I.J. Computer Network and Information Security, **2014, 11, 49-55**
Published Online October 2014 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijcnis.2014.11.07

OpenFlow Technology: A Journey of Simulation Tools

Rakesh k. Jha
Shri Mata Vaishno Devi University, Katra (J&K), 182320, India
jharakesh.45@gmail.com

Pooja Kharga, Idris Z. Bholebawa, Sangeet Satyarthi , Anuradha and Shashi Kumari
Shri Mata Vaishno Devi University, Katra (J&K), 182320, India
{ pooja.kharga, idris.Bholebawa, sangeet29382, anugupta427, swamishashi449 }@gmail.com

Abstract—This paper presents a complete guideline for developing OpenFlow infrastructure. OpenFlow is standard network protocol to manage traffic between routers and Ethernet Switches. This approach will help to create the next generation virtual network, which provides the solution for network management, flow control of packets, delay optimization etc without interfering the backbone network. Here a brief idea about all the supporting tools involved in the journey of OpenFlow has been introduced. This paper provides a solution with top to bottom approach to install OpenFlow network tools. On the basis of one by one approach user is able to solve the problem appeared during the installation with proper justifications.

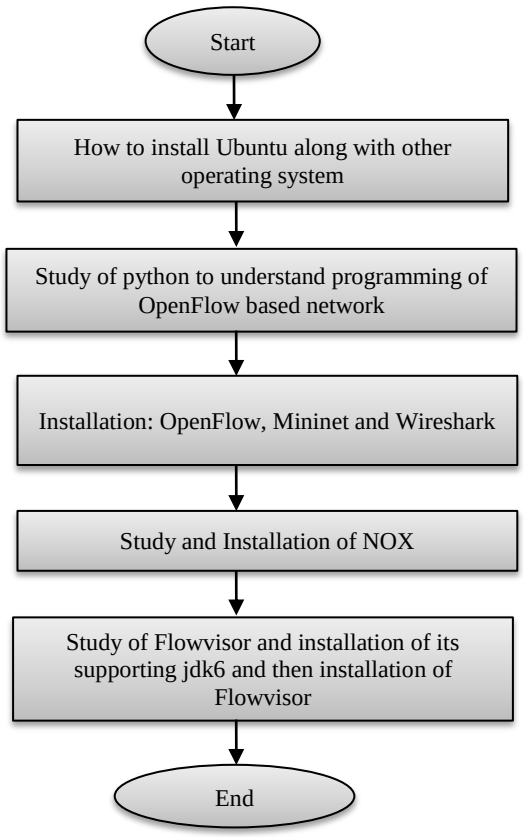
Index Terms—OpenFlow, Python, Mininet, NOX, Flowvisor.

I. INTRODUCTION

Now a day the Network should fulfil the growing demand and handle the ever growing traffic as well. To come up with new technologies for networks [14][15], we need a testbed where we can implement the experiments or changes and can check the results. To create this testbed we use Software Defined Networking (SDN). It is an approach for network virtualization where the control plane is separated from the data plane and moved to an external software controller and OpenFlow [1][2][3] is the most used SDN solution. Open source is the best platform for designing and implementing a network. Here the open source software used is Ubuntu. First we need to create a network virtually using OpenFlow [4][5] and for this Mininet [8] is used. The source code for Mininet is written in python/C++. Controllers are used in OpenFlow network to manage the flow control. NOX [10][11] is most widely used controller since it is written in C++/python. Flowvisor [13] is a special purpose OpenFlow controller which acts as transparent proxy between control and data plane.

The rest of the paper is organized as follows: Section II describes the introduction to Ubuntu and its installation. Section III discusses the application of python in OpenFlow technology. In Section IV a detailed explanation of emulator "Mininet" used in Wired OpenFlow network and its installation is outlined. Section V & VI describe the brief introduction to NOX Controller and Flowvisor, and its installation is outlined. It also gives the solution of problem appeared during installation with proper justifications. Finally Section VII concludes the whole paper.

Flow Chart of complete paper:



II. UBUNTU

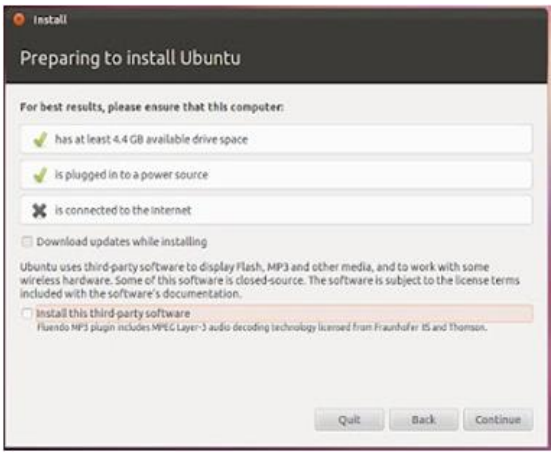
Ubuntu is an open source operating system which is distributed free.

To install Ubuntu system should have 10-20 GB of free space. First we will download its ISO file and then make it boot-able in an external disk. Connect the external disk to the computer and restart the machine and press boot button.

1. Boot from a boot-able Ubuntu USB

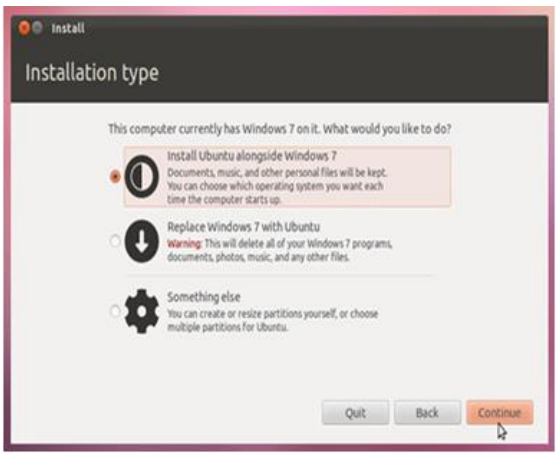
Select install Ubuntu

2. The following Ubuntu screen will appear.



Click on continue.

3. Choose installation type

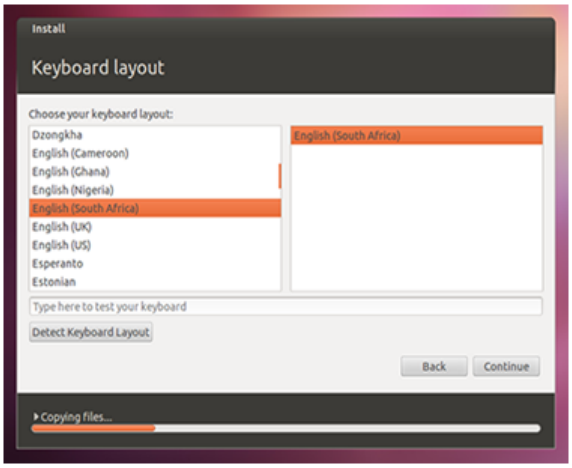


Click on continue.

4. Select location.

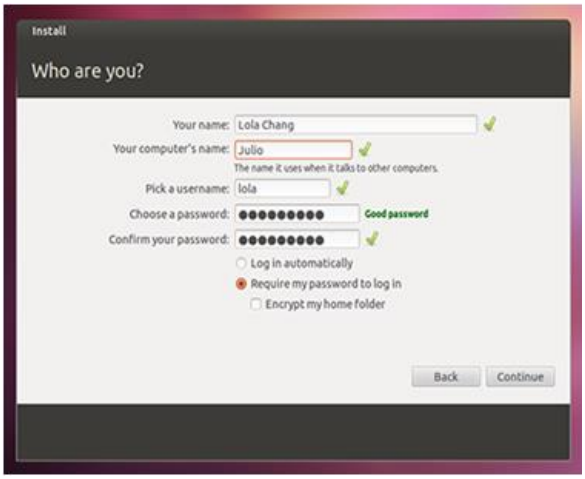
Check location and click on continue.

5. Select preferred keyboard layout.



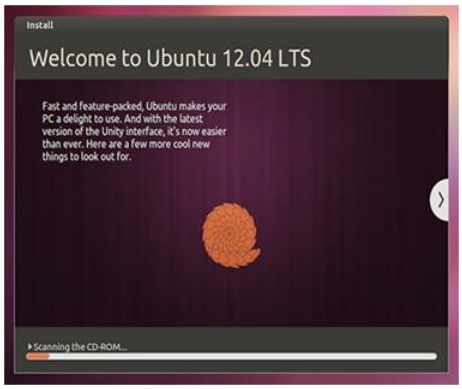
Select the language option that would be needed and click on continue.

6. Enter your login and password details



Click on continue.

7. Uploading packages



8. That's it.

All that's left is to restart computer.

After installing Ubuntu we would update it using update manager if needed.

III. Application Of Python In Openflow Based Network

Python [7] is easy to learn, powerful programming language. It has efficient high level data structures and effective approach to object-oriented programming. Python is also suitable as an extension language for customizable applications. It has mechanisms to allow a program to act both as a script and as a module to be imported and used by another python program. It also supports Graphical user interface (GUI) applications and offers much more error checking than C. Being interpreted language python can save time during program development.

Python program is executed on terminal either on python interpreter or by writing scripts. Python program runs on python interpreter by typing python on terminal and it displays '>>>' on terminal. Script is written on gedit file opened by typing gedit filename.py and after that Script is run by typing python filename.py on terminal. Since there is no compilation step, debugging python is very easy. When an error occurs, the interpreter prints an error message on terminal. As in Mininet to create any user defined topology, the program can be written in python. Most of the things used in Open Flow technology can be written with the help of python.

IV. Emulator: Mininet

Network emulator is the word used for Mininet [8]. It is a network creator; of virtual hosts, switches, controllers, and links. Mininet enables quick creation, interaction, customization and sharing of a software defined network prototype, and provides a practical conformation for implementation of the network created on hardware. It is software defined simple and inexpensive tool for experimentation on OpenFlow [6][7]. It provides a testbench for users to work concurrently on complex topologies without any need to physically develop networks. It also provides a Command Line Interface (CLI) that understands topology and is OpenFlow aware for its network wide test and debugging. Also has an extensible and easy to understand Python API for network creation. For correct system behaviour Mininet provides a smooth path, to the extent supported by our hardware performance and to experiment with topologies. Networks created on Mininet runs using real code including standard Unix/Linux network applications as well as the real Linux kernel and network stack. Because of this, the changes we make in any element of network can be directly implemented in real time physical environment with minimal changes.

Mininet supports different types of topologies, switches (ovsk, ovsl, user), hosts (cfs, proc, rt), Controllers (none, nox, ovsc, ref, remote), and tests (pingall, pingpair, iperf, all) that were defined in programs written using python language. It supports five built in network topologies: minimal, single, linear, tree and reversed. The default topology is minimal which is defined with one OpenFlow kernel switch connected to two hosts and OpenFlow reference controller but number of switches and hosts can be changed for other topologies on the command-line.

Wireshark [9] dissector is free and open source packet analyser used to capture the packets flowing in the network and extract its contents for analysis.

To install OpenFlow, Mininet and Wireshark follow given procedure:

```
$ sudo apt-get install -y git-core
$ sudo git clone git://github.com/mininet/mininet.git
$ cd ~/
$ time ~/mininet/util/install.sh
then wait for ~20 min.
Reboot to load new kernel
$ sudo reboot
```

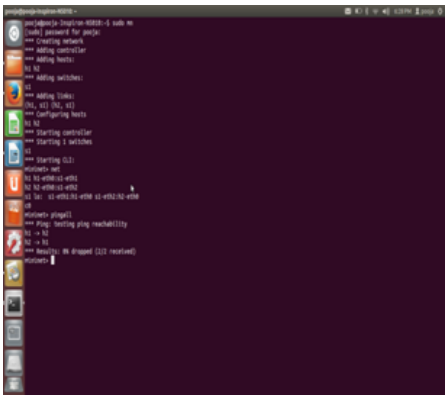


Fig 1: Running network on Mininet

As shown in Fig 1; 2 hosts are connected to a switch and the connected link are (h1, s1) and (h2, s1). Further a switch is also connected with remote controller c0.

Connectivity between hosts is verified. This is default topology which is predefined in Mininet.

Here some command-lines are given with their expected output in the form of block diagram, where boxes with 's' as name are switches and with name 'h' are hosts and the lines connecting them are links and all the switches are connected with the default controller.

$ sudo mn --topo single

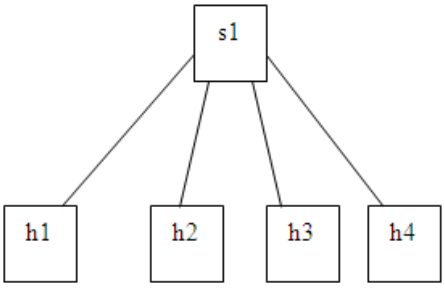


Fig 2: Single switch connected to 4 hosts

$ sudo mn --topo linear,4

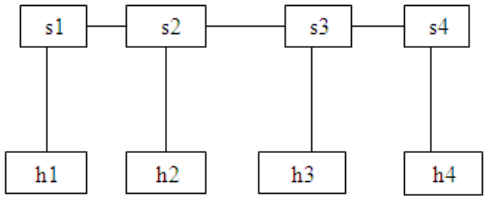


Fig 3: Linear topology of 4 switches, with one host per switch

$ sudo mn --topo tree, depth=2, fanout=2

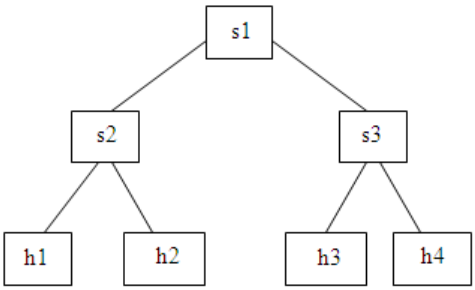


Fig 4: Tree topology with depth 2 and fanout 2

$ sudo mn –topo reversed, 4

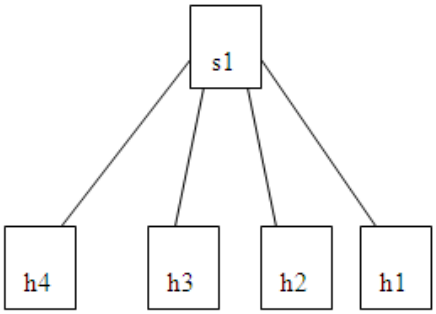


Fig 5: Single switch connected to 4 hosts, with reversed ports

We can create a number of hosts, switches and controllers as we require in our network but the number of switches to be connected to a controller is limited. This number depends on the type of switch we are using. Beside these there are tests that can give us an idea about how the network will perform in real environment. Like Ping-test checks whether a host is reachable or not to every host and iperf-test that checks what is the bandwidth of the network; which vary from system to system.

Besides these advantages, Mininet also has some limitations. It cannot exceed the CPU speed and the bandwidth iperf command shows depends on the systems hardware. It also is only compatible to Linux based OpenFlow switches or applications.

V. NOX

NOX [10] is an OpenFlow controller [15][16]. It is network controlling platform that provides high level programming interface upon which network management and control application can be built. NOX is designed to support large networks of hundreds of switches and smaller networks too. NOX controls the switches in a network through Open Flow protocol. Firstly for the new flow on the network, the packet is sent to the NOX controller then NOX passes it to other applications like how to forward the flow in network, how to modify the packets or collect statics etc. These applications are already built on NOX which reconstruct network topology, track host, provide fine grained network access control.

There are some inbuilt applications in NOX.

1. *Core application:* It provides functionalities to network application and web services.
2. *Network application:* Discovery, topology, authenticator, routing and monitoring are included in this application. Discovery keeps link between switches. Topology provides an in-memory record of all links. Authenticator provides location of host and switches.
3. *Web application:* This application is used for managing NOX through web services like web service, web server, and web service client.

In NOX, all higher functions and events are created by components or applications. A component is an encapsulation of functionality to allow declaration of dependencies, like routing can be implemented as component or any function which requires routing must declare it as dependency. NOX components can be written in either C++ or python. But python is easier for new NOX developer.

Events control all execution in NOX. There are two types of events in NOX:

1. *Core events*: Core events map directly to open flow messages received by controlled switches. They consist of data path _joint_event,

datapath_leave_event,packet_in_event,flow_mod_event, flow_removed_event, port_status_event.

2. *Application event:* Application events are part of existing NOX components. They consist of host_ event, flow_in_event and link _event.

To install NOX [12] controller, follow these steps:

```
$ sudo apt-get install libxerces-c-dev
$ sudo dpkg  -i nox-dependencies.deb
$sudo wget http://openflowswitch.org/downloads/debian/nox.list
$ sudo apt-get update
$ sudo apt-get install nox-dependencies
$ sudo apt-get install nox-dependencies
$ sudo apt-get install libtbb-dev
$ sudo apt-get install liboost-serialization-dev libboost-all-dev
$ sudo git clone git://github.com/noxrepo/nox
$cd nox
$ sudo ./boot.sh
$ sudo mkdir build
$ cd build
$ sudo ../configure
$ sudo make
$ sudo make install
Verify Install:
$ cd src
$ pwd
/home/brent/nox/build/src
$ make check
$ sudo ./nox_core -v
$ sudo ./nox_core –h
$ sudo ./nox_core
```

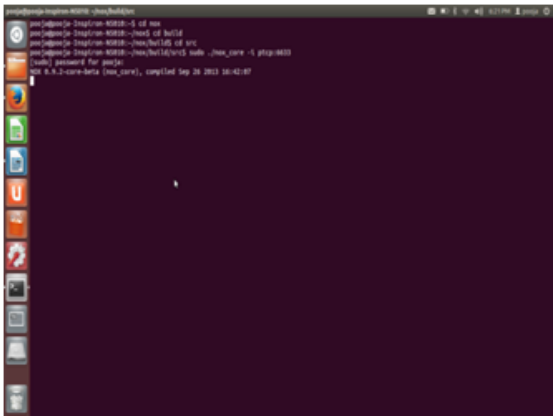


Fig 6: Running NOX on Terminal

While getting error during executing make command, just run these command and continue with the installation steps.

```
$ sudo rm -rf /var/lib/apt/lists/*
$ sudo apt-get update
$ sudo apt-get upgrade
```

The path of NOX should be at Home. If installation of NOX is done in other directory, one would get an error while installing next packages.

VI. FLOWVISOR

Flowvisor [13] allows researchers to test their new experiment on their production network at scale and with real user traffic. It works with standardizing the interface between control and data plane. It shows strong isolation between slices in network so that one slice don't interfere others. It acts as transparent layer between switches and controllers. For open flow network it act as robust virtualization layer. It is a network slicer that acts as transparent proxy between open flow switches and various network operating systems. Flowvisor containing network gives facility to control and manage some specific traffic from subset of end points. It gives great isolation between slices. Hence if one slice is performing an action or anything went wrong with it then it doesn't affect the other. In ordinary network control and data plane are logically distinct but are physically co-located. In this type they are either separated by a common protocol or there should be a clean interface planes inside the switches. Flowvisor slice the network on the basis of bandwidth, topology, forward table entry and device CPU. Slices are isolated with each other so that if one slice becomes malicious, faulty or otherwise affected then it doesn't interfere others.

When a packet arrives at the switch or router then it matches the address with flow table. Flow table consist of various flow entries. Flow entries are bit pattern, list of action, set of counter. The particular action written in flow table is performed on the packet. But if the packet doesn't matches with flow entry then packet is quid, new flow entry is sent across the network to the controller. Then controller adds the new rule to flow table to handle the qued packet. If subsequent packet comes then packet is simple forwarded without contacting the controller. Each slice has its own program control unit. Most of the time the programmer implements their own slice specific control logic as open flow controller. Flowvisor connects this controller ant open flow switch and in this way interposes between control and data plane.

To install flowvisor, pre installation of jdk 6 is mandatory.

Command for installing jdk 6

```
$ sudo add-apt-repository "deb http://archive.ubuntu.com/ubuntu hardy main multiverse"
$ sudo add-apt-repository "deb http://archive.ubuntu.com/ubuntu hardy-updates main multiverse"
$ sudo add-apt-repository "deb http://archive.canonical.com/ lucid partner"
$ sudo apt-get update
$ sudo apt-get install sun-java6-jdk
```


Err http://archive.canonical.com precise Release.gpg Something wicked happened resolving 'archive.canonical.com:http' (-5 - No address associated with hostname)

While getting an error like that during execution of command “sudo apt-get update”, just run these command:

$ sudo apt-key adv --keyserver keyserver.ubuntu.com --recv-keys 176A5C84ED67C9ED

And continue with the installation steps.

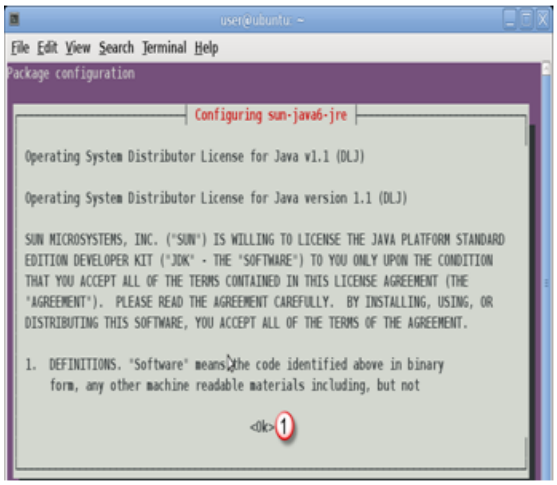
E: Could not get lock /var/lib/dpkg/lock - open (11: Resource temporarily unavailable)

E: Unable to lock the administration directory (/var/lib/dpkg/), is another processing using it?

While getting an error like that during execution of command “sudo apt-get update”, just run these command:

$ sudo rm /var/lib/dpkg/lock

Intermediate step while installation is:



Use the Tab keyboard button to move the focus to OK and then press Enter.

Use left arrow to select yes and then press Enter.

To install FlowVisor [12], follow this procedure.

Downloads

$ sudo git clone git://gitosis.stanford.edu/flowvisor.git

To install FlowVisor the user must have installed ant and sun-java6-jdk:

$ sudo apt-get -y install ant sun-java6-jdk

Installation

$ cd flowvisor/

$ sudo make CFLAGS="-I/usr/lib/jvm/java-6-sun/include -fPIC -I/usr/lib/jvm/java-6-sun/include/linux"

$ sudo make install

Running FlowVisor

$ sudo flowvisor /usr/local/etc/flowvisor/flowvisor-config.xml

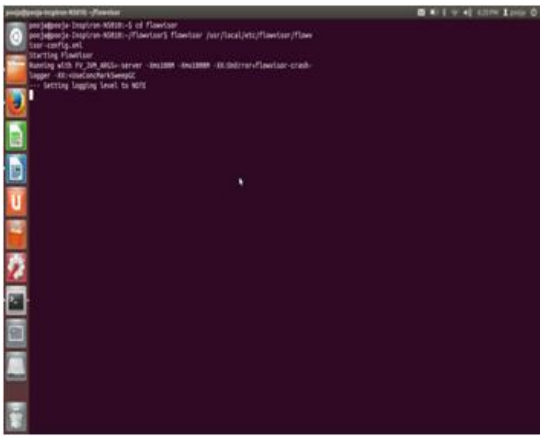


Fig 7: Running Flowvisor on Terminal

If the installation process couldn't be finished due to this alert: "respawning too fast - - DYING", do it:

$ sudo rm /usr/local/etc/flowvisor/mySSLKeyStore

Next, generate a new default configuration file with:

$ sudo fvconfig generate/usr/local/etc/flowvisor/flowvisor-config.xml

We all are getting an error while executing command “sudo apt-get update” because server doesn’t support hardy packages right now. Hardy packages are added to source.list.d during installation of jdk 6.

VII. Conclusion

OpenFlow technology is touching its peak these days. This technology is used to design an effective and real time network. Various tools are required to design and implement an OpenFlow based network. This paper includes basic idea about tools and their installation procedure. The solution for common errors is also described. The main aim of this paper is to make new user familiar with the various tools and technologies used in OpenFlow.

References

[1] OpenFlow Switch Specification Version 1.1.0 Implemented (Wire Protocol 0x02) February 28, 2011

[2] Software Defined Networking (SDN): A Reference Architecture and Open APIs, in International Conference on ICT Convergence (ICTC), Jeju Island, 2012, pp. 360-361.

[3] “Software-Defined Networking: The New Norm for Networks” at https://www.opennetworking.org/images/stories/downloads/sdn-resources/white-papers/wp-sdn-newnorm.pdf

[4] (2011, February) OpenFlow.org. [Online]. http://www.openflow.org/.

[5] Tom Anderson, Hari Balakrishnan, Guru parulkar, Larry Peterson, Jennifer Rexford, Scott shenker, Jonathan Turner Nick Mckeown. (2008, March), OPENFLOW: Enabling Innovation in campus Networks. White Paper.

[6] Simon Oechsner, Daniel Schlosser, Rastin Pries, Sebastian Goll, Phuoc Tran-Gia Michael Jarschel, "Modeling and Performance Evaluation of an OpenFlow Architecture," in Teletraffic Congress (ITC), 2011 23rd International, San Francisco, CA, 2011, pp. 1-7.

[7] (2013, July) Python/C API Reference Manual - Python v2.7.5 documentation. [Online]. http://docs.python.org/2/c-api.
[8] The Mininet Team. (2012, August) Mininet: An Instant Virtual Network on your Laptop (or other PC) - Mininet. [Online]. Available: http://www.mininet.org /.
[9] The Wireshark Team. (2013, June) Wireshark. Go Deep. [Online].http://www.wireshark.org /.
[10] "Nox ". [Online]. Available: www.noxrepo.org.
[11] Hayoung Oh, Junjie Lee, Chongkwon Kim, "A Flow-based Hybrid Mechanism to Improve Performance in NOX and wireless OpenFlow switch networks," IEEE Vehicular Technology Conference (VTC), San Francisco, CA, Sept. 2011, pp. 1-4.
[12] "Nox/Flowvisor" [Online]. Available: http://www.gta.ufrj.br/omni/index.ph.
[13] "Flowvisor". [Online]. Available: https://github.com/OPENNETWORKINGLAB/flowvisor/wiki.
[14] R. K. Jha, S V limkar, DU Dalal "A Performance Comparison of Routing Protocols (DSR and TORA) for security Issue in MANET (Mobile Ad Hoc Networks)" in IJCA special Issue on MANETs(2), 2010, pp. 78-83.
[15] Awadhesh Kumar, Prabhat Singh, Vinay Kumar, Neeraj Tyagi "Performance Analysis of AODV, CBRP, DSDV and DSR MANET Routing Protocol Using NS2 Simulation" in IJCNIS, vol.5, no.9, pp.45-50,2013. DOI: 10.5815/ijcnis.2013.09.06.
[16] H.Vignesh Ramamoorthy,D.Suganya Devi,"A New Proposal for Route Finding in Mobile AdHoc Networks", IJCNIS, vol.5, no.7, pp.1-8,2013. DOI: 10.5815/ijcnis.2013.07.01.
[17] YP Kosta, Upena D Dalal, Rakesh Kumar Jha " Security Comparison of Wired and Wireless Network with Firewall and Virtual Private Network (VPN)" in Recent Trends in information, Telecommunication and Computing (ITC), 2010 International Conference on IEEE, pp. 281-283.

Authors' Profiles

Rakesh K Jha: currently an assistant professor in school of electronics and communication department, SMVD University Katra (J&K). He is carrying out his research in WiMAX and Security issues in the laboratory ECED Lab, SMVDU. Involved research topics include WiMAX performance analysis, LBRRA, power optimization and security analysis. He has done B.Tech in Electronics & Communication from Bhopal and M.Tech from NIT Jalandhar, INDIA. Received his PhD degree from NIT Surat in 2013. Published more than 50 International Conference and Journal papers. His area of interest is Wireless communication, Communication System and computer network, and Security issues (Opti System). One concept related to router of Wireless Communication has been accepted by ITU (International Telecommunication Union) in 2010. Received young scientist author award by ITU in Dec 2010 and APAN fellowship in 2011. Also received student travel grant from COMSNET 2012. Dr. Rakesh K Jha is a member of IEEE, GISFI and SIAM, International Association of Engineers (IAENG) and ACCS (Advance Computing and Communication Society).

How to cite this paper: Rakesh k. Jha, Pooja Kharga, Idris Z. Bholebawa, Sangeet Satyarthi , Anuradha, Shashi Kumari,"OpenFlow Technology: A Journey of Simulation Tools", IJCNIS, vol.6, no.11, pp.49-55, 2014. DOI: 10.5815/ijcnis.2014.11.07

