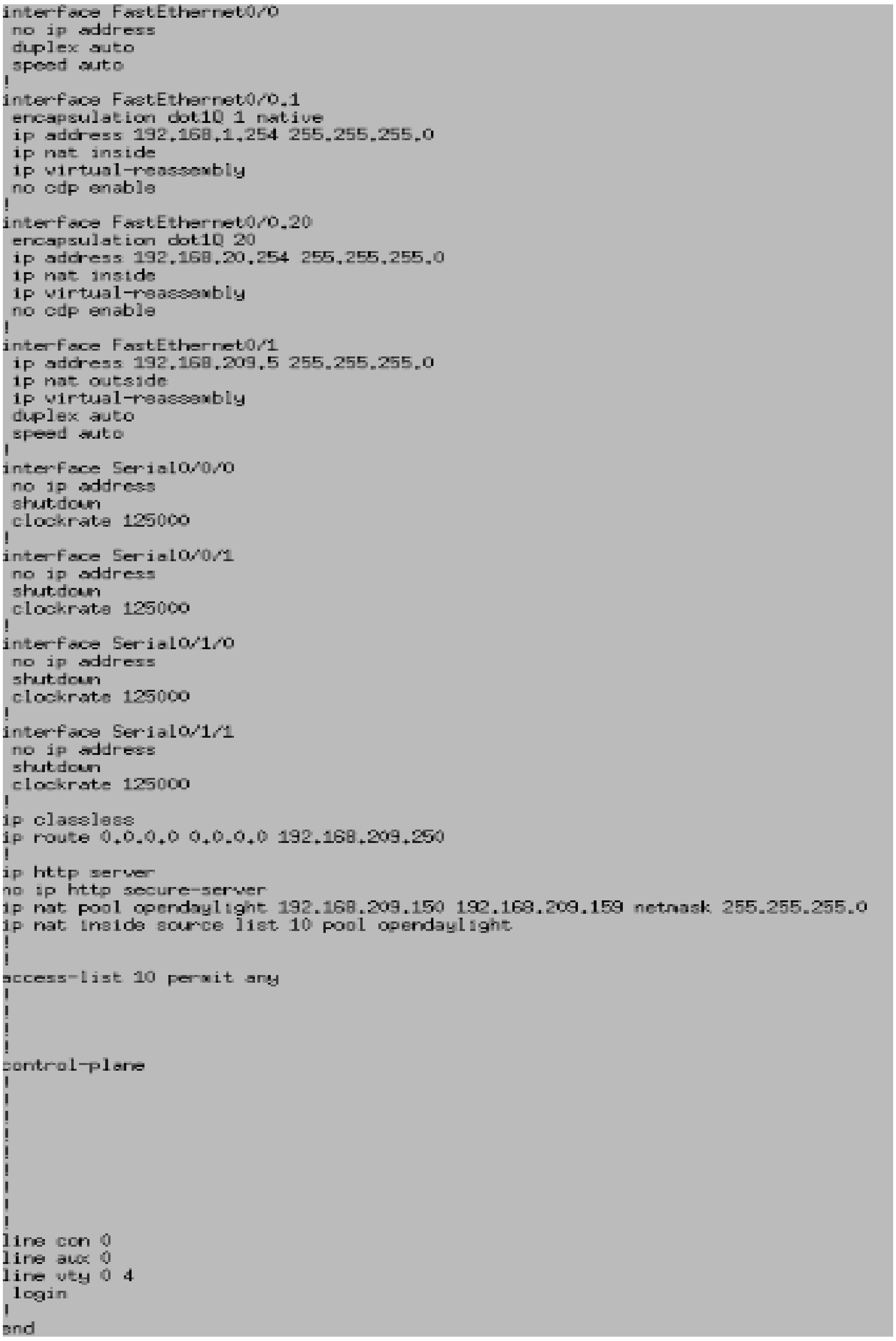
```
HP-2920-24G# sh run

Running configuration:

; J9726A Configuration Editor; Created on release #WB.15.11.0007
; Ver #03:12.15.0d:09

hostname "HP-2920-24G"
module 1 type j9726a
ip default-gateway 192.168.209.250
snmp-server community "public" unrestricted
oobm
   no ip address
   exit
router rip
   redistribute connected
   exit
vlan 1
   name "DEFAULT_VLAN"
   no untagged 5-12
   untagged 1-4,13-24,A1-A2,B1-B2
   ip address 192.168.209.11 255.255.255.0
   exit
vlan 20
   name "VLAN20"
   untagged 5-12
   ip address 192.168.20.11 255.255.255.0
   exit

HP-2920-24G#
HP-2920-24G#
HP-2920-24G#
HP-2920-24G# conf t
HP-2920-24G(config)# openflow
Invalid input: openflow
HP-2920-24G(config)# sh run

Running configuration:

; J9726A Configuration Editor; Created on release #WB.15.11.0007
; Ver #03:12.15.0d:09

hostname "HP-2920-24G"
module 1 type j9726a
ip default-gateway 192.168.209.250
snmp-server community "public" unrestricted
oobm
   no ip address
   exit
router rip
   redistribute connected
   exit
vlan 1
   name "DEFAULT_VLAN"
   no untagged 5-12
   untagged 1-4,13-24,A1-A2,B1-B2
   ip address 192.168.209.11 255.255.255.0
   exit
vlan 20
   name "VLAN20"
   untagged 5-12
   ip address 192.168.20.11 255.255.255.0
   exit
```

Figure 5.1.2.2: The configuration for HP Switch 2920-24G in terminal console.

5.1.3 Cisco Router 1841

Cisco router 1841 has been set as a role that helps to connect the SDN network with the internet. Since there are 2 VLANs from the SDN network are required to connect the internet, so that I have configured a Network Address Translation (NAT) in the Cisco router. Moreover, a static route is required to configure in the Cisco router in order to route the both VLANs' traffic out of the SDN network and go to internet.

Figure 5.1.3.1: Cisco Router 1841.



```
interface FastEthernet0/0
 no ip address
 duplex auto
 speed auto
!
interface FastEthernet0/0.1
 encapsulation dot1Q 1 native
 ip address 192.168.1.254 255.255.255.0
 ip nat inside
 ip virtual-reassembly
 no cdp enable
!
interface FastEthernet0/0.20
 encapsulation dot1Q 20
 ip address 192.168.20.254 255.255.255.0
 ip nat inside
 ip virtual-reassembly
 no cdp enable
!
interface FastEthernet0/1
 ip address 192.168.209.5 255.255.255.0
 ip nat outside
 ip virtual-reassembly
 duplex auto
 speed auto
!
interface Serial0/0/0
 no ip address
 shutdown
 clockrate 125000
!
interface Serial0/0/1
 no ip address
 shutdown
 clockrate 125000
!
interface Serial0/1/0
 no ip address
 shutdown
 clockrate 125000
!
interface Serial0/1/1
 no ip address
 shutdown
 clockrate 125000
!
ip classless
ip route 0.0.0.0 0.0.0.0 192.168.209.250
!
ip http server
no ip http secure-server
ip nat pool opendaylight 192.168.209.150 192.168.209.159 netmask 255.255.255.0
ip nat inside source list 10 pool opendaylight
!
!
access-list 10 permit any
!
!
!
!
control-plane
!
!
!
!
!
!
!
!
!
line con 0
line aux 0
line vty 0 4
 login
!
end
```

Figure 5.1.3.2: The configuration for HP Switch 2920-24G in terminal console.

5.1.4 TP-Link Wireless Access Point

In order to provide the wireless connection for the users, I have set up a TP-Link wireless access point. This access point is also acting as a Dynamic Host Control Protocol (DHCP) server which can automatic assign IP addresses for users. Based on the figure 5.1.1, the wireless access point is directly connected to the HP Switch on Port 9 with Fast Ethernet cable and it is belong to VLAN 20 with IP address 192.168.20.1.

Figure 5.1.4.1: TP-Link Wireless Access Point

5.1.5 SDN Controller

A desktop computer with Ubuntu 12.04 operating system is used as the SDN controller in my project. Based on the figure 5.1.1, the SDN Controller is directly connected to the HP Switch on Port 1. It belongs to the VLAN 1 and its IP address is 192.168.1.12.

Figure 5.1.5.1: Desktop Computer that act as SDN Controller.

5.1.6 Mobile devices and Laptop

Three devices will be using in my project system demonstration. There are 2 mobile devices, Lenovo and HTC smart phones which will connect to the wireless access point and will be assigning with dynamic IP addresses. Another device is a Lenovo Laptop which will directly connect to the HP Switch with a Fast Ethernet Cable and its IP addresses is 192.168.20.12. All these 3 devices belong to VLAN 20.

Figure 5.1.6.1: Smart Phones which will connect to the SDN Network through wireless connection.

Figure 5.1.6.2: Lenovo Laptop which will connect directly to HP Switch.

5.2 Software Setup

5.2.1 OpenDaylgiht Controller Software

OpenDaylight is an open source SDN Controller software, so that it is totally free. Below are the steps to install OpenDaylight Controller software in the Controller Desktop.

Step 1: Open terminal and type in command “apt-get update”.

Step 2: Type in “apt-get install maven git openjdk-7-jre openjdk-7-jdk”.

Step 3: Type in “git clone http://git.opendaylight.org/gerrit/p/controller.git” to download the OpenDaylight Controller.

Step 4: After the download process, go to the specific folder location by type in “cd controller/opendaylight/distribution/opendaylight”.

Step 5: Type “mvn clean install” to build the opendaylight folder.

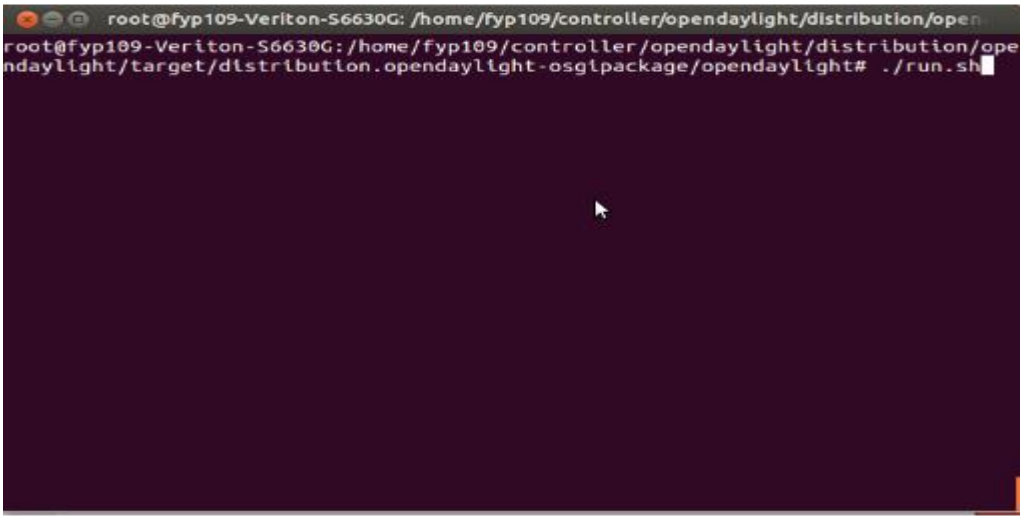
Step 6: Export the Java home by command “export JAVA_HOME=/usr/lib/jvm/java-1.7.0-openjdk-amd64”.

Step 7: Now the OpenDaylight Controller has fully installed in the Controller Desktop.

5.3 System Operation

5.3.1 Start OpenDaylight Controller Software

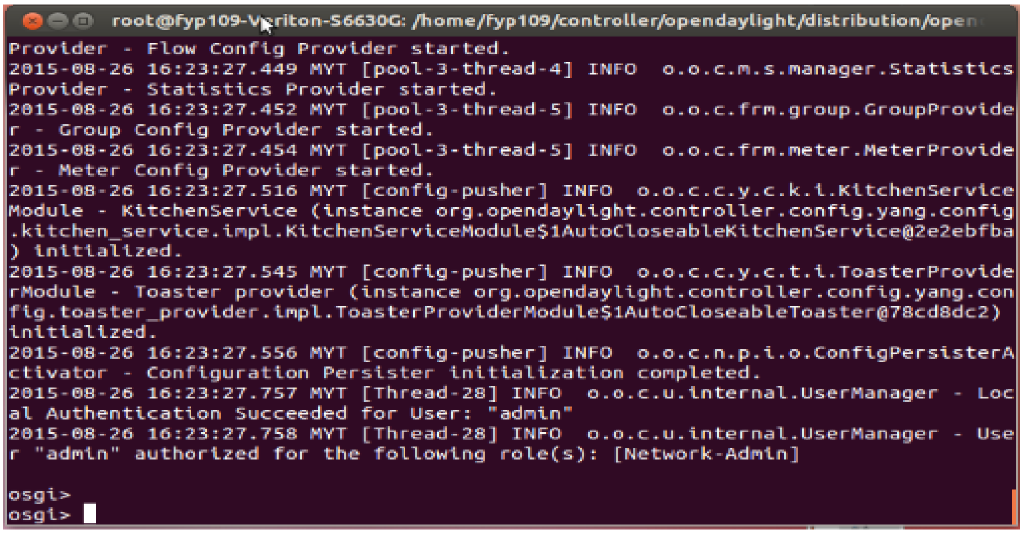
To activate and start the OpenDaylight controller, we need to go the specific location in: “/home/fyp109/controller/opendaylight/ distribution/opendaylight/target/distribution.opendaylight-osgipackage/opendaylight” and type in “./run.sh” in the terminal. Figure 5.3.1 is the command to start OpenDaylight while figure 5.3.2 is the output after the activating process.



root@fyp109-Veriton-S6630G: /home/fyp109/controller/opendaylight/distribution/open

```
root@fyp109-Veriton-S6630G:/home/fyp109/controller/opendaylight/distribution/ope
ndaylight/target/distribution.opendaylight-osgipackage/opendaylight# ./run.sh
```

Figure 5.3.1.1: Command to start the OpenDaylight Controller.



root@fyp109-Veriton-S6630G: /home/fyp109/controller/opendaylight/distribution/open

```
Provider - Flow Config Provider started.
2015-08-26 16:23:27.449 MYT [pool-3-thread-4] INFO  o.o.c.m.s.manager.Statistics
Provider - Statistics Provider started.
2015-08-26 16:23:27.452 MYT [pool-3-thread-5] INFO  o.o.c.frm.group.GroupProvide
r - Group Config Provider started.
2015-08-26 16:23:27.454 MYT [pool-3-thread-5] INFO  o.o.c.frm.meter.MeterProvide
r - Meter Config Provider started.
2015-08-26 16:23:27.516 MYT [config-pusher] INFO  o.o.c.c.y.c.k.i.KitchenService
Module - KitchenService (instance org.opendaylight.controller.config.yang.config
.kitchen_service.impl.KitchenServiceModule$1AutoCloseableKitchenService@2e2ebfba
) initialized.
2015-08-26 16:23:27.545 MYT [config-pusher] INFO  o.o.c.c.y.c.t.i.ToasterProvide
rModule - Toaster provider (instance org.opendaylight.controller.config.yang.con
fig.toaster_provider.impl.ToasterProviderModule$1AutoCloseableToaster@78cd8dc2)
initialized.
2015-08-26 16:23:27.556 MYT [config-pusher] INFO  o.o.c.n.p.i.o.ConfigPersisterA
ctivator - Configuration Persister initialization completed.
2015-08-26 16:23:27.757 MYT [Thread-28] INFO  o.o.c.u.internal.UserManager - Loc
al Authentication Succeeded for User: "admin"
2015-08-26 16:23:27.758 MYT [Thread-28] INFO  o.o.c.u.internal.UserManager - Use
r "admin" authorized for the following role(s): [Network-Admin]

osgi>
osgi>
```

Figure 5.3.1.2: Output after the activating process.

After the activating process, we can now go to the user interface of OpenDaylight. In the Google Chrome browser address bar, type in “192.168.1.12:8080” since my controller desktop is using 192.168.1.12 as the IP address. The username and password to login OpenDaylight are both “admin”. Figure 5.3.3 is the login page of OpenDaylight.

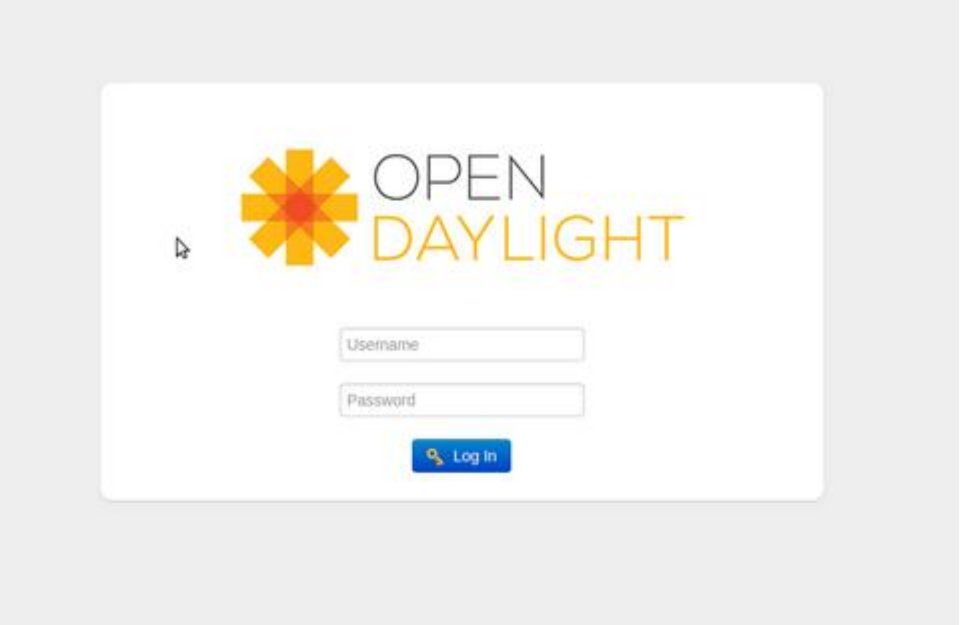
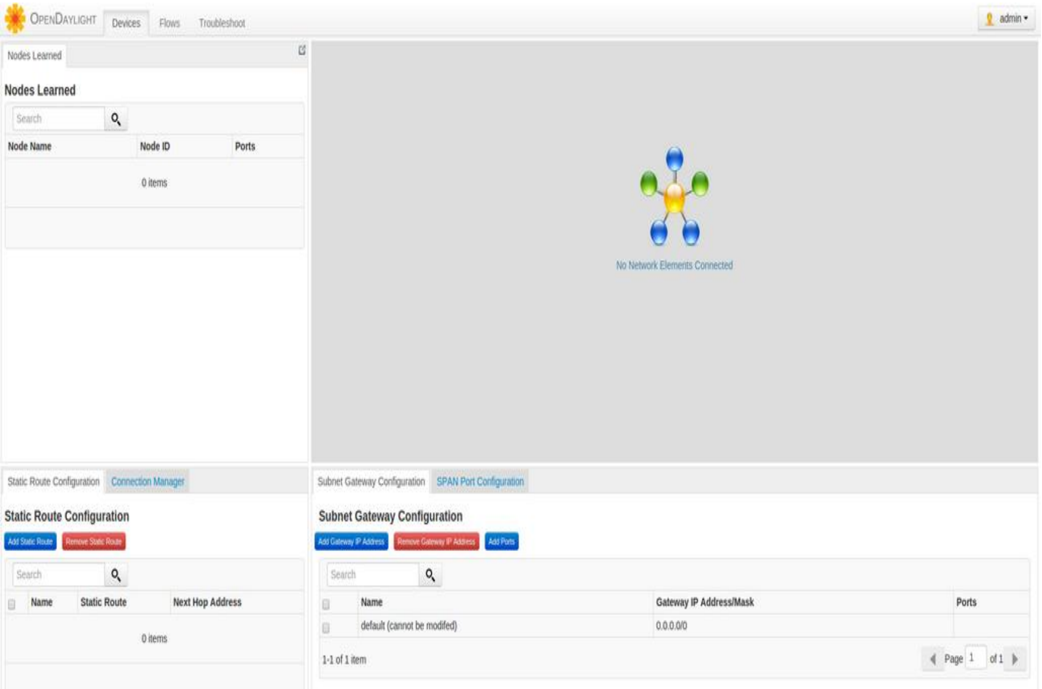


Figure 5.3.1.3: Login page of OpenDaylight.



Nodes Learned

Node Name	Node ID	Ports
	0 items	

Static Route Configuration

□	Name	Static Route	Next Hop Address
		0 items	

Subnet Gateway Configuration

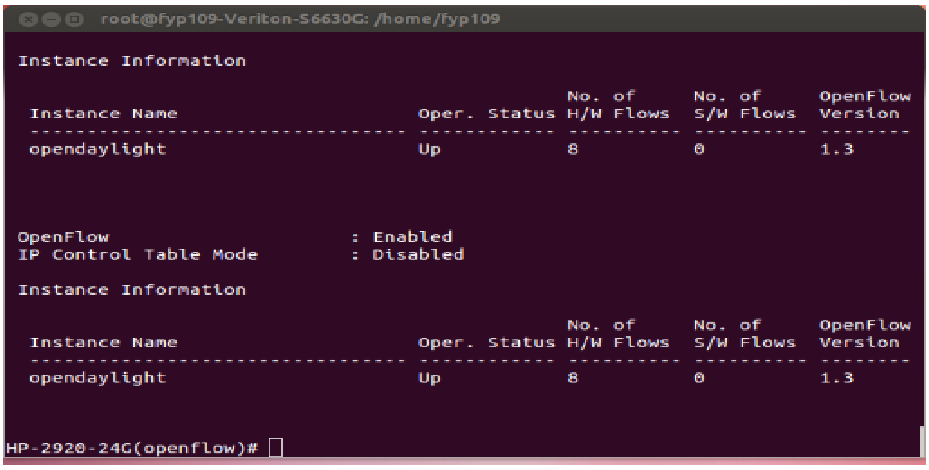
□	Name	Gateway IP Address/Mask	Ports
□	default (cannot be modifed)	0.0.0.0/0	

1-1 of 1 item   Page 1 of 1

Figure 5.3.1.4: User Interface of OpenDaylight

5.3.2 Activate Openflow Protocol in HP Switch 2920-24G

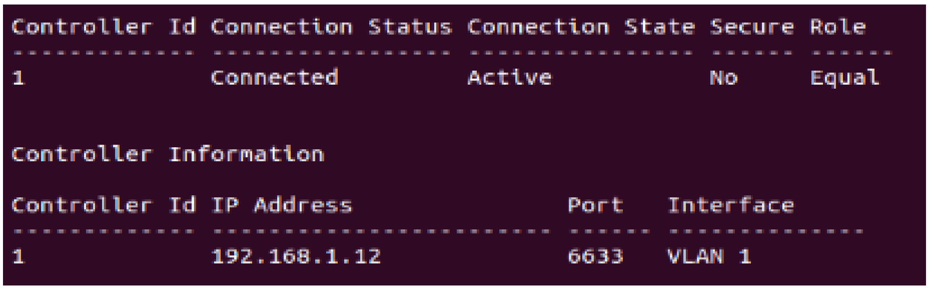
To start the SDN network and Controller Software, I had to create an Openflow instance and a Controller using the terminal console and enable them with some specific command. Below are the print screens that show the output after connected the HP Switches and the SDN Controller desktop computer. Based on the figure 5.3.1.1, the “opendaylight” operation status is up, this indicate that the Openflow protocol in the HP Switch has enabled.



```
Instance Information

                                                  No. of     No. of     OpenFlow
 Instance Name                     Oper. Status H/W Flows  S/W Flows  Version
 --------------------------------- ------------ ---------- ---------- --------
 opendaylight                      Up           8          0          1.3


OpenFlow                     : Enabled
IP Control Table Mode        : Disabled

Instance Information

                                                  No. of     No. of     OpenFlow
 Instance Name                     Oper. Status H/W Flows  S/W Flows  Version
 --------------------------------- ------------ ---------- ---------- --------
 opendaylight                      Up           8          0          1.3

HP-2920-24G(openflow)# 
```

Figure 5.3.2.1: Output that show the Openflow instance "Opendaylight" has enabled.

According to the figure 5.3.1.2, the Connection Status with “Connected” and Connection State with “Active” are indicating that the connection between controller and HP Switch has established.

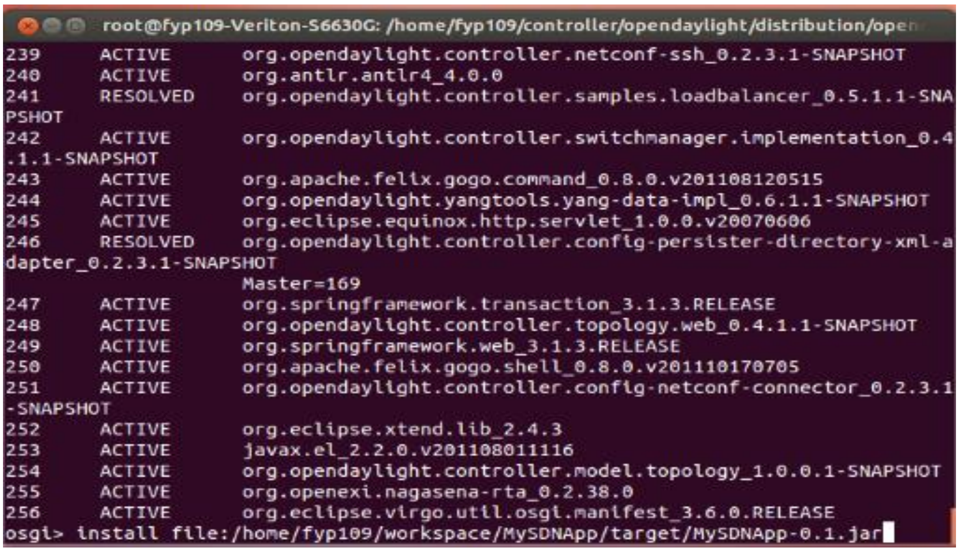


```
Controller Id Connection Status Connection State Secure Role
------------- ----------------- ---------------- ------ ------
1             Connected         Active           No     Equal


Controller Information

Controller Id IP Address               Port   Interface
------------- ------------------------ ------ --------------
1             192.168.1.12             6633   VLAN 1
```

Figure 5.3.2.2: Output that show the connection between HP Switch and SDN Controller has successful.

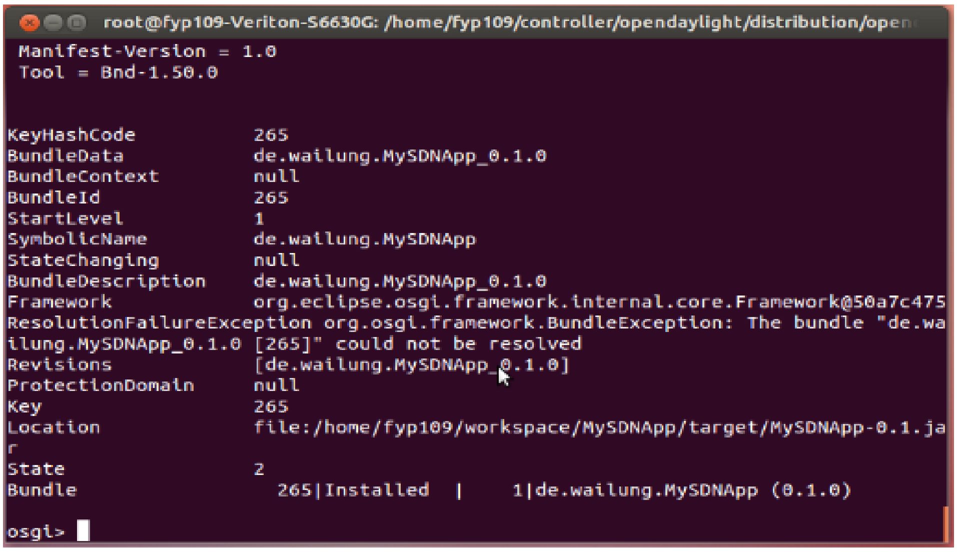
5.3.3 Install MySDNApp into Controller

After connected the SDN Controller with HP Switch, we have installed our SDN application named "MySDNApp" into the OpenDaylight Controller Software through the terminal console. After the installation process, we have started the application with command "start 265" since the application ID is 265. Figure 5.3.2.1 is the print screens of installation and activating process of the MySDNApp.


root@fyp109-Veriton-S6630G: /home/fyp109/controller/opendaylight/distribution/open

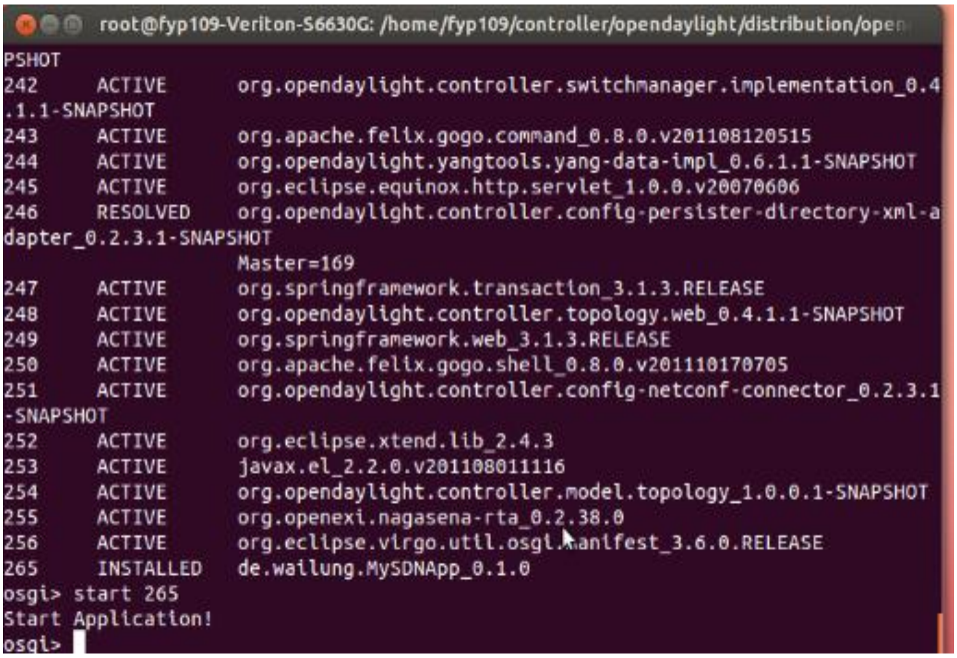
```
239     ACTIVE      org.opendaylight.controller.netconf-ssh_0.2.3.1-SNAPSHOT
240     ACTIVE      org.antlr.antlr4_4.0.0
241     RESOLVED    org.opendaylight.controller.samples.loadbalancer_0.5.1.1-SNA
PSHOT
242     ACTIVE      org.opendaylight.controller.switchmanager.implementation_0.4
.1.1-SNAPSHOT
243     ACTIVE      org.apache.felix.gogo.command_0.8.0.v201108120515
244     ACTIVE      org.opendaylight.yangtools.yang-data-impl_0.6.1.1-SNAPSHOT
245     ACTIVE      org.eclipse.equinox.http.servlet_1.0.0.v20070606
246     RESOLVED    org.opendaylight.controller.config-persister-directory-xml-a
dapter_0.2.3.1-SNAPSHOT
                    Master=169
247     ACTIVE      org.springframework.transaction_3.1.3.RELEASE
248     ACTIVE      org.opendaylight.controller.topology.web_0.4.1.1-SNAPSHOT
249     ACTIVE      org.springframework.web_3.1.3.RELEASE
250     ACTIVE      org.apache.felix.gogo.shell_0.8.0.v201110170705
251     ACTIVE      org.opendaylight.controller.config-netconf-connector_0.2.3.1
-SNAPSHOT
252     ACTIVE      org.eclipse.xtend.lib_2.4.3
253     ACTIVE      javax.el_2.2.0.v201108011116
254     ACTIVE      org.opendaylight.controller.model.topology_1.0.0.1-SNAPSHOT
255     ACTIVE      org.openexi.nagasena-rta_0.2.38.0
256     ACTIVE      org.eclipse.virgo.util.osgi.manifest_3.6.0.RELEASE
osgi> install file:/home/fyp109/workspace/MySDNApp/target/MySDNApp-0.1.jar
```

Figure 5.3.3.1: Install MySDNApp in OpenDaylight terminal console with command "install file:/home/fyp109/workspace/MySDNApp/target/ MySDNApp-0.1.jar".


root@fyp109-Veriton-S6630G: /home/fyp109/controller/opendaylight/distribution/open

```
 Manifest-Version = 1.0
 Tool = Bnd-1.50.0


KeyHashCode             265
BundleData              de.wailung.MySDNApp_0.1.0
BundleContext           null
BundleId                265
StartLevel              1
SymbolicName            de.wailung.MySDNApp
StateChanging           null
BundleDescription       de.wailung.MySDNApp_0.1.0
Framework               org.eclipse.osgi.framework.internal.core.Framework@50a7c475
ResolutionFailureException org.osgi.framework.BundleException: The bundle "de.wa
ilung.MySDNApp_0.1.0 [265]" could not be resolved
Revisions               [de.wailung.MySDNApp_0.1.0]
ProtectionDomain        null
Key                     265
Location                file:/home/fyp109/workspace/MySDNApp/target/MySDNApp-0.1.ja
r
State                   2
Bundle                    265|Installed  |    1|de.wailung.MySDNApp (0.1.0)

osgi> 
```

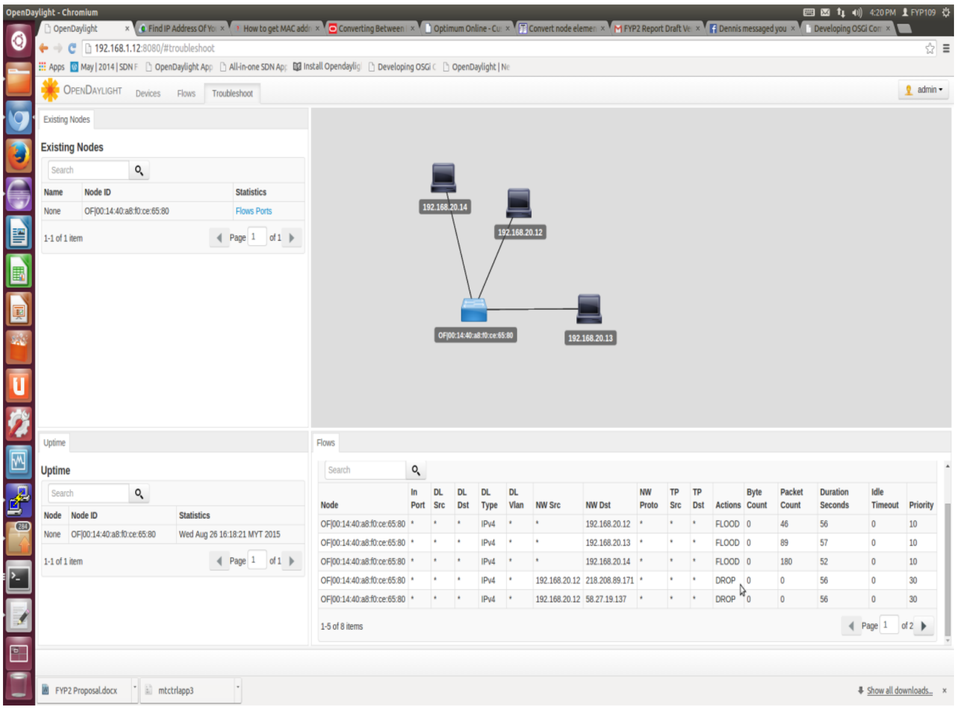
Figure 5.3.3.2: Output which show successfully installed MySDNApp.


root@fyp109-Veriton-S6630G: /home/fyp109/controller/opendaylight/distribution/open

```
PSHOT
242     ACTIVE      org.opendaylight.controller.switchmanager.implementation_0.4
.1.1-SNAPSHOT
243     ACTIVE      org.apache.felix.gogo.command_0.8.0.v201108120515
244     ACTIVE      org.opendaylight.yangtools.yang-data-impl_0.6.1.1-SNAPSHOT
245     ACTIVE      org.eclipse.equinox.http.servlet_1.0.0.v20070606
246     RESOLVED    org.opendaylight.controller.config-persister-directory-xml-a
dapter_0.2.3.1-SNAPSHOT
                    Master=169
247     ACTIVE      org.springframework.transaction_3.1.3.RELEASE
248     ACTIVE      org.opendaylight.controller.topology.web_0.4.1.1-SNAPSHOT
249     ACTIVE      org.springframework.web_3.1.3.RELEASE
250     ACTIVE      org.apache.felix.gogo.shell_0.8.0.v201110170705
251     ACTIVE      org.opendaylight.controller.config-netconf-connector_0.2.3.1
-SNAPSHOT
252     ACTIVE      org.eclipse.xtend.lib_2.4.3
253     ACTIVE      javax.el_2.2.0.v201108011116
254     ACTIVE      org.opendaylight.controller.model.topology_1.0.0.1-SNAPSHOT
255     ACTIVE      org.openexi.nagasena-rta_0.2.38.0
256     ACTIVE      org.eclipse.virgo.util.osgi.manifest_3.6.0.RELEASE
265     INSTALLED   de.wailung.MySDNApp_0.1.0
osgi> start 265
Start Application!
osgi>
```

Figure 5.3.3.3: Use "start 265" as the command to start MySDNApp.

After all the installation activating process, we can observe the SDN network topology in the OpenDaylight controller page in Google Chrome Browser.



Existing Nodes

Name	Node ID	Statistics
None	OF|00:14:40:a8:f0:ce:65:80	Flows Ports

1-1 of 1 item Page 1 of 1

Uptime

Node	Node ID	Statistics
None	OF|00:14:40:a8:f0:ce:65:80	Wed Aug 26 16:18:21 MYT 2015

1-1 of 1 item Page 1 of 1

Node	In Port	DL Src	DL Dst	DL Type	DL Vlan	NW Src	NW Dst	NW Proto	TP Src	TP Dst	Actions	Byte Count	Packet Count	Duration Seconds	Idle Timeout	Priority
OF|00:14:40:a8:f0:ce:65:80	*	*	*	IPv4	*	*	192.168.20.12	*	*	*	FLOOD	0	46	56	0	10
OF|00:14:40:a8:f0:ce:65:80	*	*	*	IPv4	*	*	192.168.20.13	*	*	*	FLOOD	0	89	57	0	10
OF|00:14:40:a8:f0:ce:65:80	*	*	*	IPv4	*	*	192.168.20.14	*	*	*	FLOOD	0	180	52	0	10
OF|00:14:40:a8:f0:ce:65:80	*	*	*	IPv4	*	192.168.20.12	218.208.89.171	*	*	*	DROP	0	0	56	0	30
OF|00:14:40:a8:f0:ce:65:80	*	*	*	IPv4	*	192.168.20.12	58.27.19.137	*	*	*	DROP	0	0	56	0	30

1-5 of 8 items Page 1 of 2

Figure 5.3.3.4: Web Page which shown the SDN network topology.

5.4 Concluding Remarks

This chapter consists of the entire hardware and software setup configuration which are very important in order to operate the SDN network. However, MySDNApp is the only main application which will implements the Bring Your Own Device (BYOD) concept into the SDN network.

6.0 System Evaluation and Discussion

6.1 System Testing and Performance Metrics

The system testing is done through separately in 3 phases. In order to test the connectivity in a network, Ping test is the best solution since it is not time consuming and simple to setup. Before we start the tests, some precautions should be done in order to ensure the accuracy of the test result. Below are precautions which must be done before the test and Table 6.1.1 is the hardware list with IP Address.

Precautions:

- Make sure all the hardware including mobile devices and laptops can connect to a Local Area Network (LAN) and Wide Area Network (WAN) without problem.
- Make sure the 2 mobiles devices can connect to the wireless access point.
- Make sure all the Fast Ethernet cables are well function.

Hardware	IP Address
HP Switch 2920-24G	192.168.1.11 (VLAN 1) 192.168.20.11 (VLAN 2)
TP-Link Wireless Access Point	192.168.20.1 (Wireless Access Point / DHCP Server)
Cisco Router 1841	Gateway for VLAN 1 - 192.168.1.254 Gateway for VLAN 20 - 192.168.20.254 Gateway for Internet - 192.168.209.250
Controller Desktop (VLAN 1)	192.168.1.12
Lenovo Laptop (VLAN 20)	192.168.20.12
HTC smart Phone (VLAN 20)	192.168.20.x (Dynamic IP assign by DHCP Server / wireless connection)
Lenovo Smart Phone (VLAN 20)	192.168.20.x (Dynamic IP assign by DHCP Server / wireless connection)

Table 6.1.1: Hardware List with IP Address.

6.2 Testing Setup and Result

6.2.1 Phase 1

In Phase 1, we will test the connectivity in the network without enables the Openflow protocol in the HP Switch. Which means in order to start a SDN network, the Openflow protocol must be enable otherwise the network will only act as a normal network. Table 6.2.1.1 is the result of Ping test in Part A.

	HP 2920-24G Switch	TP-Link Wireless Access Point	Cisco Router 1841	Controller Desktop	Lenovo Laptop	HTC Smart Phone	Lenovo Smart Phone	Google Web Site
HP 2920-24G Switch		✓	✓	✓	✓	✓	✓	✓
TP-Link Wireless Access Point	✓		✓	✓	✓	✓	✓	
Cisco Router 1841	✓	✓		✓	✓	✓	✓	✓
Controller Desktop	✓	✓	✓		✓	✓	✓	✓
Lenovo Laptop	✓	✓	✓	✓		✓	✓	✓
HTC Smart Phone	✓	✓	✓	✓	✓		✓	✓
Lenovo Smart Phone	✓	✓	✓	✓	✓	✓		✓

Table 6.2.1.1: Phase 1 Ping Test Result (✓ = success, ✗ = fail)

During Phase 1, ping tests between all devices are successful. This indicated that the network has work perfectly in both LAN and WAN (internet access).

6.2.2 Phase 2

In Phase 2, we will enable the Openflow protocol in the HP Switch and test the connectivity in the SDN network. Table 6.2.2.1 is the result of Ping test in Phase 2.

	HP 2920-24G Switch	TP-Link Wireless Access Point	Cisco Router 1841	Controller Desktop	Lenovo Laptop	HTC Smart Phone	Lenovo Smart Phone	Google Web Site
HP 2920-24G Switch		✗	✗	✓	✗	✗	✗	✗
TP-Link Wireless Access Point	✗		✗	✗	✗	✓	✓	
Cisco Router 1841	✓	✗		✓	✗	✗	✗	✓
Controller Desktop	✓	✗	✓		✗	✗	✗	✓
Lenovo Laptop	✗	✗	✗	✗		✗	✗	✗
HTC Smart Phone	✗	✓	✗	✗	✗		✓	✗
Lenovo Smart Phone	✗	✓	✗	✗	✗	✓		✗

Table 6.2.2.1: Phase 2 Ping Test Result (✓ = success, ✗ = fail)

Based on the Ping test result in Table 6.2.1.1, there are a lot of ping failure as well as ping success. If we read into details, the ping test from or to VLAN 1 are almost 100% success while the ping test from or to VLAN 20 are almost 100% fail.

According to the Chapter 4.3.2 - Create VLANs and Openflow Instance in HP Switch 2920-24G, we have assigned the VLAN 20 as a member of the Openflow instance named OpenDaylight. In this case, any network packets from or to the VLAN 20 must followed the Openflow flow table in the HP Switch. Since we did not install any flows into the Openflow table and it is totally empty, so that the Switch will automatically drop all the network packets.

From the table, we will also found that HTC and Lenovo smart phones which are belong to VLAN 20 can ping each other while they cannot ping the Lenovo Laptop which is also belongs to VLAN 20. Based on the Figure 5.1.1.1 - SDN Network Topology, the HTC and Lenovo smart phones are connected to the TP-Link Wireless Access Point through wireless connection while the Lenovo Laptop is directly connected to the HP Switch using Fast Ethernet cable. Same as the Lenovo Laptop, the access point is also connected to the HP Switch using Fast Ethernet cable. In this scenario the ping test between both smart phones are successful because the network traffic does not need to pass through HP Switch during the ping process. While the ping test between smart phones and Lenovo Laptop are fail because whoever wants to send or receive the network packet from Lenovo Laptop, the network packet must pass through the HP Switch.

6.2.3 Phase 3

In Phase 3, we will focus test whether MySDNApp can be well performing and functioning in the Controller. We will do some changes in the SDN network topology by adding one wireless device (HP Laptop) into the network. Moreover, we will also install the SDN application MySDNApp into the SDN Controller by following the instructions in Chapter 5.3.2 - Install MySDNApp to the Controller. The Ping Test in Phase 3 will start after all the preparations have done. Also, the descriptions of MySDNApp are stated in Chapter 3.2.2 - MySDNApp.

Phase 3 Preparations:

1. Add one wireless device into SDN network
2. Make sure the HP Laptop can connect LAN and WAN without problem
3. Make sure the HP Laptop can connect to the TP-Link Wireless Access Point
4. Register the MAC Address of Lenovo Laptop, Lenovo Smart Phone and HTC Smart Phone as well into MySDNApp. (No need to register HP Laptop's MAC Address)
5. Set the white list and black list web sites for particular users in MySDNApp
    - ➢ Lenovo Laptop: Can access all websites except UTAR WBLE and UTAR Home Page
    - ➢ Lenovo Smart Phone: Can access only UTAR WBLE and UTAR Home Page
    - ➢ HTC Smart Phone: Can access all websites
6. Install SDN application MySDNApp into Controller

	VLAN 1	VLAN 20	UTAR WBLE	UTAR Home Page	Other web Sites
Controller Desktop	✓	✗	✓	✓	✓
Lenovo Laptop	✗	✓	✗	✗	✓
HTC Smart Phone	✗	✓	✓	✓	✓
Lenovo Smart Phone	✗	✓	✓	✓	✗
HP Laptop	✗	✗	✗	✗	✗

Table 6.2.3.1: Phase 3 Ping Test Result (✓ = success, ✗ = fail)

From Table 6.2.3.1, all the ping tests of HP Laptop has failed because we did not register its MAC Address into the MySDNApp. So that packets from HP Laptop will drop automatically.

For the Controller, we found that all the ping tests are success except the ping test with VLAN 20. In a traditional network, the Controller should be able to ping all devices in the VLAN 20. However in SDN network, VLANs which is the member of Openflow instance are unable to communicate with the Controller's VLAN due to the security issue. In this case VLAN 1 is Controller's VLAN while VLAN 20 is the member of Opendaylight (Openflow instance in HP Switch).

The ping test results for the 3 devices are affected by the 5th Phase 3 preparations. For the Lenovo Laptop, the ping tests show that it can access any websites except UTAR WBLE and UTAR Home Page. While for the Lenovo smart phone, it can only access UTAR WBLE and UTAR Home Page. From the ping results of HTC smart phone, we know that it can access all the websites including the UTAR WBLE and UTAR Home Page.

All ping test results in those 3 phases are completely same as what we expected and it has indicated that our project is well performing and functionality.

6.3 Project Challenges

Nothing can be done easily without challenges, as well as my project. During the project development process, I have had a hard time by encountered a lot of problems. One of the main challenges for me is to setup and configuration all the network devices such as HP Switch, Cisco Router and others. At first, I need to design and create a virtual network using the simulator software which is Packet Tracer and make sure the simulation network can performed well. After that, I can only start to create a real network environment with configured network equipments based on what I have done in the simulation. During the hardware configuration process, all configurations must be 100% correct otherwise it will cause a lot of problems to the network. Moreover, to configure HP Switch 2920-24G is difficult and complicated than I expected. I have to find the HP Switch User Manual and read through it in order to configure VLAN, Openflow protocol, Trunking, and others. This is a time consuming process since I need to use more time on debugging the problems caused by hardware devices.

The other challenge that had made me suffer is related to the programming part with Java Language. As we know, one of the key points in SDN is the application. Based on my project objectives, I have to develop a SDN application using Java Language in order to implement the Bring-Your-Own-Device concepts in my project. At the beginning, I am using the HP SDN Controller Software instead of the OpenDaylight Controller Software. Although I am very familiar with Java Language programming but still I feel very strange on the SDN application programming code and there is also not much available materials from the internet. At first, I have no idea how to code a simple SDN application even a simple application which will display a message of “Hello World”. And then I have no idea on how to use the specific SDN functions. Moreover, even I have done a simple application but the testing result always reply me with unexpected or even no response.

Luckily, everything has changed since I swap the HP SDN Controller to OpenDaylight Controller. There are a lot of materials related to the OpenDaylight, I even found a tutorial on how to develop a simple SDN application which will display the “Hello World” message. Thus, it is easier for me to gain the knowledge on how to start and develop a SDN application by using the OpenDaylight SDN Controller in my project.

6.4 SWOT

Strengths

- Does not required any human intervention once the MySDNApp has installed

Weaknesses

- MySDNApp does not has any Graphical User Interface
- Need to reinstall MySDNApp whenever there are new users registered

Opportunities

- Traditional network will replace by SDN network in the future
- Innovative in SDN applications based on network users' requirements
- Provide opportunities for students to extend my current SDN system

Threats

- Network administrators are required extra knowledge related to SDN (How to manage SDN Controller Software such as HP SDN Controller, OpenDaylight Controller and etc.)
- Security issues due to no authentication process are required for users

6.5 Objectives Evaluation

All 3 objectives of my project are achieved successfully as indicated below:

1. Solve the user overloads issue on the wireless access point and routers.

   ✓ By implemented the BYOD concept in my project, MySDNApp has designed to provide the function to control the number of users. With this function, the SDN network can allow only registered users to connect.

2. Reduce network misuse among users

   ✓ To achieve the objective to reduce the misuse of network, MySDNApp has also designed to provide the functions which can control the websites accessibility among different users.

3. Develop a well designed SDN network with real equipments

   ✓ With the well developed framework over SDN in FYP Lab, students have encouraged to extend my current SDN application - MySDNApp or to develop a whole new SDN application with their innovative idea.

6.6 Concluding Remarks

Based on testing result in Chapter 6.2, it has indicated that the system is well performing as the results are highly as expected. In a nutshell, the final outcome of this project is highly successful since all 3 objectives were met.

7.0 Conclusion and Recommendation

7.1 Conclusion

Based on the study we have done for software defined networking in the pass few months, we do agree that it has a very high potential to become the future network trends. With SDN, we can develop and design different application based on our requirements to manage or solve the network problems. For example, we have developed MySDNApp with BYOD and websites restricted functions since my project objectives are to solve users overload and network misuse issues. In a nutshell, with a new solution to manage network, SDN has the abilities to help people achieve more possibilities based on their innovative ideas.

7.2 Recommendations

There are few ways to enhance my current SDN application – MySDNApp since it has some drawbacks that I still could not solve. The recommendations are indicated as below:

1. Provide the Graphical User Interface (GUI).

    - Since MySDNApp does not provide any user interface, thus we have to do some changes in the application source code in order to add a new registered user.

2. Design and create a database to store registered MAC Addresses.

    - Also, we can enhanced MySDNApp by create a database to store user’s MAC Address. A database to store registered MAC Addresses should be created when there are a large amount of users in the SDN network.

3. Create a new authentication process which uses other metrics such as student ID, staff ID and etc.

    - MySDNApp is using the MAC Address to filter unauthorized access and it is highly vulnerable to the security attack. Since the device’s MAC Address can be change easily so that unauthorized users might take this advantage to gain access to the SDN network.

8.0 References

Balaji Sivasubramanian, Erum Frahim, Richard Froom (2010) Analyzing the Cisco Enterprise Campus Architecture. [Online]. Available at: http://www.ciscopress.com/articles/article.asp?p=1608131&seqNum=3 [Accessed 8 August 2014]

Brandon Carroll (2013) Software Defined Networking vs. Traditional Networking. [Online]. Available at: http://globalconfig.net/software-defined-networking-vs-traditional/ [Accessed 5 August 2014]

Brocade (2014) *Exploring Software-Defined Networking with Brocade White Paper*. [Online]. Available at: http://www.brocade.com/downloads/documents/white_papers/exploring-sdn-wp.pdf [Accessed 15 July 2014]

Cisco (2013) *Software-Defined Networking: Why We Like It and How We Are Building On It White Paper 2013.* [Online]. Available at: http://www.cisco.com/web/strategy/docs/gov/cis13090_sdn_sled_white_paper.pdf [Accessed 27 June 2014]

Don, C., Peter, W., and Andy, R. (2012) Network Functions Virtualisation – Introductory White Paper, 1(1), pp.1-16. Available at: http://portal.etsi.org/NFV/NFV_White_Paper.pdf [Accessed 5 August 2014]

HP (2013) *Prepare for software-defined networking: Build the foundation for SDN with OpenFlow Business White Paper 2013.* [Online]. Available at: http://h20195.www2.hp.com/v2/GetPDF.aspx/4AA3-8562ENW.pdf [Accessed 15 July 2014]

HP (2014) HP VAN SDN Controller and Applications Support Matrix, vol. 1, July 2014, pp.1-20. Available at: http://h20566.www2.hp.com/portal/site/hpsc/template.BINARYPORTLET/public/kb/docDisplay/resource.process/?javax.portlet.begCacheTok=com.vignette.cachetoken&javax.portlet.endCacheTok=com.vignette.cachetoken&javax.portlet.rid_ba847bafb2a2d782fcbb0710b053ce01=docDisplayResURL&javax.portlet.rst_ba847bafb2a2d782fcbb0710b053ce01=wsrp-resourceState%3DdocId%253Demr_na-c04219768-1%257CdocLocale%253Den_US&javax.portlet.tpst=ba847bafb2a2d782fcbb0710b053ce01_ws_BI&ac.admitted=1407950258991.876444892.199480143 [Accessed 8 August 2014]

Kate Greene (2009) TR10: Software-Defined Networking. [Online]. Available at: http://www2.technologyreview.com/article/412194/tr10-software-defined-networking/ (Accessed 1 July 2014)

Lee Doyle (n.d.) Does software-defined networking requires special SDN switches? [Online]. Available at: http://searchsdn.techtarget.com/answer/Does-software-defined-networking-require-special-SDN-switches [Accessed 7 August 2014]

Mary Shacklett (2013) 10 key questions about software-defined networking (SDN). [Online]. Available at: http://www.zdnet.com/10-key-questions-about-software-defined-networking-sdn-7000015822/ [Accessed 1 July 2014]

Nick McKeown (n.d.) How SDN will shape networking. [Online]. Available at http://mvdirona.com/jrh/TalksAndPapers/NickMckeown_ON%20Summit%20NickM%2010%202011.pdf [Accessed 25 June 2014]

Open Network Foundation (2012) *Software-Defined Networking: The New Norm for Networks White Paper 2012.* [Online]. Available at: https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf [Accessed 5 July 2014]

Sterling, P. And Stan, H. (2013) *Practical Implementation of SDN & NFV in the WAN White Paper 2013*. [Online]. Available at: http://www.windriver.com/whitepapers/practical-implementation-of-sdn-and-nfv/white-paper-practical-implementation-of-sdn-and-nfv.pdf [Accessed 1 August 2014]

Stuart Miniman (2013) Networking Revolution: Software Defined Networking and Network Virtualization. [Online]. Available at: http://wikibon.org/wiki/v/Networking_Revolution:_Software_Defined_Networking_and_Network_Virtualization [Accessed 5 August 2014]

Waqas Azam (2013) Cisco PPDIOO | A Network Life Cycle. [Online]. Available at: http://www.w7cloud.com/cisco-ppdioo/ [Accessed 8 August 2014]

Wang Lei, Director of Enterprise Networking Product Mgmt Dept, Huawei Enterprise Business Group (n.d.) SDN: The Best Answer to Campus Network Challenges. [Online]. Available at: http://enterprise.huawei.com/en/about/e-journal/ict/detail/hw-311134.htm [Accessed 2 August 2014]

WWW.OPENMANIAK.COM (2006) What is Vyatta? [Online]. Available at: http://openmaniak.com/vyatta.php [Accessed 5 August 2014]

Appendix

Appendix 1

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 1
Student Name & ID: Khoo Wai Lung - 1207252	
Supervisor: Dr. Liew Seoung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

1. WORK DONE - Check the network equipment in FYP lab - Quick review on my FYP 1 report
2. WORK TO BE DONE - Appointment with Dr. Liew and discuss the progress
3. PROBLEM ENCOUNTERED - None
4. SELF EVALUATION OF THE PROGRESS - Progress is ok

____________________________ ____________________________

Supervisor’s signature Student’s signature

Appendix 2

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 4
Student Name & ID: Khoo Wai Lung - 1207252	
Supervisor: Dr. Liew Seoung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

1. WORK DONE - Created a network with 2 Vlans and integrate OpenDaylight controller with HP Switch
2. WORK TO BE DONE - Connect the network to internet by using NAT method - Reinstall OS version in controller desktop
3. PROBLEM ENCOUNTERED - OpenDaylight controller cannot connect to the HP Switch due to OS version not supported
4. SELF EVALUATION OF THE PROGRESS - Progress is slower than expected

Supervisor's signature

Student's signature

Appendix 3

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 5
Student Name & ID: Khoo Wai Lung - 1207252	
Supervisor: Dr. Liew Seoung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

1. WORK DONE - OpenDaylight controller has successfully connected to HP Switch and SDN network is working fine
2. WORK TO BE DONE - Connect the network to internet by using NAT method
3. PROBLEM ENCOUNTERED - Network created cannot connect to internet after NAT configuration
4. SELF EVALUATION OF THE PROGRESS - Progress is slow, need more effort.

_______________________________ _______________________________

Supervisor's signature Student's signature

Appendix 4

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 7
Student Name & ID: Khoo Wai Lung – 1207252	
Supervisor: Dr. Liew Soung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

1. WORK DONE - Finally the network has successfully connected to internet
2. WORK TO BE DONE - Start to develop MySDNApp
3. PROBLEM ENCOUNTERED - None
4. SELF EVALUATION OF THE PROGRESS - Progress is slow, need more efforts

Supervisor’s signature

Student’s signature

Appendix 5

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 10
Student Name & ID: Khoo Wai Lung – 1207252	
Supervisor: Dr. Liew Soung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

1. WORK DONE - Added packet forwarding function in MySDNApp
2. WORK TO BE DONE - Add BYOD function in MySDNApp - Add websites blocking function in MySDNApp
3. PROBLEM ENCOUNTERED - None
4. SELF EVALUATION OF THE PROGRESS - Overall progress is slow, need more efforts

_______________________________ _______________________________

Supervisor's signature Student's signature

Appendix 6

(Project II)

Trimester, Year: Year 3 Trimester 3	**Study week no.:** 11
Student Name & ID: Khoo Wai Lung – 1207252	
Supervisor: Dr. Liew Soung Yue	
Project Title: A Bring-Your-Own-Device Framework over Software Defined Networking	

5. WORK DONE - Added BYOD and websites blocking functions in MySDNApp - Implement MySDNApp in the OpenDaylight Controller
6. WORK TO BE DONE - Start to debug application errors
7. PROBLEM ENCOUNTERED - None
8. SELF EVALUATION OF THE PROGRESS - Overall progress is ok, satisfied.

____________________________ ____________________________

Supervisor’s signature Student’s signature