File: hacking/arp-poisoning.pdf

File owner: Lysander Miller

PERSON-IN-THE-MIDDLE VIA ARP SPOOFING

EXECUTION:

1. **What is Kali's main interface's MAC address? (The main interface is probably called eth0, but check ifconfig to be sure.):**

   00:0c:29:c4:55:be

2. **What is Kali's main interface's IP address?**

   192.168.152.128

3. **What is Metasploitable's main interface's MAC address?**

   00:0c:29:5a:40:b4

4. **What is Metasploitable's main interface's IP address?**

   192.168.152.129

5. **Show Kali's routing table. (Use "netstat -r" to see it with symbolic names, or "netstat -rn" to see it with numerical addresses.)**

   netstat -r gave me the following output:

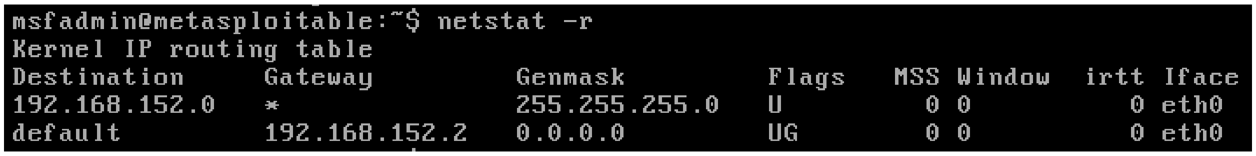
   ```
   Kernel IP routing table

   Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface

   default         192.168.152.2   0.0.0.0         UG        0 0          0 eth0

   192.168.152.0   0.0.0.0         255.255.255.0   U         0 0          0 eth0
   ```

6. **Show Kali's ARP cache. (Use "arp" or "arp -n".)**

arp got me the following output:

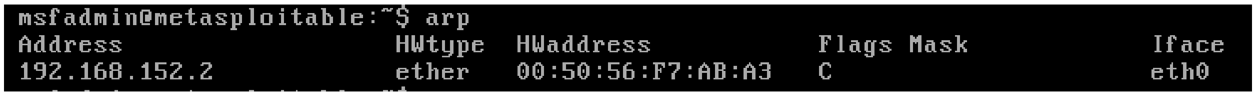
```
Address              HWtype  HWaddress          Flags Mask          Iface

192.168.152.2          ether   00:50:56:f7:ab:a3   C                  eth0
```

7. **Show Metasploitable's routing table.**



```
msfadmin@metasploitable:~$ netstat -r
Kernel IP routing table
Destination     Gateway         Genmask         Flags   MSS Window  irtt Iface
192.168.152.0   *               255.255.255.0   U         0 0          0 eth0
default         192.168.152.2   0.0.0.0         UG        0 0          0 eth0
```

8. **Show Metasploitable's ARP cache.**



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.152.2            ether   00:50:56:F7:AB:A3   C                     eth0
```

9. **Suppose the user of Metasploitable wants to get the CS338 sandbox page via the command "curl http://cs338.jeffondich.com/". To which MAC address should Metasploitable send the TCP SYN packet to get the whole HTTP query started? Explain why.**

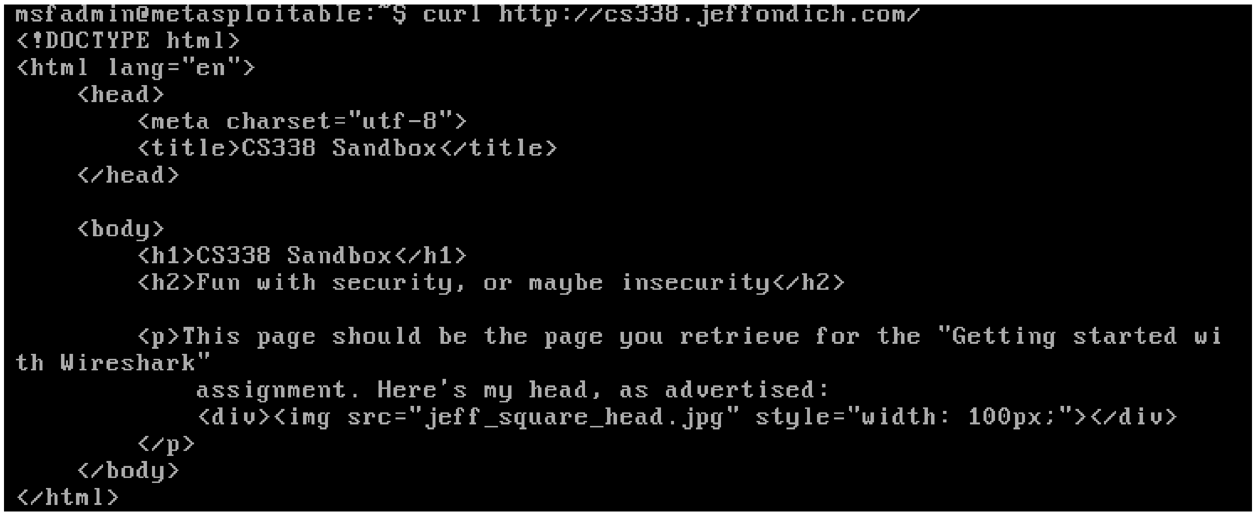
   Metasploitable should send the TCP SYN packet to MAC address 00:50:56:F7:AB:A3. In order to get this, I did the below:

   a) I saw that the only gateway on Metasploitable's routing table was 192.168.152.2. So, I knew Metasploitable would have to send the packet to the MAC address associated with 192.168.152.2.
   b) By looking at the ARP cache, I saw that the MAC address associated with IP address 192.168.152.2 is 00:50:56:F7:AB:A3.

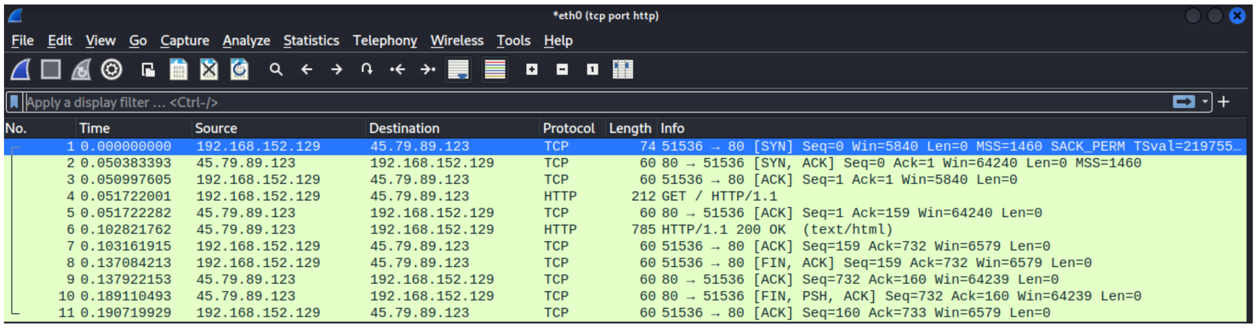
10. **Fire up Wireshark on Kali. Start capturing packets for "tcp port http". On Metasploitable, execute "curl http://cs338.jeffondich.com/". On Kali, stop**

capturing. Do you see an HTTP response on Metasploitable? Do you see any captured packets in Wireshark on Kali?

I do see an HTTP response on Metasploitable. I also see captured packets in Wireshark on Kali.



```
msfadmin@metasploitable:~$ curl http://cs338.jeffondich.com/
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>CS338 Sandbox</title>
    </head>

    <body>
        <h1>CS338 Sandbox</h1>
        <h2>Fun with security, or maybe insecurity</h2>

        <p>This page should be the page you retrieve for the "Getting started wi
th Wireshark"
            assignment. Here's my head, as advertised:
            <div><img src="jeff_square_head.jpg" style="width: 100px;"></div>
        </p>
    </body>
</html>
```



*eth0 (tcp port http)
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

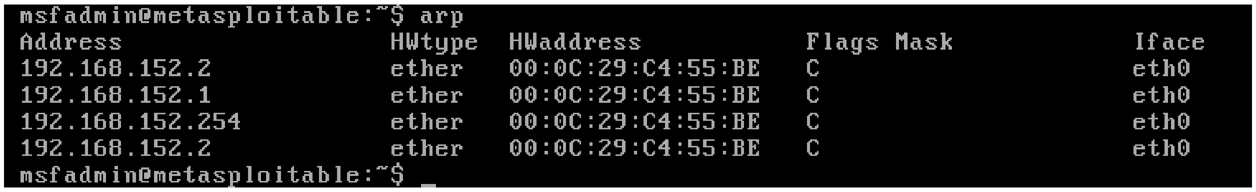
```
No.  Time         Source           Destination      Protocol Length Info
   1 0.000000000  192.168.152.129  45.79.89.123     TCP        74 51536 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSval=219755…
   2 0.050383393  45.79.89.123     192.168.152.129  TCP        60 80 → 51536 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
   3 0.050997605  192.168.152.129  45.79.89.123     TCP        60 51536 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
   4 0.051722001  192.168.152.129  45.79.89.123     HTTP      212 GET / HTTP/1.1
   5 0.051722282  45.79.89.123     192.168.152.129  TCP        60 80 → 51536 [ACK] Seq=1 Ack=159 Win=64240 Len=0
   6 0.102821762  45.79.89.123     192.168.152.129  HTTP      785 HTTP/1.1 200 OK  (text/html)
   7 0.103161915  192.168.152.129  45.79.89.123     TCP        60 51536 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0
   8 0.137084213  192.168.152.129  45.79.89.123     TCP        60 51536 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0
   9 0.137922153  45.79.89.123     192.168.152.129  TCP        60 80 → 51536 [ACK] Seq=732 Ack=160 Win=64239 Len=0
  10 0.189110493  45.79.89.123     192.168.152.129  TCP        60 80 → 51536 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0
  11 0.190719929  192.168.152.129  45.79.89.123     TCP        60 51536 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0
```

11. **Now, it's time to be Mal (who will, today, merely eavesdrop). Use Ettercap to do ARP spoofing (also known as ARP Cache Poisoning) with Metasploitable as your target. There are many online tutorials on how to do this (here's one). Find one you like, and start spoofing your target. NOTE: most of these tutorials are showing an old user interface for Ettercap, which may make them confusing. The steps you're trying to take within Ettercap are:**
    - **Start sniffing (*not* bridged sniffing) on eth0**
    - **Scan for Hosts**
    - **View the Hosts list**
    - **Select your Metasploit VM from the Host List**
    - **Add that host as Target 1**
    - **Start ARP Poisoning (including Sniff Remote Connections)**

- **Do your stuff with wireshark and Metasploitable**
- **Stop ARP Poisoning**

I'll post some screenshots on Slack of how I got Ettercap to do these things. Honestly, I don't know who redesigned this user interface to make it so much harder to do things, but they did. (Common enough in the Linux UI world.)
So, to wrap up this step: start the ARP poisoning. You will keep the ARP poisoning attack active until you are done with your AITM attack. (Realistically, you will probably start and stop ARP poisoning several times as you gradually figure out what's going on while doing the steps below.)

12. **Show Metasploitable's ARP cache. How has it changed?**



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.152.2            ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.1            ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.254          ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.2            ether   00:0C:29:C4:55:BE   C                     eth0
msfadmin@metasploitable:~$ _
```

Before I began the ARP poisoning, Metasploitable's ARP cache just contained address 192.168.152.2 with MAC address 00:50:56:F7:AB:A3. Now, address 192.168.152.2 appears twice in the ARP and, each time, has a MAC address of 00:0C:29:C4:55:BE (Kali's MAC address). Also, the ARP now has additional addresses (192.168.152.1 and 192.168.152.254) each with the MAC address 00:0C:29:C4:55:BE.

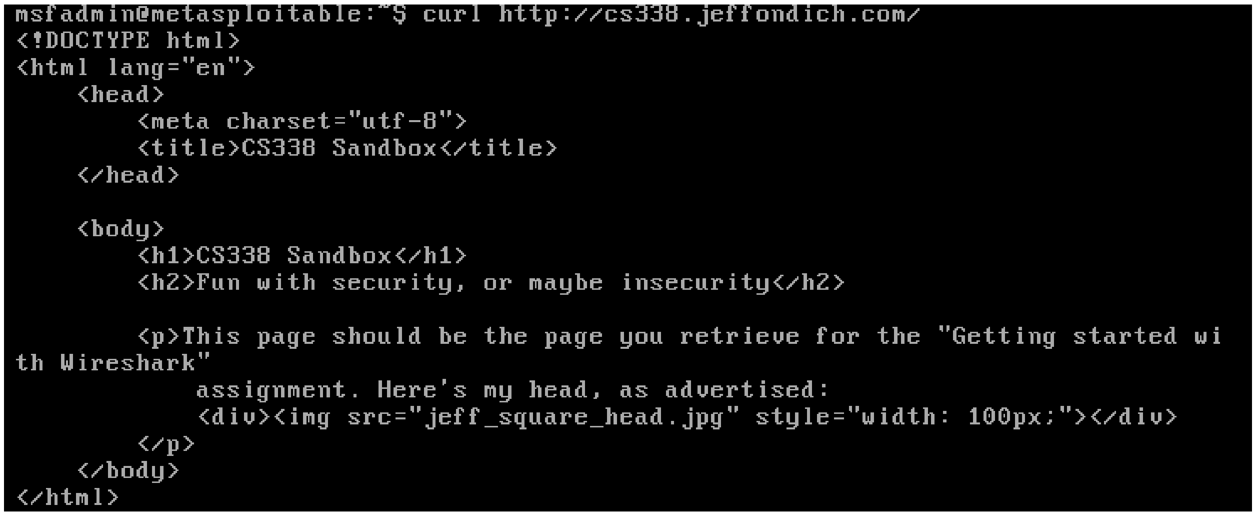
13. **Without actually doing it yet, predict what will happen if you execute "curl http://cs338.jeffondich.com/" on Metasploitable now. Specifically, to what MAC address will Metasploitable send the TCP SYN packet? Explain why.**

I predict that Metasploitable will send the TCP SYN packet to MAC address 00:0C:29:C4:55:BE (Kali's MAC address). I think this is what will happen because, after Metasploitable gets the IP address 192.168.152.2 from the routing table, it will go to the ARP cache and see that the MAC address associated with IP address 192.168.152.2 is 00:0C:29:C4:55:BE. So, Metasploitable will send the TCP SYN packet to the MAC address 00:0C:29:C4:55:BE.

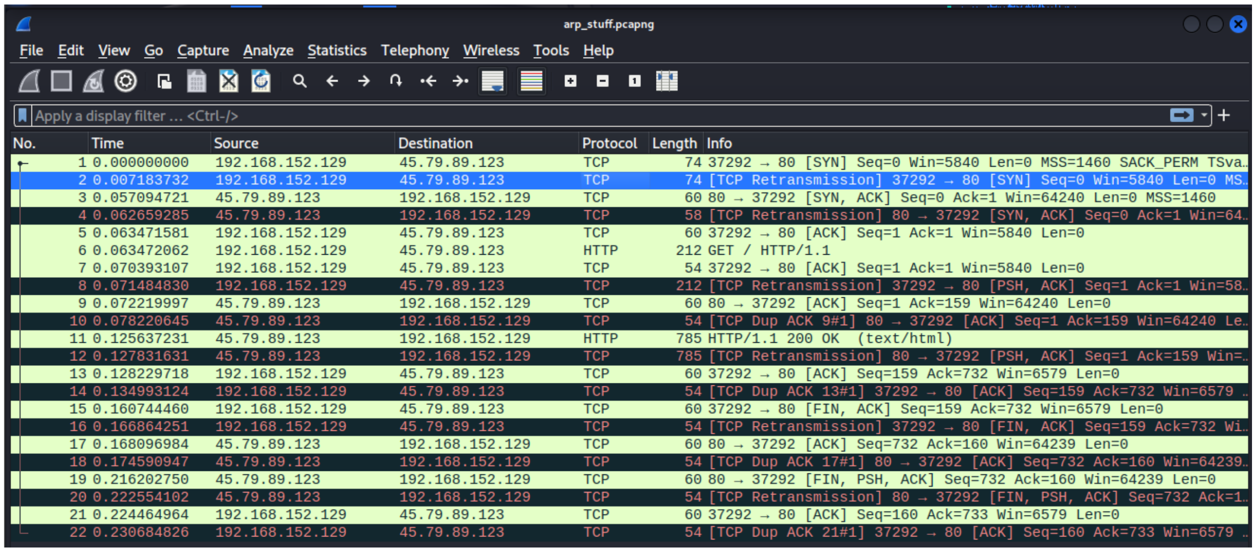
14. **Start Wireshark capturing "tcp port http" again.**
15. **Execute "curl http://cs338.jeffondich.com/" on Metasploitable. On Kali, stop capturing. Do you see an HTTP response on Metasploitable? Do you see captured**

packets in Wireshark? Can you tell from Kali what messages went back and forth between Metasploitable and cs338.jeffondich.com?

I do see an HTTP response on Metasploitable and captured packets in Wireshark.



```
msfadmin@metasploitable:~$ curl http://cs338.jeffondich.com/
<!DOCTYPE html>
<html lang="en">
    <head>
        <meta charset="utf-8">
        <title>CS338 Sandbox</title>
    </head>

    <body>
        <h1>CS338 Sandbox</h1>
        <h2>Fun with security, or maybe insecurity</h2>

        <p>This page should be the page you retrieve for the "Getting started wi
th Wireshark"
            assignment. Here's my head, as advertised:
            <div><img src="jeff_square_head.jpg" style="width: 100px;"></div>
        </p>
    </body>
</html>
```



arp_stuff.pcapng
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter … <Ctrl-/>

```
No.  Time         Source           Destination      Protocol Length Info
   1 0.000000000  192.168.152.129  45.79.89.123     TCP       74 37292 → 80 [SYN] Seq=0 Win=5840 Len=0 MSS=1460 SACK_PERM TSva…
   2 0.007183732  192.168.152.129  45.79.89.123     TCP       74 [TCP Retransmission] 37292 → 80 [SYN] Seq=0 Win=5840 Len=0 MS…
   3 0.057094721  45.79.89.123     192.168.152.129  TCP       60 80 → 37292 [SYN, ACK] Seq=0 Ack=1 Win=64240 Len=0 MSS=1460
   4 0.062659285  45.79.89.123     192.168.152.129  TCP       58 [TCP Retransmission] 80 → 37292 [SYN, ACK] Seq=0 Ack=1 Win=64…
   5 0.063471581  192.168.152.129  45.79.89.123     TCP       60 37292 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
   6 0.063472062  192.168.152.129  45.79.89.123     HTTP     212 GET / HTTP/1.1
   7 0.070393107  192.168.152.129  45.79.89.123     TCP       54 37292 → 80 [ACK] Seq=1 Ack=1 Win=5840 Len=0
   8 0.071484830  192.168.152.129  45.79.89.123     TCP      212 [TCP Retransmission] 37292 → 80 [PSH, ACK] Seq=1 Ack=1 Win=58…
   9 0.072219997  45.79.89.123     192.168.152.129  TCP       60 80 → 37292 [ACK] Seq=1 Ack=159 Win=64240 Len=0
  10 0.078220645  45.79.89.123     192.168.152.129  TCP       54 [TCP Dup ACK 9#1] 80 → 37292 [ACK] Seq=1 Ack=159 Win=64240 Le…
  11 0.125637231  45.79.89.123     192.168.152.129  HTTP     785 HTTP/1.1 200 OK  (text/html)
  12 0.127831631  45.79.89.123     192.168.152.129  TCP      785 [TCP Retransmission] 80 → 37292 [PSH, ACK] Seq=1 Ack=159 Win=…
  13 0.128229718  192.168.152.129  45.79.89.123     TCP       60 37292 → 80 [ACK] Seq=159 Ack=732 Win=6579 Len=0
  14 0.134993124  192.168.152.129  45.79.89.123     TCP       54 [TCP Dup ACK 13#1] 37292 → 80 [ACK] Seq=159 Ack=732 Win=6579 …
  15 0.160744460  192.168.152.129  45.79.89.123     TCP       60 37292 → 80 [FIN, ACK] Seq=159 Ack=732 Win=6579 Len=0
  16 0.166864251  192.168.152.129  45.79.89.123     TCP       54 [TCP Retransmission] 37292 → 80 [FIN, ACK] Seq=159 Ack=732 Wi…
  17 0.168096984  45.79.89.123     192.168.152.129  TCP       60 80 → 37292 [ACK] Seq=732 Ack=160 Win=64239 Len=0
  18 0.174590947  45.79.89.123     192.168.152.129  TCP       54 [TCP Dup ACK 17#1] 80 → 37292 [ACK] Seq=732 Ack=160 Win=64239…
  19 0.216202750  45.79.89.123     192.168.152.129  TCP       60 80 → 37292 [FIN, PSH, ACK] Seq=732 Ack=160 Win=64239 Len=0
  20 0.222554102  45.79.89.123     192.168.152.129  TCP       54 [TCP Retransmission] 80 → 37292 [FIN, PSH, ACK] Seq=732 Ack=1…
  21 0.224464964  192.168.152.129  45.79.89.123     TCP       60 37292 → 80 [ACK] Seq=160 Ack=733 Win=6579 Len=0
  22 0.230684826  192.168.152.129  45.79.89.123     TCP       54 [TCP Dup ACK 21#1] 37292 → 80 [ACK] Seq=160 Ack=733 Win=6579 …
```

From Kali, I can tell that Metasploitable set up a connection to cs338.jeffondich.com before sending a GET request to cs338.jeffondich.com requesting /. cs338.jeffondich.com proceeded to send the html text associated with /. Then, the two closed the connection.

16. **Explain in detail what happened. How did Kali change Metasploitable's ARP cache? (If you want to watch the attack in action, try stopping the AITM attack by**

selecting "Stop mitm attack(s)" from Ettercap's Mitm menu, starting a Wireshark capture for "arp", and restarting the ARP poisoning attack in Ettercap.)

After we started the ARP poisoning attack in Ettercap, Kali proceeded to send out ARP messages which basically said “the mac address for IP address ___ is 00:0c:29:c4:55:be.” So, when Metasploitable started looking for the MAC addresses associated with an IP address, no matter what the IP address was, Metasploitable heard that the MAC address was 00:0c:29:c4:55:be and, thus, proceeded to store this MAC address in its ARP cache.

17. **If you wanted to design an ARP spoofing detector, what would you have your detector do? (As you think about this, consider under what circumstances your detector might generate false positives.)**

    One could design an ARP spoofing detector that checks if, within an ARP cache, multiple different IP addresses are associated with the same MAC address. However, this could generate false positives if, for example, someone is using ARP spoofing techniques for legitimate purposes. For example, if a server goes down, a backup server might send out gratuitous ARP requests (like those used in ARP poisoning attacks) in order to redirect traffic from the downed server to the backup server (source: https://en.wikipedia.org/wiki/ARP_spoofing).

SYNTHESIS

1. **Explain in detail Mal's strategy for intercepting the traffic between Alice and Bob. Use any of your observations from the Execution section to clarify your explanation. But be careful not to just reiterate all the steps, and not to focus on specific tools. (For example, I would not expect you to refer to Ettercap in this explanation, since it is merely one of many available tools for generating suitable ARP messages.) Your goal here is to explain to a technical audience (e.g., other CS majors who have not studied security) what Mal is up to, and how ARP cache poisoning works.**

    When Alice wants to send a packet to Bob, she must first check the DNS system to determine the IP address of bob.com. Let’s say she gets 123.123.123.123.

    Now, she can’t just send her packet directly to 123.123.123.123. Instead, she must send her packet to another IP address which will then send it along to another IP address and so on until the packet eventually gets to Bob. So, in order to figure out which IP address Alice should send her packet to (thus starting the bucket-brigade of packet-passing), she must look at her routing table. Suppose she gets the IP address 10.10.10.10

Now, in order to send the packet to IP address 10.10.10.10, Alice needs the MAC address of IP address 10.10.10.10. In order to find the MAC address of IP address 10.10.10.10, Alice can check her ARP cache. If it's not in her ARP cache, she can send out an ARP request basically asking "what is the MAC address of IP address 10.10.10.10?" Eventually, she'll get the MAC address of IP address 10.10.10.10. Let's say the MAC address is 01:23:45:67:89:ab.

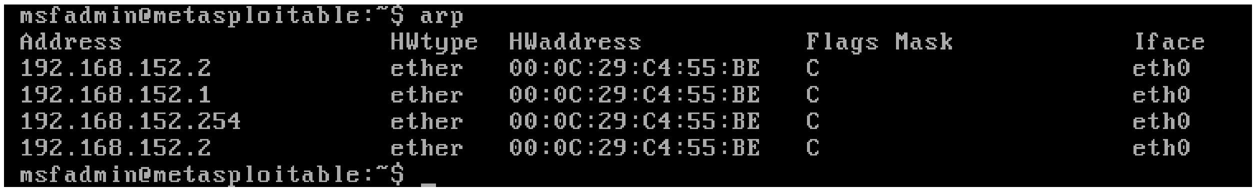
Now, Alice will proceed to send 01:23:45:67:89:ab her packet for IP address 10.10.10.10. Eventually, Alice's packet will find its way to Bob.

However, in an ARP poisoning attack, Mal will do one of two things:

1) Mal will send out something called gratuitous ARP requests. These are unsolicited broadcast messages which basically say "the mac address for IP address ___ is ____." These messages will have different IP addresses from one another but the MAC addresses will always be Mal's mac address.

2) Mal will wait until someone sends out an ARP request asking "what is the MAC address of IP address ____?" Then, Mal will respond that the MAC address of that IP address is Mal's MAC address.

As a result of Mal's actions, Alice's ARP cache will end up looking something like the image below:



```
msfadmin@metasploitable:~$ arp
Address                  HWtype  HWaddress           Flags Mask            Iface
192.168.152.2            ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.1            ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.254          ether   00:0C:29:C4:55:BE   C                     eth0
192.168.152.2            ether   00:0C:29:C4:55:BE   C                     eth0
msfadmin@metasploitable:~$ _
```

As you can see, all of the IP addresses have the same HWaddress (MAC address). Thus, when Alice goes to send a message to Bob at bob.com, after she gets the IP address for bob.com and the IP addresses of the first member of the packet-passing bucket brigade, she looks in her ARP cache and gets the MAC address associated with Mal's machine. Thus, she sends her packet to Mal's machine. This allows Mal to read Alice's message, modify Alice's message, or not send Alice's message along (thus preventing it from getting to Bob).

2. **From Alice's perspective, is this attack detectable? If not, why not? If so, how would Alice's setup need to change to detect the attack?**

If she checks her ARP cache, Alice will see that multiple IP addresses are associated with the same MAC address. This could allow Alice to detect the ARP poisoning. However, as I discussed earlier, sometimes ARP spoofing techniques are used for legitimate purposes. For example, if a server goes down, a backup server might send out gratuitous ARP requests (like those used in ARP poisoning attacks) in order to redirect traffic from the downed server to the backup server (source: https://en.wikipedia.org/wiki/ARP_spoofing). Thus, Alice might get some false positives.

Additionally, most people don't often check their ARP cache. Thus, Alice might want to change her setup so that it automatically checks the ARP cache for her.

3. **From Bob's perspective, is this attack detectable?**

   It is not.

4. **Could Alice or Bob detect and/or prevent this attack if the website in question was using HTTPS instead of HTTP? Explain.**

   If the website in question was using HTTPS instead of HTTP, Mal could still get packets but wouldn't be able to read or modify them. After all, as we discussed in class, HTTPS prevents MITM attacks.