ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

Reliable SDN Network Architecture

Parvathy S Parthan, Dr. N.Guruprasad

Department of Computer Science and Engineering, New Horizon College of Engineering
Bengaluru, Karnataka, India-560103

ABSTRACT: Software Defined Networking (SDN) has a very high demand in today's business world as it addresses most of the business challenges in cost-effective method and bringing more revenue. Today's business demand for a programmable, resilient, agile network requirement made the SDN approach widely acceptable. SDN follows policy automation architecture. The idea behind SDN is to reduce manual effort with the help of SDN controller pushing the configuration to each network element and direct the traffic flow, instead of configuring each of them manually. This project will demonstrate the implementation of SDN network and analyze the flow with Wireshark.

I. INTRODUCTION

SDN -software define network, is an approach in networking that allows the network admin to build a network, which is programmable and can control changes and manage network behaviour dynamically via open interfaces.

Software Defined Network is programmable, less complex, less expensive and can be integrated with business applications. The key benefit of SDN is separation of data plane and control plane. The intelligence of routers and switches with the help of routing and switching capabilities makes the boxes more expensive. The operating systems create forwarding table to forward the packet from source to destination. Forwarding table is in the control plane which is the master brain of routers and switches. The data plane is only to push the data using the instructions from control plane. When the SDN separates the control plane from data plane, which makes the hardware really cheap. The control plane will be made in a separate server with controller. The router will just forward the packet as per the instructions from controller where the control plane resides.

II. SDN OVERVIEW

SDN can be categorized according to user's demand for application, whether application related with northbound api or south bound api. SDN architecture can be categorized in to two

1. Service configuration
2. Service definition

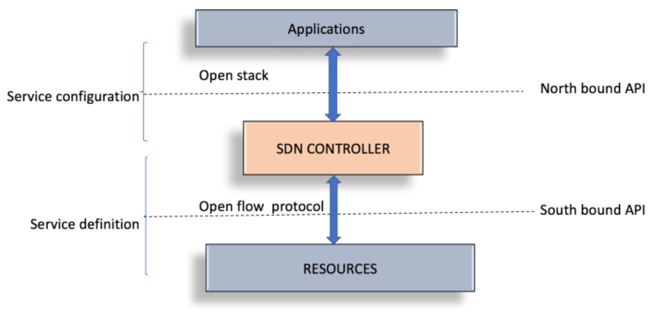


Figure 1. SDN Overview

Service configuration is to reduce the risk in configuring and operating complex network. Service definition will enable the process of defining new network behaviors of network application and sdn controller can communicate using open

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

stack protocol. The north bound api act as an intermediate between the network application and sdn controller while south bound api act as an intermediate between sdn controller and the network resources. The protocol using to communicate between these controllers and the networking resource is openflow, netconf, snmp etc

A. SERVICE CONFIGURATION

Service configuration is a higher level of network abstraction, such as switch, router, firewall and load balancer

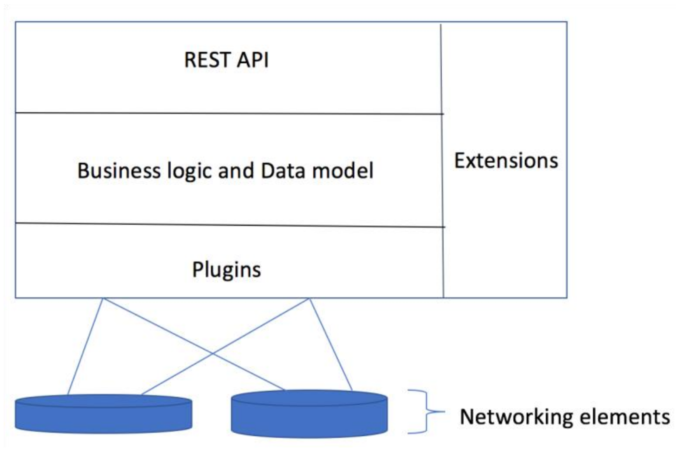


Figure 2. Service Configuration

REST/HTTP API will allow the application to interact with controller for hosting applications. Plugins which will model network abstractions, that will adapt the bits and pieces of information to actual target requirement on the back end of the network. Business logic and data modelling where the controller allocation, management, concepts of layer2 VPN taking place. It also consists of libraries for address allocation, key generation etc. some of the plugins available are cisco nexus, juniper mx, etc. It is based on open stack protocol which is at the northbound interface of the controller.

B. SERVICE DEFINITION

The main responsibility of service definition is the data plane abstraction like port forwarding, flow table entries, metering etc. it is an interface for manipulating abstractions.

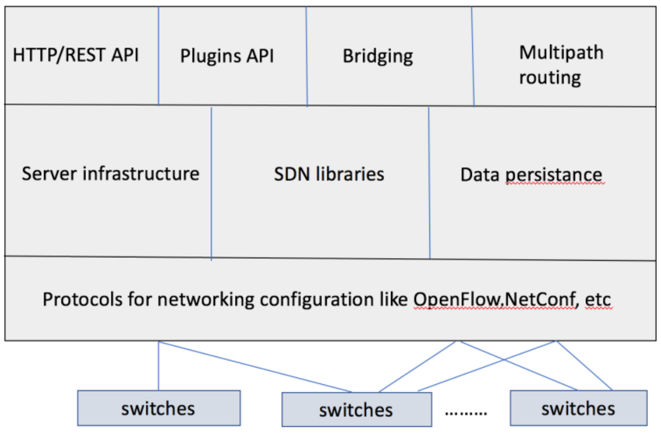


Figure 3. Service Definition

III. CUREENT VS SDN DESIGN

A. EXISTING

The current network infrastructure will work according to routing/forwarding decisions of various protocols like BGP, OSPF, EIGRP, STP etc. Both the forwarding table creating and data forwarding is under the responsibility of the network element which is usually error-prone and demands manual effort. This can be simplified by introducing network function virtualization in SDN.

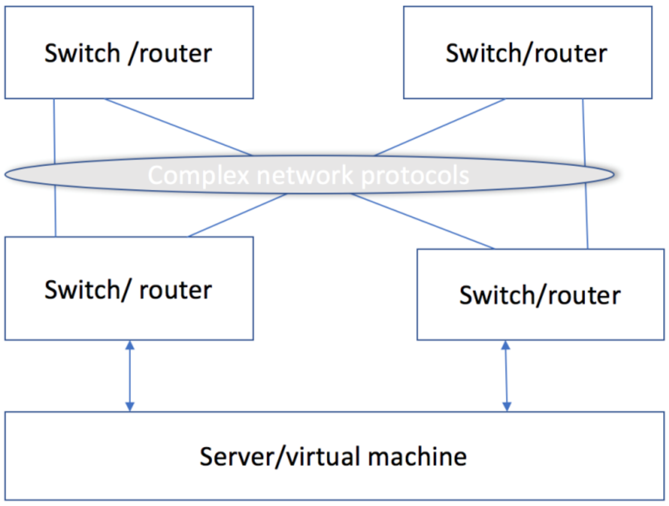


Figure 4. Existing Network Designs

When a network element need to communicate with other network element there will be complex networking protocol acting in between the two elements which will increase the complexity of the network. If any failure occurs, then the network element need to be reconfigured manually. This will result delay in network operations and performance. In these scenarios, the network element is very expensive because of the modules used in the router to creates network configuration protocols. The brain of the network is control plane; data plane will follow the instructions from control plane.

B. SDN NETWORKS

The network will run on SDN controller protocols like openflow. The concept of SDN is taking out the brain of the network element such that it will only need to forward the packets. This will make any network element less expensive and avoid overhead of creating control plane for making forwarding decisions. Centralized controller resides in a separate server controls all the network components will reduce the complexity. If any error happens then the Network Engineer need the access the controller to maintain and manage the network without going to each and every network element.

IV. LAYERS OF SDN

We can consider this as three layers one is orchestration of configuration and control plane interaction which is north bound interface that is the communication between the network applications to the controller and there will be a interaction between control plane and resource and the interaction with in the control plane is called east west protocol.

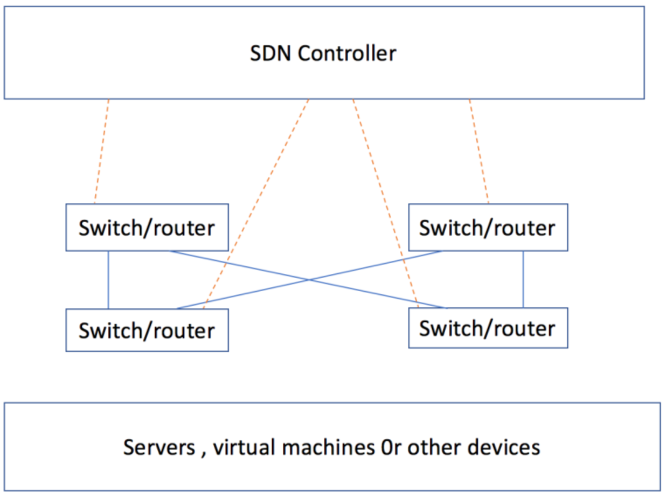


Figure 5. Proposed Network Design

There are three layers in SDN architecture

1. Orchestration of configuration
2. Control plane interaction with northbound interfaces communicating between network applications to the controller
3. Interaction between control plane and resources. The interaction within the control plane is called east west protocol

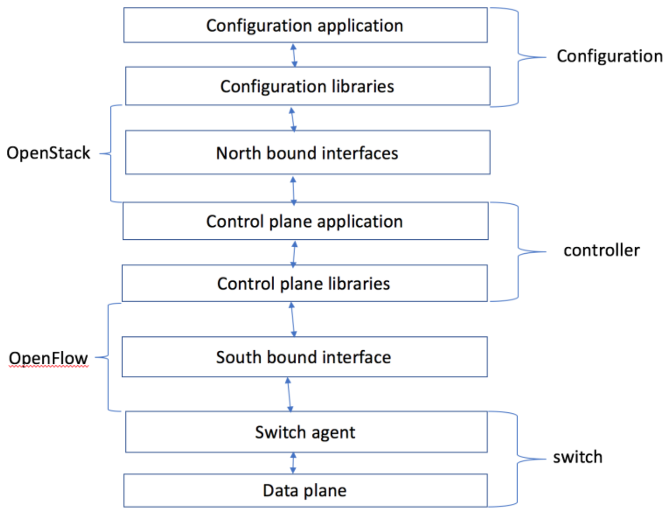


Figure 6. Layers in SDN Architecture

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

A. SWITCH

Switch, consist of two elements switch agent and data plane. Switch agent will manage the southbound interface protocol, it will act as intermediate between the data plane and southbound interface. It will translate from SBI (South bound interface) to data plane and vice versa. If any event occur in the data plane switch agent will inform the SBI.

B. DATA PLANE

Packet abstraction methods like header classification, header modification, flow statistics, flow metering and output handling are taking place in data plane.

C. SOUTH BOUND INTERFACE

It act as an intermediate with the underlying resource and the controller. The controller send instructions to the white box switch through the SBI. SBI is a protocol of network which will handle authentication, authorization, accounting, version negotiation, dead peer detection, capability discovery, queries and asynchronous events etc

D. CONTROL PLANE LIBRARIES

Control plane libraries will normalize SBI variations such that hiding the details to reduce the complexity. It also presents a unified data model to the application distributed data structure, topology discovery and protocol handler. These are some of the application libraries present in control plane libraries.

E. CONTROL PLANE APPLICATIONS

It will provide unique features like multipath forwarding, flood/broadcast suppression, source rate limiting etc. Some of the control plane applications are Ethernet bridge, IP router, application load balancer, firewall etc.

F. NORTH BOUND INTERFACE

Provide an interface between controller and application layer. It will use either HTTP or REST API's to communicate, and it uses open stack protocol for this communication. Capability discovery, version negotiation, authentication, authorization are done by using network protocols

V. SDN ENABLING TECHNOLOGIES

1. SDN Controllers
2. South bound protocols
3. Northbound API's
4. Commodity network switches
5. Overlay protocols
6. Network function virtualization

VI. SDN NETWORK IMPLEMENTATION

A. MININET

Mininet is a network emulator. It can emulate a complete network of switches, links, and host on a single physical or virtual machine. Mininet uses light weight virtualization method to make a single system look like a complete network. By using Mininet we are creating a network with software. The network consists of switches, hosts, links and controllers. In this project, I will be creating a networking elements using mininet.

Which resembles the original hardware network element and we can deploy it on real hardware. By using mininet we can easily interact with network using the mininet command line interface, we can customize it, we can even share the same with others. It is a great way to develop, share and do project with openflow and SDN in system

B. BENEFITS OF USING MININET

1. We can create custom topologies using mininet.
2. We can execute real programs that runs on linux kernel can be easily be created with mininet
3. With the help of simple python codes, we can create the network topologies as per our needs.
4. Easy to interact with the network created by using mininet.
5. Open source project and also under active development to add more features.

C. MININET LIMITATIONS

We can run mininet using the Linux kernel only. Mininet won't provide an open flow controller, if we need then we need to customize it using other application

Table 1: Useful commands and its description.

Sl. No	commands	Description
1	Sudo mn -h	It will display healp message
2	Sudo wireshark&	Starts the wireshark in background
3	Sudo mn	Create a simple topology with 2hosts, and 1 switch.
4	Sudo mn -c	It will clean all the previous topology.
5	Sudo mn –test pingpair	It will run oneregresstion test
6	Sudo mn –test iperf	It will run one iperf host on one side and run other iperf host on other side and parsed the bandwidth between each.
7	Sudo mn –controller remote	It will add one remote controller
8	Sudo mn –topo=single,4	It will create a topology with one switch, one controller and 4 hosts.
9	Sudo mn –topo=linear , 4	It will create four swtch and four host which are linearly connected each other.
10	Sudo mn –topo= tree ,2,2	It will create a topology like tree structure with 4 host and 2 switches , where each switch shares 2 hosts and one controller to the switches
11	Sudo mn –controller remote –ip 127.0.0.1 –port 6633	It will customise a remote controller with ip 127.0.0.1 and port no 6633

12	sudo mn -x	It will start one xterm for each hostand switch, pass the -x option. the xterms will pop up, with automatically set window names
13	sudo mn --switch ovsk --test iperf	To run the user-space switch: it will create a OVSk switch , and will add new functionality especially when software requirement is not critical.
14	sudo mn --test none	To record the time to set up and tear down a topology
15	sudo mn --innamespace --switch user	It will show everything on its namespace.

Table 2:Mininet CLI commands

SL NO	CLI commands	Description
1	help	Display all mininet CLI commands
2	nodes	Display nodes
3	net	Display links
4	dump	Dump information about all the nodes will be displayed
5	h1 ifconfig -a	Run a command on a host process
6	S1 ifconfig -a	Run a command on a switch process , it will display all details of s1
7	h1 ps -a	Print the process list from a host process.
8	s1 ps -a	Print the process list from a host process.
9	h1 ping -c1 h2	Shows the open flow of the packet.
10	pingall	Check the connectivity by pinging all host form each other.
11	Exit	It will exit the mininet
12	Iperf	Testing tcp bandwidth between h1 and h2.
13	xterm h2	It will display the h2 nodes xterm.
14	h1 arp	To check ARP tables on each host.
15	dpctl dump-flows	To see the contents of the flow tables on all switches .
16	dpctl del-flows	To clear all flow tables on all switches.
17	link s1 h1 down	It will down the link between the s1 and h1.
18	link s1 h1 up	It will up the link between the s1 and h1.
19	py 'hello ' + 'world'	It will print the python output as hello world
20	py locals()	Print the accessible local variables
21	py dir(s1)	the methods and properties available for a node, using the dir() function:

D. ORACLE VM VIRTUAL BOX

Oracle VM virtual box is a cross platform virtualization application that allows multiple operating systems to run inside the virtual box. We can run linux and windows on mac or we can run linux on windows pc. Here we will be using linux on windows pc. Virtual box can be run in small embedded system or cloud environment or data center deployment etc.

E. WHY VIRTUALIZATION IN SDN

1. We can run multiple OSs simultaneously
2. Easy software installation. for example, it is very easier to install a mail server than a normal system.
3. Testing and disaster recovery once installed. A VM and its virtual hard disks can be considered a ‘container’ that can be arbitrarily frozen, waken up, copied, backed up and transport between hosts.
4. It will provide a snapshot feature, which will allow user to share a particular state of a VM that need to be back to the state.
5. Infrastructure consideration virtualization will reduce hardware cost. Instead of running many physical computers we can run many virtual machines in one single VM box.

F. TERMINOLOGY

Every virtual machine has guest OS, host OS and virtual machine OS. Here the host OS is windows 10. Which is the host machine where the VM box is running. Guest OS is the OS of the virtual box which is the guest machine running on the host machine. The virtual machine OS is the OS of the virtual environment that we are creating using the virtual box.

G. FEATURES OF ORACLE VM VIRTUAL BOX

1. Portability
2. No hardware virtualization required.
3. Guest additions, shared folders, seamless windows, 3D virtualization.
4. Support both 32-bit and 64-bit host virtual box is also called ‘hosted’ hypervisor or ‘type2’ hypervisor.
5. Great hardware support (guest multiprocessing, USB device support, hardware compatibility, full ACPI support, multi-screen resolutions, built in ISCSI support, etc)

H. OPENFLOW CONTROLLER

VM groups- Enable the user to organize and control VM collectively as well as individually.

Open flow controller is general purpose software running on a commodity system. It basically of the shelf control hardware, that is, one machine running one server manages a handful of openflow switches. Controller is not a piece of specializes hardware, it is an actual hardware. Some of the opensourse controllers are POX, OPENDAYLIGHT, etc. there are two types of conversation between openflow switch and openflow controller

1. Synchronous communication: this communication initiates by the controller, sometimes the application of the controller needs to learn some information.
2. Asynchronous communication: unexpected message from the switch or any messages from the switch

Open flow controller is the brain of SDN, it will run on the control plane and programs the flows on the network switches based on the control plane logics. the controller runs on a server that this server is capable of reaching all the openflow switches in the network. the controller will be maintained and controlled by an administrator for the ease of use

I. OPENFLOW SWITCHES

Open flow switch is a typical networking switch but it doesn’t have the intelligent protocol. It has only the forwarding properties, and follows controller instructions to forward thepackets. It runs the software to bare minimum to connect with the openflow controller and install and analyse the flows. The open flow switch can be categorized in to two.

1. Pure open flow switch: the switch which supports only open flow protocol.
2. Hybrid open flow switch: it will support traditional Ethernet protocols in addition to the network protocols.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

J. OPENFLOW PROTOCOL

The openflow protocol is a standardized protocol for interacting with the forwarding behaviors of switches from multiple vendors. That is a communication interface defined between the control and forwarding layers of SDN architecture. Openflow protocol can be analyzed through the topology in this project.

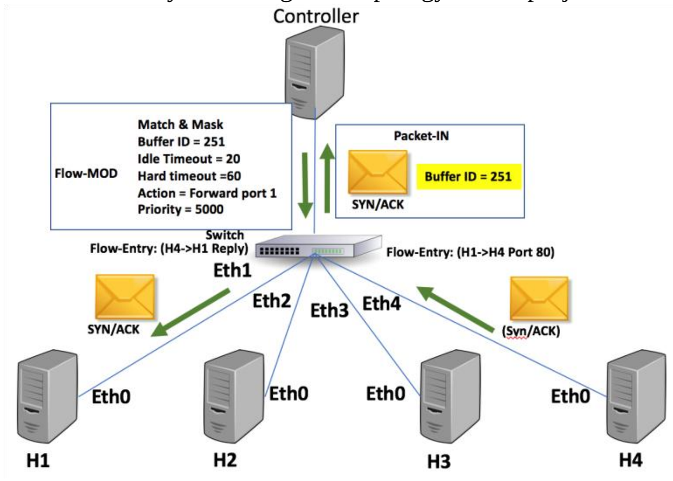


Figure 7. Openflow Protocol

1. If we want to ping from h1 to h3, then from h1, h1 starts its conversation SYN msg. it will send a SYN msg to s1.
2. S1 recevies this SYN msg and check its flow table to find the destination h3. Since it is the first transfer there will not be the destination path. This is called table miss. Then it will contact the controller for its help by sending PACKETIN msg consist of a buffer id, let's say a buffer id=250. By using the buffer id, it notifies the controller what to do with the packet. We can send eithr the encrypted full msg or else contain only the refernce packet.
3. While receving the PACKETIN msg. controller will check the routing protocols and find the path. And send a PACKETOUT msg with buffered and action to do with the packet.Controller will aso send a FLOWMOD msg along with the PACKETOUT msg. The FLOWMOD msg consist match and mask, that will contain the details about buffer id, idle timeout, hard timeout, action, priority.

   Timeout: Timeout tells the switch how long to store a flow entry. Idle timeout 20sec: means if there is no matching HTTP request for 20sec then remove this flow entry. Hard timeout 60sec tells no matter whether the packet is alive or not, it will remove from the flow entry

   Priority: priority is a very important concern. If two flow entries are storied and one with higher priority and other with lower priority. the switch will ignore the lower priority

   Action: It informs the switch that any tcp port which requires the IP MAC of h1 to the IP MAC of h3, send all of those to the port4

   Buffer id: It will instruct the switch that the packet send with this buffer id 250 release the packet from the buffer and applied the actions in this message

   Actions: We can perform multiple actions. We can change multiple headers ip's, TCP port etc, push, pop and swap mpls labels, we can also take actions like flooding outputs, dropping the packet, instruct the switch to send matching packet to the controller, or we can make the switch to use the normal non openflow packet processing.

4. Along with this FLOW MOD, PACKETOUT msg the controller will make a flow entry in the switches flow table.
5. According to the revised flow table the s1 find the path to h3 and send packet through the path.
6. After receiving the packet, h3 will send the response back to the h1 , the s1 will again check the path from h3 to h1 , table miss occurs since it is the first packet . all the above process repeats

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

7. Switch will get the complete path from h1 to hh3 and h3 to h1 in the flow table. And further packet transfer is done by the switch since it has the path.

K. POX CONTROLLER

In this project, I am using the pox controller. Pox controller will help us by providing a frame work for communicating with SDN switches using either the OPENFLOW or OVS DB protocol. We can create an OPENFLOW SDN controller with the help of pox controller. We can directly use POX as an SDN controller because it comes already with a bundle of components.

L. POX COMPONENTS

When the pox controller get started the pox components get invoked. Pox components is nothing but the application python program, which is bundled with the pox controller, these components implement the functionality of the network. Like all SDN controller the pox will allow user to program their own application that uses the controller as the intermediately between the network application and network equipment.

M. GETTING STARTED WITH POX

1. Download the pox controller
2. Invoke the controller and get started by selecting pox components to run.
3. Test controller by pingall
4. Check flow table
5. Exit the controller by control+z

N. WIRESHARK

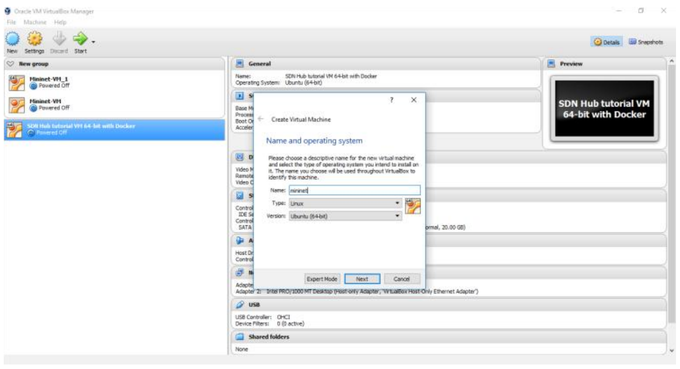
Wireshark is a free open source platform for packet analyser. We can use this platform to perform the network troubleshooting, communication protocol development etc. Earlier it was known as ethereal, in 2006 it was renamed as Wireshark. It "understands" the structure of different networking protocols and capture the data.

1. Data can be captured from a live network connection
2. Different types of networks can be analysed, including all types of network like Ethernet, IEEE 802.11, PPP, and loopback.
3. we can browse the captured data by GUI.
4. we can modify the captured data using programming
5. Various settings like timers and filters can be set to ensure triggered traffic

VII. CONFIGURATION STEPS OF AN SDN CONTROLLER IN OPEN SOURCE MININET EMULATOR

1. Download and install oracle VM virtual box
2. Set up the Mininet network simulator VM
3. Using the Mininet network simulator to create a network topology by python programing
4. Download and run the pox controller
5. Test the controller connectivity by pingall command

A. Setting up mininet vm.


Create Virtual Machine

Name and operating system

Please choose a descriptive name for the new virtual machine and select the type of operating system you intend to install on it. The name you choose will be used throughout VirtualBox to identify this machine.

Name: mininet
Type: Linux
Version: Ubuntu (64-bit)

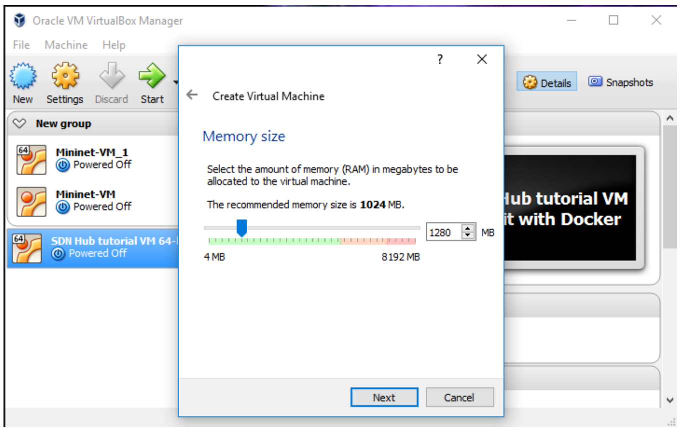
General
Name: SDN Hub tutorial VM 64-bit with Docker
Operating System: Ubuntu (64-bit)

USB
USB Controller: OHCI
Device Filters: 0 (0 active)

Shared folders
None

Figure 8. Setting up Mininet VM: Step-1

For setting up Mininet we need to import the Mininet VM to the virtualbox. Then set the name and operating system details.


Create Virtual Machine

Memory size

Select the amount of memory (RAM) in megabytes to be allocated to the virtual machine.

The recommended memory size is **1024** MB.

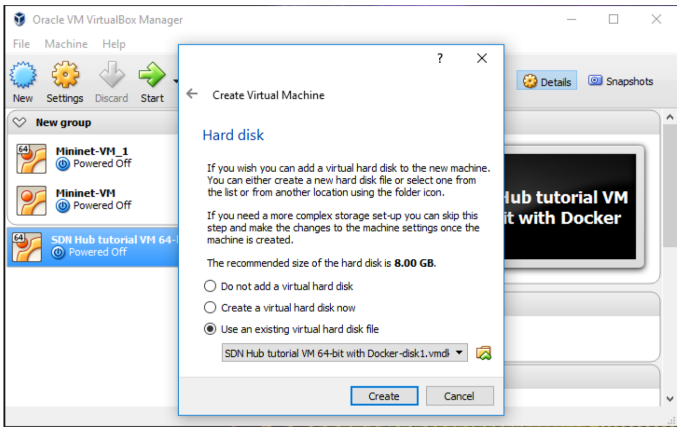
1280 MB

4 MB 8192 MB

Figure 9. Setting up Mininet VM: Step-2

After setting-up the os and name, provide appropriate memory size to run the Mininet VM

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017



Create Virtual Machine

Hard disk

If you wish you can add a virtual hard disk to the new machine. You can either create a new hard disk file or select one from the list or from another location using the folder icon.

If you need a more complex storage set-up you can skip this step and make the changes to the machine settings once the machine is created.

The recommended size of the hard disk is **8.00 GB**.

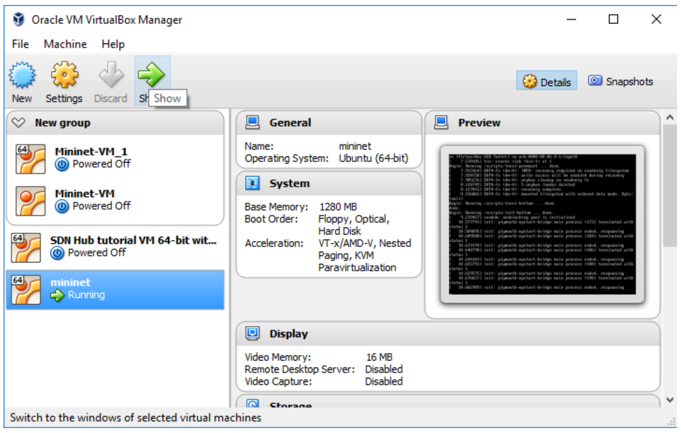
- Do not add a virtual hard disk
- Create a virtual hard disk now
- [✓ Use an existing virtual hard disk file]

SDN Hub tutorial VM 64-bit with Docker-disk1.vmdk

Create Cancel

Figure 10. Setting up Mininet VM: Step-3

Select the hard disk to run the Mininet environment. Then by clicking the create button, Mininet will be created.


Oracle VM VirtualBox Manager
File Machine Help
New Settings Discard Show
Details Snapshots
New group
Mininet-VM_1
Powered Off
Mininet-VM
Powered Off
SDN Hub tutorial VM 64-bit wit...
Powered Off
mininet
Running
Preview
Storage
Switch to the windows of selected virtual machines

General

Name:	mininet
Operating System:	Ubuntu (64-bit)

System

Base Memory:	1280 MB
Boot Order:	Floppy, Optical, Hard Disk
Acceleration:	VT-x/AMD-V, Nested Paging, KVM Paravirtualization

Display

Video Memory:	16 MB
Remote Desktop Server:	Disabled
Video Capture:	Disabled

Figure 11. Setting up Mininet VM: Step-4

Now the mininet VM is running and it can create our network topology for testing the performance of SDN network.

ISSN: 2350-0328
International Journal of Advanced Research in Science, Engineering and Technology
Vol. 4, Issue 3 , March 2017

B. Starting wireshark

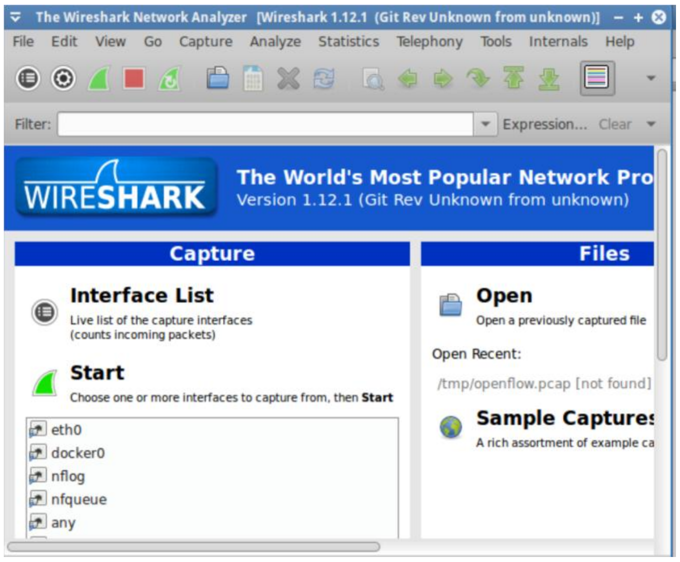

The Wireshark Network Analyzer [Wireshark 1.12.1 (Git Rev Unknown from unknown)]
File Edit View Go Capture Analyze Statistics Telephony Tools Internals Help
Filter:
Expression... Clear
WIRESHARK
The World's Most Popular Network Pro
Version 1.12.1 (Git Rev Unknown from unknown)
Capture
Files
Interface List
Live list of the capture interfaces (counts incoming packets)
Start
Choose one or more interfaces to capture from, then Start
eth0
docker0
nflog
nfqueue
any
Open
Open a previously captured file
Open Recent:
/tmp/openflow.pcap [not found]
Sample Captures
A rich assortment of example ca

Figure 12. Setting up Wireshark

By typing the Sudo wireshark& in the terminal will create a wireshark that will run in the background.

C. Starting Pox controller

The Wireshark captures and analyse OpenFlow packets.

VIII. CONCLUSION

As we know that for every technology there should be a strong backbone network. Technology transformation is dramatically and a resilient, agile network with next-gen technology support is the base for every network. SDN approach is reliable, efficient, error free, resilient, cost effective in network design and operation. The above approach showcased an SDN implementation demonstration with packet analysis with Wireshark.

REFERENCES

[1]K. Hyojoon, N. Feamster, "Improving network management with software defined networking", Communications Magazine IEEE, vol. 51, pp. 114-119, 2013.
[2]W. Anjing, M. Iyer, R. Dutta, G. N. Rouskas, I. Baldine, "Network Virtualization: Technologies Perspectives and Frontiers", Lightwave Technology Journal of, vol. 31, pp. 523-537, 2013.
[3]"VXLAN: A Framework for Overlaying Virtualized Layer 2 Networks over Layer 3 Networks", RFC 7348 (Informational), [online] Available: http://datatracker.ietf.org/doc/draft-mahalingam-dutt-dcops-vxlan.
[4]"Open Networking Foundation Software-Defined Networking: The New Norm for Networks", ONF White Paper
[5] B. Raghavan et al., "Software-defined internet architecture: Decoupling architecture from infrastructure", Proc. 11th ACM Workshop Hot Topics Netw., pp. 43-48, 2012.