Homework 6

Iman Tabrizian
ECE1508

March 4, 2019

In this lab I learned how to implement a VNF (firewall) to protect a wordpress application. I learned that a lot of technology is required to enable simple software-based network function. I learned the basics about iptables' chains and tables and how to enable SNAT using SNAT.

What I did in this lab was implementation of a software-based firewall application. Through the means of SDN I redirected the traffic to the VM containing the VNF (firewall) and applied made the traffic to go through the snort. In this way I was able to apply simple rules to the traffic and provide better security for the Wordpress application.

Part 1

show the results of the ping tests from h1 to h2 and to h3.

```
ubuntu@netsoft17-h1:~$ ping -c 4 192.168.200.11
PING 192.168.200.11 (192.168.200.11) 56(84) bytes of data.
64 bytes from 192.168.200.11: icmp_seq=1 ttl=64 time=512 ms
64 bytes from 192.168.200.11: icmp_seq=2 ttl=64 time=4.25 ms
64 bytes from 192.168.200.11: icmp_seq=3 ttl=64 time=4.80 ms
64 bytes from 192.168.200.11: icmp_seq=4 ttl=64 time=5.05 ms

--- 192.168.200.11 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3004ms
rtt min/avg/max/mdev = 4.251/131.731/512.819/220.021 ms
```

```
ubuntu@netsoft17-h1:~$ ping -c 4 192.168.200.12
PING 192.168.200.12 (192.168.200.12) 56(84) bytes of data.
64 bytes from 192.168.200.12: icmp_seq=1 ttl=64 time=12.2 ms
64 bytes from 192.168.200.12: icmp_seq=2 ttl=64 time=4.64 ms
64 bytes from 192.168.200.12: icmp_seq=3 ttl=64 time=4.46 ms
64 bytes from 192.168.200.12: icmp_seq=4 ttl=64 time=4.71 ms

--- 192.168.200.12 ping statistics ---
4 packets transmitted, 4 received, 0% packet loss, time 3005ms
rtt min/avg/max/mdev = 4.469/6.509/12.217/3.297 ms
```

Part 2

include screenshots clearly showing the correctness of your implementation: a routing table from either h2 or h3, and the relevant iptables entry for turning h1 into a NAT router

```
sudo iptables -t nat -A POSTROUTING -o eth0 -j MASQUERADE
sudo iptables -P FORWARD ACCEPT

Chain POSTROUTING (policy ACCEPT)
target     prot opt source               destination
MASQUERADE  all  --  172.17.0.0/16        anywhere
MASQUERADE  all  --  anywhere             anywhere

ubuntu@netsoft17-h2:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.200.10  0.0.0.0         UG    0      0        0 br1-internal
10.12.125.0     *               255.255.255.0   U     0      0        0 eth0
172.17.0.0      *               255.255.0.0     U     0      0        0 docker0
192.168.200.0   *               255.255.255.0   U     0      0        0 br1-internal

ubuntu@netsoft17-h3:~$ route
Kernel IP routing table
Destination     Gateway         Genmask         Flags Metric Ref    Use Iface
default         192.168.200.10  0.0.0.0         UG    0      0        0 br1-internal
10.12.125.0     *               255.255.255.0   U     0      0        0 eth0
172.17.0.0      *               255.255.0.0     U     0      0        0 docker0
192.168.200.0   *               255.255.255.0   U     0      0        0 br1-internal
```

Part 3.3

inport

Part 3.4

reject tcp any any -¿ any 80 (content:"inject"; nocase; msg:"accessed forbidden pages!!"; sid:5000000;)

The first of the rule tells the action to be applied to the matching rule in this case *reject*. The second word tells which protocol does this rule apply to. In this case the protocol is *ip*. The third word tells what is the source ip address. In this case source ip address is *any*. The fourth word specifies the source port number. In this case source port number is *any*. The part after arrow specifies the destination specification. Also, the direction of the arrow indicates the direction of interest. The first word after arrow is destination ip address which in this case is *any*. The second word after arrow is the destination port number which in this case is *80*. The part in the paranthesis specifies the specific options regarding the rule. The *nocase* option is used to deactivate any case sensitivity in the content rule. The *msg*

specifies the message to be printed along the packet dump. The *sid* is used for specifying a particular snort rule. This can be used by external plugins to identify snort rules.

```
ubuntu@netsoft17-h3:~$ sudo ovs-ofctl show br1
sudo: unable to resolve host netsoft17-h3
OFPT_FEATURES_REPLY (xid=0x2): dpid:00005e5d9fa3fa49
n_tables:254, n_buffers:256
capabilities: FLOW_STATS TABLE_STATS PORT_STATS QUEUE_STATS ARP_MATCH_IP
actions: output enqueue set_vlan_vid set_vlan_pcp strip_vlan mod_dl_src mod_dl_dst mod_n
 1(br1-internal): addr:de:8d:6b:f2:38:9b
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
 2(netsoft17-h3-ne): addr:72:ff:e1:b2:55:c7
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
 3(snort-1): addr:26:76:28:3c:e4:56
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
 4(snort-2): addr:52:45:c1:f6:32:79
     config:     0
     state:      0
     speed: 0 Mbps now, 0 Mbps max
 LOCAL(br1): addr:5e:5d:9f:a3:fa:49
     config:     PORT_DOWN
     state:      LINK_DOWN
     speed: 0 Mbps now, 0 Mbps max
OFPT_GET_CONFIG_REPLY (xid=0x4): frags=normal miss_send_len=0

ubuntu@netsoft17-h3:~$ sudo ovs-ofctl dump-flows br1
sudo: unable to resolve host netsoft17-h3
NXST_FLOW reply (xid=0x4):
 cookie=0x0, duration=122.956s, table=0, n_packets=145, n_bytes=18294, idle_age=1, in_po
 cookie=0x0, duration=111.446s, table=0, n_packets=0, n_bytes=0, idle_age=111, priority=
 cookie=0x0, duration=101.133s, table=0, n_packets=0, n_bytes=0, idle_age=101, priority=
 cookie=0x0, duration=187.244s, table=0, n_packets=407, n_bytes=38408, idle_age=1, prior
 cookie=0x0, duration=144.414s, table=0, n_packets=0, n_bytes=0, idle_age=144, priority=
 cookie=0x0, duration=6513.161s, table=0, n_packets=3626, n_bytes=388664, idle_age=123,
```