Indian Journal of Science and Technology, Vol 9(32), DOI: 10.17485/ijst/2016/v9i32/100195, August 2016
ISSN (Print) : 0974-6846
ISSN (Online) : 0974-5645

Software Defined Networks (S.D.N): Experimentation with Mininet Topologies

Deepak Kumar* and Manu Sood

Department of Computer Science, Himachal Pradesh University, Summer Hill, Shimla – 171005, Himachal Pradesh, India; deepak.cs339@gmail.com, soodm_67@yahoo.com

Abstract

Objectives: Traditional networks were complex and difficult to manage. S.D.N is the future of programmable networks. Our objective is to find out how host communicates in realistic networking environment. **Methods/Statistical Analysis**: In order to find out how actual network works. We have used an emulator i.e. mininet. We have compared various mininet topologies. Finally we have used one of the topology to find out how host communicate in mininet environment. We have made one of host as the server and other may serve as a clients. Client host will request to the server to download particular. We have used the wget linux feature to make this possible. **Findings:** Today S.D.N has become an important part of networking. So it is important to emulate its behaviour. Mininet provides us with the inbuilt hosts, switches and controller. Each of these hosts assigned with unique IP address. Each host also corresponds to xterm display option. We can use these xterm display to make particular host as a server. Similarly we can use other xterm display for client hosts to connect to the server through wget commands. Our finding are novel because this works has not been done before. The works we have done shows that how client request and how server respond. When request is made by client to download particular file. One of the advantage of using emulator is that, application that works on emulator can be easily deployable in realistic networks. As S.D.N is a new platform so it enable the chances for the innovation of new research. As each past work helps us to make further research. So this work also benefit other researcher somewhere. **Application/ Improvements:** We can make use of S.D.N in various areas like in campus area, Labs, cellular networks etc. There is always a chance of improvement and betterment, as S.D.N is new paradigm so chances of improvement are much more. As there are risk of security. So there are wide areas of improvements.

Keywords: Mininet, OpenFlow, Openvswitch, SDN, Virtualization

1. Introduction

Software Defined Networking is rapidly growing that has changed the current networking by enabling programmatic control, easier management and chances of new innovations[1-5]. All of these benefits are due to centralized control which makes it flexible. Early networks were making use of ethane[6] and 4D[7] approach for the network control and management. To make scalable network the architecture must include local controller. The main purpose of the local controller is to run applications closer to underlying hardware devices in data plane. The controller that resides in control plane can dynamically implement policy or rules. According to these rules forwarding plane devices perform operations like identification of packet, acceptance/rejection and forwarding of packet etc[8]. We have various available choices of controller for SDN like POX[9], NOX[10], Onix[11], beacon[12] and Floodlight[13]. The networks OS controls the data plane devices through narrow interface called as OpenFlow[14] which defines the forwarding of low level data plane devices. SDN not only handles a complicated tasks but it does so in distributive way running in a rapidly changing environment. Modern data center includes thousands of switches and hundred thousands of hosts. Therefore SDN must be replicated across multiple servers[15]. To implement SDN is costly task because SDN implementation requires SDN supportive hardware resources. Being a student it is very difficult

*Author for correspondence

to buy these hardware devices to implement working of SDN. There are various open source network emulators and simulator available online. Some of the available emulators are Mininet[16], DieCast[17] and ModelNet[18] etc. Most used network simulator is NS-2[19]. The one of the limitation of the NS-2 is it does not provides us the actual networking environment.

The implementation done on NS-2 cannot be easily deployable in realistic networks. So we have used Mininet emulator because it provide realistic environment to the user. We can say that Mininet acts as a testing platform for the SDN. It helps us in rapidly prototyping the large networks with constrained resources on a single computer. Mininet provides us realistic networking environment at very cheap cost. Mininet is beneficial because it provide accuracy in performance. It is easy in use and also provides scalability. Mininet supports various topologies that help us in creation of thousands of nodes and can perform testing on them. We can modify the behavior of these topologies according to our need. Mininet support light weight virtualization that provides us the virtual network which is similar to the real network, running real kernel, switch and application code etc. Mininet is based on Command Line Interface (CLI). Mininet switches support Open Flow controller for software defined networking and custom topology. To start Mininet in Linux platform we enter command in terminal as: **Sudo mn**

In order to run Mininet as root we must use the **Sudo** keyword to run the Mininet. It starts the Mininet and creates the network by adding controller, hosts, and switches & adds links between them. This command initially creates two hosts h1, h2 and switch s1.

2. System Requirement for Working with Mininet

We can work on Mininet at various platforms like Windows, Linux and MAC. We have on Linux (Ubuntu 13.10). In Ubuntu 13.10 X server is preinstalled also gnome terminal and SSH is built in guarantee that the global best is achieved.

- Laptop/Computer with at least 2 GB RAM and at least 6-8 GB of free hard disk space
- Mininet Controller (POX[9], NOX[10], Beacon[12], FloodLight[13], Maestro etc.).
- Java/Python language support.
- Mininet

3. Installation of Mininet

Mininet creates a network of virtual hosts, switches, controllers and links. Also Mininet supports research learning, testing that helps us to develop network on a single laptop or computer. We can simply install Mininet on Ubuntu 13.10 using command: **Sudo apt-get install Mininet**[23]. This command easily installs the Mininet in Ubuntu 13.10. If this command does not work, then firstly update the working platform with command: **Sudo apt-get update.** This command will update the platform. We can use the same command again to install the Mininet i.e. **Sudo apt-get install Mininet.** There are also other ways to install Mininet in Ubuntu 13.10. But this way is the most simplistic one.

Installation of Controller

Table 1. OS Types and versions Other Requirements

OS Type	OS Version	Virtualization Software	X Server	Terminal
Windows	7,8	Virtualbox	Xming	Putty
Windows	XP	Virtualbox	Xming	Putty
Mac	OS X 10.7-10.8 Lion/Mountain	Virtualbox	Download and install XQuartz	Terminal.app (built.in)
Mac	OS X 10.5-10.6 Leopard/Snow Leopard	Virtualbox	X11(Install from OS X main system DVD, preferred), or download XQuartz	Terminal.app (built.in)
Linux	Ubuntu 13.10	Virtualbox	X server already installed	Gnome terminal + SSH built in

We also have choice of installing the controller according to our requirement. We have various options of controller like POX[9], NOX[10], FloodLight[13], Onix[11] and Beacon[12] etc. We can choose anyone of these. Each of these controller works on different platforms like POX/NOX controller supports python whereas FloodLight controller supports java platform. Therefore implementation of the SDN is platform independent[20]. We can install the controller of our choice using command given below: **Sudo apt-get install POX/FloodLight**[23]. Mininet by default includes OVCS controller and openvswitch[21]. But we can use controller of our own choice. There are also others way to install controller.

4. Basic Command Syntax while Working with Mininet.

- **$:** It precedes commands that typed at shell prompt.
- **Mininet>:** It precedes commands that typed at mininet's CLI (Command Line Interface).
- **#:** It precedes commands that are typed at a root shell prompt.

Some important mininet CLI Commands

We can take help in Mininet to make use of other commands. Mininet started with options that determine the network topology to be created[22].

- In order to see the help menu, we use the Mininet command mn with option -h as: **Sudo mn -h**
- To clear Mininet we can use command as: **Sudo mn –c**

Once we enter Mininet we also have some CLI commands. These are as follows:

- **Mininet>help**: This command show various options which we can write on Mininet CLI.
- **Mininet>nodes:** This command shows the nodes available in the current network of Mininet as shown in figure below. When we use this command it shows us the available nodes are c0, h1, h2, s1. These nodes are available for the Mininet default minimal topology.
- **Mininet>dump:** This command shows the dump information about all nodes available in the current Mininet network.
- **Mininet> h1 ping h2**: This command tests the connectivity between host h1 and h2. This command will keep checking the connectivity between hosts until we stop the command.
- **Mininet> h1 ping -c1 h2**: This command checks for the connection between host h1 and h2 for one packet**.**
- **Mininet> h1 ifconfig -a:** This command shows us the host's h1-eth0 and loopback (lo) interfaces. The interface h1-eth0 is not seen by the main Linux system, when we run the ifconfig command. Because it is specific to host of network.
- **Mininet> s1 ifconfig -a:** As switch runs by default as a root in network. Therefore running command on a switch is same as running command on a normal terminal.
- **Mininet>pingall:** This command checks the connectivity/reachability among all hosts in the network.

To start mininet xterm display option:

To open xterm display in Mininet we simply use command as follows.

- **$ Sudo mn –x**: This is the basic xterm display command when we run this command on Mininet it startup xterm display for host h1, h2, switch s1, and controller c0.
- **Mininet> xterm h1 h2:** The command **xterm h1 h2** open the xterm terminal for the host h1 and h2. Enter command in xterm of h1 i.e. **ping 10.0.0.2**. It will start checking the connectivity between host h1 and h2. Similarly we can ping the host h1 from h2 using command in h2 xterm terminal **ping 10.0.0.1.**

5. Various Topologies in Mininet

In mininet we have various topologies like minimal, single, reversed, linear, tree topology etc.

- **Minimal:** It is the most basic topology with two hosts and one switch. To run minimal topology we simply run the following command in the terminal window i.e. **Sudo mn --topo minimal**
- **Single Topology:** It is the simple topology with one switch and N hosts. To run this topology we run following command in terminal window i.e. S**udo mn – topo single,3**
- **Reversed Topology:** It is similar to the single connection but order of connection between hosts and switch is reversed. To run reversed topology we use the command in terminal window i.e. **Sudo mn –topo reversed,3**

- **Linear Topology:** It is the connection between the N hosts and N switches. We can run this topology by writing the following command in terminal window i.e. **Sudo mn –topo linear,3**
- **Tree Topology:** A tree topology is a multilevel topology with N levels and two hosts per switch. To run this topology we can use the following command in terminal window i.e. **Sudo mn –topo tree,3**

6. Comparison of Mininet Topologies

We have run various topologies over Mininet to see their startup, stop and completion time in seconds. **First** we run minimal topology which simply creates two hosts and one switch. Its startup, stop and completion times are 1 sec, 0.5 sec and 3.563 sec respectively.

Table 2. Mininet topologies comparison

Topology	No. of Host	Switches	Startup (sec)	Stop (sec)	Completion (sec)
Minimal	2	1	1	0.5	3.563
Single,50	50	1	9	5	14.234
Reversed, 50	50	1	9	5	14.234
Linear,10	10	10	4.5	2	7.233
Tree,4,4	256	85	168.4	83.9	253.34
Tree,3	8	7	4	2	6.456

Second we run single topology which simply creates 50 hosts and 1 switch. Its startup, stop and completion times are 9 sec, 5 sec and 14.234 sec. **Third** we run reversed topology which simply creates 50 hosts and 1 switch. Its startup, stop and completion times are 9 sec, 5 sec and 14.234 sec. **Four** we run linear topology which simply creates 10 hosts and 10 switches. Its startup, stop and completion times are 4.5 sec, 2 sec and 7.233 sec. **Five** we run Tree, 4,4 which simply creates 256 hosts and 85 switch. Its startup, stop and completion times are 168.4 sec, 83.9 sec and 253.34 sec. **Six** we run Tree, 3 which simply creates 8hosts and 7 switches. Its startup, stop and completion times are 4 sec, 2 sec and 6.456 sec.

7. Graphical Representation of Mininet Topologies

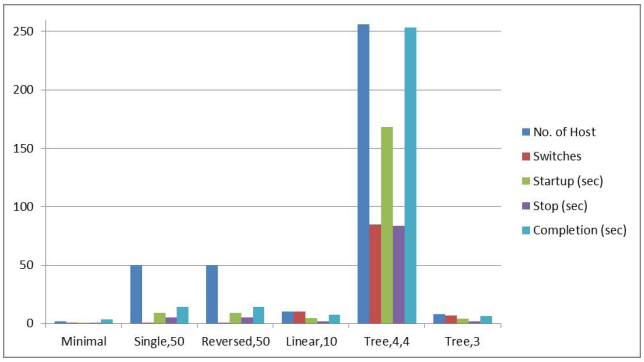

Figure 1. Graphical representation of mininet topologies.

8. Experimentation with Mininet

We have run various topologies over Mininet. When we use any of topology, it creates host and switches. For example: If we use linear topology i.e. S**udo mn –topo linear, 4.**This topology creates 4 hosts (h1, h2, h3, h4) and 4 switches (s1, s2, s3, s4). When we use **pingall** statement in Mininet[22], it checks connection between each of created host. When we ping between hosts in network, flow of packet takes place; it means that there exists reachability between hosts. Therefore, we think if there is connectivity between hosts, there are also chances of sending files between hosts with in Mininet[22] network. For making this really happen (i.e. sending of files between hosts) we have used wget feature of Linux in Mininet. We can use any of topology.

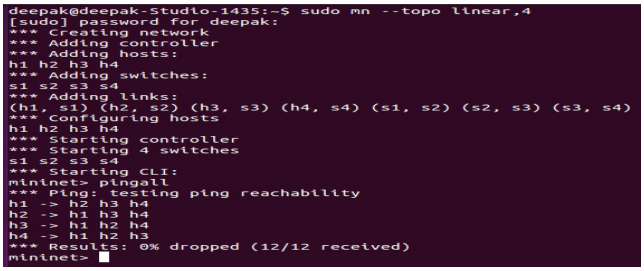

Figure 2. CLI commands for linear topology to ping all hosts in the network.

To send file between hosts, one option is to make one host as a server and other hosts as client. Since we are sending file between virtual hosts so all operations must be occur within mininet environment.

In this environment client host will request for particular file to the server. When one of the client host request

for particular file, firstly there must establish a connection between client host and server host.

We are using the linear topology to perform this operation: **Sudo mn –topo linear, 4**.This topology will create 4 hosts and 4 switches. Each of host is assigned with unique IP address. **For example:** host h1 is assigned with 10.0.0.1, host h2 is assigned with 10.0.0.2, host h3 is assigned with 10.0.0.3 and host h4 is assigned with 10.0.0.4

9. Steps to Perform Operation

Open terminal: The first obvious step is to open the terminal. When we open the terminal it looks like below figure.

Figure 3. Open Terminal.

- Run the topology: The second step is to run the topology. We have used linear topology to perform this operation i.e. **Sudo mn –topo linear, 4.**

Figure 4. CLI command for linear topology.

- After running **Sudo mn –topo linear, 4** topology. Mininet is started with 4 hosts and 4 switches. (After this everything will run inside Mininet).

Figure 5. Startup of Mininet for linear topology.

- Now to test connectivity between hosts we will use the **pingall**[22] command in Mininet.
- Now open the xterm terminal for each of the host (i.e. h1, h2, h3, h4) by entering command in Mininet terminal i.e. **Mininet> xterm h1 h2 h3 h4**

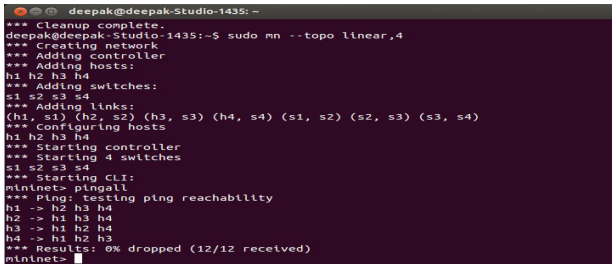

Figure 6. Pingall command for CLI linear topology.

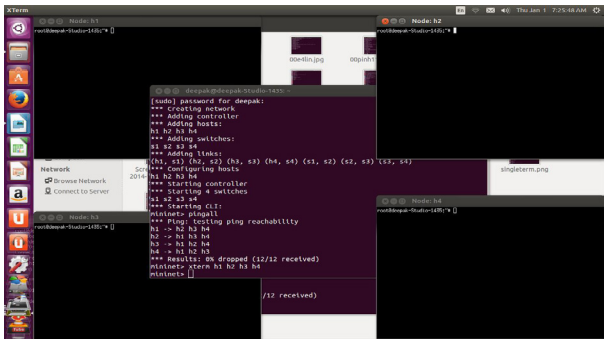

Figure 7. Xterm terminal for host h1, h2, h3, h4.

Now we have 4 xterm terminals for host h1, h2, h3 and h4. We have choice to make any of host as a server and rest of host would be act like client. And they will request to server for particular file to download. In this topology we are making host h3 as a HTTP server. To do this go to h3 xterm terminal and enter the command: **python -m SimpleHTTPServer 80 &** Now h3 xterm act as a server and other xterm will acts as a client.

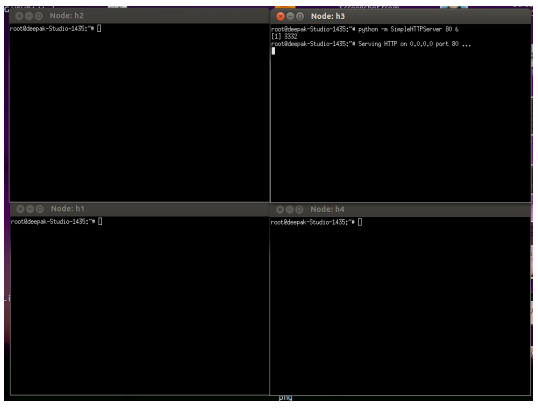

Figure 8. Host h3 as a HTTP server.

- From xterm h2 we can request server to download file. To initiate connection between client and server. We simply use the wget command, which is the inbuilt feature of the Linux. To make

connection to the server we simply use command in xterm h2: **wget 10.0.0.3.**

This will show that connection is established between host h2 and server (host h3) and host h2 download the by default text file i.e. index.html

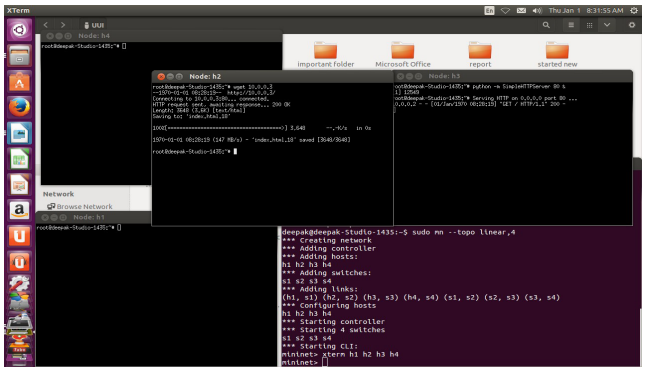

Figure 9. Host h2 as client downloading the index.html file from host h3 as server.

- When we click on the file index.html it will open browser with address i.e. **file:///home/deepak/index.html** which includes the whole directory of the system. But what if clients wants to download particular file stored in the system.

In this example we have file Dpk.py stored in the Download folder. So the main task is to make request to server to download that file i.e. **Dpk.py**, For this to make really happen what we need to do is we again have to use the Linux feature i.e. wget command to make request to the server for particular what we want to download.

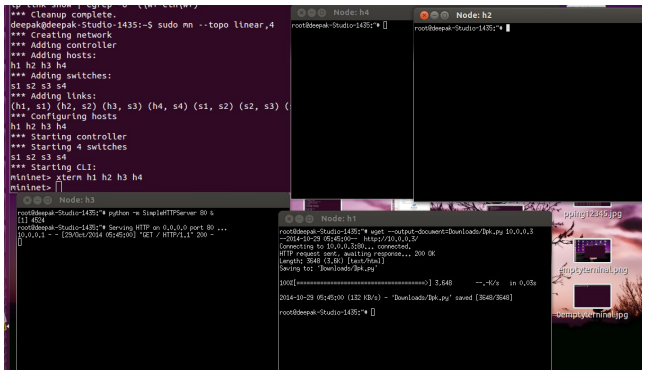

Figure 10. Host h2 requesting the server h3 for downloading file Dpk.py available in Download directory.

In our example we are using **–output-document=FILE command**[24]. For proper usage of wget commands with the Mininet tools we already have mentioned which host is server, and rest of other are client.

The last important step to make connection and to download file from server we will type command i.e. **wget –output-document=Downloads/Dpk.py 10.0.0.3** (where 10.0.0.3 is the address of h3 host, which acts as a HTTP server at post 80). When we used this command we would be able to download file (Dpk.py) from the server.

10. Conclusion

Mininet provides us the platform to understand how actual SDN works, by providing virtual network closer to realistic network. It supports to run unmodified network applications code on small as well as very large scale networks. Being a student, it is the most cost effective and cheaper way to study the behavior of SDN. Since it supports various topologies, therefore we can create our own application on Mininet by modifying source code existing topologies. The application those are developed on various platforms can be easily deployable in SDN. Therefore we can say SDN is platform independent. Mininet supports system level integration tests, which are iterative and easier to implement. The work that has been done on Mininet can be performed in same way on SDN network. So this way we can create prototype networks and later we can implement them in realistic networks.

Add Recent References including Those Published in www.indjst.org

11. References

1. Dixit A, Hao F, Mukherjee S, Lakshman TV, Kompella R. Towards an Elastic Distributed SDN Controller HotSDN'13, Hong Kong, China, 2013 Aug.
2. Casado M, Freedman MJ, Shenker S. Ethane: Taking Control of the Enterprise ACM SIGCOMM. 2007.
3. Greenberg A, Hjalmtysson G, Maltz DA. A clean slate 4D approach to network control and management SIGCOMM CCR. 2005.
4. Lakshman TV, Nandagopal T, Ramjee R, Sabnani K, Woo T. The SoftRouter Architecture ACM HOTNETS. 2004.
5. McKeown N, Anderson T, Balakrishnan H, Parulkar G. Openflow: enabling innovation in campus networks SIGCOMM CCR. 2008.
6. Casado M, Freedman MJ, Pettit J. Ethane: Taking Control of the Enterprise SIGCOMM'07. 2007; 27–31.
7. Greenberg A, Hjalmtysson G, Maltz DA, Myers A, Rexford J. A clean slate 4D Approach to network control and management. ACM SIGCOMM Computer Communications Review. 2005; 35(5):41–54.

8. Salsano S, Ventre PL, Prete P, Siracusano G. OSHI - Open Source Hybrid IP/SDN networking (and its emulation on Mininet and on distributed SDN testbeds) Univ. of Rome Tor Vergata, (2) Consortium GARR (3) CREATE-NET
9. POX [ONLINE] . Available from: http://www.noxrepo.org/pox/about-pox/
10. Gude N, Koponen T, Pettit J, Pfaff B, Casado M. McKeown N. NOX: Towards an Operating System for Networks. ACM SIGCOMM Computer Communication Review. 2008; 38(3).
11. Koponen T, Casado M, Gude N, Stribling J, Poutievski L, Zhu M, Ramanathan R. Onix: A Distributed Control Platform for Large Scale Production Networks, OSDI 10. 2010.
12. Erickson D. The Beacon OpenFlow controller. IN Proceedings of HotSDN2013.
13. Floodlight, [ONLINE]Available from: http://floodlight.openflowhub.org Date accessed 04/04/2016
14. The OpenFlow switch. Available from: http://www.openflowswitch.org. Date accessed 05/04/2016
15. Scott RC, Wundsam A, Zarifis K, Shenker S. What, Where, and When: Software Fault Localization for SDN Electrical Engineering and Computer Sciences University of California at Berkeley, 2012 Jul.
16. Lantz B, Heller B, McKeown N. A Network in a Laptop: Rapid Prototyping for Software-Defined Networks. HotNets ACM. 2010
17. Gupta D, Vishwanath KV, Vahdat A. DieCast: Testing Distributed Systems with an Accurate Scale Model. ACM Transactions on Computer Systems 29, 2011; 4:1-4:48 Scale
18. Vahdat A, Yocum K, Walsh K, Mahadevan P, Kostic D. Scalability and Accuracy in a Large- Network Emulator.
19. The network simulator - ns-2. Available from: http://www.isi.edu/nsnam/ns/.. Date accessed 07/04/2016
20. Need of new networking architecture. Available from: https://www.opennetworking.org/sdn-resources/sdn-library
21. Open vSwitch: An Open Virtual Switch. Available from: http://openvswitch.org/ Date accessed 02/04/2016
22. Introduction to Mininet. Available from: https://github.com/mininet/mininet/wiki/Introduction-to-Mininet
23. Mininet. Available from: http://mininet.org Date accessed 02/04/2016 Wget commands. Available from: http://linux.about.com/od/commands/l/blcmdl1_wget.htm Date accessed 03/04/2016