

東海大學

資訊工程研究所

碩士論文

指導教授：楊朝棟博士

運用 OpenFlow 於雲端虛擬交換器監控系統之實作

Implementation of a Virtual Switch Monitor System Using

Openflow on Cloud

研究生：陳煒勝

中華民國一零二年一月

東海大學碩士學位論文考試審定書

東海大學資訊工程學系 研究所

研究生 陳 煒 勝 所提之論文

運用 OpenFlow 於雲端虛擬交換器監控

系統之實作

經本委員會審查，符合碩士學位論文標準。

學位考試委員會

召 集 人



簽章

委

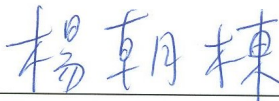
員

伍朝欽

李冠憬

賴冠州

指 導 教 授



簽章

中華民國 102 年 1 月 17 日

# 摘要

OpenFlow 是一種新興的網路通訊協定，藉由分離控制層與網路層，使網路的效率能夠更好，也能更加的實現真正的 QoS 功能，將網路的需求留給真正需要使用，或是系統中所定義優先權較高的人所使用，而 OpenFlow 的實現，除了依靠通訊協定本身之外，還需要實體或虛擬的交換器，以及擔當流量控制角色的控制器來輔助，才有辦法形成一個完整的系統。當開始使用 OpenFlow 時，交換器或路由器提供了什麼功能已經不是重點，或者是說，OpenFlow 所著重的，就是把原本廠商結合好的專屬作業系統以及功能切分開，將系統對使用者開放，讓使用者自行撰寫所需要的功能，舉凡是 RIP、OSPF、EGP 等路由協定，或是防火牆、QoS、防毒、NAT 等等功能，只要你有概念，都可以用軟體的方式實現在有支援 OpenFlow 的交換器或路由器上面。本論文主要著重點在於，建立一個 OpenFlow 交換器監控系統，能夠發現網路上所有的封包及流量。藉由我們所建立的網頁版流量控制系統，將 QoS 政策能夠輕易的設定到支援 OpenFlow 的交換器上，控制每個 IP 的流量優先權，使網路管理人員可以輕鬆的管理整個網路。

# Abstract

OpenFlow mechanism is a next generation networking protocol. It speeds up network performance by separating the control layer and the data layer. It can implement the real QoS function. Users who really need network speed can get their resources, or decided by the priority which system defined. To implement OpenFlow mechanism, we need two elements, the switch which supports OpenFlow, whatever it is physical switch or virtual switch, and a controller to send flow setting packet, to control the switch flow table. When you start using OpenFlow, the switch or router provide function, like RIP, OSPF, EGP routing protocol, or firewall, QoS, Anti Virus, NAT, is not important. Because OpenFlow focus on provide a standard Application Programming Interface, let users can design function which they want, and do not use the manufacturers good proprietary operating system and functions. OpenFlow allows users to freely choose vendors, not limited by specific vendors and specific functions. The main goal of this thesis is to create a OpenFlow switch monitor system. It can find out all host and traffic pass through switched under controller, and provide a simple web page which the network administrator can modify each flow priority. Allow network administrator to manage the whole network more easily.

## 致謝詞

首先誠摯的感謝指導教授楊朝棟博士，老師悉心的教導使我得以一窺網路以及雲端運算領域的深奧，不時的討論並指點我正確的方向，使我在這些年中獲益匪淺。老師對學問的嚴謹更是我輩學習的典範。

再來要感謝的是各位協助我論文更加完美的各位口試委員，感謝呂芳懌老師，也是當初我在大學部做專題的時候的指導老師，呂老師對研究方法的要求十分嚴格，但也因為如此，使得我們這組後來在專題評審時，獲得許多的讚賞，也感謝呂老師在作業系統、資料庫等課程的教學，奠定了我的許多基礎知識，才有辦法在後來的研究所課程中如魚得水的輕鬆。也感謝賴冠州教授、伍朝欽教授、李冠憬教授不辭辛苦地前來東海當我的口試委員，為我的論文提出了許多的建議，使得本來鬆散的文章結構，重整之後看起來總算像了個樣。

本論文的完成另外亦得感謝高效能實驗室的各位學弟妹及學長姐的大力協助，尤其是陳宏彥學長，從高中就認識他是我人生中莫大的幸運，使我從 C/C++、Linux、各種伺服器架設等等的領域皆越來越得心應手，黃智霖學長也常常提醒我一些技術上的重點，陳柏翰學長及黃冠龍協助處理實驗室的大小事務，幫我們減輕了不少負擔，這篇論文的完成也要感謝蘇奕偉以及楊耀佑，沒有他們的幫忙我就沒辦法完成這份論文，另外還有更多更多的人要感謝，他們不只是在論文上，更是生活上的幫助，使我可以順利的完成論文。

兩年半裡的日子，實驗室里共同的生活點滴，學術上的討論、言不及義的閒扯、讓人又愛又怕的宵夜、趕作業的革命情感，感謝眾位學長姐、同學、學弟妹的共同砥礪，你/妳們的陪伴讓兩年半的研究生活變得絢麗多彩。

最後，謹以此文獻給一直支持我，我所摯愛的雙親。

# Table of Contents

摘要	I
Abstract	II
致謝詞	III
Table of Contents	IV
List of Figures	VI
List of Tables	VIII
<b>1 Introduction</b>	<b>1</b>
1.1 Motivation . . . . .	1
1.2 Contributions . . . . .	2
1.3 Thesis Organization . . . . .	2
<b>2 Background and Related Work</b>	<b>3</b>
2.1 Virtual LAN . . . . .	3
2.2 OpenFlow . . . . .	4
2.3 OpenvSwitch . . . . .	5
2.4 Software Defined Network . . . . .	6
2.5 Cloud Computing . . . . .	8
2.6 Virtualization . . . . .	9
2.6.1 Xen's Architecture . . . . .	15
2.6.2 KVM's Architecture . . . . .	16
2.7 Related Work . . . . .	19
<b>3 System Implementation</b>	<b>21</b>
3.1 System Architecture . . . . .	21
3.1.1 OpenFlow Testbed . . . . .	22
3.1.2 Network Configuration . . . . .	24
3.1.3 OpenvSwitch configuration . . . . .	25
3.1.4 Virtual Machine Configuration . . . . .	26
3.2 System Setup . . . . .	27
<b>4 Experimental Results</b>	<b>32</b>
4.1 Experimental Environment . . . . .	32
4.2 Experimental Results and Discussion . . . . .	36

---

<b>5</b>	<b>Conclusions and Future work</b>	<b>47</b>
	<b>Bibliography</b>	<b>49</b>

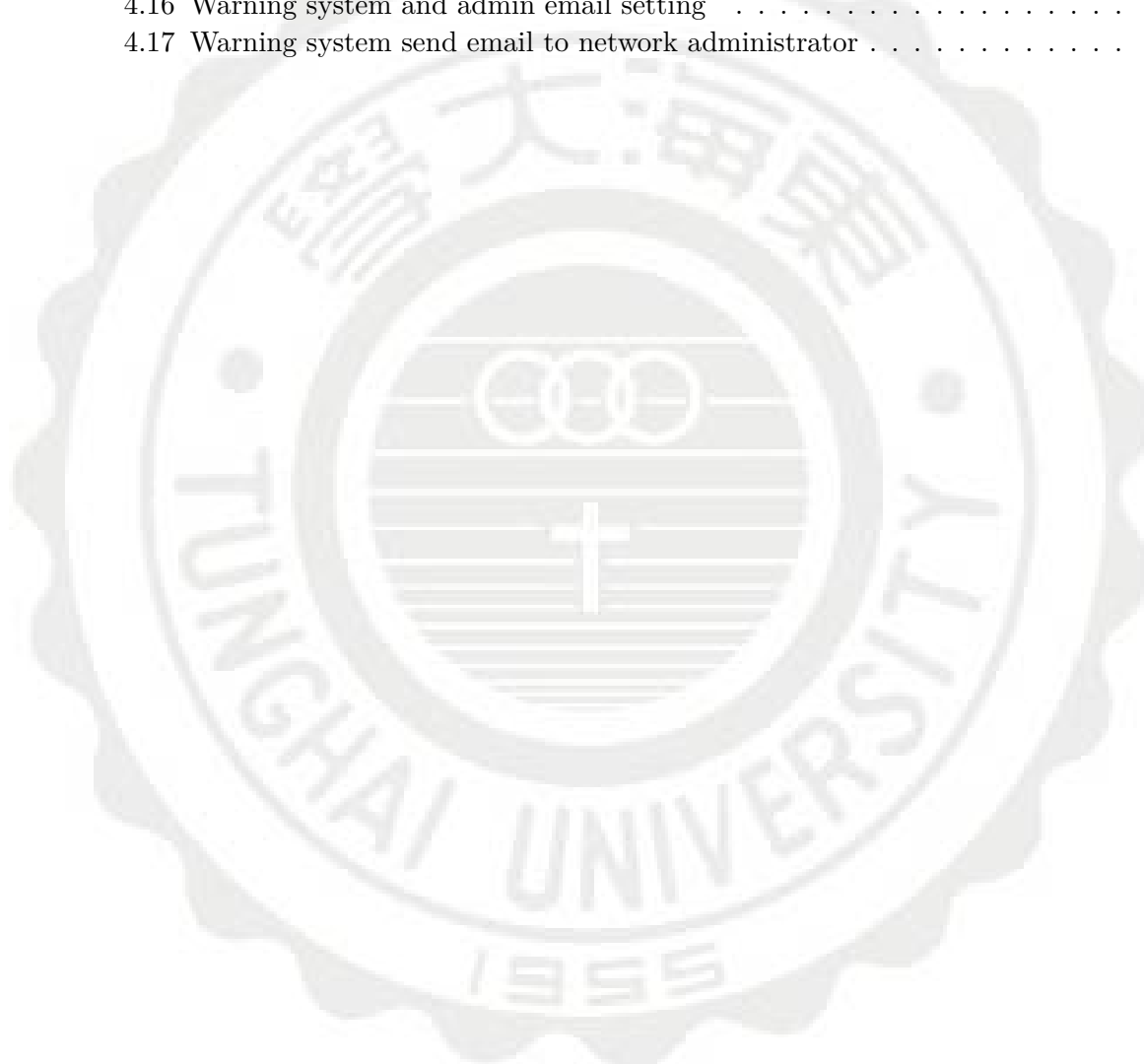


# List of Figures

2.1	Virtual LAN	4
2.2	OpenFlow Architecture	5
2.3	OpenvSwitch Architecture	6
2.4	SDN Architecture	8
2.5	Virtualization diagram	10
2.6	The general operation system	11
2.7	The virtualization operation system	12
2.8	Para-virtualization	13
2.9	Host Virtualization	14
2.10	Host and Xen's hypervisor type	15
2.11	Domain0 and DomainU	16
2.12	Intel virtualization technology	17
2.13	KVM's architecture	18
3.1	Network Layers	21
3.2	DC OpenFlow	22
3.3	OpenFlow Testbed	23
3.4	Network Type 1	23
3.5	Network Type 2	24
3.6	GSM 7352S2 configures	24
3.7	OpenvSwitch installation	25
3.8	OpenvSwitch check and setup controller ip	25
3.9	OpenvSwitch tool relation	26
3.10	Dashboard of Floodlight OpenFlow Controller	28
3.11	Check Switched status	29
3.12	Web page of Indigo OpenFlow firmware for Netgear GSM7352S2	30
3.13	Indigo OpenFlow firmware to view flow table at switch	31
4.1	Experimental Environment	33
4.2	Web control interface	34
4.3	Web control, hide MAC without IP	35
4.4	Web control, set priority	35
4.5	Web control, set flow to switch	36
4.6	Experimental Result	37
4.7	Web Setting 1	38
4.8	Web Setting Host 1 Priority to 19999	39
4.9	Web Setting Host 1 Priority to 5000	40
4.10	Web Setting Host 1 Priority to 5000	41



4.11 Web Setting Host 2 Priority to 19999 . . . . .	41
4.12 QoS Result 1 . . . . .	42
4.13 QoS Result 2 . . . . .	42
4.14 Network traffic without QoS policy . . . . .	43
4.15 Network with QoS policy after 10 minutes . . . . .	44
4.16 Warning system and admin email setting . . . . .	45
4.17 Warning system send email to network administrator . . . . .	46



# List of Tables

2.1	Comparison of Xen and KVM . . . . .	18
4.1	Hardware specification . . . . .	32
4.2	OpenFlow Match Field . . . . .	34
4.3	Using iperf with different probe method . . . . .	36

# Chapter 1

## Introduction

### 1.1 Motivation

At Internet just begins, users who dial up to a site, chat and transform file with other user, user need to pay phone bill. Next, users use Modem and dial a special number to the ISP whose speed rate is 14.4kbps to 52.2bps. From now, Internet becomes more and more user using. But user using ISP connect to Internet, user need to pay double bills, phone bill and Internet connection cost, virtually negates the willingness of the part of the population connected to the Internet. At this time, the Internet has been begun to flourish, many people create their own HomePage profile, sharing small file and image like MIDI file, GIF file, JPEG file, etc. The BBS also developed at this time, because in low connection speed, if people want to send some information to other, the fattest way is through text. After Modem is xDSL technology, the network speed began to progress to Mb level. In Taiwan, the most using methods are ADSL and VDSL, the highest speed can reach 100Mbps, when user has a convenient and unlimited amount of Internet plan, Naturally, user and company began to develop the architecture on which the service, and the cloud is showed.

With the continuous development of cloud technology, people's daily life is close to the cloud service. These large number of cloud service provide people convenient environment, Since ancient times, messaging and information exchange path changed. Many cloud vendors started budding, like Google, Amazon, Microsoft, Yahoo, Apple, etc. All these vendors provide different cloud service, like message passing(MSN), video

call(skype, Google+ Hangout), file sharing(Dropbox, Google drive) and many other service still developing. Overturn the computer always need in stand-alone operating. Now, you just open your browser at any computer with it, you can get all file and data that you store in cloud, even not need to install software, you still can do all your work like edit documents, spreadsheets, presentations, and more. In past time, you always using these file at some place already install software, then you can edit it. But now, just put these file to cloud, you got no trouble and everything is fine.

## 1.2 Contributions

In this generation, we use cloud service in everyday life, whether send message, transmit photo, or chat with VoIP software. All of these service use cloud service and network. But sometime we will encounter situations like click some file to download, but the server has no response or show blank page, or program pop-up a message box, show "Cannot connect to server now, please try again later." . Most of these situations are caused by the network. The cloud service vendor did not predict such a large network traffic, its also because the original network design is lack of flexibility and QoS mechanisms. This thesis focused on the cloud service vendor network and use OpenFlow as access layer switch for QoS mechanisms. Goal is to reduce the load of the core layer switches and QoS devices, dispose of network traffic from the access layer.

## 1.3 Thesis Organization

In chapter 2, we describe the techniques used and some background knowledge. Chapter 3 describes the system architecture and key algorithms which this thesis is used. Chapter 4 makes some experiments for our proposed system. Chapter 5 are conclusions and future work of this thesis.

## Chapter 2

# Background and Related Work

### 2.1 Virtual LAN

The full name of Virtual LAN is Virtual Local Area Network, some time we say VLAN or 802.1Q. IEEE announce 802.1Q in 1999. It provide data separation and security between network traffic of Ethernet. Using VLAN Tagging to share a physical interface for multiple VLAN, and keep message secure.

VLAN has three types:

- Port-Based VLAN: Also called Static VLAN. Each physical interface access only one VLAN, user specify at configuration file.
- MAC-Based VLAN: When user connect to switch interface, switch send the mac of connected user, to a VLAN Management Policy Server (VMPS), network administrator can assign VLAN to user.
- Protocol-Based VLAN: If we have a network, that running multiple protocol, like Novell IPX, AppleTalk, TCP/IP, we can use protocol-based VLAN separate each kind protocol to each VLAN.

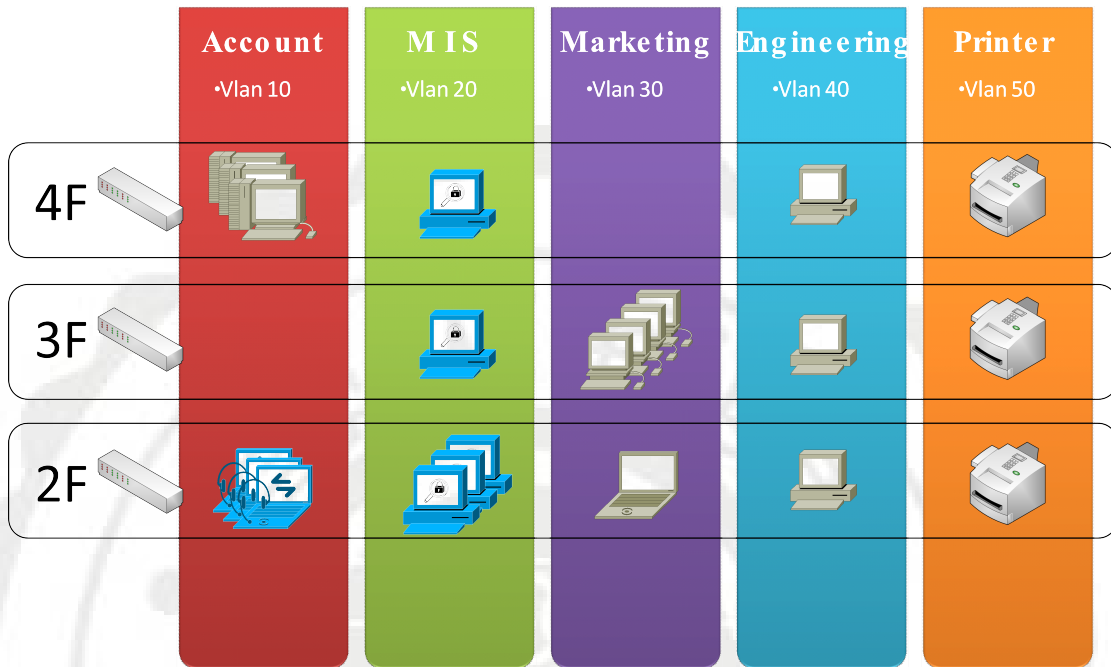


FIGURE 2.1: Virtual LAN

The figure 2.1 shows we using port-based VLAN, we can separate each network flow of department, prevent different department get other department's data. VLAN also keeps network more clean. Because separate VLAN also separate broadcast domain. Minimal the collision event.

## 2.2 OpenFlow

OpenFlow is a layer 2 protocol that separate data plan and control plan for better performance. OpenFlow gives a remote controller permission to modify the action of network device, just like normal router and switch usually do. But normal router or switch processing huge throughput then CPU loading will increase, it because switch need to decide the packet path for each packet, when CPU loading increase, process packets will be little slow, increase more, slow more, until cpu loading full, machine crash. OpenFlow allow the path of packets can be determined by software running on PC or router, allow more complex traffic management then before. Network admin can management network better than using access control lists (ACLs) or routing protocol. OpenFlow controller using a secure channel communicate with OpenFlow switch, sending the define message. OpenFlow protocol was released on February 28, 2011.

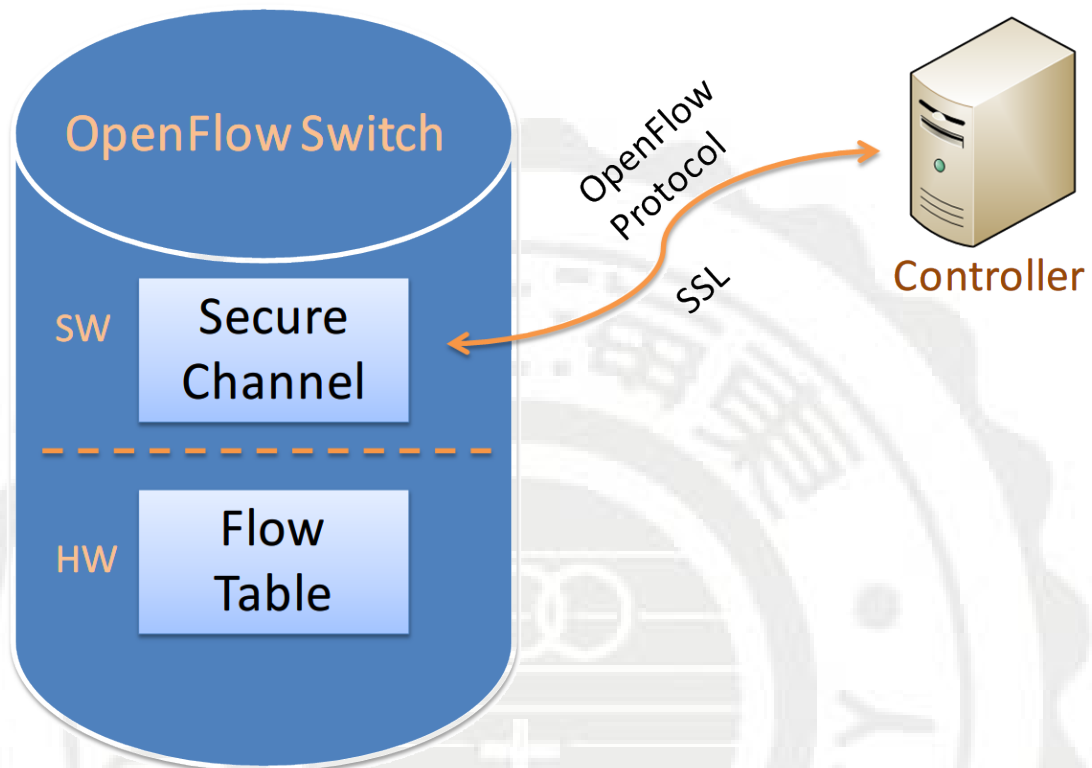


FIGURE 2.2: OpenFlow Architecture

### 2.3 OpenvSwitch

OpenvSwitch is a production quality, multilayer virtual switch licensed under the open source Apache 2.0 license. It is designed to enable massive network automation through programmatic extension, while still supporting standard management interfaces and protocols (e.g. NetFlow, sFlow, SPAN, RSPAN, CLI, LACP, 802.1ag). In addition, it is designed to support distribution across multiple physical servers similar to VMware's vNetwork distributed vswitch or Cisco's Nexus 1000V

As figure 2.3 shown, we can see OpenvSwitch split to parts. At top is ovs-vswitchd,

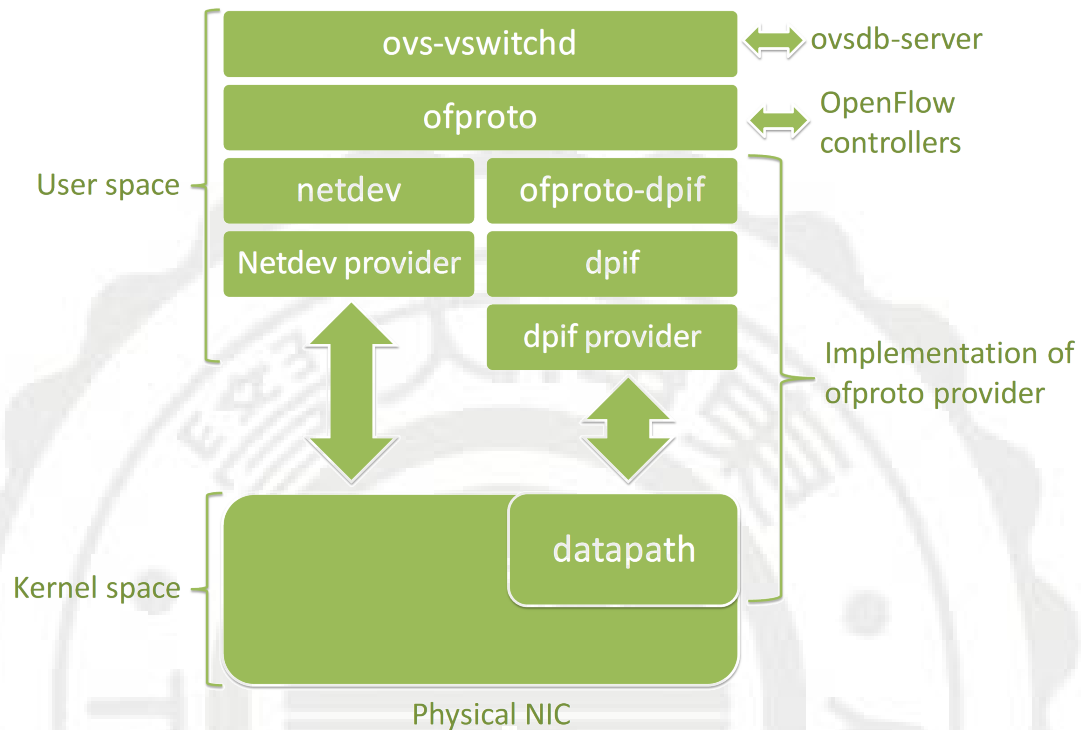


FIGURE 2.3: OpenvSwitch Architecture

## 2.4 Software Defined Network

In recent year, network technology has been little grow and change, but since Software Defined Network published at 2011, many things change. The core technology of Software Defined Network is OpenFlow, it is a software can running at normal operating system, let user change network architecture and control network flow. Because it separate smart system(control layer) and real data transmission(data layer), after this point, how to stream network data, where the packet should go, are no longer a router or a switch directly specified, but a manager of datacenter or network system administrator control. In simple term, Software Defined Netowrk give more permission of network control, from device vendor to user. Let user directly to management with the system characteristic and system function, this act not only saves money, but also, provide more flexibility to satisfy business needs.

Software-defined network is proposed in March 2011 by the Open Network Foundation, the research originally led by Stanford University and the University of California at Berkeley. After the establishment of the Foundation, the idea transformed into a commercial product, order to achieve a software-defined network. At begin, OpenFlow



is a function when the investigator wants to free experiment, and manufacturers do not need to publish their own source or show the inside of device. Add OpenFlow to ethernet switch, router or wireless access point, provide a standard Application programming interface(API) for program, then the experiment can be continue.

Software-defined network has following advantages:

- Any developer can program the device, provide the flexibility of network using, operating, and sales model.
- Users faster access to the desired function, without equipment suppliers such features into their own product lines.
- Software-defined network implement the virtualization of network, combine network, calculate and storage, control whole IT environment.

With Software-defined network, the network operating system can running at any personal computer or any kinds of server, and not modify kernel or system nodule. Switch is still responsible for the actual packet-switched. But OpenFlow protocol will run between switch and controller. When switch get a packet without action record, it will send packet to controller, decide what action should do with this packet, and send action message to switch. So switch do action by itself after get the message.

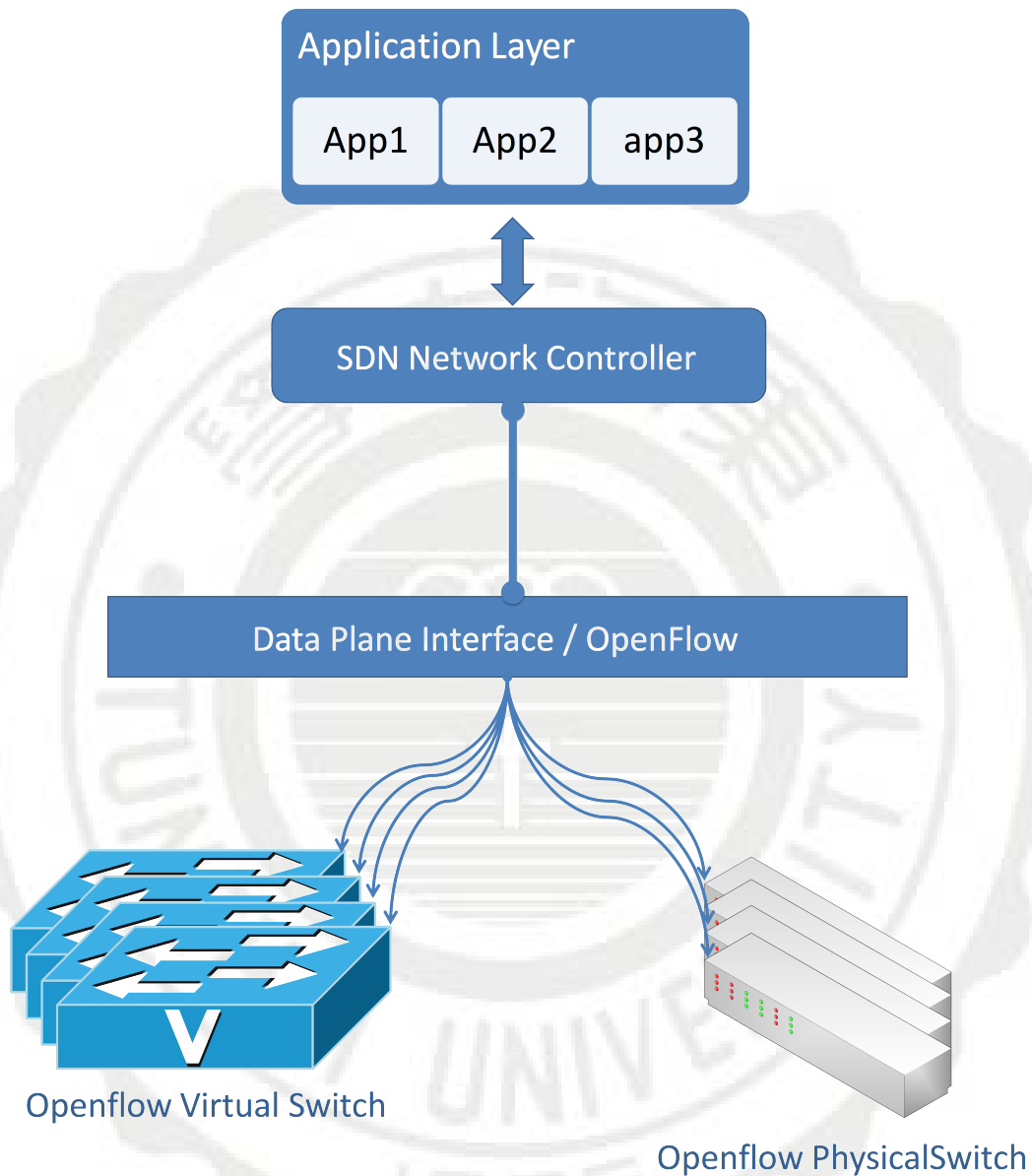


FIGURE 2.4: SDN Architecture

## 2.5 Cloud Computing

Cloud computing is a computing approach based on the Internet. In this way, resources can be shared by the required hardware and software available to computers and other devices. Users no longer need to understand the "cloud" in the details of the infrastructure, do not possess the necessary professional knowledge, without direct control. Cloud computing describes a new Internet-based services to increase IT use and delivery models, usually involving the Internet is easy to provide dynamic and often is a virtual

extension of the resource. The cloud is network, Internet a metaphor. Cloud computing can be considered include the following levels of service: infrastructure as a service (IaaS), Platform as a Service (PaaS) and Software as a Service (SaaS).

- Infrastructure as a Service (IaaS): Users can follow the required level of computer and network equipment and other resources, to the service provider subscription service, and may require changes to settings, and service provider by users of the CPU, memory, Disk space, network load to calculate the costs.
- Platform as a Service (PaaS): development of services vendors who rented to a computer, this computer has all the necessary hardware and software developers environment; or to provide application developers to market, in accordance with the amount of traffic with the use of resources Developer fees.
- Software as a Service (SaaS): the software stored in the data center to provide users network access services, according to period or pay-per-order the type of charge.

## 2.6 Virtualization

Virtualization technology is due to present a single host more and more powerful hardware performance, if only a single server implementation of the tasks seem too much idle time, so multiple hosts by the hardware virtualization technology, the original value Line by more than one virtual host, after the service, placed on a single powerful server is running, but also makes virtualization virtual machine after the machine easier to control than real checks and controls, more flexible configuration and can be anywhere in the world And can achieve real-time transfer of virtual machines to ensure uninterrupted service. The virtualization diagram shown on Figure 2.5.

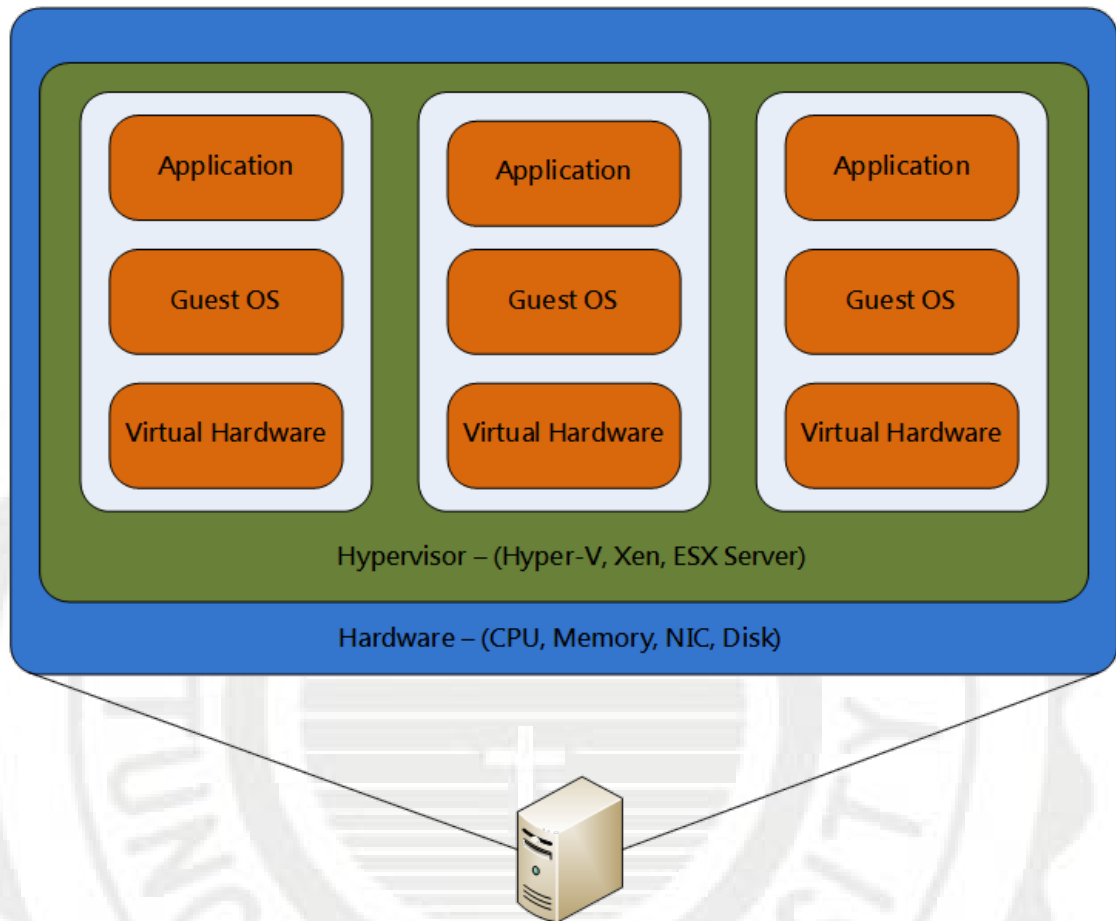


FIGURE 2.5: Virtualization diagram

The virtualization technology is an internal access control by CPU, in a real operating system, applications and users between the placements of a administrator to control the entire virtual machine CPU process, to enable Guest OS CPU think that they have full rights Implement their own programs.

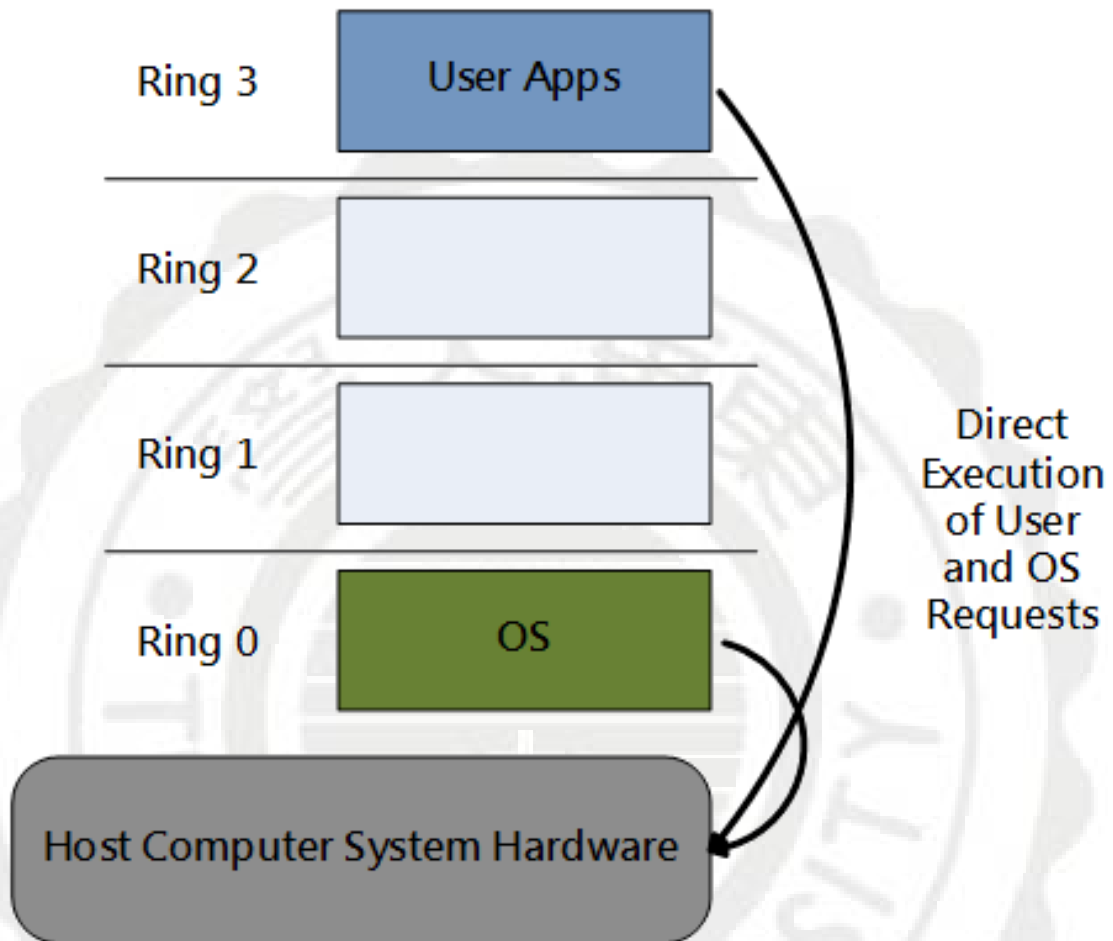


FIGURE 2.6: The general operation system

The case of the general operation of the operating system shown on Figure 2.6, the user's program is the implementation of the Ring 3 in the CPU part, and the implementation of the operating system and then operate in Ring 0 in the control of CPU and the hardware, the hardware is a direct implementation. By the operating system and user application are to the instructions.

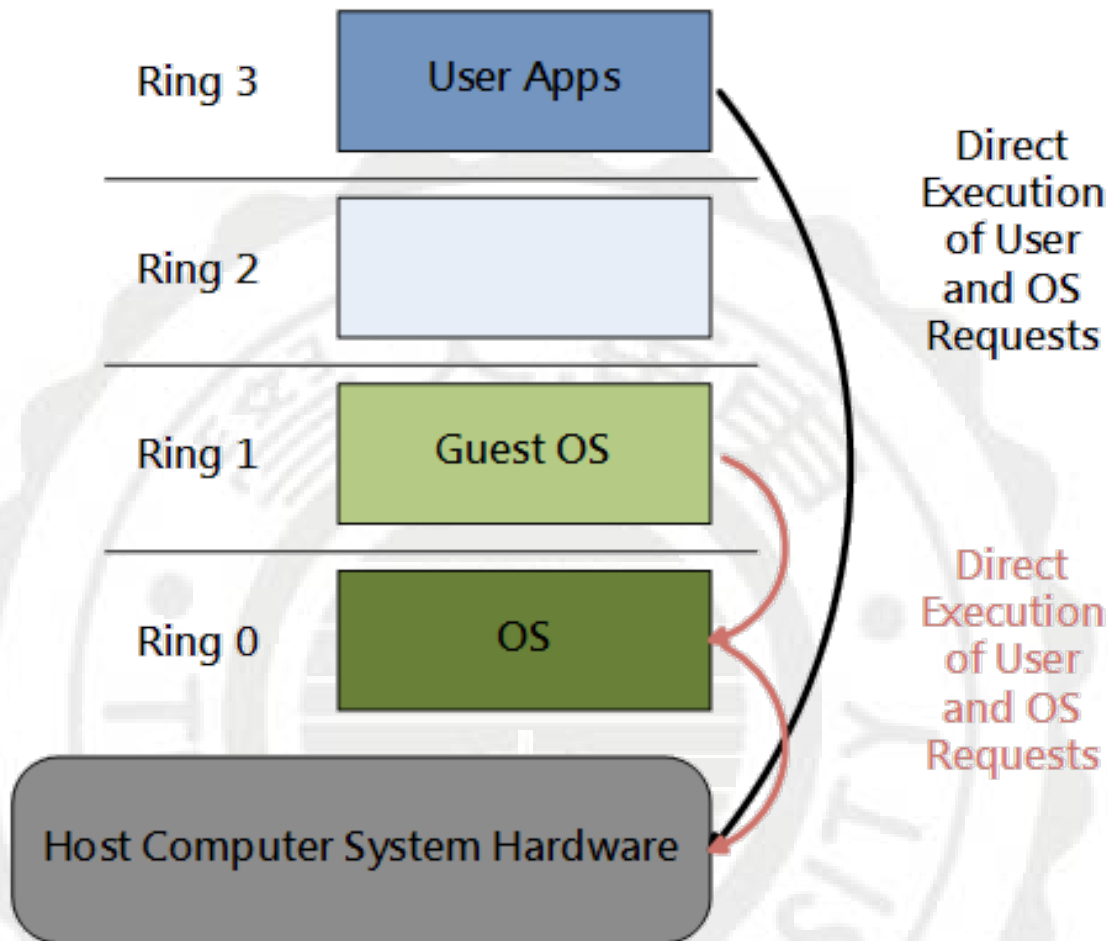


FIGURE 2.7: The virtualization operation system

Figure 2.7 shows the virtualization operation system. User application is still part of the implementation of the Ring 3, and the virtual operating system out (Guest OS) into the implementation of the Ring 1, was originally part of the operating system should become a Virtual Machine Manager (VMM) of the holding, Guest OS is not to be executed directly to the CPU instruction execution, but to use Virtual Machine Manager made after translation to CPU and hardware for the implementation of the action.

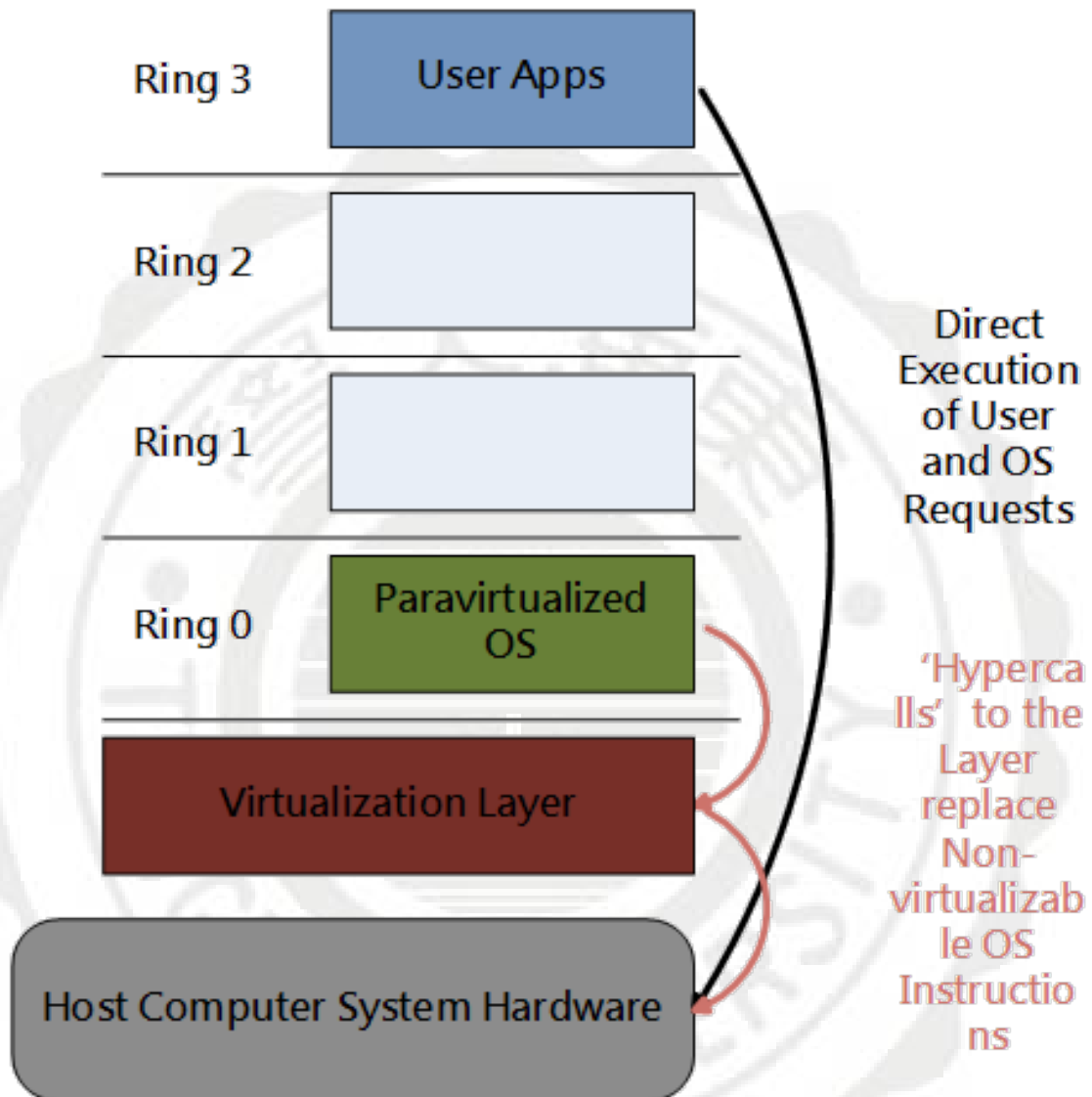


FIGURE 2.8: Para-virtualization

Para-virtualization with the full virtualization of the difference is that full virtualization of the Guest OS does not need to make any changes to the line-up will have all the hardware that they own rights, but so are the underlying needs of the operation command VMM to assist the conversion, resulting in the implementation of efficiency will be somewhat less, and some low performance virtualization to solve the problem, because the Guest OS does not need to go through the operation of translation at this time, but later issued directly through the bottom of the virtual layer Hardware, eliminating the need for a conversion step, is the performance has improved, but the disadvantage of this method is that the core of the operating system must be modified so that the underlying hardware, operating systems and virtual step instructions, you

need the operation of the operating system software with the virtual The results can be achieved this way, the difficulty of this method lies. Xen is the University of Cambridge Computer Institute in the GNU (General Public License) of the GPL (General Public License) authorized the release of the free software, which aims to achieve high performance mainframe virtualization technologies to enable a single host can be modeled as multi-Taiwan (heterogeneous operating system) hosts. For the public in source code, a variety of programs and related technology, ongoing development, also is one of open source virtualization platform, the main program. The development of the Xen VMM (Virtual Machine Monitor) software for the effective and safe use of high-end mainframe computer of the resource is shown on Figure 2.9. With the rising performance of the host CPU, and memory prices and other factors, causing the host idle rate, combined with low PC into the mainstream, corporate owned dramatic increase in the number of hosts, resulting in increased operating costs. To save costs, it will be split into a complex virtual host host the increasing demand.

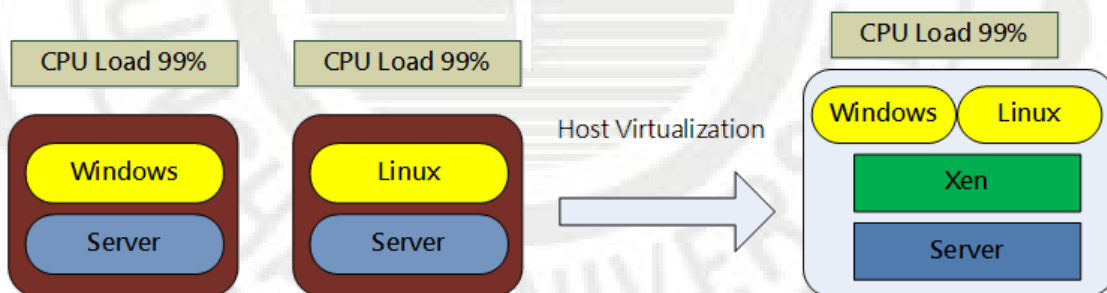


FIGURE 2.9: Host Virtualization

KVM (Kernel-based Virtual Machine) is a virtualization solution for Linux on x86 hardware containing virtualization extensions Technology (Intel VT or AMD-V). It consists of a loadable kernel module "kvm.ko", which provides the core virtualization infrastructure and a specific processor module, supports KVM intel.ko module or KVM amd.ko module. KVM on a machine can run multiple virtual machines. Each virtual machine has its own virtualized hardware, such as: network card, disk, video card .... Host of desktop processors generally the average utilization rate of about 15-20% will be hosting the DC(Data Center) the use of space, power and related maintenance costs much higher than the virtual host, so the host virtualization Technology helps enterprises or research institutions to reduce costs. Although there are some virtualization into the host, if the calculation of its associated costs down, may not reduce the overall



cost and other remarks, but did not reduce the host system vendors support the trend of virtualization, but gradually risen to the mainstream.

### 2.6.1 Xen's Architecture

Host virtualization software generally divided into two kinds of Host OS and Xen's hypervisor as shown in Figure 2.10, Host OS layer deployed in the virtual Windows, Linux and other operating systems, and then install the virtualization layer on top of other operating systems, virtualization Layer below the operating system, known as Host OS, the top of the OS called the Guest OS. The Xen's hypervisor is installed directly on the host, the other want to deploy the operating system installed on it, and to cut the resources required for Host OS, better performance, CPU, Memory, Network, Storage and other resource management are more Easy. The use of Xen's hypervisor and VMM(Virtual Machine Monitor) architecture. Main purpose of efficient and safe control of the host CPU, Memory and other resources[?].

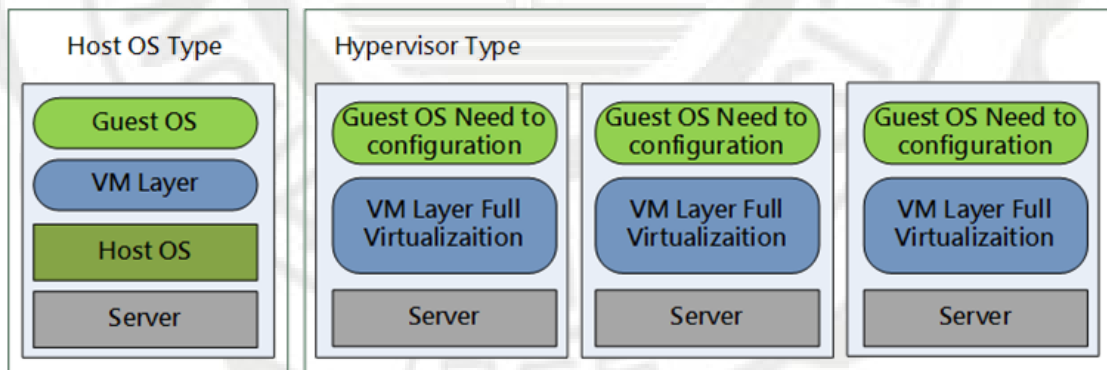


FIGURE 2.10: Host and Xen's hypervisor type

Xen's hypervisor used is divided into Para-Virtualization and Full-Virtualization. Para-Virtualization in the Guest OS kernel must do the appropriate amendments, such as the Linux and other open source OS, its core can be modified for Xen and adjustments made in particular to reduce the burden and improve performance. And Full-Virtualization in the Guest OS cannot be amended, more suitable for a similar Windows installation. Processor vendor Intel Virtualization Technology(Intel VT) and the "AMD Virtualization(AMD-V) also support virtualization, with which the host CPU can be virtual environment in the semi-direct install Windows. There are also Windows in Para-virtualization drivers running on the environment.

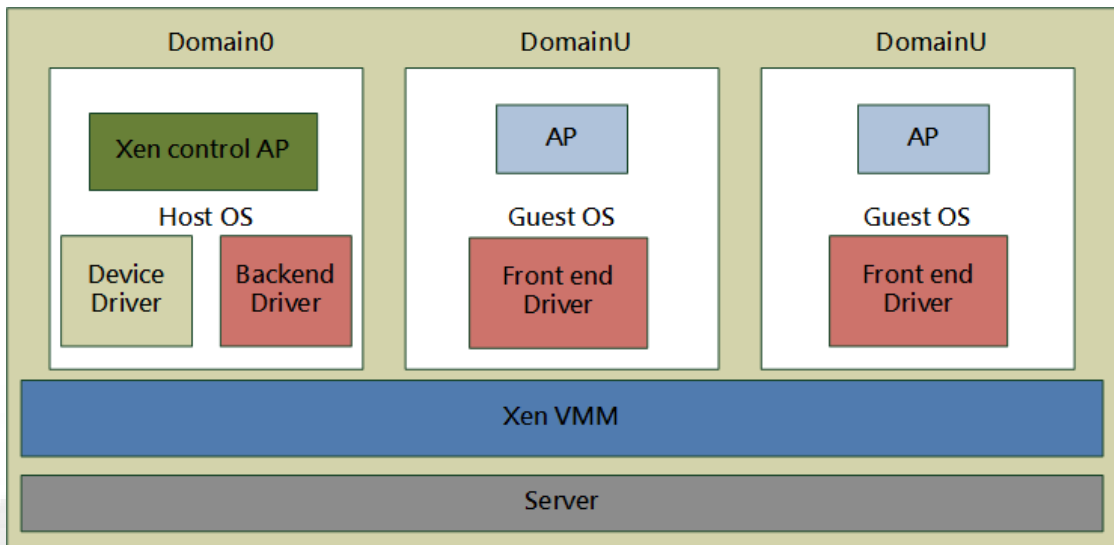


FIGURE 2.11: Domain0 and DomainU

Xen management in the virtual host, use the Domain to do management unit, Domain is divided into two types as shown in Figure 2.11, one of which is managed by the Domain0, play like the Host OS role, a Xen Control of AP, to manage another type of DomainU. DomainU installed on the field Guest OS and AP, in the use of physical resources, must be through Domain0 took the deal, cannot directly call the hardware drivers. Xen in the industry, the American have been led by Novell SUSE Linux Server(SLES) and Red Hat Enterprise Linux(RHEL) and other commercial Linux version used. In addition, Oracle also introduced a virtualization product Oracle VM, which Sun Microsystems released xVM Server and other products. It can be seen, Xen virtualization software on the host, has been widely supported by the system vendors.

### 2.6.2 KVM's Architecture

Kernel-based Virtual Machine(KVM) is a Linux core, a part of the framework, the current structure of native virtualization support KVM hardware-assisted virtualization is supported by the CPU, Intel virtualization technology called VT(Virtualization Technology, as shown on Figure 2.12) or AMD's AMD-V Technology in Linux through the two CPU module to support two different KVM (Intel: kvm-intel.ko; AMD: kvm-amd.ko). In RHEL5 update4 automatically according to `/proc/cpuinfo` of flag to select the appropriate CPU module, this script file stored in `/etc/sysconfig/modules/kvm.modules` [36].

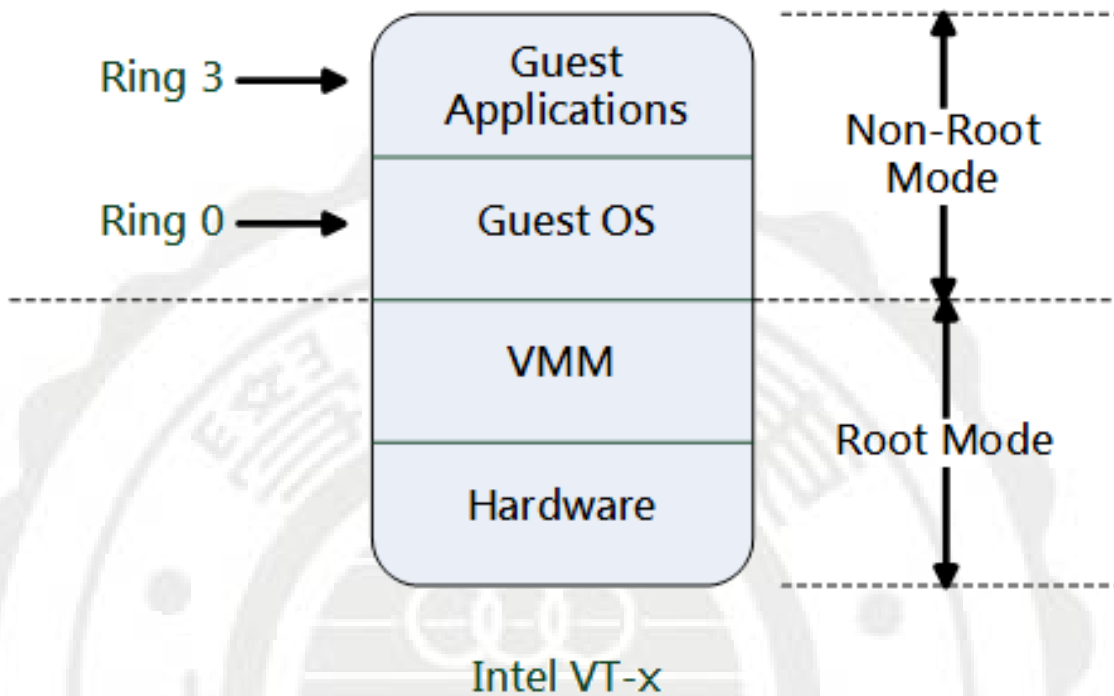


FIGURE 2.12: Intel virtualization technology

Support for the Para-virtualization, such as he now supports Linux and Windows, Para-virtual network device drivers, and the balloon (on the memory technology VMM-virtual memory manager) has done for Linux Guest's CPU optimization. KVM is currently only operating in the i386/x86\_64 the CPU on the system, such as PowerPC and IA64 are still in development stage. Linux's core team in Linux 2.6.20 (February 2007) version of KVM will be included. FreeBSD Kernel module approach also supports KVM. However, KVM alone cannot be completed virtualization must also do something with the QEMU device simulation and the following GNU software:

- KVM kernel module: GPLv2
- KVM user module: LGPLv2
- QEMU virtual CPU core library and QEMU PC system emulator: LGPL
- Linux user mode QEMU emulator: GPL
- BIOS files (bios.bin 、vgabios.bin and vgabios-cirrus.bin): LGPLv2 or later

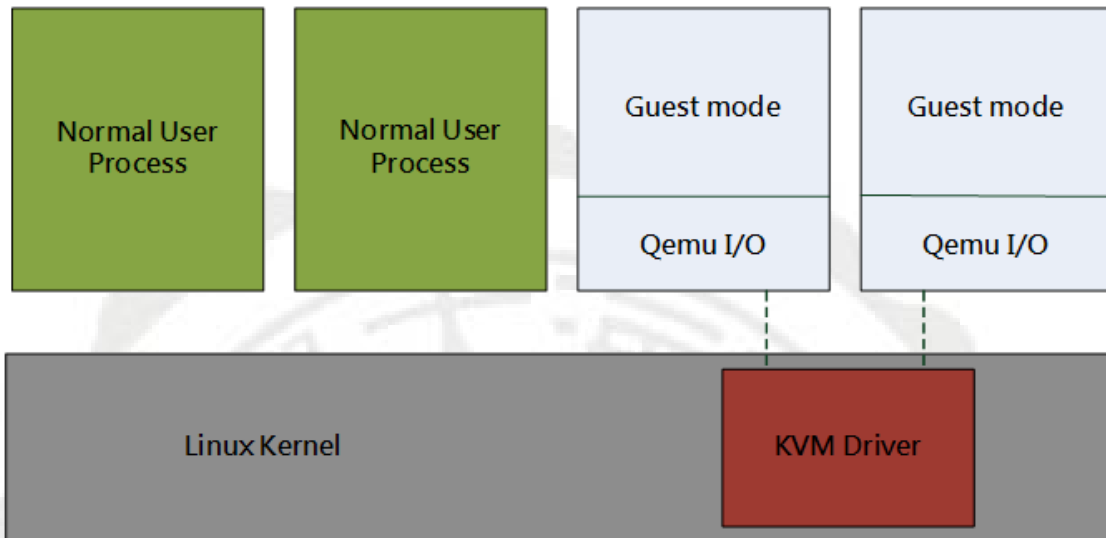


FIGURE 2.13: KVM's architecture

Show in Figure 2.13, KVM's architecture consists of two parts:

- Kernel Device Driver (managing the virtualization hardware) - Used to manage and simulation Virtual Machine hardware.
- User space process - qemu is a PC hardware emulator, after the modified KVM become kqemu.

	Virtualization	Advantages	Kernel integrity	Hardware dependencies
Xen	Para-Virtualization Full-Virtualization (need CPU suppose)	CPU performance better	Kernel 2.6.23 was added	Does not have Intel VT-x or AMD-V
KVM	Full-Virtualization (need CPU suppose)	I/O performance better	Kernel 2.6.20 was added	Must have the Intel VT-x or AMD-V

TABLE 2.1: Comparison of Xen and KVM

Table 2.1 shown, this thesis will compare the advantages:

- Scalability and elasticity
- Availability and reliability
- Manageability and interoperability
- Accessibility and portability

Finally, select the KVM-based virtualization platform for the article.

## 2.7 Related Work

In recent years, performance improvement management process technology advances make it possible to try to use the virtual machine (VM) computing platform. Many studies have been implemented through the virtual network environment, reduce system costs. Data transmission between server nodes often appear in parallel and distributed computing systems, high cost of the network may cause significant loss of performance throughout the system.

Blanco want to know that how OpenFlow switch improve the network speed. And how OpenFlow protocol how to enable flow isolation and resource slicing. They using a linux based PC to simulate OpenFlow switch and measure the packet switch speed at OpenFlow switch, layer-2 Ethernet switch and layer-3 IP router. [5] Their conclusion is using a linux based PC to be a OpenFlow switch. Its performance is good. But the performance of OpenFlow switch at packet size at 64-bytes and 128-bytes are little worse, packet size larger then 128-bytes the performance as good as hardware layer-2 switch. It also suggest that if we went a flow has more performance, we should use hash table at OpenFlow.

Pisa at their work show that if combine Xen and OpenFlow, using the characteristic of OpenFlow, data and control plane separation, the packet lost rate will be decrease. They using OpenFlow network environment at virtual machine migration, is also reduce the number of the dirtied page at the process of migration, decrease the downtime at migration.[40]

Hayoung trying to use OpenFlow to improve performance of NOX and wireless OpenFlow switch, to prevent Access Point failure, make sure another AP will take off the traffic.[35] Some people also using OpenFlow to Academic Network, like Rostami design a prototype OpenFlow-enabled network using gigabit ethernet switch, they use ATCA switch platform to build it, but finally find out the bottleneck is in OpenFlow switches.[43] But Ferkouss trying to use OpenFlow at a 100 Gigabit network with TCAM and OpenFlow 1.1, got good performance.[12] At video streaming area, Egilmez using OpenFlow to do QoS routing to improve the video stream quality.[8] Rotsos tested

OpenFlow, find out the performance of OpenFlow switch depends on applied actions and firmware.[44]



## Chapter 3

# System Implementation

### 3.1 System Architecture

Original network are divided to three layers, core layer, distributed layer, and access layer, like Figure 3.1. Usually network flow will aggregation to core layer, then doing some action to flow, like firewall, QoS, VoIP, monitor, but these network flow aggregation to core layer always being a huge amount. More large amount of network flow, the device which process these flow should be more powerful. Our system implementation is doing these thing, which usually doing at core layer, shift to access layer or distributed layer. To solve the longstanding problem at network.

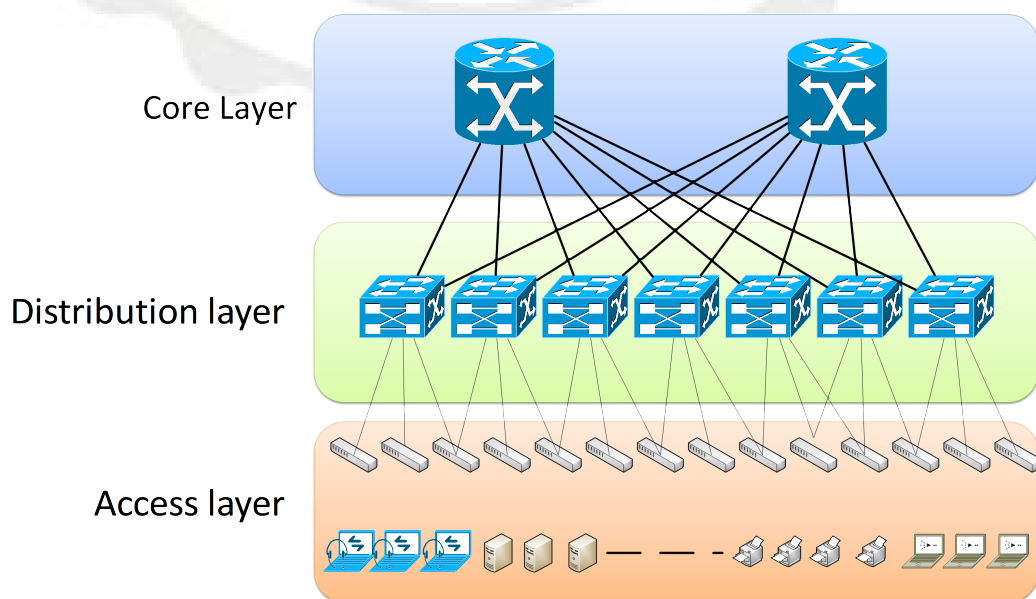


FIGURE 3.1: Network Layers

The system of this thesis using OpenvSwitch play the role of the access layer switch. This work also using three Netgear GSM7352S2 play the role of the distributed switch. Figure 3.2 Using sFlow to monitor network flow, and show network flow to admin console, let network administrator can doing some action to the OpenFlow switch, like drop packet, forward packet to the port which administrator specified, and change some header of the packet.

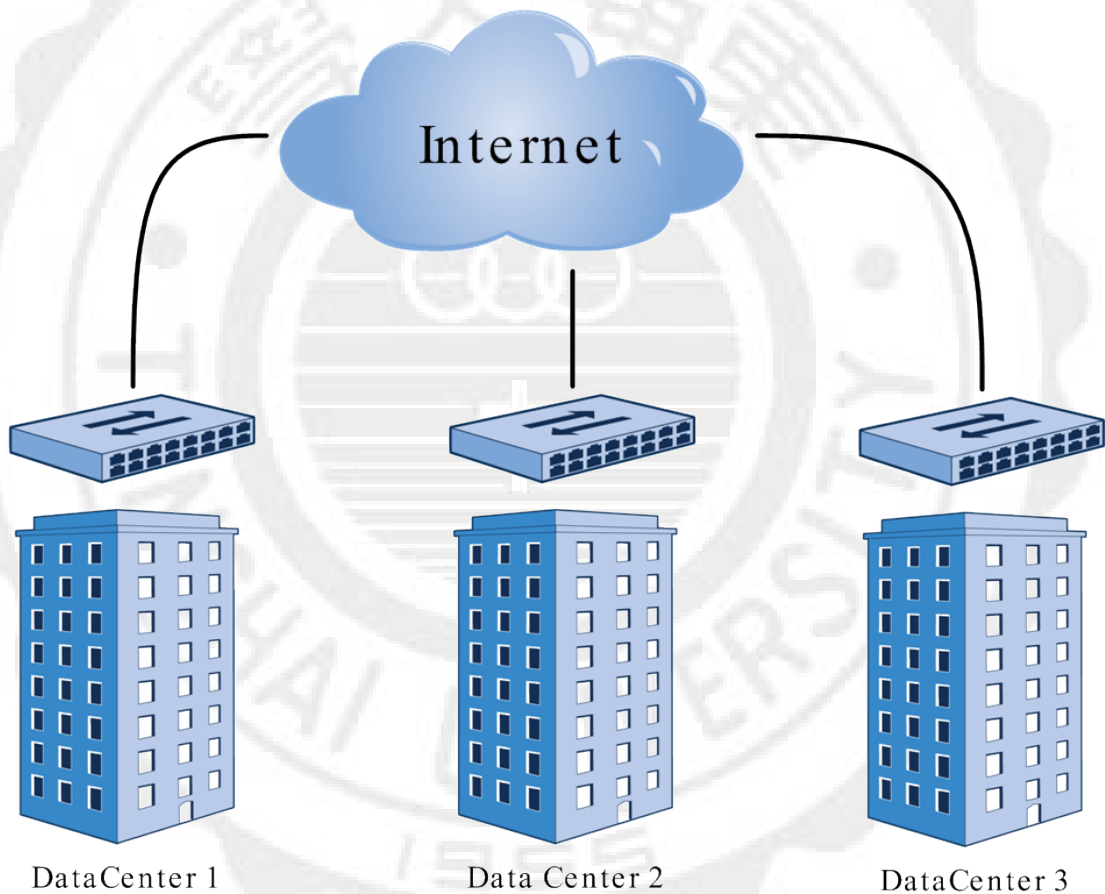


FIGURE 3.2: DC OpenFlow

### 3.1.1 OpenFlow Testbed

This section is showing our real OpenFlow Testbed. We have three Netgear GSM7352 switch and one SMC 8524T Gigabit switch. Netgear switch already change firmware to Indigo Open Source firmware, it is develop to support high rate for high port counts for OpenFlow. SMC switch is used to emulate traditional network as a normal layer 2 switch. PC 2, 3, 4 (Figure 3.3) are all have a dual port NIC, each port connect to one SMC switch and three Netgear switches in Type 2 mode. The testbed network have two



type for experimental, Type 1 is VM mode ( Figure 3.4 ), three Netgear switches connect to PC 2,3,4 each NIC, this mode is used to create VM to test OpenvSwitch function and emulate Data center at different places. Type 2 is traffic mode ( Figure 3.5 ), this mode is used to measure Netgear switches OpenFlow function, and its transmission efficiency.



FIGURE 3.3: OpenFlow Testbed

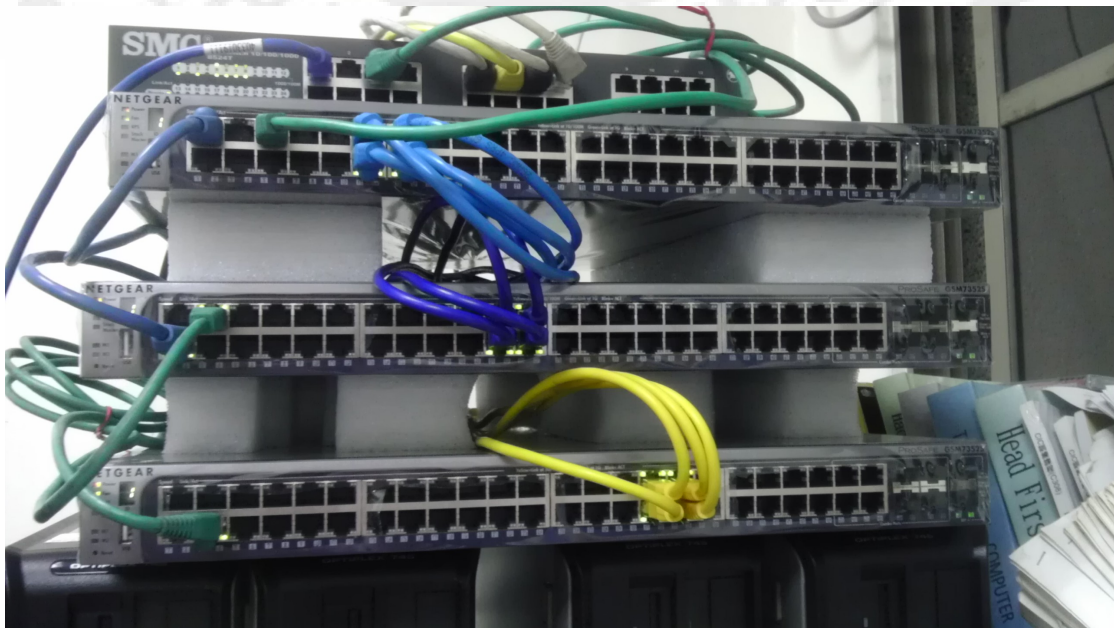


FIGURE 3.4: Network Type 1

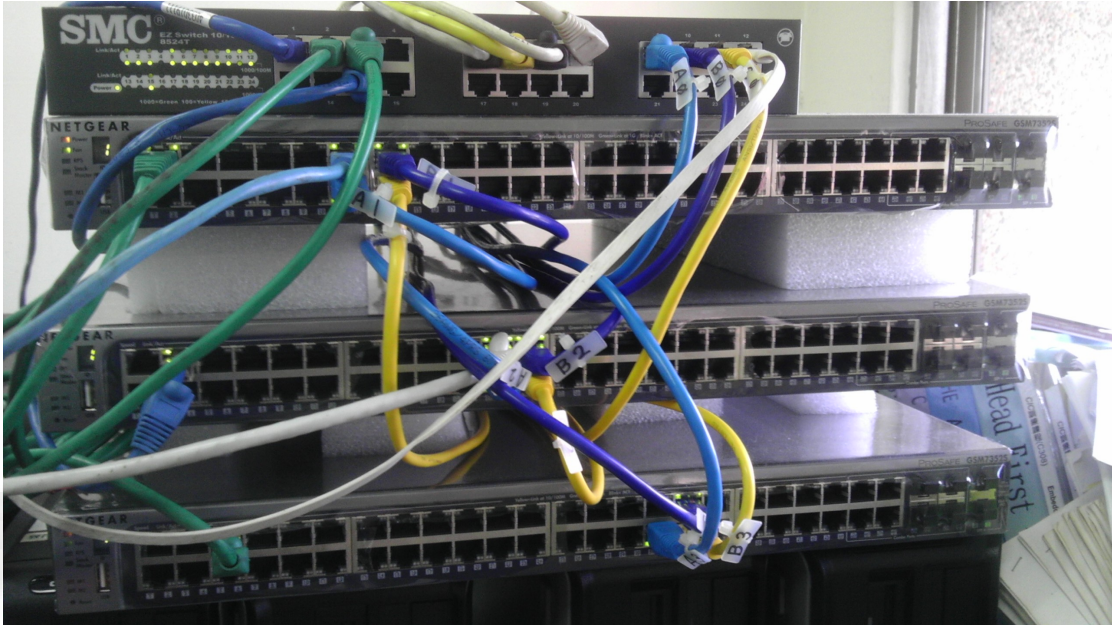


FIGURE 3.5: Network Type 2

### 3.1.2 Network Configuration

First, we should upload a firmware which suppose OpenFlow to these three switch GSM3752S2, then we using the serial console to access the switch which suppose OpenFlow, type these command to set what ip switch should be, and where the controller is.

```
cli
config set switch_ip $WHICH_IP_SWITCH_SHOWLD_HAVE
config set controller_ip CONTROLLER_IP
config set controller_port 6633
config set system_ref some_system_name
quit
```

FIGURE 3.6: GSM 7352S2 configures

After we setting these information to the switch, we can use the tool of OpenFlow protocol to control the switch, like add-flow or del-flow at switch

### 3.1.3 OpenvSwitch configuration

Because OpenvSwitch not the official option of KVM yet, so we need to install it manually. Type these commands at a Ubuntu Linux system to install OpenvSwitch.

```
$ apt-get install openvswitch-datapath-source  
bridge-utils  
$ module-assistant auto-install openvswitch-d  
atapath  
$ apt-get install openvswitch-brcompat openvs  
witch-common
```

FIGURE 3.7: OpenvSwitch installation

After installing OpenvSwitch, we need to check if it is really installed or not. We need to install OpenvSwitch, then set up the controller IP to the OpenvSwitch, let it can be managed and configured.

```
Check install  
  
$ ovs-vsctl show  
ovs_version: "1.4.0+build0"  
  
$ ovs-vsctl set-controller br-int tcp:$CONTROL  
LER_IP:6633
```

FIGURE 3.8: OpenvSwitch check and setup controller IP

At Figure 3.9 we depict OpenvSwitch each component to a figure. We can see there are two processes running at system user space, `ovs-vswitchd` is a process that communicates with OpenFlow controller and `ovsdb`, `ovs-server` is the `ovsdb` location, it stores all OpenvSwitch settings, and notifies datapath at kernel space if needed.

We also have these user space tools, help us to set up OpenvSwitch, `ovs-vsctl` is a command to manage the `ovsdb`, let users create bridges, specify bridge port mapping, etc. `ovs-dpctl` is a tool to manage datapath, most information is showing the status through netlink, but it can operate the flow in datapath also. `ovs-ofctl` is the management tool of OpenvSwitch, change settings in process of `ovs-vswitchd`. `ovs-appctl` is a management tool of `ovs-vswitchd`, using Process ID (PID) of `ovs-vswitchd` to control it or dump information.

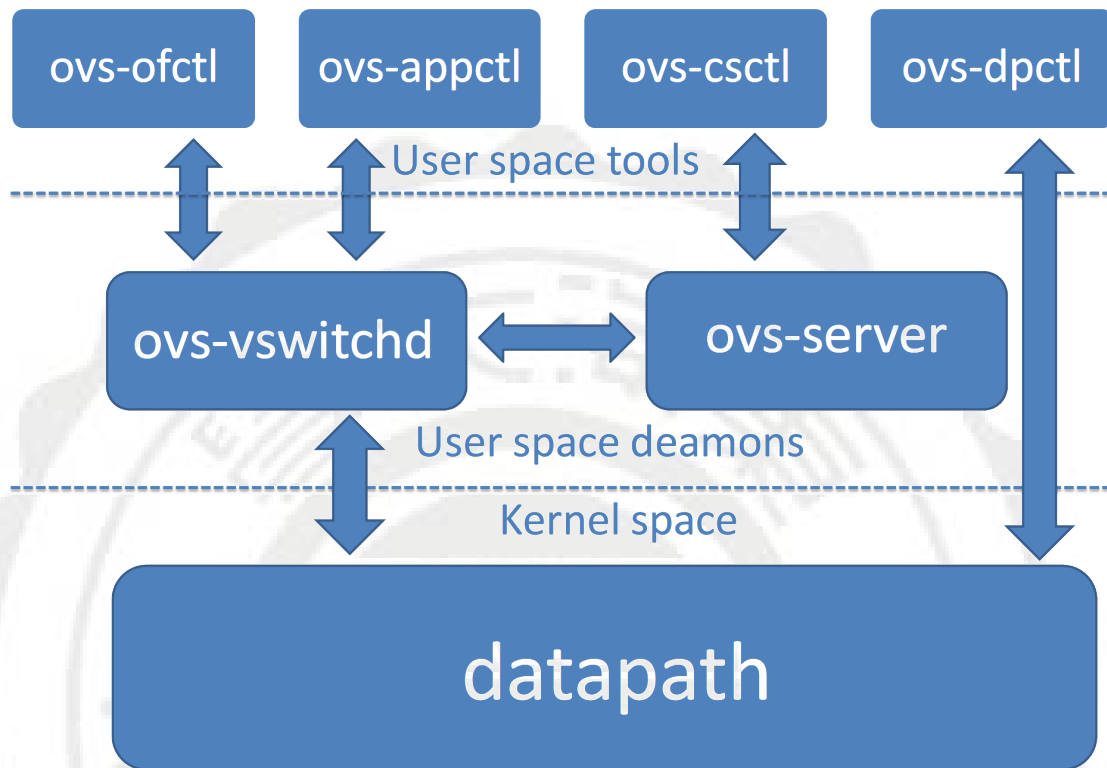


FIGURE 3.9: OpenvSwitch tool relation

### 3.1.4 Virtual Machine Configuration

We running some VM at physical machine to simulate actions that normal user usually do, like browser websites, using skype to talk with friend, download lots of small or a large files, etc. We want to close the user's real situation and promote the user experience.

This is out VM configure file using libvirt XML format.

```
<domain type='qemu'>
  <name>QEmu-fedora-i686</name>
  <uuid>c7a5fddb-cdaf-9455-926a-d65c16db1809</uuid>
  <memory>219200</memory>
  <currentMemory>219200</currentMemory>
  <vcpu>2</vcpu>
  <os>
    <type arch='i686' machine='pc'>hvm</type>
    <boot dev='cdrom' />
  </os>
  <devices>
    <emulator>/usr/bin/qemu-system-x86_64</emulator>
    <disk type='file' device='cdrom'>
```

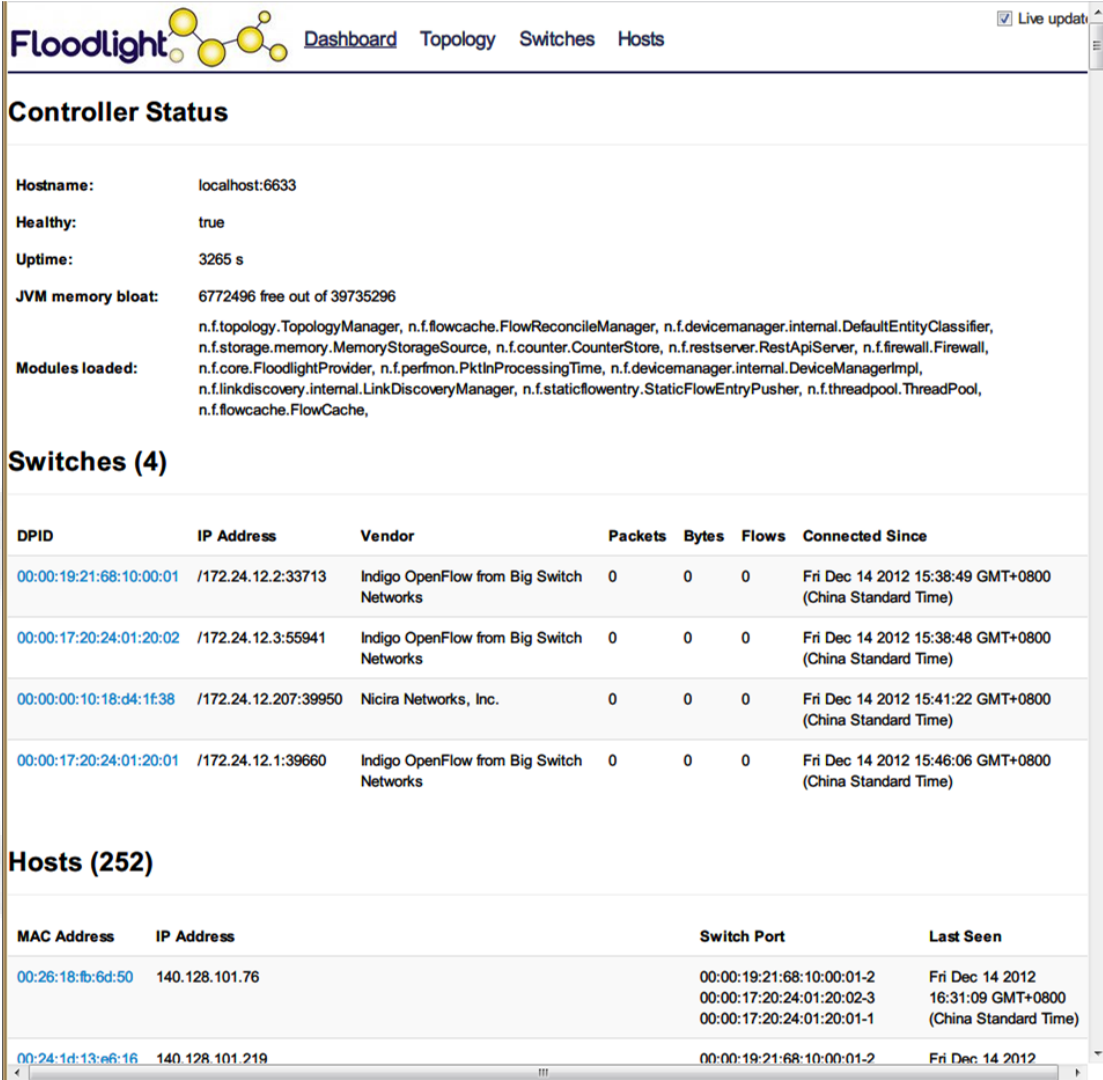
```
<source file='/home/user/boot.iso'/>
<target dev='hdc'/>
<readonly/>
</disk>
<disk type='file' device='disk'>
<source file='/home/user/fedora.img'/>
<target dev='hda'/>
</disk>
<interface type='network'>
<source network='default'/>
</interface>
<graphics type='vnc' port='-1'/>
</devices>
</domain>
```

---

VM SETTING 3.1: QEMU x86\_64 VM XML using libvirt

## 3.2 System Setup

After entire environment configured, we can use the browser to see several pages, like Floodlight OpenFlow controller ( Figure 3.10 ) and Indigo Open source OpenFlow firmware's web page ( Figure 3.12 ). At floodlight web page, we can see some tab at the top, Switch tab is showing how much OpenFlow switches are connect to this controller, and click switch path id, you can see more detail information of this switch ( Figure 3.11 ), like how much ports, link status of each port, transmit and receive packets and bytes. Host tab is showing all host ever connect to switches, even it just ARP request, floodlight controller will record it and show it to host tab. Topology tab is using Scalable Vector Graphics(SVG) to draw whole network topology to graph, it will connect host and switches, connect with line to showing which switch can reach which host. Dashboard tab is combine switch tab and host tab, direct showing these two tab at same page, let user has the system overview.



The screenshot displays the Floodlight OpenFlow Controller dashboard. At the top, there is a navigation bar with the Floodlight logo and links for Dashboard, Topology, Switches, and Hosts. A 'Live update' checkbox is visible in the top right corner.

### Controller Status

**Hostname:** localhost:6633  
**Healthy:** true  
**Uptime:** 3265 s  
**JVM memory bloat:** 6772496 free out of 39735296  
**Modules loaded:** n.f.topology.TopologyManager, n.f.flowcache.FlowReconcileManager, n.f.devicemanager.internal.DefaultEntityClassifier, n.f.storage.memory.MemoryStorageSource, n.f.counter.CounterStore, n.f.restserver.RestApiServer, n.f.firewall.Firewall, n.f.core.FloodlightProvider, n.f.perfmon.PktInProcessingTime, n.f.devicemanager.internal.DeviceManagerImpl, n.f.linkdiscovery.internal.LinkDiscoveryManager, n.f.staticflowentry.StaticFlowEntryPusher, n.f.threadpool.ThreadPool, n.f.flowcache.FlowCache

### Switches (4)

DPID	IP Address	Vendor	Packets	Bytes	Flows	Connected Since
00:00:19:21:68:10:00:01	/172.24.12.2:33713	Indigo OpenFlow from Big Switch Networks	0	0	0	Fri Dec 14 2012 15:38:49 GMT+0800 (China Standard Time)
00:00:17:20:24:01:20:02	/172.24.12.3:55941	Indigo OpenFlow from Big Switch Networks	0	0	0	Fri Dec 14 2012 15:38:48 GMT+0800 (China Standard Time)
00:00:00:10:18:d4:1f:38	/172.24.12.207:39950	Nicira Networks, Inc.	0	0	0	Fri Dec 14 2012 15:41:22 GMT+0800 (China Standard Time)
00:00:17:20:24:01:20:01	/172.24.12.1:39660	Indigo OpenFlow from Big Switch Networks	0	0	0	Fri Dec 14 2012 15:46:06 GMT+0800 (China Standard Time)

### Hosts (252)

MAC Address	IP Address	Switch Port	Last Seen
00:26:18:fb:6d:50	140.128.101.76	00:00:19:21:68:10:00:01-2 00:00:17:20:24:01:20:02-3 00:00:17:20:24:01:20:01-1	Fri Dec 14 2012 16:31:09 GMT+0800 (China Standard Time)
00:24:1d:13:e6:16	140.128.101.219	00:00:19:21:68:10:00:01-2	Fri Dec 14 2012

FIGURE 3.10: Dashboard of Floodlight OpenFlow Controller

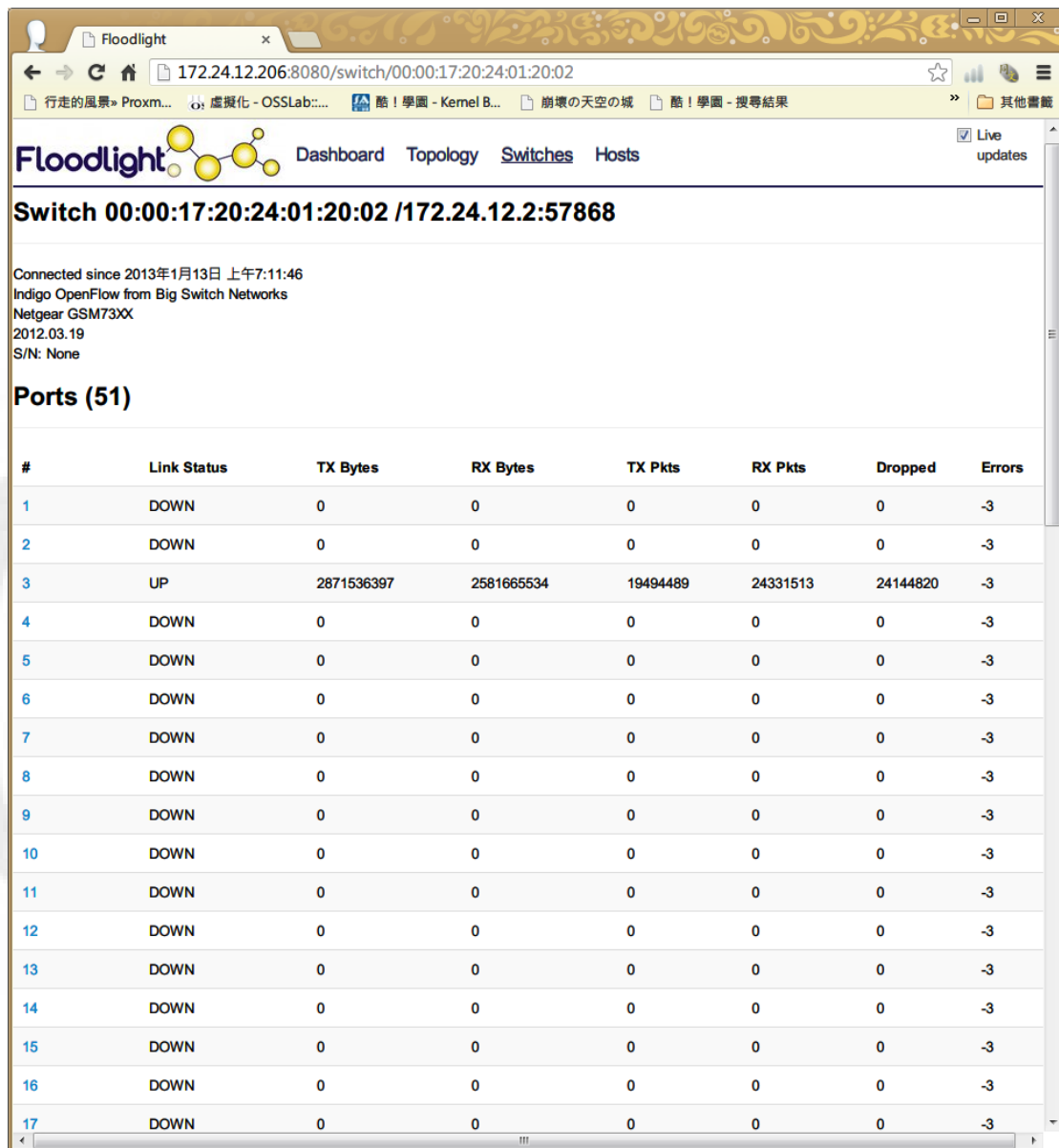


FIGURE 3.11: Check Switched status

Indigo open source OpenFlow firmware web page ( Figure 3.12 ) also provide user to view and modify some variable, like setting MAC address, IP address, and specify OpenFlow controller IP and port. Go to the monitor tab and click flow table option, we can see how much flow are setting to this switch or click detail can see full flow rule. Figure 3.13

The screenshot displays the Netgear web interface for the OpenFlow System Configuration of a Netgear GSM7352S2 switch. The interface includes a top navigation bar with 'System', 'Monitoring', 'Maintenance', and 'Help' tabs. The 'System' tab is active, and the 'Status' section is expanded to show 'OpenFlow System Configuration'. The main configuration area is titled 'OpenFlow System Configuration' and contains the following settings:

- Switch MAC (running config):** E0:46:9A:4E:A0:61
- Override Preprogrammed MAC:**  Use preprogrammed MAC,  Override preprogrammed MAC
- MAC address for override:** [Empty text box]
- Switch IP address (running config):** 172.24.12.1 /255.255.255.0
- Using DHCP for switch IP:**  Disable DHCP,  Enable DHCP,  Only use DHCP
- Configured IP address for this switch:** [Empty text box]
- Netmask for this IP address:** 255.255.255.0
- Configured Gateway IP address:** [Empty text box]
- Current Gateway (running config):** [Empty text box]
- OpenFlow Controller IP address:** 172.24.12.206
- OpenFlow Controller TCP port:** 6633
- ofprotocol options (advanced):** [Empty text box]
- Datapath ID of this switch:** 172024012001
- System Ref:** hpc\_OF-1
- Host Name:** [Empty text box]
- Management Mode:**  Out-of-band,  In-band
- Datapath Management Port:**  Fixed port (Port: [Empty text box]),  Any port
- Fall open/close:**  open,  closed
- Log level:** none

FIGURE 3.12: Web page of Indigo OpenFlow firmware for Netgear GSM7352S2



The screenshot shows the Netgear OpenFlow Flow Table interface. The browser address bar displays `172.24.12.1/cgi-bin/flowtable?displey=full`. The page title is "NETGEAR" and the firmware version is "2012.03.19-iods: User mode. GSM73XX". The navigation menu includes "System", "Monitoring", "Maintenance", and "Help". The "Monitoring" section is active, and the "Flow Table" tab is selected.

The "OpenFlow Flow Table" section has two radio buttons: "Normal Display" (selected) and "Detailed Display". A "Refresh Table" button is also present. A note indicates "Note: TO=Time Out".

Entry	Entry	Entry
<p>Entry 1</p> <pre> actions          : (table)   1              : (table)     action_name   : output     len           : 8     max_len       : 65535     port          : 14     type          : 0 byte_count       : 0 cookie           : 4503599627370496 created          : 615487795 dl_dst           : 00:10:18:d4:16:91 dl_src           : 00:10:18:d4:1f:39 dl_type          : 2048 dl_vlan          : 65535 dl_vlan_pcp     : 0 hard_timeout     : 0 idle_timeout     : 5 in_port          : 12 nw_dst           : 0.0.0.0 nw_dst_mask      : 0.0.0.0 nw_proto         : 0 nw_src           : 0.0.0.0 nw_src_mask      : 0.0.0.0 nw_tos           : 0 out_port         : 65535 packet_count     : 153625 priority         : 100 tp_dst           : 0 tp_src           : 0 used             : 615530004 wildcards        : 4194288 </pre>	<p>Entry 2</p> <pre> actions          : (table)   1              : (table)     action_name   : output     len           : 8     max_len       : 65535     port          : 12     type          : 0 byte_count       : 0 cookie           : 4503599627370496 created          : 615487795 dl_dst           : 00:10:18:d4:1f:39 dl_src           : 00:10:18:d4:16:91 dl_type          : 2048 dl_vlan          : 65535 dl_vlan_pcp     : 0 hard_timeout     : 0 idle_timeout     : 5 in_port          : 14 nw_dst           : 0.0.0.0 nw_dst_mask      : 0.0.0.0 nw_proto         : 0 nw_src           : 0.0.0.0 nw_src_mask      : 0.0.0.0 nw_tos           : 0 out_port         : 65535 packet_count     : 153625 priority         : 100 tp_dst           : 0 tp_src           : 0 used             : 615530004 wildcards        : 4194288 </pre>	

Copyright © 1996-2011 Netgear ®

FIGURE 3.13: Indigo OpenFlow firmware to view flow table at switch

## Chapter 4

# Experimental Results

### 4.1 Experimental Environment

This work is using OpenvSwitch as access layer switch and three Netgear GSM7352 as disturbed layer switch, we using sFlow to collect all packet at our network.

Model	DELL OptiPlex 745
CPU	Intel Core 2 6400 2.13Ghz
Memory	DDR2 667MHz 512MB x 2
Disk	160GB
Hypervisor	KVM 1:84+dfsg-0ubuntu16+1.0+noroms+0ubuntu14.3
Virtual Switch	OpenvSwitch 1.4.0-1Ubuntu1.3
Linux	Ubuntu 12.04 amd64 server edition
Kernel	3.2.0-23-generic
Hardware Switch	Netgear GSM7352S2 x 3

TABLE 4.1: Hardware specification

Table 4.1 is our experimental environment, we have four Dell OptiPlex 745, one for controller, three for VM host and OpenvSwitch, each Dell OptiPlex 745 has Intel Core 2 6400, 1GB of RAM, 160GB of hard disk, and all of Dell OptiPlex 745 install Ubuntu 12.04 amd64 server version as Operating System, hypervisor is KVM, virtual switch is OpenvSwitch. OpenvSwitch built in sFlow, so we just setting up the environment to get flow data. In this work, we use pmacct to collect data from sFlow agent, and we write a shell script to show how many host in network now, let network administrator can set OpenFlow to OpenFlow switch (OpenvSwitch and GSM7352S2).

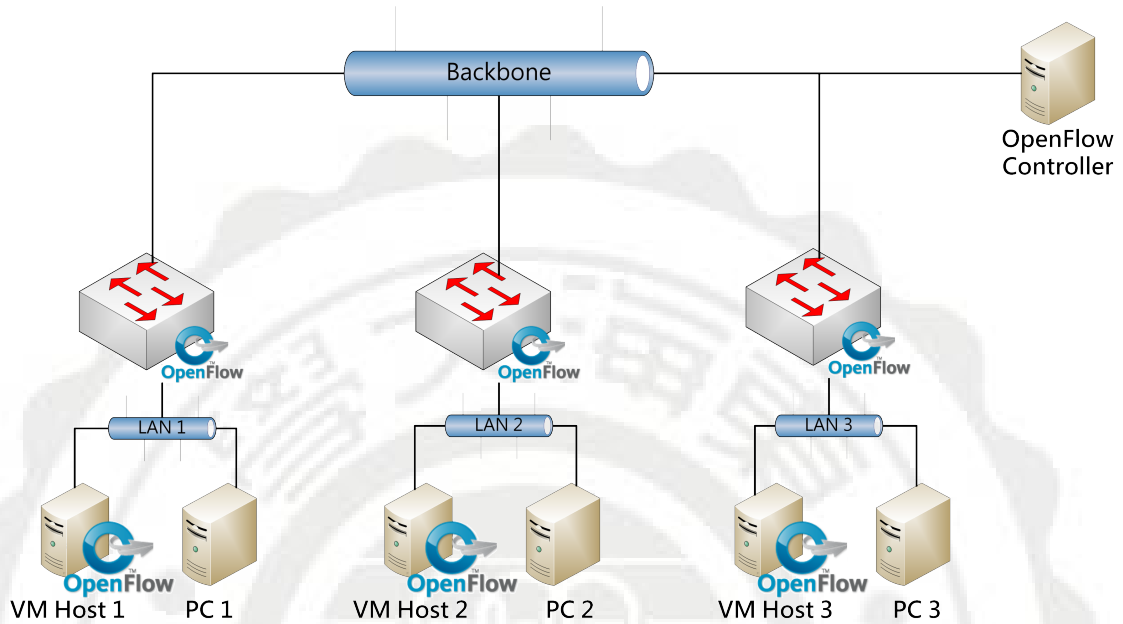


FIGURE 4.1: Experimental Environment

We depict our experimental environment at figure 4.1, it shown three LAN and one backbone with a OpenFlow controller. Each LAN has a VM host that install OpenvSwitch to support OpenFlow, to simulate a cloud computing environment using VM, and a personal computer just install normal Operating System like Windows XP, Linux, Mac OS, etc., simulate a network without OpenFlow built in.

We using the flag at OpenFlow packet to control the function we need to use then reach out target. Before set flag, we need to using match function to match which flow we want to set flag. Match field showing at table 4.2.

We design a WEB control interface (figure 4.2) to control our switch, by set the flag of priority in OpenFlow packet, reach the QoS target. The page also user friendly, it can hide MAC without IP like figure 4.3. This function can reduce complexity to user, let user can operate out system more easy. When user want to set priority to network, just click which IP or MAC, the text box will appear at left, 32768 (lowest priority) is default value when network flow set to switch, then click the text box and enter a number (small number has higher priority)( figure 4.4), then user click send to set priority to switch (figure 4.5), there will a animation from flow to switch id at left. The result will showing at next section.

Ingress port
Metadata
Ethernet Source
Ethernet Destination
Ethernet Type
VLAN id
Priority
MPLS Label
MPLS Traffic class
IPv4 Source
IPv4 Destination
IPv4 protocol or ARP opcode
IPv4 ToS bits
TCP / UDP / SCTP Source Port or ICMP Type
TCP / UDP / SCTP Destination Port or ICMP Code

TABLE 4.2: OpenFlow Match Field

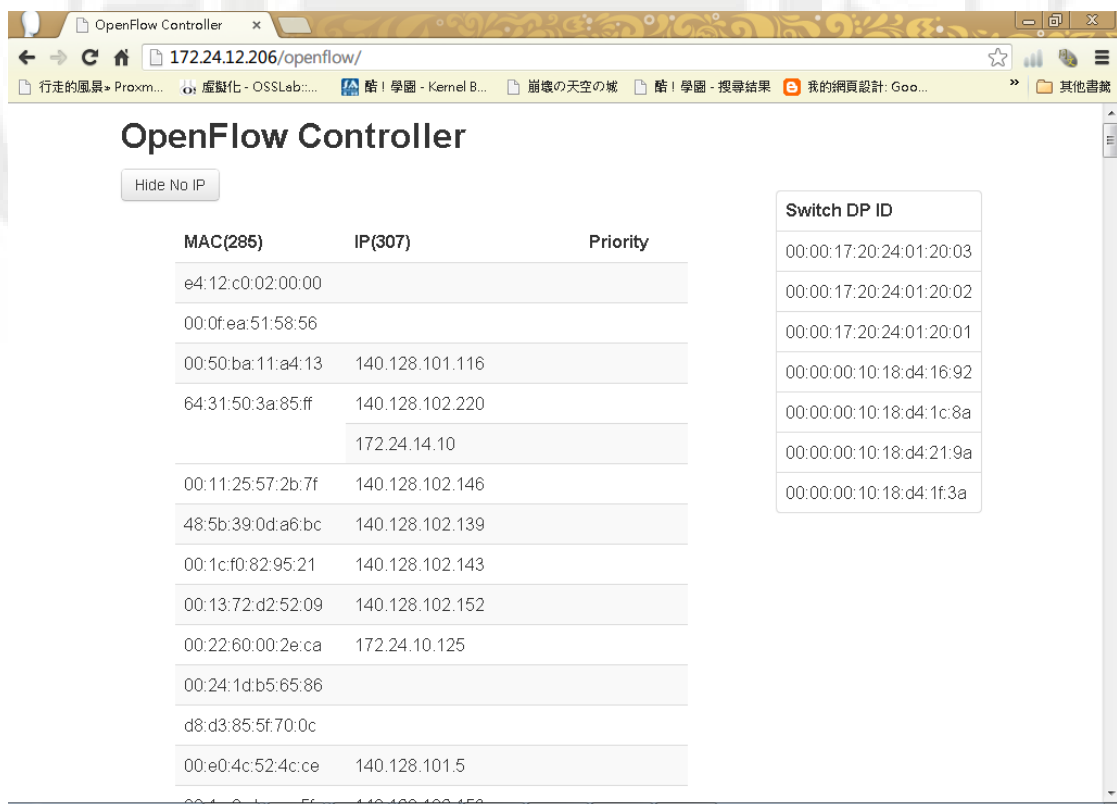


FIGURE 4.2: Web control interface

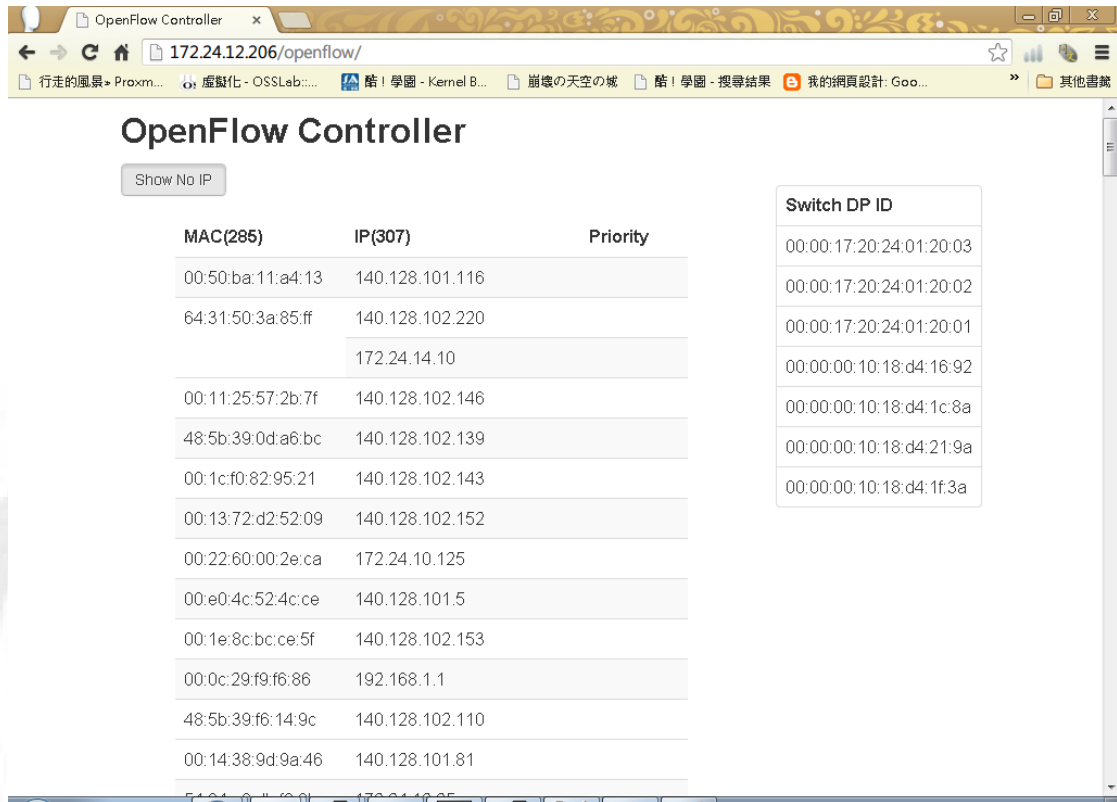


FIGURE 4.3: Web control, hide MAC without IP

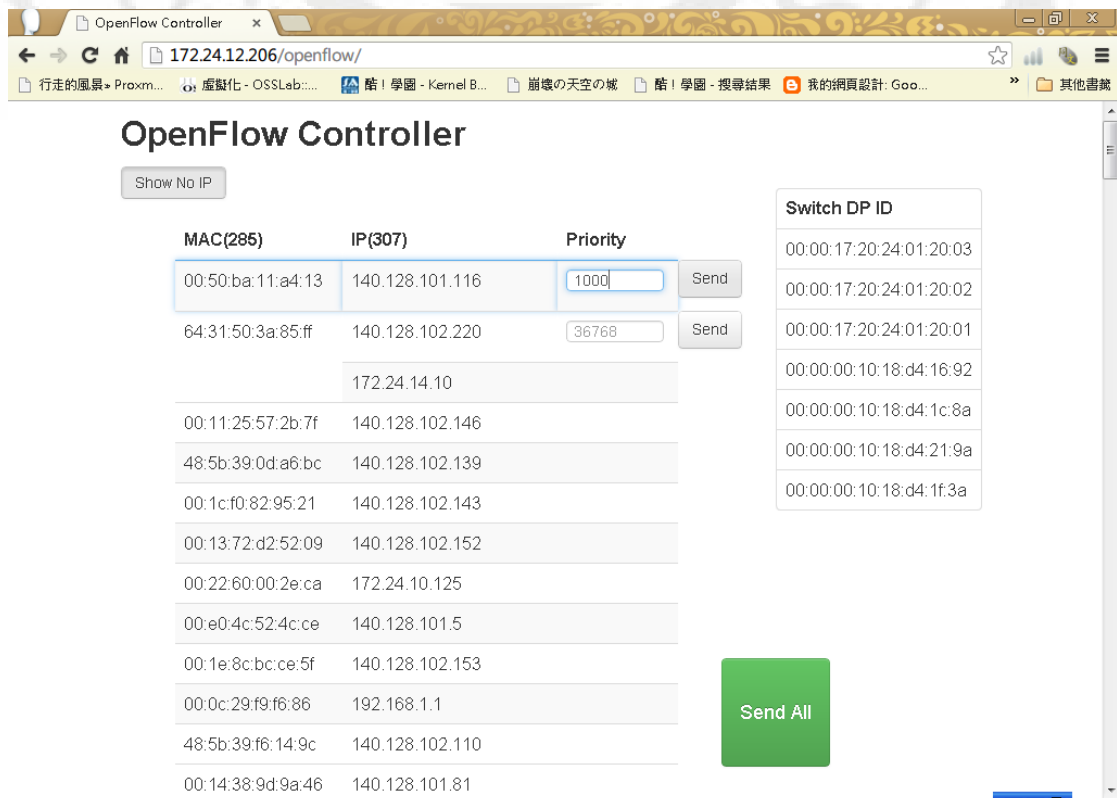


FIGURE 4.4: Web control, set priority

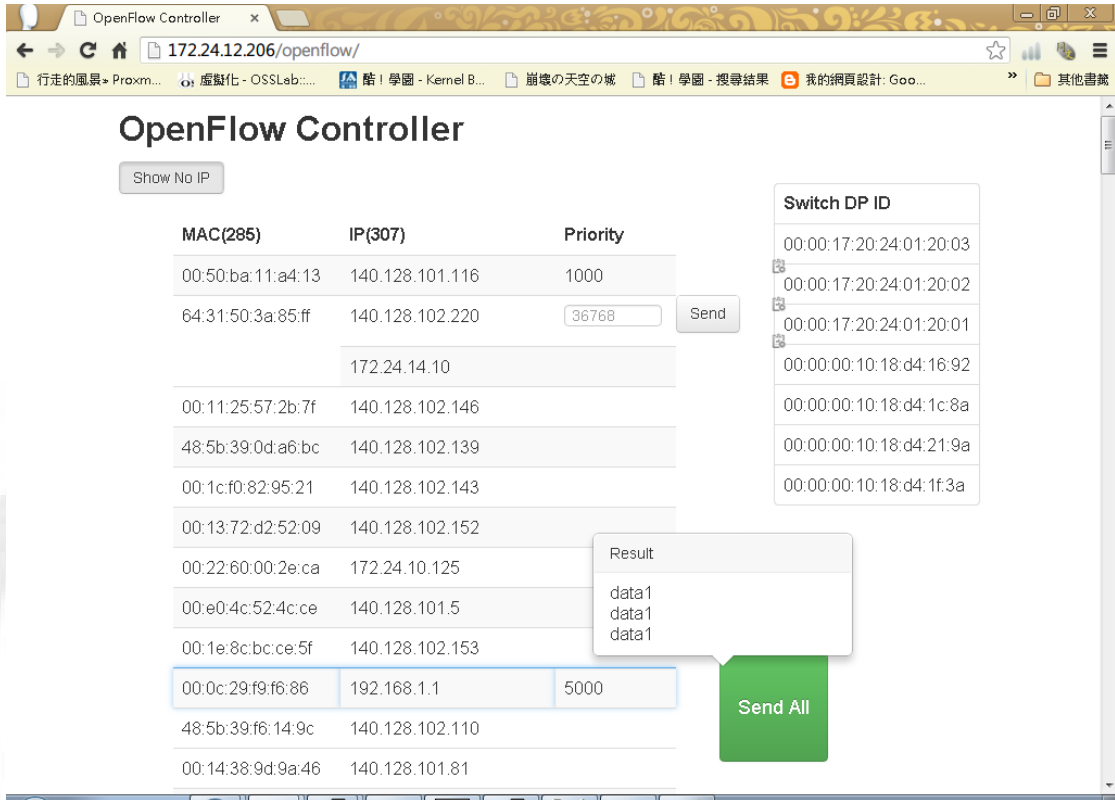


FIGURE 4.5: Web control, set flow to switch

## 4.2 Experimental Results and Discussion

First, we using iperf, its a tool to create TCP and UDP data stream and measure throughput of network, it has a parameter named stdin, let user can specify the packet content. In this work, we use parameter stdin to fix the packet size, and measure the performance between different packet size, the result is shown at table 4.3 and Figure 4.6

Packets sizes (bytes)	64	96	128	256	512	1024	1500
normal bridge	389	618	645	815	901	930	952
OpenFlow	412	630	648	820	904	933	959
switch	268	420	589	813	902	935	955

TABLE 4.3: Using iperf with different probe method

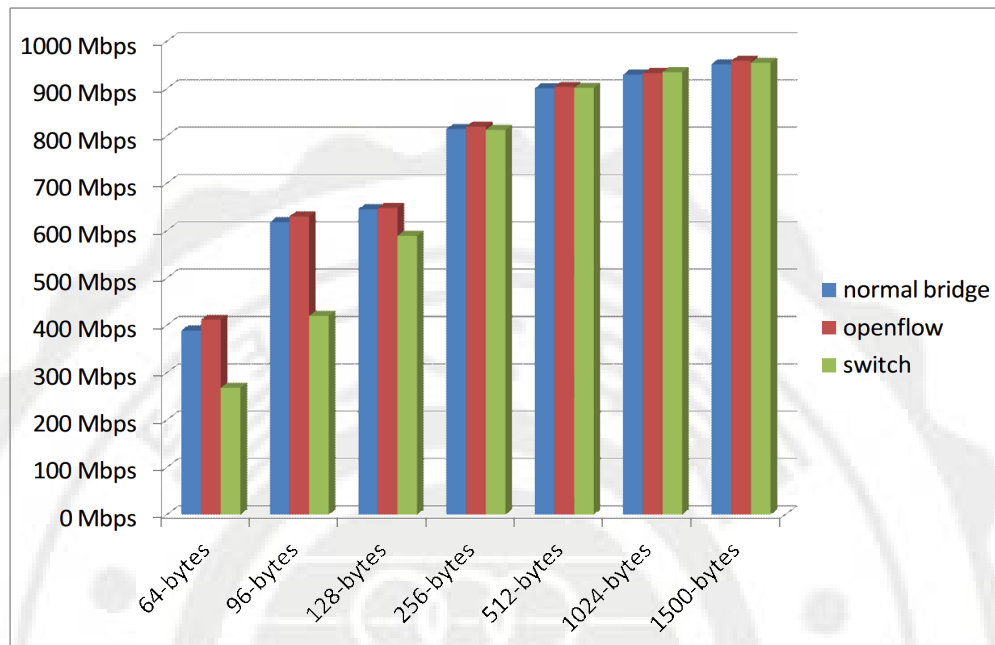


FIGURE 4.6: Experimental Result

We can see the green line of switch, is lower than other at 64-bytes to 128-bytes. But after 256-bytes, the three methods just have a little amount of difference. Guess its because the three methods process their header, and the amount of packet. More packets the protocol needs to process more header, it needs to split the packet to view where the packet from and where it should go, the protocol design pros and cons is shown at here.

After experimental with different protocol probe, next experimental focus on create QoS policies to limit the bandwidth from large amount network flow, from figure 4.7 we can see, there has 9 hosts at this experimental, prefix IP with 10.0.x.x are our experimental network with virtual machine.

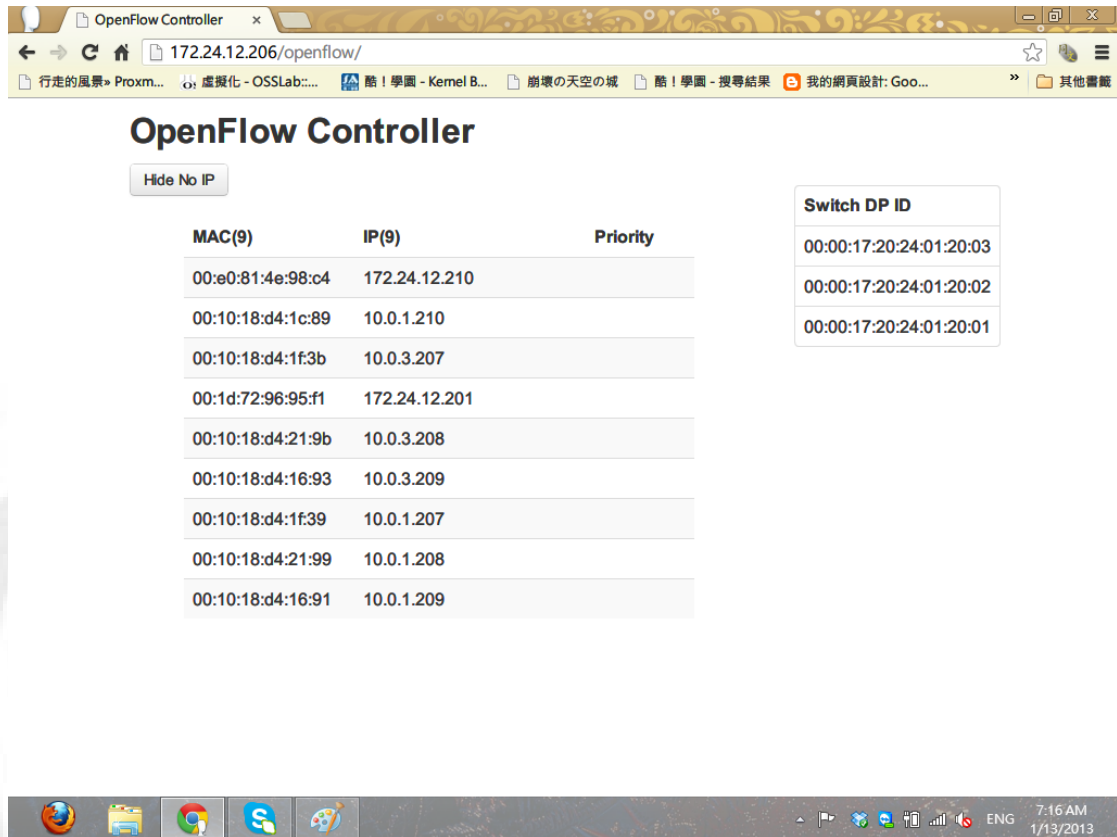


FIGURE 4.7: Web Setting 1

At figure 4.8 we setting host 1 with priority 19999, that will limit bandwidth to 200Mbps.



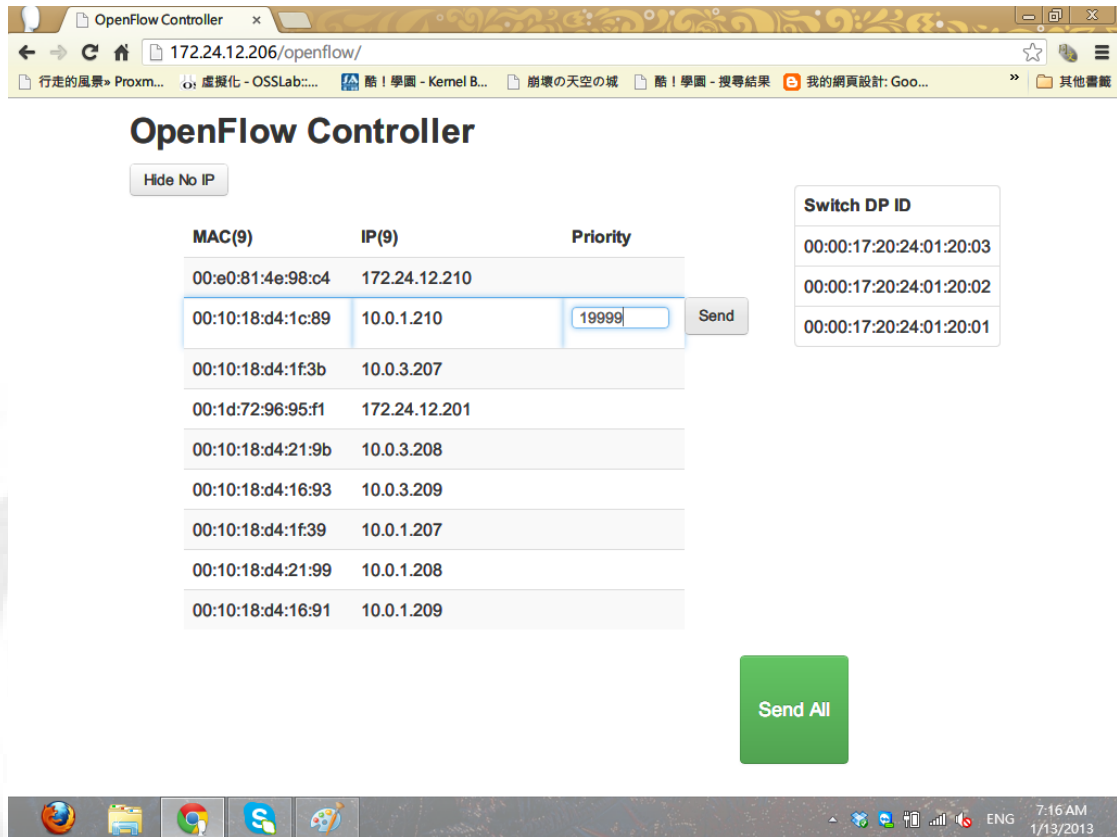


FIGURE 4.8: Web Setting Host 1 Priority to 19999

At figure 4.9 we setting host 1 with priority 5000, that will increase limit bandwidth to 550Mbps.

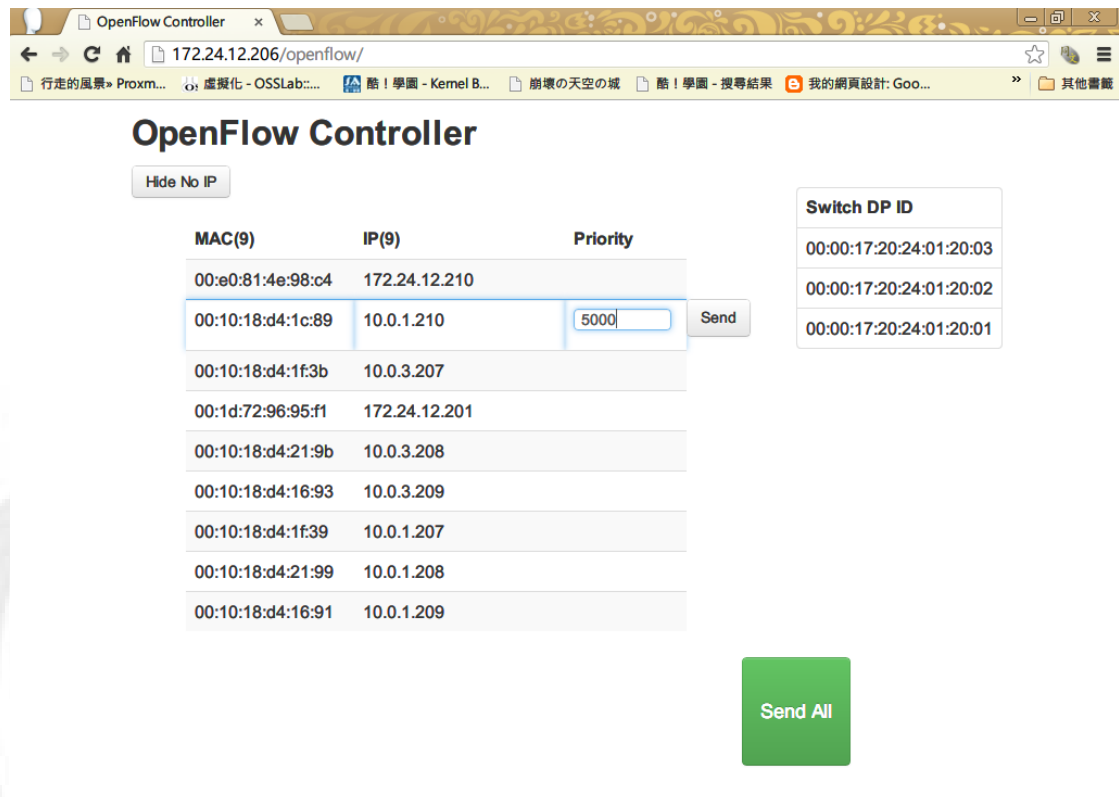


FIGURE 4.9: Web Setting Host 1 Priority to 5000

Below two figure 4.10 and figure 4.11 also setting priority to 5000 and 19999, but set to different host.

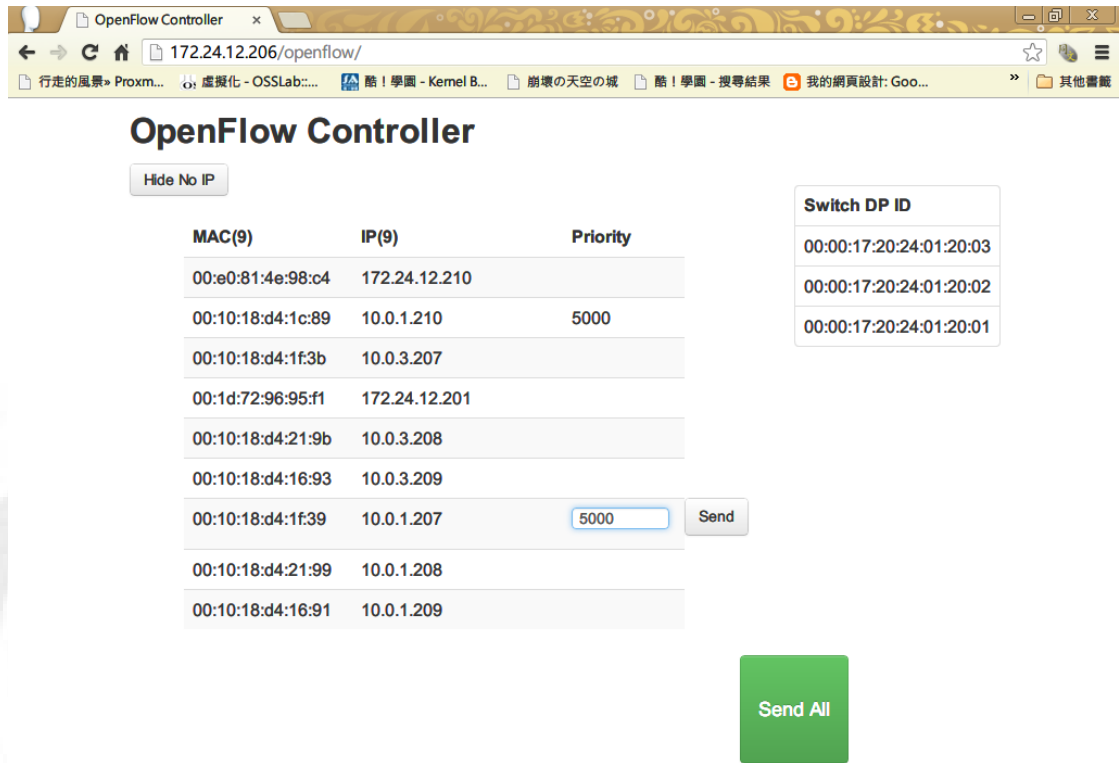


FIGURE 4.10: Web Setting Host 1 Priority to 5000

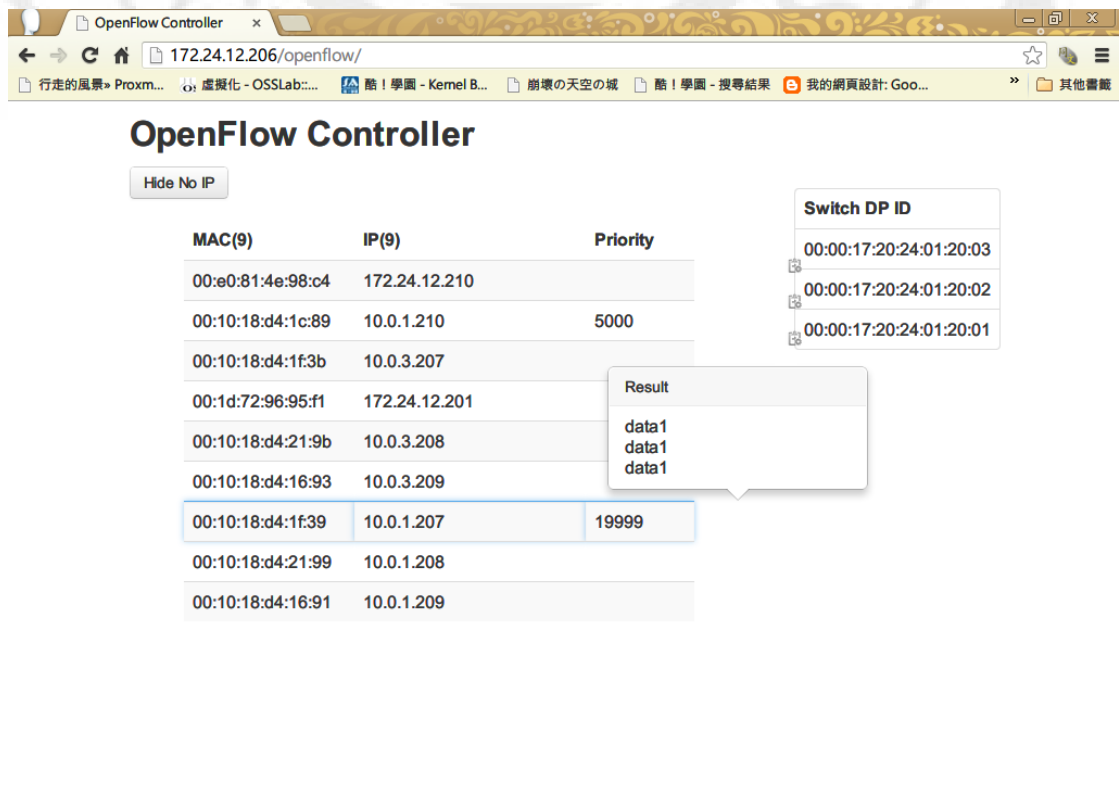


FIGURE 4.11: Web Setting Host 2 Priority to 19999

After setting priority, let's check the real effect to network traffic. Figure 4.12 and Figure 4.13 show the effect. Host 1 setting priority 19999 than 5000, so the speed from 1Gbps decrease to 200Mbps cause priority set to 19999, than increase to 550Mbps cause priority set to 5000. Host 2 inverse this process, network speed from 1Gbps decrease to 200Mbps than increase to 550Mbps.

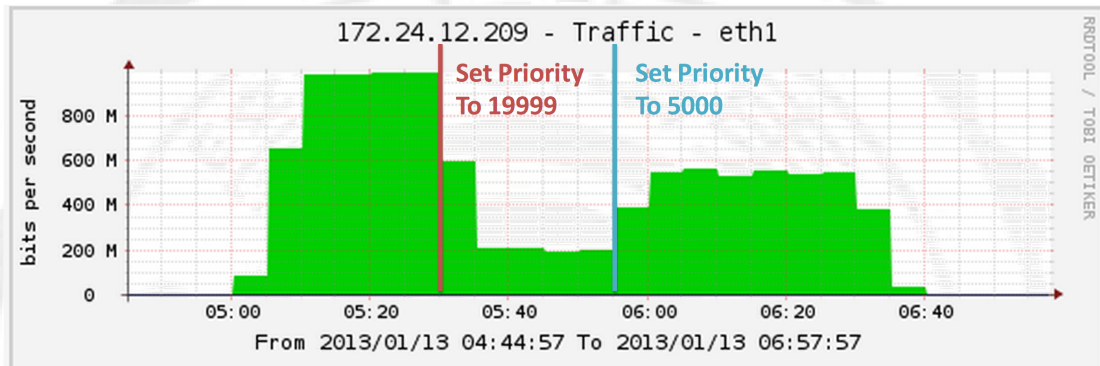


FIGURE 4.12: QoS Result 1

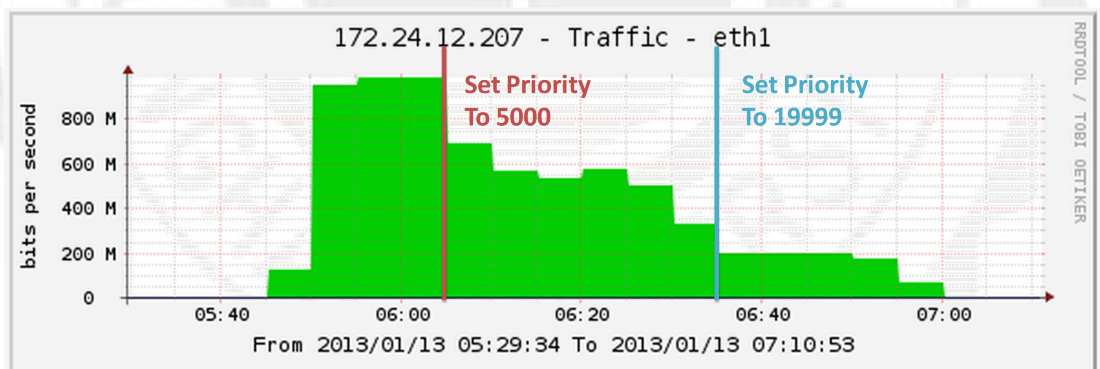


FIGURE 4.13: QoS Result 2

After QoS experimental, we trying to use QoS setting to a scenario we setting. Think a scenario that a user downloading a large file from HTTP protocol, he hold the most of network traffic, nearly 900M bps, but if now has another user want to use FTP to download some file to install machine, it will keep in low speed and long time. With OpenFlow switch, it can match packet with different port, we can realize it to protocol, cause different protocol usually use different port, and it usually fixed. When switch match packet with defined port, it will set the packet in a queue, we set three queue, default queue is set to full speed ( 1000M bps ),FTP queue set to 550M bps and HTTP queue set to 100Mbps. Our experimental has two step, first step, We use default queue to all packet, then start HTTP protocol, let it download with full speed, then start

ftp download after 5 minutes, and monitor network speed, full experimental time is 20 minutes . Second step, we process experimental like first step for 10 minuses, after 10 minutes we apply the QoS policy, to check the effect with QoS policy.

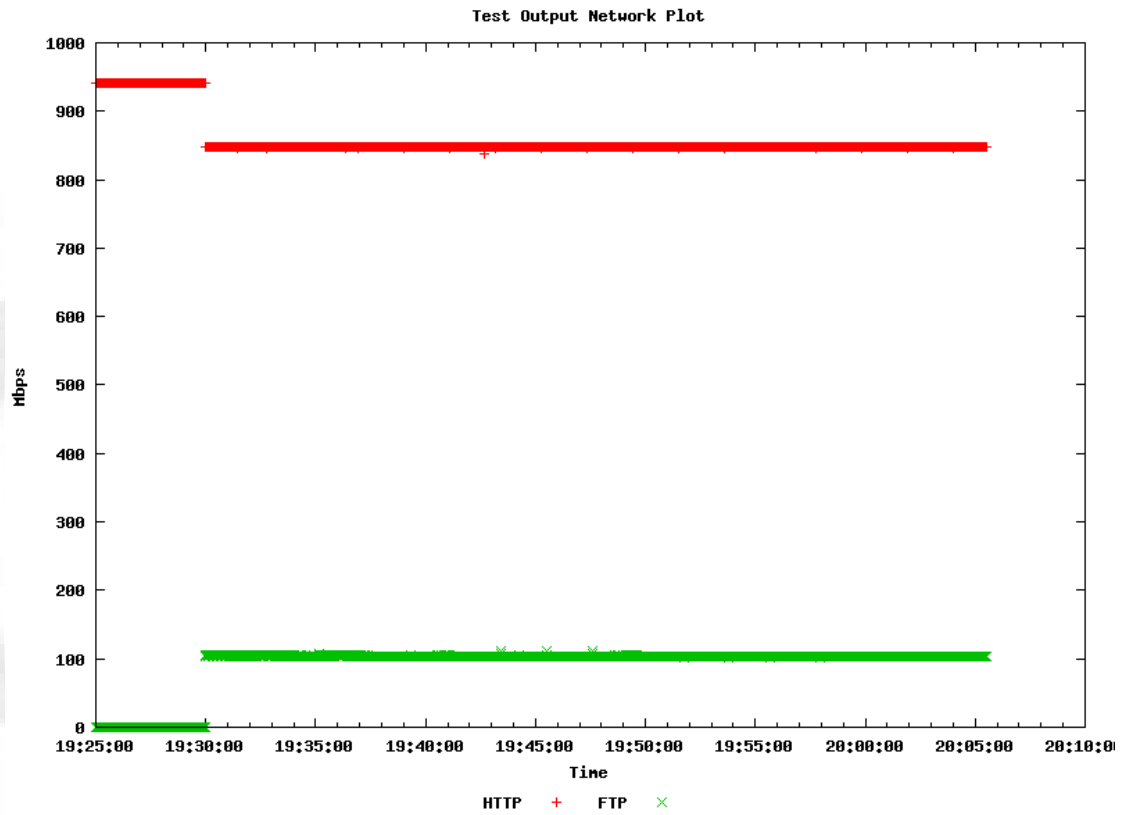


FIGURE 4.14: Network traffic without QoS policy

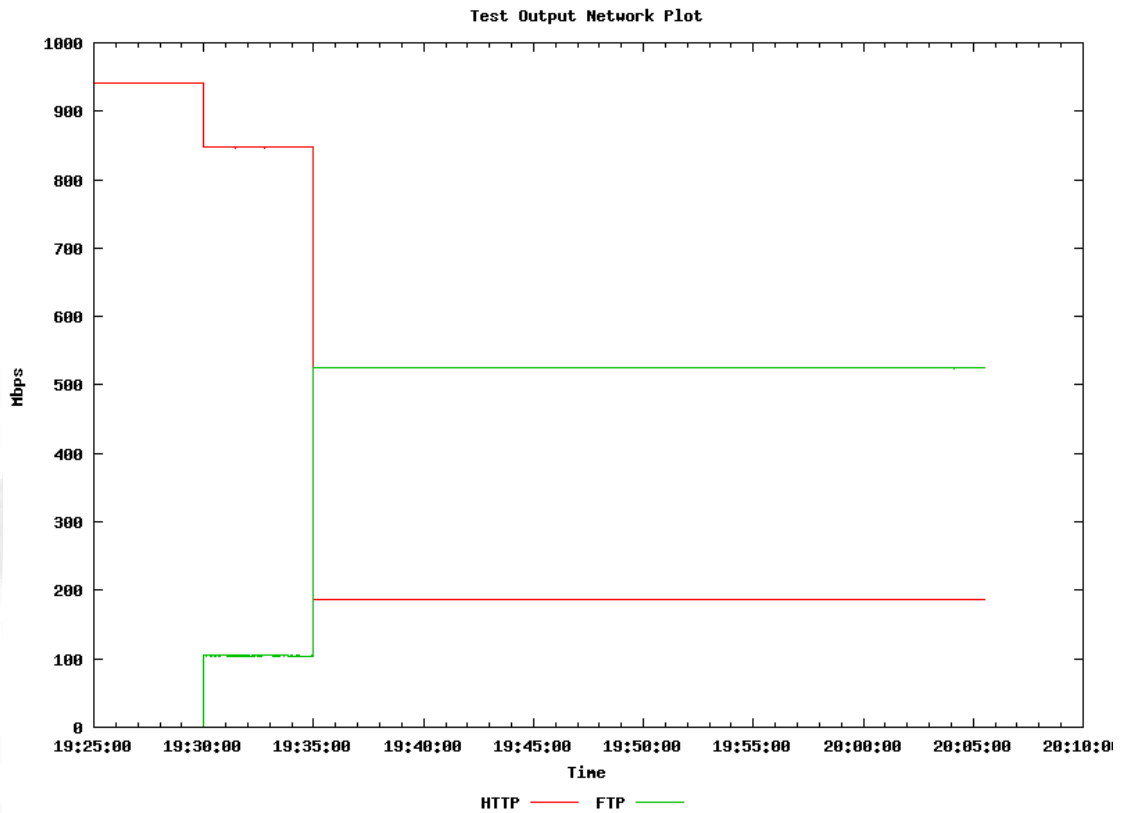


FIGURE 4.15: Network with QoS policy after 10 minutes

We can see the traditional network like Figure 4.14, The FTP speed is always lower than HTTP, about 100M bps, but the HTTP traffic is much higher, is 830M bps. But when we enable the QoS policy at Figure 4.15, we can see the traffic, after 10 minutes, the QoS policy be applied to network, HTTP traffic is decrease to 200M bps and the FTP traffic increase to 550M bps.

As a monitor system, our system provide warning function, network administrator set a upper bound for each port, when port traffic reach the upper bound, our system will markup which port and all host under it.(Figure 4.16) System also send a warning message through email, notice network administrator to check the network. (Figure 4.17)

## OpenFlow Controller

**Switch DP ID**  
 00:00:17:20:24:01:20:03  
 00:00:17:20:24:01:20:02  
 00:00:17:20:24:01:20:01

Upper Bound

MAC(214)	IP(239)	Priority
52:54:00:39:65:5d	140.128.102.195	
48:5b:39:f6:16:78	140.128.102.113	
48:5b:39:0d:a6:42	140.128.102.142	
b8:27:eb:06:53:a3	140.128.101.64	<input style="width: 50px;" type="text" value="36768"/> <input type="button" value="Send"/>
00:0c:6e:a5:43:fd	140.128.101.23	
00:25:90:58:9e:30		
00:0e:0c:5a:f4:a0	140.128.98.41	
f4:6d:04:9e:2c:d1	140.128.101.35	
00:25:90:4b:2f:b3	140.128.98.44	
52:54:00:1e:2e:3e	140.128.101.11	
00:00:48:d3:1a:37		
14:d6:4d:0d:3e:38	140.128.102.201	
30:85:a9:3c:aa:61	140.128.102.82	
52:54:00:12:34:60	140.128.101.187	
00:18:f3:a4:84:3b		
6c:62:6d:46:68:4a	140.128.102.140	

FIGURE 4.16: Warning system and admin email setting

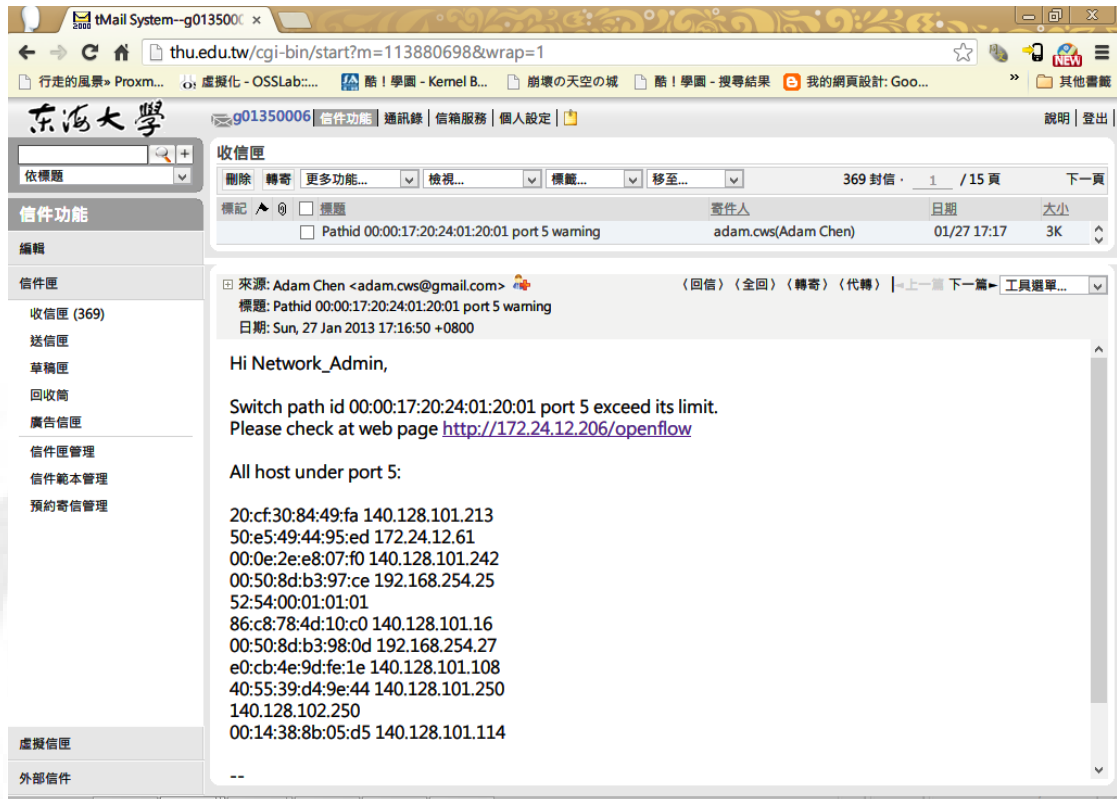


FIGURE 4.17: Warning system send email to network administrator



## Chapter 5

# Conclusions and Future work

In this thesis, a Virtual Switch Monitor System Using OpenFlow on cloud computing environments, first we measure the speed of normal bridge, OpenFlow and normal switch, we found that speed is lower before packet size 128-bytes, but after 256-bytes these three methods speed just little amount of difference. Believe its because header, more packet more process time to deal to packet flow to which port. After 256-bytes the speed little amount different because just few packet header need to process, and the OpenFlow is design to line-rate and depart controller layer to remote, let switch focus on processing data flow, Performance is not showing at this time because we just use same ip and mac doing test. If using it at complex environment with much more different IP and MAC, it should showing its power.

After compare different protocol, we trying use OpenFlow's feature, set its flag to do QoS, at traditional network, firewall and QoS always be put at backbone, but backbone always have large amount network traffic, if we want to process these network traffic as firewall or QoS, the hardware of firewall or QoS device need to be very powerful, also mean spent more money. But with OpenFlow-enabled switch, we can reduce and process network flow at frontend, where network traffic be generated, using OpenFlow to separate data plane and control plane, we can control a single policy, and act OpenFlow-enabled switch like firewall or QoS device, spend less money.

We develop a web interface to control the entire environment like OpenFlow controller, OpenFlow-enabled switch and user interface. By the experimental result, we success control the network traffic, reduce the network utilization from source.

Compare with traditional network, OpenFlow showing itself will not decrease the speed, IT maybe can trying move the QoS or firewall service from backbone to the end switch to decrease the pressure of device or server. At future, we will continued development this system, let user can add QoS or firewall policy at same page, and simplify the process of using.



# Bibliography

- [1] K. Adams and O. Agesen. A comparison of software and hardware techniques for x86 virtualization. In *Proceedings of the 12th international conference on Architectural support for programming languages and operating systems, ASPLOS-XII*, pages 2–13, New York, NY, USA, 2006. ACM.
- [2] G. Antichi, A. Di Pietro, S. Giordano, G. Procissi, and D. Ficara. Design and development of an openflow compliant smart gigabit switch. In *Global Telecommunications Conference (GLOBECOM 2011), 2011 IEEE*, pages 1–5, dec. 2011.
- [3] P. Barham, B. Dragovic, K. Fraser, S. Hand, T. Harris, A. Ho, R. Neugebauer, I. Pratt, and A. Warfield. Xen and the art of virtualization. In *Proceedings of the nineteenth ACM symposium on Operating systems principles, SOSP '03*, pages 164–177, New York, NY, USA, 2003. ACM.
- [4] F. Bellard. Qemu, a fast and portable dynamic translator. *USENIX 2005 Annual Technical Conference*, FREENIX Track:41, 2005.
- [5] A. Bianco, R. Birke, L. Giraudo, and M. Palacin. Openflow switching: Data plane performance. In *Communications (ICC), 2010 IEEE International Conference on*, pages 1–5, may 2010.
- [6] Z. Bozakov and V. Sander. Openflow: A perspective for building versatile networks. In A. Clemm and R. Wolter, editors, *Network-Embedded Management and Applications*, pages 217–245. Springer New York, 2013.
- [7] P. Du, M. Chen, and A. Nakao. Port-space isolation for multiplexing a single ip address through open vswitch. In T. Magedanz, A. Gavras, N. Thanh, and J. Chase, editors, *Testbeds and Research Infrastructures. Development of Networks and Communities*, volume 46 of *Lecture Notes of the Institute for Computer Sciences, Social*

- Informatics and Telecommunications Engineering*, pages 113–122. Springer Berlin Heidelberg, 2011.
- [8] H. Egilmez, B. Gorkemli, A. Tekalp, and S. Civanlar. Scalable video streaming over openflow networks: An optimization framework for qos routing. In *Image Processing (ICIP), 2011 18th IEEE International Conference on*, pages 2241–2244, sept. 2011.
- [9] W. Emenecker and D. Stanzione. Hpc cluster readiness of xen and user mode linux. In *Cluster Computing, 2006 IEEE International Conference on*, pages 1–8, sept. 2006.
- [10] F. Farias, J. Salvatti, E. Cerqueira, and A. Abelem. A proposal management of the legacy network environment using openflow control plane. In *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 1143–1150, april 2012.
- [11] T. Feng, J. Bi, and H. Hu. Openrouter: Openflow extension and implementation based on a commercial router. In *Network Protocols (ICNP), 2011 19th IEEE International Conference on*, pages 141–142, oct. 2011.
- [12] O. Ferkouss, I. Snaiki, O. Mounaouar, H. Dahmouni, R. Ben Ali, Y. Lemieux, and C. Omar. A 100gig network processor platform for openflow. In *Network and Service Management (CNSM), 2011 7th International Conference on*, pages 1–4, oct. 2011.
- [13] N. Fernandes, M. Moreira, I. Moraes, L. Ferraz, R. Couto, H. Carvalho, M. Campista, L. Costa, and O. Duarte. Virtual networks: isolation, performance, and trends. *annals of telecommunications - annales des télécommunications*, 66:339–355, 2011.
- [14] A. Giorgetti, F. Cugini, F. Paolucci, and P. Castoldi. Openflow and pce architectures in wavelength switched optical networks. In *Optical Network Design and Modeling (ONDM), 2012 16th International Conference on*, pages 1–6, april 2012.
- [15] C. Huang, G. Zheng, L. Kalé, and S. Kumar. Performance evaluation of adaptive mpi. In *Proceedings of the eleventh ACM SIGPLAN symposium on Principles and practice of parallel programming*, PPOPP '06, pages 12–21, New York, NY, USA, 2006. ACM.

- [16] M. Jarschel, S. Oechsner, D. Schlosser, R. Pries, S. Goll, and P. Tran-Gia. Modeling and performance evaluation of an openflow architecture. In *Teletraffic Congress (ITC), 2011 23rd International*, pages 1 –7, sept. 2011.
- [17] H. Jin, D. Pan, J. Liu, and N. Pissinou. Openflow based flow level bandwidth provisioning for cicq switches. In *INFOCOM, 2011 Proceedings IEEE*, pages 476 – 480, april 2011.
- [18] Y. Kanaumi, S. Saito, and E. Kawai. Toward large-scale programmable networks: Lessons learned through the operation and management of a wide-area openflow-based network. In *Network and Service Management (CNSM), 2010 International Conference on*, pages 330 –333, oct. 2010.
- [19] Y. Kanaumi, S. Saito, E. Kawai, S. Ishii, K. Kobayashi, and S. Shimojo. Deployment and operation of wide-area hybrid openflow networks. In *Network Operations and Management Symposium (NOMS), 2012 IEEE*, pages 1135 –1142, april 2012.
- [20] G. Kecskemeti, G. Terstyanszky, P. Kacsuk, and Z. Nemeth. An approach for virtual appliance distribution for service deployment. *Future Generation Computer Systems*, 27(3):280 – 289, 2011.
- [21] J. Kempf, S. Whyte, J. Ellithorpe, P. Kazemian, M. Haitjema, N. Beheshti, S. Stuart, and H. Green. Openflow mpls and the open source label switched router. In *Teletraffic Congress (ITC), 2011 23rd International*, pages 8 –14, sept. 2011.
- [22] A. Kertesz and P. Kacsuk. Grid interoperability solutions in grid resource management. *Systems Journal, IEEE*, 3(1):131 –141, march 2009.
- [23] D. Kotani, K. Suzuki, and H. Shimonishi. A design and implementation of openflow controller handling ip multicast with fast tree switching. In *Applications and the Internet (SAINT), 2012 IEEE/IPSJ 12th International Symposium on*, pages 60 – 67, july 2012.
- [24] A. Köpsel and H. Woesner. Ofelia – pan-european test facility for openflow experimentation. In W. Abramowicz, I. Llorente, M. Surrige, A. Zisman, and J. Vayssière, editors, *Towards a Service-Based Internet*, volume 6994 of *Lecture Notes in Computer Science*, pages 311–312. Springer Berlin Heidelberg, 2011.

- [25] Y. Li, Y. Yang, M. Ma, and L. Zhou. A hybrid load balancing strategy of sequential tasks for grid computing environments. *Future Generation Computer Systems*, 25(8):819 – 828, 2009.
- [26] J. Liang, Z. Lin, and Y. Ma. Of-nedl: An openflow networking experiment description language based on xml. In F. Wang, J. Lei, Z. Gong, and X. Luo, editors, *Web Information Systems and Mining*, volume 7529 of *Lecture Notes in Computer Science*, pages 686–697. Springer Berlin Heidelberg, 2012.
- [27] H. Lin, L. Sun, Y. Fan, and S. Guo. Apply embedded openflow mpls technology on wireless openflow - openroads. In *Consumer Electronics, Communications and Networks (CECNet), 2012 2nd International Conference on*, pages 916 –919, april 2012.
- [28] F. Long, Z. Sun, Z. Zhang, H. Chen, and L. Liao. Research on tcam-based openflow switch platform. In *Systems and Informatics (ICSAI), 2012 International Conference on*, pages 1218 –1221, may 2012.
- [29] C. Matthews and Y. Coady. Virtualized recomposition: Cloudy or clear? In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, CLOUD '09, pages 38–43, Washington, DC, USA, 2009. IEEE Computer Society.
- [30] S. Mehdi, J. Khalid, and S. Khayam. Revisiting traffic anomaly detection using software defined networking. In R. Sommer, D. Balzarotti, and G. Maier, editors, *Recent Advances in Intrusion Detection*, volume 6961 of *Lecture Notes in Computer Science*, pages 161–180. Springer Berlin Heidelberg, 2011.
- [31] D. Milojić andić and, I. M. Llorente, and R. S. Montero. Opennebula: A cloud management tool. *Internet Computing, IEEE*, 15(2):11 –14, march-april 2011.
- [32] A. B. Nagarajan, F. Mueller, C. Engelmann, and S. L. Scott. Proactive fault tolerance for hpc with xen virtualization. In *Proceedings of the 21st annual international conference on Supercomputing*, ICS '07, pages 23–32, New York, NY, USA, 2007. ACM.

- [33] H. Nguyen Van, F. Dang Tran, and J.-M. Menaud. Autonomic virtual resource management for service hosting platforms. In *Proceedings of the 2009 ICSE Workshop on Software Engineering Challenges of Cloud Computing*, CLOUD '09, pages 1–8, Washington, DC, USA, 2009. IEEE Computer Society.
- [34] K. Ogawa, K. Higuchi, and S. Chaki. Fault recovery performance analysis of functionally distributed transport networking system. In S. Caballé, F. Xhafa, and A. Abraham, editors, *Intelligent Networking, Collaborative Systems and Applications*, volume 329 of *Studies in Computational Intelligence*, pages 281–295. Springer Berlin Heidelberg, 2011.
- [35] H. Oh, J. Lee, and C. Kim. A flow-based hybrid mechanism to improve performance in nox and wireless openflow switch networks. In *Vehicular Technology Conference (VTC Fall), 2011 IEEE*, pages 1–4, sept. 2011.
- [36] H. Oi and F. Nakajima. Performance analysis of large receive offload in a xen virtualized system. In *Computer Engineering and Technology, 2009. ICCET '09. International Conference on*, volume 1, pages 475–480, jan. 2009.
- [37] e. A. P. Luszczek. Introduction to the hpc challenge benchmark suite. *LBL-57493*, 2005.
- [38] J. K. D. S. Patrícia Takako Endo, Glauco Estácio Gonçalves. A survey on open-source cloud computing solutions. *Workshop em Clouds, Grids e Aplicações*, VIII: 3–16, 2011.
- [39] V. Philip, Y. Gourhant, and D. Zeglache. Openflow as an architecture for e-node b virtualization. In R. Popescu-Zeletin, K. Jonas, I. Rai, R. Glitho, and A. Villafiorita, editors, *e-Infrastructure and e-Services for Developing Countries*, volume 92 of *Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering*, pages 49–63. Springer Berlin Heidelberg, 2012.
- [40] P. Pisa, N. Fernandes, H. Carvalho, M. Moreira, M. Campista, L. Costa, and O. Duarte. Openflow and xen-based virtual network migration. In A. Pont, G. Pujolle, and S. Raghavan, editors, *Communications: Wireless in Developing Countries and Networks of the Future*, volume 327 of *IFIP Advances in Information and Communication Technology*, pages 170–181. Springer Berlin Heidelberg, 2010.

- [41] Qumranet. White paper: Kvm kernel-based virtualization drive. *Qumranet, Tech. Rep*, 2006.
- [42] H. Raj and K. Schwan. High performance and scalable i/o virtualization via self-virtualized devices. In *Proceedings of the 16th international symposium on High performance distributed computing, HPDC '07*, pages 179–188, New York, NY, USA, 2007. ACM.
- [43] A. Rostami, T. Jungel, A. Koepsel, H. Woesner, and A. Wolisz. Oran: Open-flow routers for academic networks. In *High Performance Switching and Routing (HPSR), 2012 IEEE 13th International Conference on*, pages 216–222, june 2012.
- [44] C. Rotsos, N. Sarrar, S. Uhlig, R. Sherwood, and A. Moore. Oflops: An open framework for openflow switch evaluation. In N. Taft and F. Ricciato, editors, *Passive and Active Measurement*, volume 7192 of *Lecture Notes in Computer Science*, pages 85–95. Springer Berlin Heidelberg, 2012.
- [45] S. Sharma, D. Staessens, D. Colle, M. Pickavet, and P. Demeester. Enabling fast failure recovery in openflow networks. In *Design of Reliable Communication Networks (DRCN), 2011 8th International Workshop on the*, pages 164–171, oct. 2011.
- [46] H. Shimonishi and S. Ishii. Virtualized network infrastructure using openflow. In *Network Operations and Management Symposium Workshops (NOMS Wksp), 2010 IEEE/IFIP*, pages 74–79, april 2010.
- [47] H. Shimonishi, H. Ochiai, N. Enomoto, and A. Iwata. Building hierarchical switch network using openflow. In *Intelligent Networking and Collaborative Systems, 2009. INCOS '09. International Conference on*, pages 391–394, nov. 2009.
- [48] D. Simeonidou, R. Nejabati, and S. Azodolmolky. Enabling the future optical internet with openflow: A paradigm shift in providing intelligent optical network services. In *Transparent Optical Networks (ICTON), 2011 13th International Conference on*, pages 1–4, june 2011.
- [49] J. Smith and R. Nair. The architecture of virtual machines. *Computer*, 38(5):32–38, may 2005.



- [50] S. Soltesz, H. Pötzl, M. E. Fiuczynski, A. Bavier, and L. Peterson. Container-based operating system virtualization: a scalable, high-performance alternative to hypervisors. *SIGOPS Oper. Syst. Rev.*, 41(3):275–287, Mar. 2007.
- [51] B. Sotomayor, R. Montero, I. Llorente, and I. Foster. Virtual infrastructure management in private and hybrid clouds. *Internet Computing, IEEE*, 13(5):14–22, sept.-oct. 2009.
- [52] A. Tootoonchian, M. Ghobadi, and Y. Ganjali. Opentm: Traffic matrix estimator for openflow networks. In A. Krishnamurthy and B. Plattner, editors, *Passive and Active Measurement*, volume 6032 of *Lecture Notes in Computer Science*, pages 201–210. Springer Berlin Heidelberg, 2010.
- [53] F. Turck, Y. Kiriha, and J.-K. Hong. Management of the future internet: Status and challenges. *Journal of Network and Systems Management*, 20:616–624, 2012.
- [54] C. Vaishampayan, S. Bidkar, S. Mehta, D. Bhamare, R. Vaishampayan, and A. Gumaste. Demonstrating openflow over a carrier ethernet switch router (cesr) - a services perspective. In *Networks and Optical Communications (NOC), 2012 17th European Conference on*, pages 1–5, june 2012.
- [55] A. Voellmy and P. Hudak. Nettle: Taking the sting out of programming network routers. In R. Rocha and J. Launchbury, editors, *Practical Aspects of Declarative Languages*, volume 6539 of *Lecture Notes in Computer Science*, pages 235–249. Springer Berlin Heidelberg, 2011.
- [56] C. A. Waldspurger. Memory resource management in vmware esx server. *SIGOPS Oper. Syst. Rev.*, 36(SI):181–194, Dec. 2002.
- [57] J. Werner. Description of network research enablers on the example of openflow. In T. Tronco, editor, *New Network Architectures*, volume 297 of *Studies in Computational Intelligence*, pages 167–177. Springer Berlin Heidelberg, 2010.
- [58] J. Werner. Description of network research enablers on the example of openflow. In T. Tronco, editor, *New Network Architectures*, volume 297 of *Studies in Computational Intelligence*, pages 167–177. Springer Berlin Heidelberg, 2010.

- [59] P. Willmann, J. Shafer, D. Carr, A. Menon, S. Rixner, A. Cox, and W. Zwaenepoel. Concurrent direct network access for virtual machine monitors. In *High Performance Computer Architecture, 2007. HPCA 2007. IEEE 13th International Symposium on*, pages 306–317, feb. 2007.
- [60] R. Wolter. A brief history of network programmability and related fields. In A. Clemm and R. Wolter, editors, *Network-Embedded Management and Applications*, pages 23–57. Springer New York, 2013.
- [61] V. Yazici, M. Sunay, and A. Ercan. Architecture for a distributed openflow controller. In *Signal Processing and Communications Applications Conference (SIU), 2012 20th*, pages 1–4, april 2012.
- [62] Y. Yiakoumis and M. Kobayashi. Innovating in your network with openflow: A hands-on tutorial. In *High Performance Interconnects (HOTI), 2010 IEEE 18th Annual Symposium on*, pages 121–122, aug. 2010.
- [63] Y. Yu, C. Shanzhi, L. Xin, and W. Yan. A framework of using openflow to handle transient link failure. In *Transportation, Mechanical, and Electrical Engineering (TMEE), 2011 International Conference on*, pages 2050–2053, dec. 2011.
- [64] B. Zhang, X. Wang, R. Lai, L. Yang, Z. Wang, Y. Luo, and X. Li. Evaluating and optimizing i/o virtualization in kernel-based virtual machine (kvm). In C. Ding, Z. Shao, and R. Zheng, editors, *Network and Parallel Computing*, volume 6289 of *Lecture Notes in Computer Science*, pages 220–231. Springer Berlin Heidelberg, 2010.
- [65] D. Zhang, L. Liu, L. Hong, H. Guo, T. Tsuritani, J. Wu, and I. Morita. Experimental demonstration of obs/wson multi-layer optical switched networks with an openflow-based unified control plane. In *Optical Network Design and Modeling (ONDM), 2012 16th International Conference on*, pages 1–6, april 2012.
- [66] X. Zhang and Y. Dong. Optimizing xen vmm based on intel: Virtualization technology. In *Internet Computing in Science and Engineering, 2008. ICICSE '08. International Conference on*, pages 367–374, jan. 2008.
- [67] H. Zhong, K. Tao, and X. Zhang. An approach to optimized resource scheduling algorithm for open-source cloud systems. In *ChinaGrid Conference (ChinaGrid), 2010 Fifth Annual*, pages 124–129, july 2010.