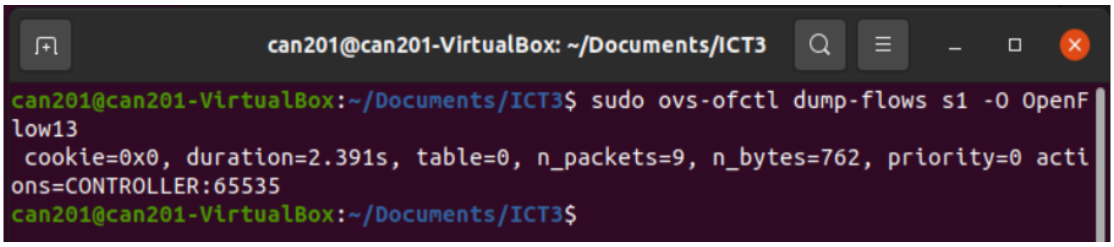
CAN201 In-Class-Test 3

Student Name: Junhao Huang   Student ID: 2256792

Q1. What is the initial flow entry (before attacking) installed in the switch of this lab?

cookie=0x0, duration=2.391s, table=0, n_packets=9, n_bytes=762, priority=0 actions=CONTROLLER:65535



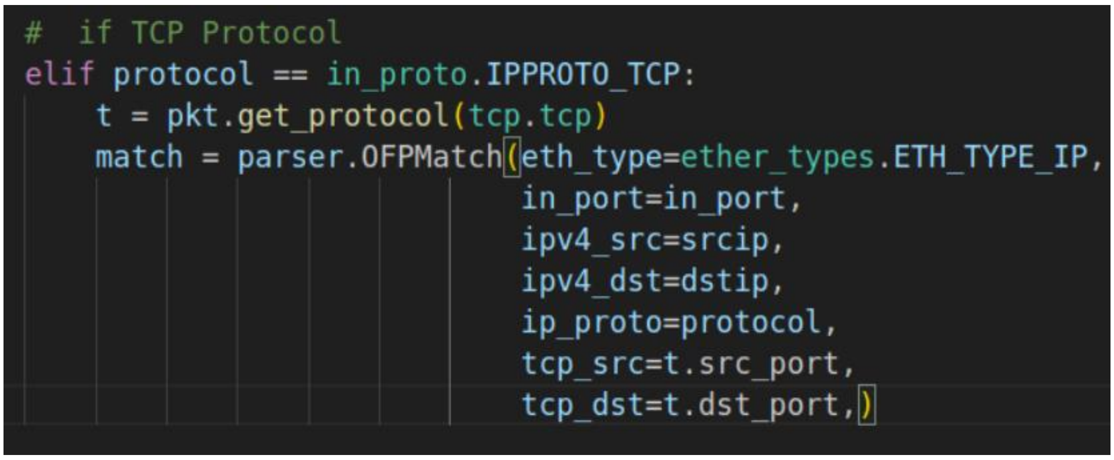
```
can201@can201-VirtualBox:~/Documents/ICT3$ sudo ovs-ofctl dump-flows s1 -O OpenF
low13
 cookie=0x0, duration=2.391s, table=0, n_packets=9, n_bytes=762, priority=0 acti
ons=CONTROLLER:65535
can201@can201-VirtualBox:~/Documents/ICT3$
```

Q2. Explain what this initial flow entry does.

According to the priority which is 0, this means that it is the last option that will only be executed when all other flow table entries fail to match the packet. Therefore, initially, it is the only rule existing in the flow table to cope with the coming packets which are not matched by other flow entries will be sent to the controller. This configuration allows the controller to perform further processing of those unknown packets.

Q3. Show the flow 'match' rule (in the lab11.py) used to cope with the above flooding traffic.

According to the flooding command,the -S parameter represents the SYN flag of TCP and is used for the initial synchronization when a TCP connection is established.
This code defines a flow table entry that matches whether the incoming packet is TCP or not, and checks the source IP, destination IP, source port, and destination port.



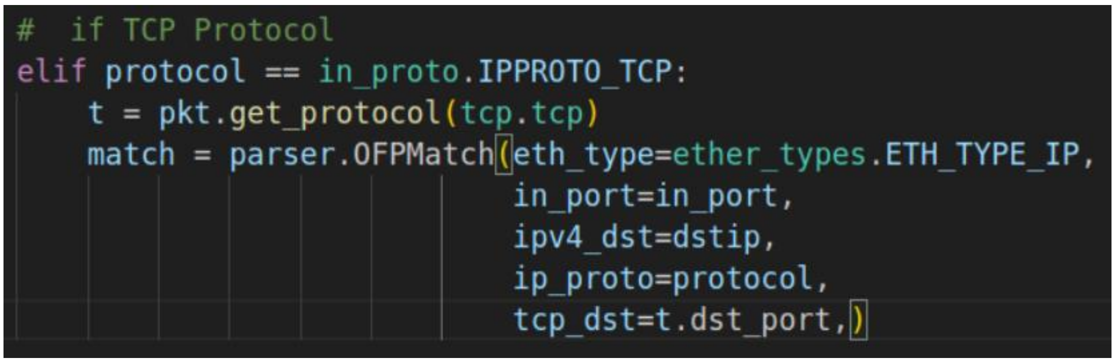
```
#  if TCP Protocol
elif protocol == in_proto.IPPROTO_TCP:
    t = pkt.get_protocol(tcp.tcp)
    match = parser.OFPMatch(eth_type=ether_types.ETH_TYPE_IP,
                            in_port=in_port,
                            ipv4_src=srcip,
                            ipv4_dst=dstip,
                            ip_proto=protocol,
                            tcp_src=t.src_port,
                            tcp_dst=t.dst_port,)
```

Q4. Explain how the flooding command exhausts the switch's flow table?

If we do not modify the matching rules in the source code, and executing the flooding command in the Mininet terminal, it will send a large number of packets with random source IP and source port, attempting to initiate new connections. Since each coming packet requiring the creation of new flow table entry in the switch, the attack can quickly exhaust the switch's flow table space by filling with certain types of connections. As the flow table fills up, the switch will be unable to create flow table entries for new connections, and impacting normal network traffic. Therefore, by

Q5. Revise the flow 'match' rule in the original lab11.py file to solve the vulnerability.

By removing `ipv4_src=srcip` and `tcp_src=t.src_port` in the TCP matching rules in the source code, the switch will no longer match TCP packets based on the source IP address and source port, but only match the destination IP address and destination port.



```
#  if TCP Protocol
elif protocol == in_proto.IPPROTO_TCP:
    t = pkt.get_protocol(tcp.tcp)
    match = parser.OFPMatch(eth_type=ether_types.ETH_TYPE_IP,
                            in_port=in_port,
                            ipv4_dst=dstip,
                            ip_proto=protocol,
                            tcp_dst=t.dst_port,)
```

End of the report