**Prelab3 - Introduction to SDN and OpenFlow Building a Firewall**

1. [10 pts] Use your own words to answer the following questions. For each

question, provide your reference from either the list of References above or

from within this document:

a) What is Software Defined Networking?

Ans. Software Defined Networking or SDN is a Network layer paradigm where instead of the traditional handling of both the data and the control plane happening locally on the router, the two are decoupled and control plane/routing is handled by a central “controller”, determining the path packets take from host A to host B by calculating and updating the flow table dynamically. While the data plane continues to be handled locally by the routers.

Reference: Software Defined Networking & OpenFlow in the document and book Computer Networking: Chapter 4

b) What is OpenFlow?

Ans. OpenFlow is a popular SDN protocol. It is used to control the communication between controller and switches. We use OpenFlow within Mininet in this lab.

Reference: OpenFlow 1.3 Overview in the document

c) What is Mininet?

Ans. Mininet is a network emulator tool used to design and test large scale networks.

Reference: Prelab 1 pdf

d) What is the purpose of a Controller in Mininet?

Ans. The controller in Mininet is used to determine the rules and flow of packets through our emulated network moving via switches.

Reference: Mininet and its Controller: in the document

e) What is the POX Controller? How is it different from the Mininet default

controller?

Ans. POX is SDN remote controller. It is different from Mininet’s default controller in that we get to write its implementation which gives us greater flexibility and tools to design and test our network it is also written in python.

Reference: Mininet and its Controller: in the document and [POX WIKI](https://intronetworks.cs.luc.edu/auxiliary_files/mininet/poxwiki.pdf)

2. [4 pts] Invoking a Controller in Mininet:

In the Mininet environment different controllers can be activated:

● “sudo mn” invokes Mininet along with its internal controller

● “sudo mn --controller=remote” invokees a remote controller

a. What command would you use to invoke the remote controller with IP

Address 127.0.0.1 and port 8900?

Ans. sudo mn –controller=remote, ip = 127.0.0.1, port = 8900

b. After running “sudo mn” in a terminal, run the command “dump” after

Mininet has finished setting up. What is the IP address and Port number of

the controller being used by Mininet? Include a timestamped screenshot of

the output of the command and circle in red separately the IP address and

Port number being used by the controller.

Ans. IP address: 127.0.0.1, port number: 6633

A screenshot of a computer

Description automatically generated

3. [4 pts] Default Topologies

a. What command invokes the controller to create a tree topology where the

“root” switch is connected to 3 other switches and each of these switches is

connected to 3 unique hosts?

Ans. sudo mn --controller=remote,ip = 127.0.0.1,port=8900 --topo=tree,depth=2,fanout=3

A computer screen shot of a program

Description automatically generated

b. Using a drawing tool, draw an image of this topology, labeling the switches

A diagram of computer network

Description automatically generatedand hosts with the IP addresses assigned to their interfaces.

Ans.

4. [4 pts] POX Controller

a. Why are we using the POX controller (instead of the default controller) to implement the firewall?

Ans. We are using POX because it’s simple and we can write the firewall from scratch hence giving us more flexibility, also POX is python based.

b. Add the POX controller to your drawing in 3b using arrows to show the communication between the devices and the controller.

A diagram of a remote controller

Description automatically generatedAns.

5. [4 pts] OpenFlow Messages

a) Show a timestamped screenshot of the terminal running your single command to invoke Mininet with the topology and the output of Ping.

Ans. A screenshot of a computer

Description automatically generated

b) In another screenshot circle in red (if any) of the following packets that are displayed in Wireshark: Packet-In, Packet-Out, Flow-Mod and Flow-Expired

A screenshot of a computer

Description automatically generatedAns.

c) Explain the purpose of each OpenFlow message that you found.

Ans. i. Packet\_in: This is produced when the switch receives a packet that it doesn’t know what to do with, as such the switch sends the packet to the controller for further inspection.

ii. Packet\_out: This is produced by the controller to inform them on how to deal with the packet.

iii. Flow\_mod: This is produced by the controller to inform the switches on how to modify their flow table whether by adding, deleting or modifying a particular flow.

6. [6 pts] Requirements, Guidance and Getting Started with the Firewall:

a) How will you “install” the rule in the switch for 30 seconds?

Ans. We can install a rule for a fixed length of 30 seconds by setting the value of hard\_timeout (int) to 30 seconds in an ofp\_flow\_mod message. If instead we want the switch to keep remembering the rule in case it encounters activity for the rule and forget the rule when there is no activity for 30 seconds then, we will set the value of idle\_timeout(int) to 30 seconds

b) What is packet flooding?

Ans. Packet flooding is done when the switch broadcasts a received packet out of all of its ports except the one where it received the packet from. This is done usually when the switch doesn’t recognise the destination address.

c) Draw an image of a switch with 4 ports. Show a packet entering on one port

A diagram of a switch

Description automatically generatedand then show packet flooding.

Ans.

7. [5 pts] Basic Firewall Rules (Table 1):

a. Briefly explain the meaning for Rules 1, 3, 4 and 9 from Table 1. A sentence or two for each rule is sufficient.

Ans.

Rule 1: It means that if we receive packet that can have any source and destination hosts or IP address and is using ARP protocol, the firewall will accept and allow the packet to pass.

Rule 3: It means that if we receive a TCP message from any workstation host(identified by its IP address), intended for any workstation, the firewall will accept and allow the packet to pass.

Rule 4: It means that if we receive a TCP message from any workstation host(identified by its IP address) in the IT subnet, intended for Laptop 1 or 2, the firewall will accept and allow the packet to pass.

Rule 9: It means that if we receive a UDP message from the DNS server, intended for any workstation or laptop, the firewall will accept and allow the packet to pass.

b. Give an example of a packet transfer using the TCP protocol which is dropped according to Rule 10 by the firewall. The packet originates from Workstation3 and is transferred to another host. Write the destination host name and its IP address.

Ans. If we send a TCP message from Workstation3 to the DNSserver(200.20.2.3), it will be dropped according to rule 10 as DNSserver in other rules only accept UDP messages.

Destination Hostname: DNSserver

Destination IP address: 200.20.2.3

c. If you were to run ping between Workstation3 and Printer, do you expect it to work? Explain your answer based on the rules in Table 1.

Ans. I do expect it to work, as we know ping uses ICMP protocol and according to rule 2, any ICMP message between any two hosts is allowed by the firewall.

d. If you were to run iperf between Laptop1 and WebServer, do you expect it to

work? Explain your answer based on the rules in Table 1.

Ans. No, I don’t expect it to work, as we know iperf uses either UDP or TCP protocol. By looking at Table 1 we can see, WebServer only accepts messages in rule 1 and rule where the protocol specifies is ARP and ICMP respectively. Hence the UDP or TCP iperf messages from Laptop1 will be dropped by the firewall.

8. [6 pts] Write pseudocode for your Firewall using the rules in Table 1. Use

the host names instead of the IP addresses in your pseudocode. e.g., use Laptop1

rather than 200.20.2.9, according to the standard names in Table 2. Check

pox/pox/forwarding/l2\_learning.py to identify the key functions needed in your

firewall.

Ans.

def launch():

# launch firewall

connect\_switch()

def connect\_switch():

#connect and listen to switch

check\_pkt(pkt)

def accept\_packet():

#implements accepting packet

def drop\_packet():

drop(pkt)

def isWorkstation(ip):

if(ip==ws1 || ip==ws2 || ip==ws3 || ip==ws4):

return true

return false

def isITWorkstation(ip):

if(ip==ws3 || ip==ws4):

return true

return false

isLaptop(ip):

if(ip==Laptop1 || ip==Laptop2):

return true

return false

isServer2(ip):

if(ip==Serv2):

return true

return false

isDNSserver(ip):

if(ip==ServDNS):

return true

return false

def check\_pkt(pkt):

# main func determines what to do with packet

src\_ip = pkt.sourceIP

dest\_ip = pkt.destIP

protocol = pkt.protocol

# using if-elif to determine what to do with pkt acc to rules also when # #implemented in order also preserves priority

if protocol== ARP:

accept\_packet()

elif protocol == ICMP:

accept\_packet()

elif protocol == TCP && isWorkstaion(src\_ip) && isWorkstation(dest\_ip)

accept\_packet()

elif protocol == TCP && isITWorkstaion(src\_ip) && isLaptop(dest\_ip)

accept\_packet()

elif protocol == TCP && isLaptop(src\_ip) && isITWorkstation(dest\_ip)

accept\_packet()

elif protocol == TCP && isLaptop(src\_ip) && isServer2(dest\_ip)

accept\_packet()

elif protocol == TCP && isServer2(src\_ip) && isLaptop(dest\_ip)

accept\_packet()

elif protocol == UDP && (isWorkstation(src\_ip) || isLaptop(src\_ip)) && isDNSserver(dest\_ip)

accept\_packet()

elif protocol == UDP && isDNSserver(src\_ip) && (isWorkstation(dest\_ip) || isLaptop(dest\_ip))

accept\_packet()

else:

drop\_packet()

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Rule**  **#** | **Src Host** | **SRC IP** | **Dst Host** | **DST IP** | **Protocol** | **Action** |
| 1 |  | Any |  | Any | ARP | accept |
| 2 |  | Any IPv4 |  | Any IPv4 | ICMP | accept |
| 3 | Any Workstation | 200.20.2.4-200.20.2.7 | Any Workstation | 200.20.2.4-200.20.2.7 | TCP | accept |
| 4 | IT Department  workstations | 200.20.2.6  200.20.2.7 | Laptops (1 and 2) | 200.20.2.8  200.20.2.9 | TCP | accept |
| 5 | Laptops (1 and 2) | 200.20.2.8  200.20.2.9 | IT Department  workstations | 200.20.2.6  200.20.2.7 | TCP | accept |
| 6 | Laptops (1 and 2) | 200.20.2.8  200.20.2.9 | Server2 | 200.20.2.1 | TCP | accept |
| 7 | Server2 | 200.20.2.1 | Laptops (1 and 2) | 200.20.2.8  200.20.2.9 | TCP | accept |
| 8 | Any Workstation or Laptop | 200.20.2.4-  200.20.2.9 | DNSserver | 200.20.2.3 | UDP | accept |
| 9 | DNSserver | 200.20.2.3 | Any Workstation or Laptop | 200.20.2.4-  200.20.2.9 | UDP | accept |
| 10 |  | Any |  | Any | - | drop |

9. [20 pts] Code complete - Testing the Basic Firewall

a. pingall:

● Run the pingall command. Which of the firewall rules contributed to the result?

Explain your answer(s).

● Include a timestamped screenshot of the results.