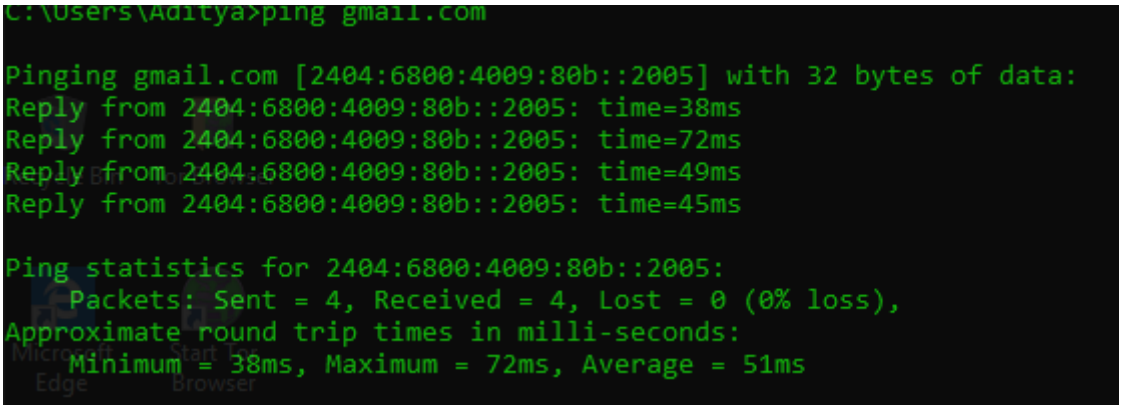
Name: - Bhalsod Aditya M. **Enrollment No: - 175690693001**
Subject: - Computer Networking (CN) **Subject Code: - 4649301**

Lab 2:

Objective: To make the students aware about and learn the detailed use of the following OS level TCP/IP diagnostic and troubleshooting commands: ping, ns lookup:

Ping: The ping command is a command prompt command used to test the ability of the source computer to reach a specified destination computer. The ping command is usually used as a simple way to verify that a computer can communicate over the network with another computer or network device.

The ping command operates by sending *Internet Control Message Protocol (ICMP) Echo Request* messages to the destination computer and waiting for a response. How many of those responses are returned, and how long it takes for them to return, are the two major pieces of information that the ping command provides.



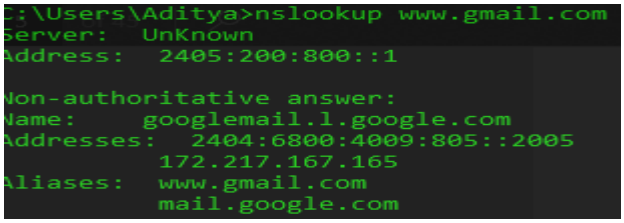
Item	**Explanation**
-t	Using this option will ping the *target* until you force it to stop by using ctrl+c.
-a	This ping command option will resolve, if possible, the hostname of an IP address *target*.
-n (count)	This option sets the number of ICMP Echo Requests to send, from 1 to 4294967295. The ping command will send 4 by default if **-n** isn't used.
-l *size*	Use this option to set the size, in bytes, of the echo request packet from 32 to 65,527. The ping command will send a 32-byte echo request if you don't use the **-l** option.
-f	Use this ping command option to prevent ICMP Echo Requests from being fragmented by routers between you and the *target*. The **-f** option is most often used to troubleshoot Path Maximum Transmission Unit (PMTU) issues.
-i *TTL*	This option sets the Time to Live (TTL) value, the maximum of which is 255.
-v *TOS*	This option allows you to set a Type of Service (TOS) value. Beginning in Windows 7, this option no longer functions but still exists for compatibility reasons.
-r *count*	Use this ping command option to specify the number of hops between your computer and the *target* computer or device that you'd like to be recorded and displayed. The maximum value for *count* is 9, so use the tracert command instead if you're interested in viewing all the hops between two devices.
-s *count*	Use this option to report the time, in Internet Timestamp format, that each echo request is received and echo reply is sent. The maximum value for *count* is 4, meaning that only the first four hops can be time stamped.
-w *timeout*	Specifying a *timeout* value when executing the ping command adjusts the amount of time, in milliseconds, that ping waits for each reply. If you don't use the **-w** option, the default timeout value of 4000 is used, which is 4 seconds.
-R	This option tells the ping command to trace the round trip path.

-S *src addr*	Use this option to specify the source address.
-p	Use this switch to ping a hyper-v *Network Virtualization* provider address.
-4	This forces the ping command to use IPv4 only but is only necessary if *target* is a hostname and not an IP address.
-6	This forces the ping command to use IPv6 only but as with the **-4**option, is only necessary when pinging a hostname.
target	This is the destination you wish to ping, either an IP address or a hostname.
/?	Use the help switch with the ping command to show detailed help about the command's several options.

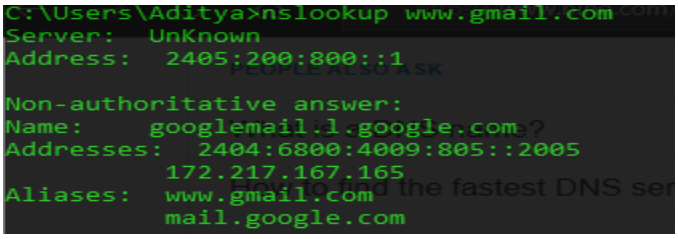
Ns lookup: To illustrate the use of nslookup we are going to use it to:
Find the IP address of a host.
Find the domain name of an IP address.
Find mail servers for a domain.
These are probably the most common usage scenarios.

Find the IP address of a host.

To find the ip address of a host e.g. www.google.com type: **nslookup** www.google.com at a command prompt.

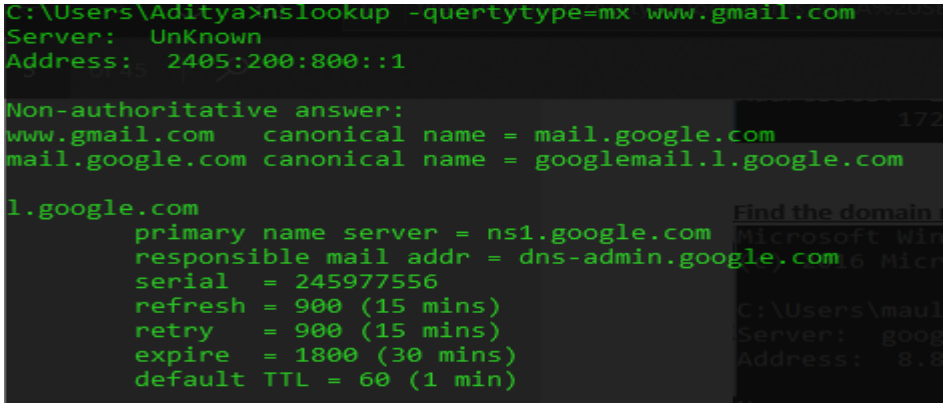


Find the domain name of an IP address.



Find mail servers for a domain.

Type **nslookup -querytype=mx domain name**



Lab 3:

Objective: To make the students aware about and learn the detailed use of the following OS level TCP/IP diagnostic and troubleshooting commands: ipconfig, arp, netstat, tracert, telnet:

Ipcofig:
Ipconfig is a DOS utility that can be used from MS-DOS and the Windows Command Line to display the network settings currently assigned and given by a network. This command can be utilized to verify a network connection as well as to verify your network settings.

/all	Display full configuration information.
/release	Release the IPV4 address for the specified adapter.
/release6	Release the IPV6 address for the specified adapter.
/renew	Renew the IPv4 address for the specified adapter.
/renew6	Renew the IPv6 address for the specified adapter.
/flushdns	Purges the DNS Resolver cache.
/registerdns	Refreshes all DHCP leases and re-registers DNS names.
/displaydns	Display the contents of the DNS Resolver cache.
/showclassid	Displays all the DHCP class IDs allowed for adapter.
/setclassid	Modifies the DHCP class id.
/showclassid6	Displays all the IPv6 DHCP class IDs allowed for an adapter.
/setclassid6	Modifies the IPv6 DHCP class id.

Eg.

```
C:\Users\Aditya>ipconfig /all

Windows IP Configuration

   Host Name . . . . . . . . . . . . : Aditya
   Primary Dns Suffix  . . . . . . . :
   Node Type . . . . . . . . . . . . : Hybrid
   IP Routing Enabled. . . . . . . . : No
   WINS Proxy Enabled. . . . . . . . : No

Ethernet adapter Ethernet:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek PCIe FE Family Controller
   Physical Address. . . . . . . . . : C8-5B-76-67-6B-B7
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Ethernet adapter Npcap Loopback Adapter:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Npcap Loopback Adapter
   Physical Address. . . . . . . . . : 02-00-4C-4F-4F-50
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::30db:8113:f467:e5ba%6(Preferred)
   Autoconfiguration IPv4 Address. . : 169.254.229.186(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.0.0
```

```
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 822214732
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-52-93-4C-C8-5B-76-67-6B-B7
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
                                       fec0:0:0:ffff::3%1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Ethernet adapter VirtualBox Host-Only Network:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : VirtualBox Host-Only Ethernet Adapter
   Physical Address. . . . . . . . . : 0A-00-27-00-00-07
   DHCP Enabled. . . . . . . . . . . : No
   Autoconfiguration Enabled . . . . : Yes
   Link-local IPv6 Address . . . . . : fe80::3d3c:4d14:4a35:54a5%7(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.56.1(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
   DHCPv6 IAID . . . . . . . . . . . : 1024065575
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-52-93-4C-C8-5B-76-67-6B-B7
   DNS Servers . . . . . . . . . . . : fec0:0:0:ffff::1%1
                                       fec0:0:0:ffff::2%1
                                       fec0:0:0:ffff::3%1
   NetBIOS over Tcpip. . . . . . . . : Enabled

Wireless LAN adapter Local Area Connection* 1:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter
   Physical Address. . . . . . . . . : 7A-45-61-EB-52-B9
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter Local Area Connection* 2:

   Media State . . . . . . . . . . . : Media disconnected
   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Microsoft Wi-Fi Direct Virtual Adapter #2
   Physical Address. . . . . . . . . : 78-45-61-EB-52-B9
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes

Wireless LAN adapter WiFi:

   Connection-specific DNS Suffix  . :
   Description . . . . . . . . . . . : Realtek RTL8723BE Wireless LAN 802.11n PCI-E NIC
   Physical Address. . . . . . . . . : 78-45-61-EB-52-B9
   DHCP Enabled. . . . . . . . . . . : Yes
   Autoconfiguration Enabled . . . . : Yes
   IPv6 Address. . . . . . . . . . . : 2405:204:8585:f1ce:4:6adb:611f:c5c3(Preferred)
   Temporary IPv6 Address. . . . . . : 2405:204:8585:f1ce:2c77:3f80:8d0c:d6f7(Preferred)
```

```
   Link-local IPv6 Address . . . . . : fe80::4:6adb:611f:c5c3%3(Preferred)
   IPv4 Address. . . . . . . . . . . : 192.168.43.126(Preferred)
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Lease Obtained. . . . . . . . . . : Tuesday, April 30, 2019 11:37:46 PM
   Lease Expires . . . . . . . . . . : Wednesday, May 1, 2019 1:35:51 AM
   Default Gateway . . . . . . . . . : fe80::5a8:30b9:f5b:cfc5%3
                                       192.168.43.1
   DHCP Server . . . . . . . . . . . : 192.168.43.1
   DHCPv6 IAID . . . . . . . . . . . : 58213729
   DHCPv6 Client DUID. . . . . . . . : 00-01-00-01-24-52-93-4C-C8-5B-76-67-6B-B7
   DNS Servers . . . . . . . . . . . : 2405:200:800::1
                                       192.168.43.1
                                       2405:200:800::1
   NetBIOS over Tcpip. . . . . . . . : Enabled

C:\Users\Aditya>
```

arp:

arp displays and modifies entries in the Address Resolution Protocol (ARP) cache, which contains one or more tables that are used to store IP addresses and their resolved Ethernet or Token Ring physical addresses. There is a separate table for each Ethernet or Token Ring network adapter installed on your computer. Used without parameters, **arp**displays help.

You can use the **arp** command to view and modify the ARP table entries on the local computer. This may display all the known connections on your local aream network segment (if they have been active and in the cache). The **arp** command is useful for viewing the ARP cache and resolving address resolution problems.

Syntax (Inet means Internet address)

arp [**-a** [*InetAddr*] [**-N** *IfaceAddr*]] [**-g** [*InetAddr*] [**-N** *IfaceAddr*]] [**-d** *InetAddr* [*IfaceAddr*]] [**-s** *InetAddr EtherAddr* [*IfaceAddr*]]

Here are the switch definitions:

-a [*InetAddr*] [-N*IfaceAddr*] : Displays current ARP cache tables for all interfaces. To display the ARP cache entry for a specific IP address, use **arp -a** with the *InetAddr* parameter, where *InetAddr* is an IP address. To display the ARP cache table for a specific interface, use the **-N** *IfaceAddr* parameter where *IfaceAddr* is the IP address assigned to the interface. The **-N** parameter is case-sensitive.

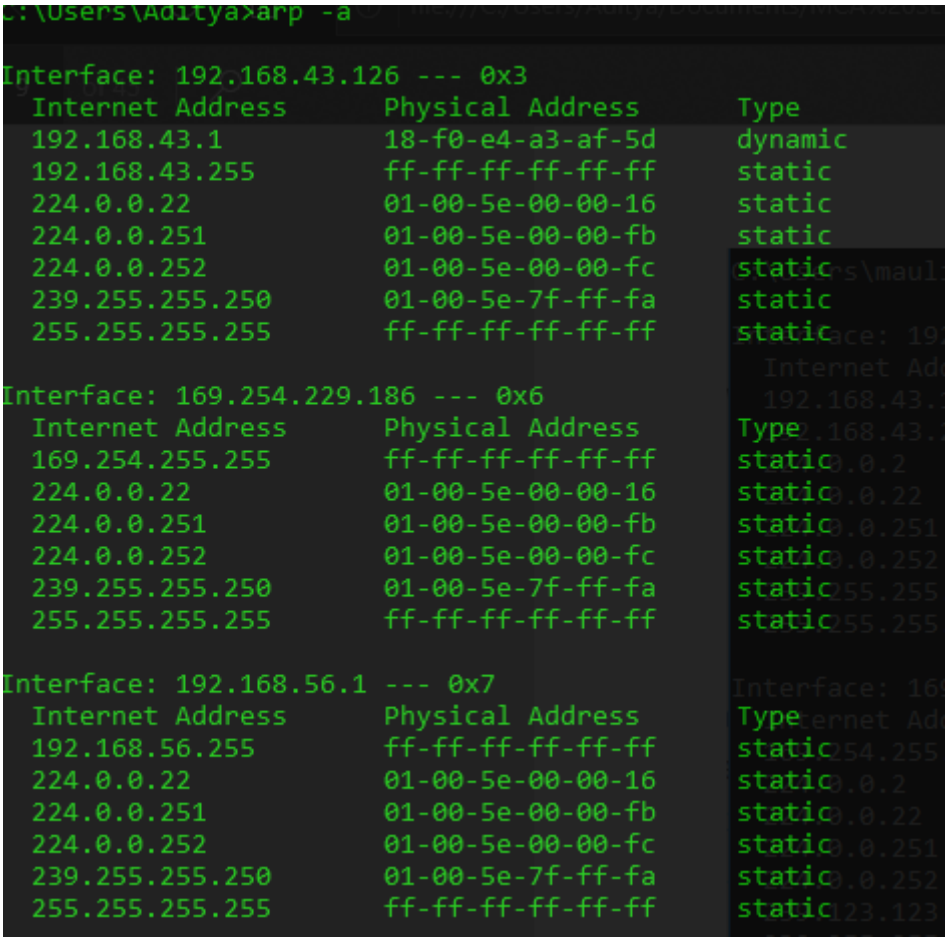
-g [*InetAddr*] [-N*IfaceAddr*] : Identical to **-a**.

-d*InetAddr* [*IfaceAddr*] : Deletes an entry with a specific IP address, where *InetAddr* is the IP address. To delete an entry in a table for a specific interface, use the *IfaceAddr* parameter where *IfaceAddr* is the IP address assigned to the interface. To delete all entries, use the asterisk (*) wildcard character in place of *InetAddr*. So "arp -d *" will flush your ARP cache.

-s*InetAddr EtherAddr* [*IfaceAddr*] : Adds a static entry to the ARP cache that resolves the IP address *InetAddr* to the physical address *EtherAddr*. To add a static ARP cache entry to the table for a specific interface, use the *IfaceAddr*parameter where *IfaceAddr* is an IP address assigned to the interface. **/?:** Displays help at the command prompt.

Using arp on Windows

To run the arp command in Windows click START> RUN> CMD. Now enter 'arp -a' at the > prompt:



```
C:\Users\Aditya>arp -a

Interface: 192.168.43.126 --- 0x3
  Internet Address      Physical Address      Type
  192.168.43.1          18-f0-e4-a3-af-5d     dynamic
  192.168.43.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 169.254.229.186 --- 0x6
  Internet Address      Physical Address      Type
  169.254.255.255       ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static

Interface: 192.168.56.1 --- 0x7
  Internet Address      Physical Address      Type
  192.168.56.255        ff-ff-ff-ff-ff-ff     static
  224.0.0.22            01-00-5e-00-00-16     static
  224.0.0.251           01-00-5e-00-00-fb     static
  224.0.0.252           01-00-5e-00-00-fc     static
  239.255.255.250       01-00-5e-7f-ff-fa     static
  255.255.255.255       ff-ff-ff-ff-ff-ff     static
```

There are two types of ARP entries- static and dynamic. Most of the time, the computer will use dynamic ARP entries. This means that the ARP entry (the Ethernet MAC to IP address link) has been learned (usually from the default gateway) and is kept on a device for some period of time, as long as it is being used. A static ARP entry is the opposite of a dynamic ARP entry. With a static ARP entry, the computer is manually entering the link between the Ethernet MAC address and the IP address. Software in your computer will predefine these static entries such as multicast addresses and broadcast addresses. Because of management headaches and the lack of significant negatives to using dynamic ARP entries, dynamic ARP entries are used most of the time.

Netstat:

Netstat Command Syntax

netstat [-a] [-b] [-e] [-f] [-n] [-o] [-p *protocol*] [-r] [-s] [-t] [-x] [-y] [*time_interval*] [/?]

Option	**Explanation**
netstat	Execute the netstat command alone to show a relatively simple list of all active TCP connections which, for each one, will show the local IP address (your computer), the foreign IP address (the other computer or network device), along with their respective port numbers, as well as the TCP state.
-a	This switch displays active TCP connections, TCP connections with the listening state, as well as UDP ports that are being listened to.

-b	This netstat switch is very similar to the -o switch listed below, but instead of displaying the PID, will display the process's actual file name. Using b over -o might seem like it's saving you a step or two but using it can sometimes greatly extend the time it takes netstat to fully execute.
-e	Use this switch with the netstat command to show statistics about your network connection. This data includes bytes, unicast packets, nonunicast packets, discards, errors, and unknown protocols received and sent since the connection was established.
-f	The -f switch will force the netstat command to display the Fully Qualified Domain Name(FQDN) for each foreign IP addresses when possible.
-n	Use the -n switch to prevent netstat from attempting to determine host name for foreign IP addresses. Depending on your current network connections, using this switch could considerably reduce the time it takes for netstat to fully execute.
-o	A handy option for many troubleshooting tasks, the -o switch displays the process identifier (PID) associated with each displayed connection. See the example below for more about using netstat o.
-r	Execute netstat with **-r** to show the IP routing table. This is the same as using the route command to execute **route print**.
-s	The **-s** option can be used with the netstat command to show detailed statistics by protocol. You can limit the statistics shown to a particular protocol by using the **-s**option and specifying that protocol, but be sure to use **-s** before **-p** protocol when using the switches together.
-t	Use the **-t** switch to show the current TCP
	chimney offload state in place of the typically displayed TCP state.
-x	Use the **-x** option to show all NetworkDirect listeners, connections, and shared endpoints.
-y	The **-y** switch can be used to show the TCP connection template for all connection. You cannot use **-y** with any other netstat option.
time_interval	This is the time, in seconds, that you'd like the netstat command to re-execute automatically, stopping only when you use ctrl-c to end the loop.

/?	Use the help switch to show details about the netstat command's several options.
-p	Use the **-p** switch to show connections or statistics only for a particular *protocol*. You can not define more than one *protocol* at once, nor can you execute netstat with **-p** without defining a *protocol*.
protocol	When specifying a *protocol* with the **-p** option, you can use **tcp**, **udp**, **tcpv6**, or **udpv6**. If you use **s** with **-p** to view statistics by protocol, you can use **icmp**, **ip**, **icmpv6**, or **ipv6** in addition to the first four I mentioned.

```
C:\Users\Aditya>netstat -s -p tcp -f

TCP Statistics for IPv4

  Active Opens                            = 4994
  Passive Opens                           = 743
  Failed Connection Attempts              = 726
  Reset Connections                       = 1878
  Current Connections                     = 9
  Segments Received                       = 491061
  Segments Sent                           = 424757
  Segments Retransmitted                  = 5053

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:5939         Aditya:57062           ESTABLISHED
  TCP    127.0.0.1:5939         Aditya:61522           ESTABLISHED
  TCP    127.0.0.1:57062        Aditya:5939            ESTABLISHED
  TCP    127.0.0.1:57069        Aditya:57070           ESTABLISHED
  TCP    127.0.0.1:57070        Aditya:57069           ESTABLISHED
  TCP    127.0.0.1:61522        Aditya:5939            ESTABLISHED
  TCP    192.168.43.126:57108   40.90.189.152:https    ESTABLISHED
  TCP    192.168.43.126:57423   a23-64-140-152.deploy.static.akamaitechnologies.com:https  CLOSE_WAIT
  TCP    192.168.43.126:61517   test.activity.windows.com:https  TIME_WAIT
  TCP    192.168.43.126:61519   168.63.153.205:https   TIME_WAIT
  TCP    192.168.43.126:61520   168.63.153.205:https   TIME_WAIT
  TCP    192.168.43.126:61523   52.109.124.5:https     TIME_WAIT
  TCP    192.168.43.126:61524   test.activity.windows.com:https  ESTABLISHED
```

Tracert:

The tracert command is a Command Prompt Command that's used to show several details about the path that a packet takes from the computer or device you're on to whatever destination you specify. You might also sometimes see the tracert command referred to as the trace route command or traceroute command.

Tracert Command Syntax:

tracert [**-d**] [**-h** *MaxHops*] [**-w** *TimeOut*] [**-4**] [**-6**] *target* [**/?**]

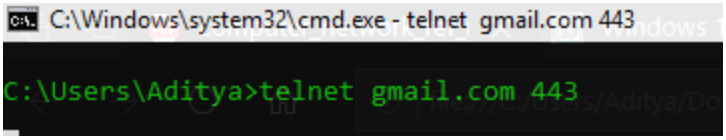
Item	Description
-d	This option prevents tracert from resolving IP addresses to hostname, often resulting in much faster results.
-h *MaxHops*	This tracert option specifies the maximum number of hops in the search for the *target*. If you do not specify *MaxHops*, and a *target* has not been found by 30 hops, tracert will stop looking

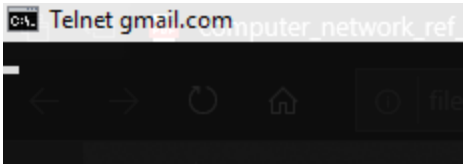
-w TimeOut	You can specify the time, in milliseconds, to allow each reply before timeout using this tracert option.
-4	This option forces tracert to use IPv4 only.
-6	This option forces tracert to use IPv6 only.
target	This is the destination, either an IP address or hostname.
/?	Use the help switch with the tracert command to show detailed help about the command's several options.

```
C:\Users\Aditya>tracert www.gmail.com

Tracing route to googlemail.l.google.com [2404:6800:4009:805::2005]
over a maximum of 30 hops:

  1     3 ms     2 ms     2 ms  fe80::20d2:4eff:fedc:809a
  2     *        *        *     Request timed out.
  3    36 ms    41 ms    44 ms  2405:200:320:1504::2
  4    45 ms    39 ms    40 ms  2405:200:801:b00::a9a
  5    44 ms    42 ms    36 ms  2405:200:801:b00::ab2
  6    66 ms    61 ms    55 ms  2405:200:801:200::31d
  7    52 ms    61 ms    40 ms  2001:4860:1:1::270
  8    62 ms    49 ms    49 ms  2001:4860:0:115b::1
  9    50 ms    46 ms    55 ms  2001:4860:0:1::22c7
 10    44 ms    55 ms    48 ms  bom05s09-in-x05.1e100.net [2404:6800:4009:805::2005]

Trace complete.

C:\Users\Aditya>
```

Telnet: telnet is a protocol to provide communication over the Internet or a LAN a using a virtual terminal connection.
It is installed by default on Linux and older Mac operating systems, but must be installed on Windows and macOS High Sierra 10.13 and later.
The terminal provides a way to remotely log on to *another* device, just as if you were sitting in front of it and using it like any other computer. This method of communication is, of course, done via Telnet.

To connect to a Telnet server, you need to enter a command that follows this syntax: Telnet host portnumber





Lab 4a:

Objective: Implement a simple TCP socket based client server program in Python in which the client connects to the server. The server displays the ip address and port number of client and sends an acknowledgement message back to client. The client displays the received acknowledgement message on screen

server.py - C:\Users\Aditya\Desktop\server.py (3.7.2)

File Edit Format Run Options Window Help

```
    import socket
host = "localhost"
port =5555
sok = socket.socket()#also valid instead of following statement
#sok = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
sok.bind((host,port))
print("server start")
sok.listen(1)
client,addr = sok.accept()
print("client address: ", addr)
msg = "Hello client how are you?"
client.send(msg.encode())
client.send(b'Bye')
client.close()
print("Close")
```

client.py - C:\Users\Aditya\Desktop\client.py (3.7.2)

File Edit Format Run Options Window Help

```
import socket
host="localhost"
port = 5555
sok = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
sok.connect((host,port))
msg = sok.recv(1024) #1024 size of buffer
while msg:
    print(msg.decode())
    msg = sok.recv(1024)
sok.close()
```

output :-

C:\Windows\system32\cmd.exe

```
C:\Users\Aditya\Desktop>python server.py
server start
client address:  ('127.0.0.1', 63660)
Close

C:\Users\Aditya\Desktop>
```

C:\Windows\system32\cmd.exe

```
Microsoft Windows [Version 10.0.17763.437]
(c) 2018 Microsoft Corporation. All rights reserved.

C:\Users\Aditya>cd desktop

C:\Users\Aditya\Desktop>python client.py
Hello client how are you?
Bye

C:\Users\Aditya\Desktop>
```

Lab 4b:

Objective: Implement a simple UDP socket based client server program in Python in which the client connects to the server. The server displays the ip address and port number of client and sends an acknowledgement message back to client. The client displays the received acknowledgement message on screen.

server.py - C:\Users\Aditya\Desktop\server.py (3.7.2)

File Edit Format Run Options Window Help

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
# For UDP
udp_host = socket.gethostname()
# Host IP
udp_port = 12345
# specified port to connect
#print type(sock) ============> 'type' can be used to see type
# of any variable ('sock' here)
sock.bind((udp_host,udp_port))
while True:
    print ("Waiting for client...")
    data,addr = sock.recvfrom(1024)
    #receive data from client
    print ("Received Messages:",data," from",addr)
```

client.py - C:\Users\Aditya\Desktop\client.py (3.7.2)

File Edit Format Run Options Window Help

```
import socket

sock = socket.socket(socket.AF_INET,socket.SOCK_DGRAM)
# For UDP
udp_host = socket.gethostname()
# Host IP
udp_port = 12345
# specified port to connect
msg = "Hello Python!".encode()
print ("UDP target IP:", udp_host)
print ("UDP target Port:", udp_port)
sock.sendto(msg,(udp_host,udp_port))
# Sending message to UDP server
```

output:

C:\Windows\system32\cmd.exe - python server.py

```
C:\Users\Aditya\Desktop>python server.py
Waiting for client...
Received Messages: b'Hello Python!'  from ('169.254.229.186', 51150)
Waiting for client...
Received Messages: b'Hello Python!'  from ('169.254.229.186', 51151)
Waiting for client...
```

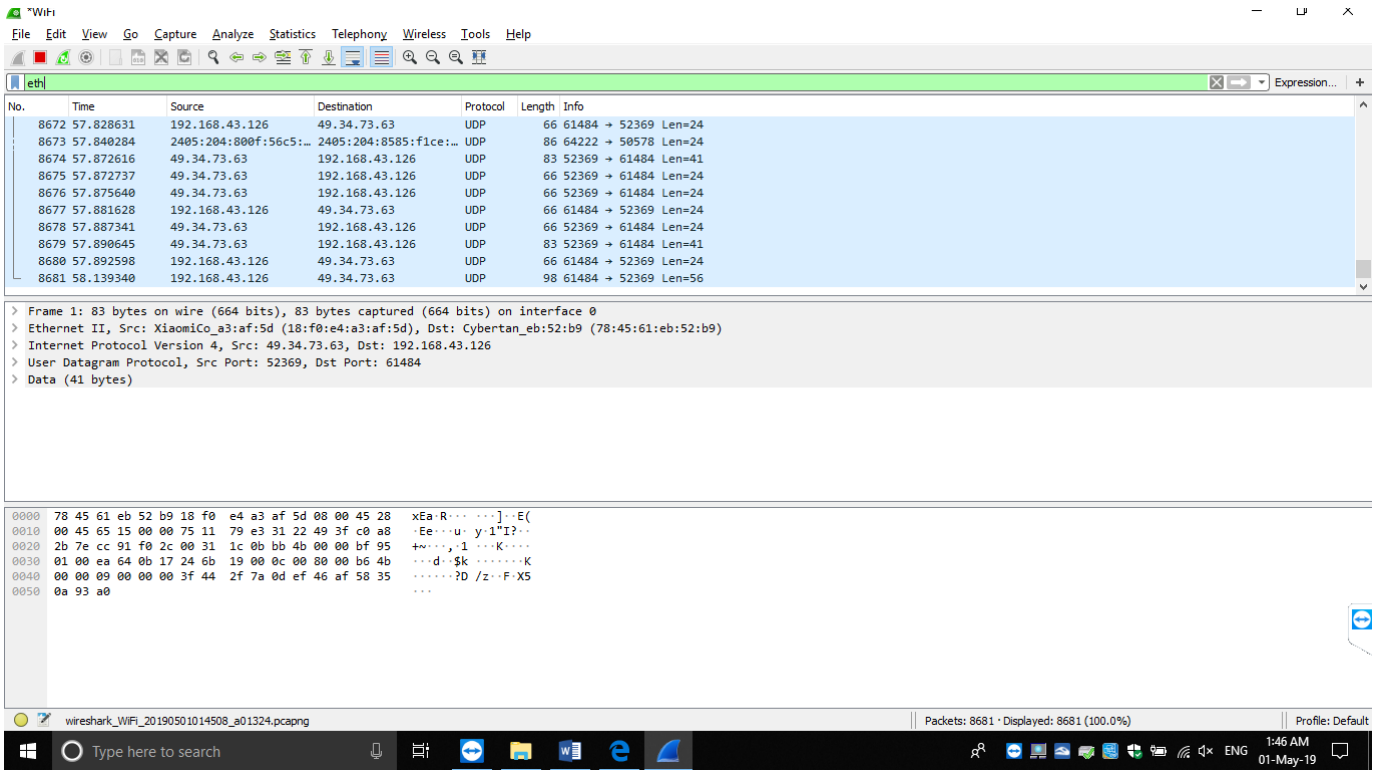
C:\Windows\system32\cmd.exe

```
C:\Users\Aditya\Desktop>python client.py
UDP target IP: Aditya
UDP target Port: 12345

C:\Users\Aditya\Desktop>python client.py
UDP target IP: Aditya
UDP target Port: 12345

C:\Users\Aditya\Desktop>
```

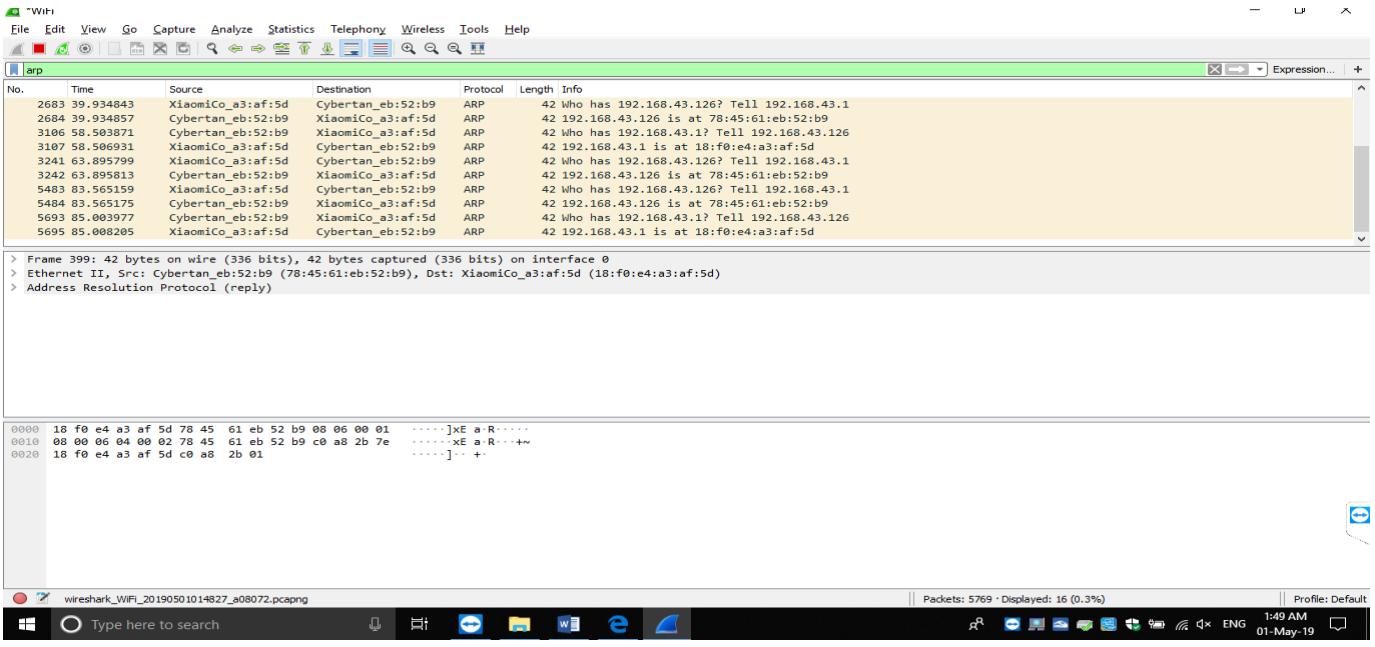
Lab 6:

Objective: Learn and use view level filters and capture level filters in Wireshark for different traffic types like Ethernet, ARP, IP, TCP, UDP, DNS, HTTP, etc. For each captured category, observe how different headers are encapsulated within each other. Eg. TCP encapsulated within IP, HTTP encapsulated within TCP, etc.

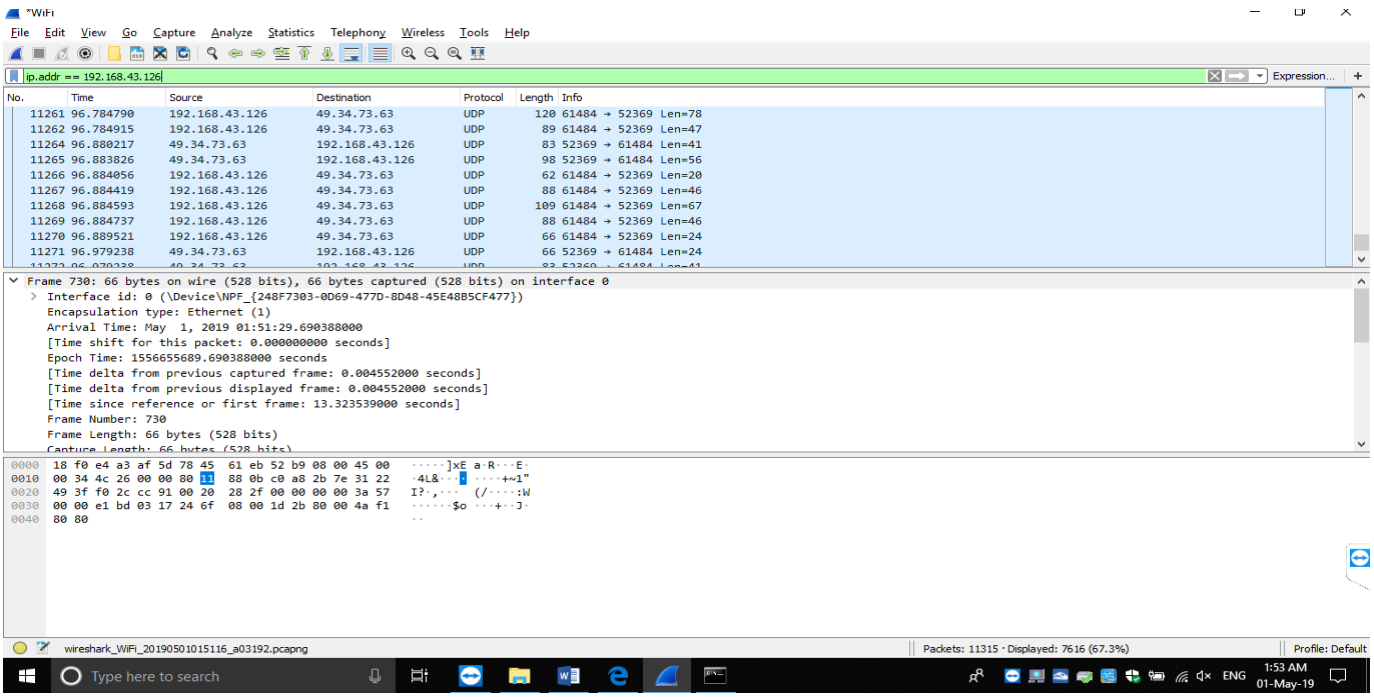
ETHERNET



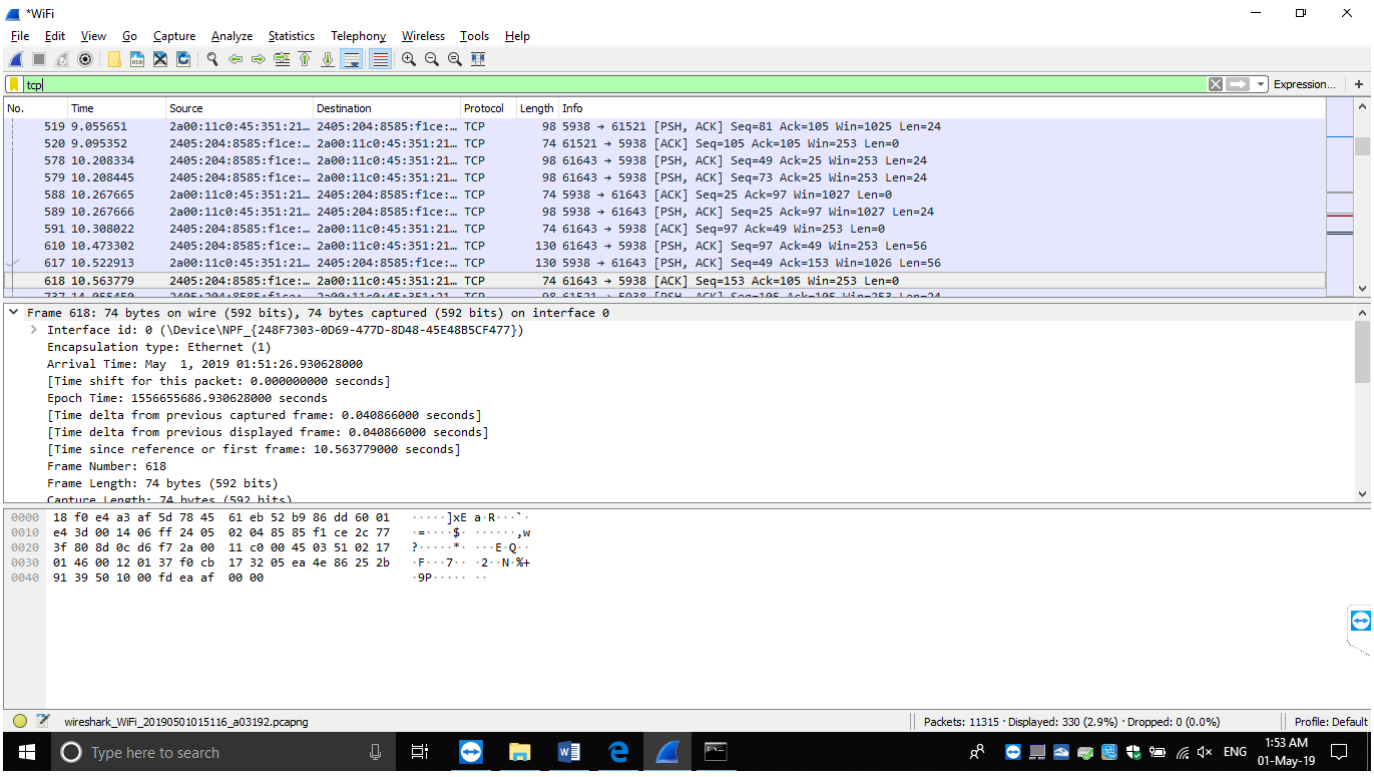
ARP



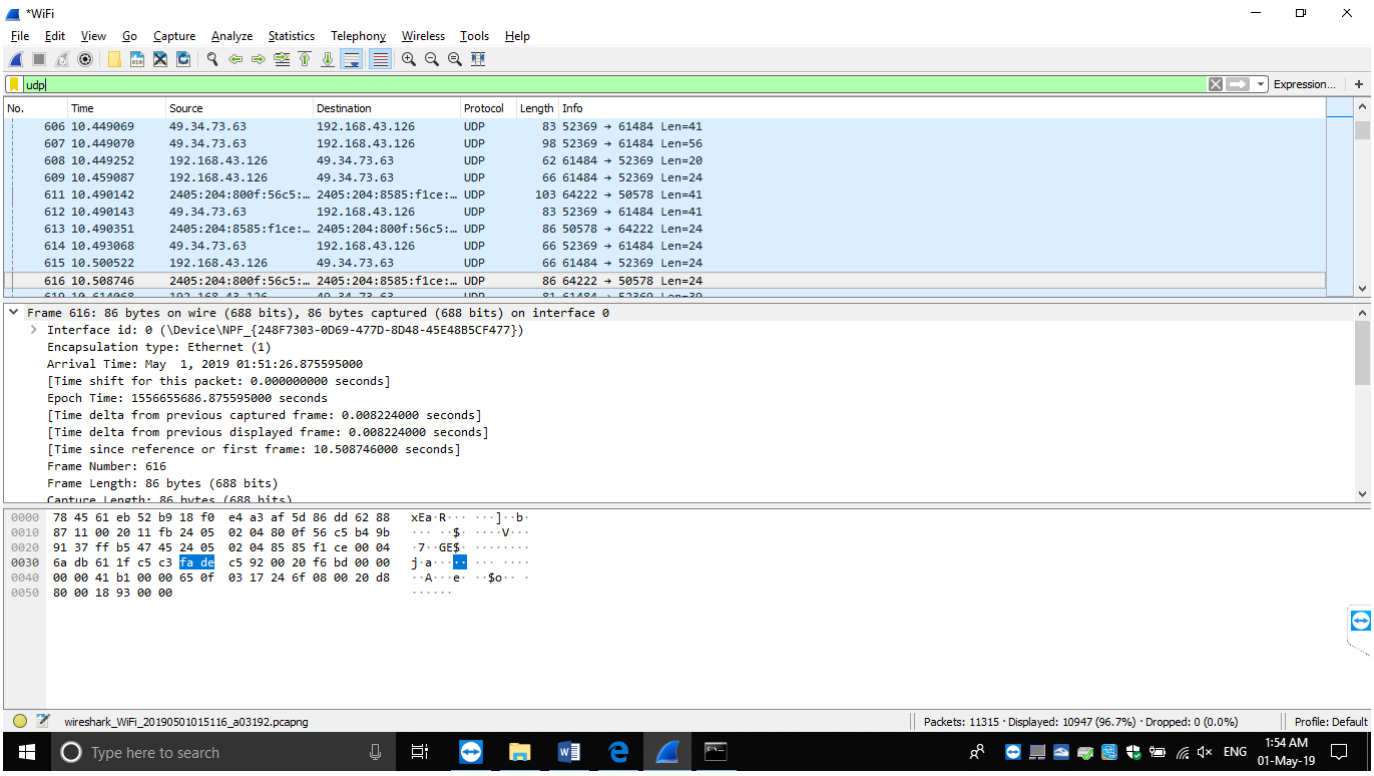
IP


*WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
ip.addr == 192.168.43.126
No. Time Source Destination Protocol Length Info
11261 96.784790 192.168.43.126 49.34.73.63 UDP 120 61484 → 52369 Len=78
11262 96.784915 192.168.43.126 49.34.73.63 UDP 89 61484 → 52369 Len=47
11264 96.880217 49.34.73.63 192.168.43.126 UDP 83 52369 → 61484 Len=41
11265 96.883826 49.34.73.63 192.168.43.126 UDP 98 52369 → 61484 Len=56
11266 96.884056 192.168.43.126 49.34.73.63 UDP 62 61484 → 52369 Len=20
11267 96.884419 192.168.43.126 49.34.73.63 UDP 88 61484 → 52369 Len=46
11268 96.884593 192.168.43.126 49.34.73.63 UDP 109 61484 → 52369 Len=67
11269 96.884737 192.168.43.126 49.34.73.63 UDP 88 61484 → 52369 Len=46
11270 96.889521 192.168.43.126 49.34.73.63 UDP 66 61484 → 52369 Len=24
11271 96.979238 49.34.73.63 192.168.43.126 UDP 66 52369 → 61484 Len=24
Frame 730: 66 bytes on wire (528 bits), 66 bytes captured (528 bits) on interface 0
Interface id: 0 (\Device\NPF_{248F7303-0D69-477D-8D48-45E48B5CF477})
Encapsulation type: Ethernet (1)
Arrival Time: May 1, 2019 01:51:29.690388000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1556655689.690388000 seconds
[Time delta from previous captured frame: 0.004552000 seconds]
[Time delta from previous displayed frame: 0.004552000 seconds]
[Time since reference or first frame: 13.323539000 seconds]
Frame Number: 730
Frame Length: 66 bytes (528 bits)
0000 18 f0 e4 a3 af 5d 78 45 61 eb 52 b9 08 00 45 00
0010 00 34 4c 26 00 00 80 11 88 0b c0 a8 2b 7e 31 22
0020 49 3f f0 2c cc 91 00 20 28 2f 00 00 00 00 3a 57
0030 00 00 e1 bd 03 17 24 6f 08 00 1d 2b 80 00 4a f1
0040 80 80
wireshark_WiFi_20190501015116_a03192.pcapng
Packets: 11315 · Displayed: 7616 (67.3%)
Profile: Default
Type here to search
ENG
1:53 AM
01-May-19

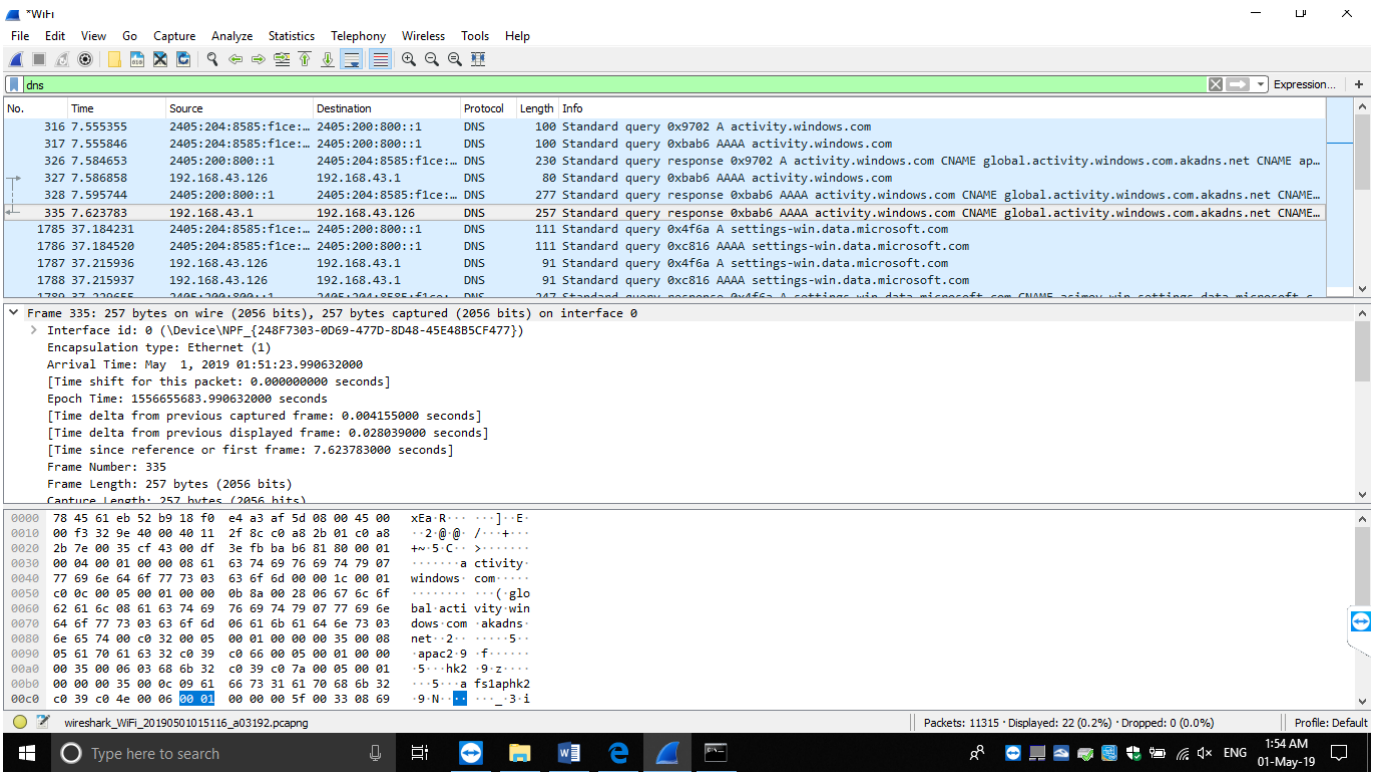
TCP


*WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
tcp
No. Time Source Destination Protocol Length Info
519 9.055651 2a00:11c0:45:351:21… 2405:204:8585:f1ce:… TCP 98 5938 → 61521 [PSH, ACK] Seq=81 Ack=105 Win=1025 Len=24
520 9.095352 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 74 61521 → 5938 [ACK] Seq=105 Ack=105 Win=253 Len=0
578 10.208334 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 98 61643 → 5938 [PSH, ACK] Seq=49 Ack=25 Win=253 Len=24
579 10.208445 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 98 61643 → 5938 [PSH, ACK] Seq=73 Ack=25 Win=253 Len=24
588 10.267665 2a00:11c0:45:351:21… 2405:204:8585:f1ce:… TCP 74 5938 → 61643 [ACK] Seq=25 Ack=97 Win=1027 Len=0
589 10.267666 2a00:11c0:45:351:21… 2405:204:8585:f1ce:… TCP 98 5938 → 61643 [PSH, ACK] Seq=25 Ack=97 Win=1027 Len=24
591 10.308022 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 74 61643 → 5938 [ACK] Seq=97 Ack=49 Win=253 Len=0
610 10.473302 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 130 61643 → 5938 [PSH, ACK] Seq=97 Ack=49 Win=253 Len=56
617 10.522913 2a00:11c0:45:351:21… 2405:204:8585:f1ce:… TCP 130 5938 → 61643 [PSH, ACK] Seq=49 Ack=153 Win=1026 Len=56
618 10.563779 2405:204:8585:f1ce:… 2a00:11c0:45:351:21… TCP 74 61643 → 5938 [ACK] Seq=153 Ack=105 Win=253 Len=0
Frame 618: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
Interface id: 0 (\Device\NPF_{248F7303-0D69-477D-8D48-45E48B5CF477})
Encapsulation type: Ethernet (1)
Arrival Time: May 1, 2019 01:51:26.930628000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1556655686.930628000 seconds
[Time delta from previous captured frame: 0.040866000 seconds]
[Time delta from previous displayed frame: 0.040866000 seconds]
[Time since reference or first frame: 10.563779000 seconds]
Frame Number: 618
Frame Length: 74 bytes (592 bits)
0000 18 f0 e4 a3 af 5d 78 45 61 eb 52 b9 86 dd 60 01
0010 e4 3d 00 14 06 ff 24 05 02 04 85 85 f1 ce 2c 77
0020 3f 80 8d 0c d6 f7 2a 00 11 c0 00 45 03 51 02 17
0030 01 46 00 12 01 37 f0 cb 17 32 05 ea 4e 86 25 2b
0040 91 39 50 10 00 fd ea af 00 00
wireshark_WiFi_20190501015116_a03192.pcapng
Packets: 11315 · Displayed: 330 (2.9%) · Dropped: 0 (0.0%)
Profile: Default
Type here to search
ENG
1:53 AM
01-May-19

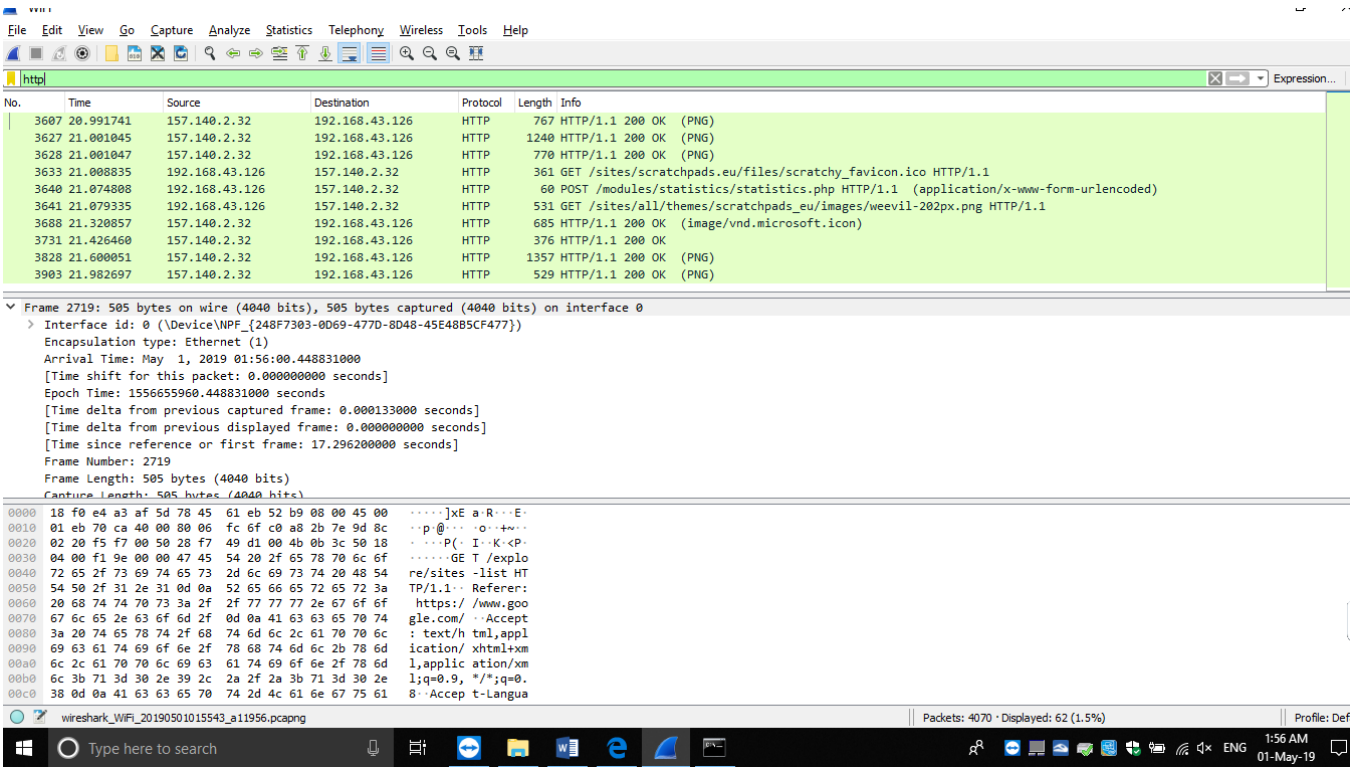
UDP


*WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
udp
No. Time Source Destination Protocol Length Info
606 10.449069 49.34.73.63 192.168.43.126 UDP 83 52369 → 61484 Len=41
607 10.449070 49.34.73.63 192.168.43.126 UDP 98 52369 → 61484 Len=56
608 10.449252 192.168.43.126 49.34.73.63 UDP 62 61484 → 52369 Len=20
609 10.459087 192.168.43.126 49.34.73.63 UDP 66 61484 → 52369 Len=24
611 10.490142 2405:204:800f:56c5:... 2405:204:8585:f1ce:... UDP 103 64222 → 50578 Len=41
612 10.490143 49.34.73.63 192.168.43.126 UDP 83 52369 → 61484 Len=41
613 10.490351 2405:204:8585:f1ce:... 2405:204:800f:56c5:... UDP 86 50578 → 64222 Len=24
614 10.493068 49.34.73.63 192.168.43.126 UDP 66 52369 → 61484 Len=24
615 10.500522 192.168.43.126 49.34.73.63 UDP 66 61484 → 52369 Len=24
616 10.508746 2405:204:800f:56c5:... 2405:204:8585:f1ce:... UDP 86 64222 → 50578 Len=24
Frame 616: 86 bytes on wire (688 bits), 86 bytes captured (688 bits) on interface 0
Interface id: 0 (\Device\NPF_{248F7303-0D69-477D-8D48-45E48B5CF477})
Encapsulation type: Ethernet (1)
Arrival Time: May 1, 2019 01:51:26.875595000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1556655686.875595000 seconds
[Time delta from previous captured frame: 0.008224000 seconds]
[Time delta from previous displayed frame: 0.008224000 seconds]
[Time since reference or first frame: 10.508746000 seconds]
Frame Number: 616
Frame Length: 86 bytes (688 bits)
0000 78 45 61 eb 52 b9 18 f0 e4 a3 af 5d 86 dd 62 88
0010 87 11 00 20 11 fb 24 05 02 04 80 0f 56 c5 b4 9b
0020 91 37 ff b5 47 45 24 05 02 04 85 85 f1 ce 00 04
0030 6a db 61 1f c5 c3 fa de c5 92 00 20 f6 bd 00 00
0040 00 00 41 b1 00 00 65 0f 03 17 24 6f 08 00 20 d8
0050 80 00 18 93 00 00
wireshark_WiFi_20190501015116_a03192.pcapng
Packets: 11315 · Displayed: 10947 (96.7%) · Dropped: 0 (0.0%)
Profile: Default
Type here to search
ENG 1:54 AM 01-May-19

DNS


*WiFi
File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
dns
No. Time Source Destination Protocol Length Info
316 7.555355 2405:204:8585:f1ce:... 2405:200:800::1 DNS 100 Standard query 0x9702 A activity.windows.com
317 7.555846 2405:204:8585:f1ce:... 2405:200:800::1 DNS 100 Standard query 0xbab6 AAAA activity.windows.com
326 7.584653 2405:200:800::1 2405:204:8585:f1ce:... DNS 230 Standard query response 0x9702 A activity.windows.com CNAME global.activity.windows.com.akadns.net CNAME ap...
327 7.586858 192.168.43.126 192.168.43.1 DNS 80 Standard query 0xbab6 AAAA activity.windows.com
328 7.595744 2405:200:800::1 2405:204:8585:f1ce:... DNS 277 Standard query response 0xbab6 AAAA activity.windows.com CNAME global.activity.windows.com.akadns.net CNAME...
335 7.623783 192.168.43.1 192.168.43.126 DNS 257 Standard query response 0xbab6 AAAA activity.windows.com CNAME global.activity.windows.com.akadns.net CNAME...
1785 37.184231 2405:204:8585:f1ce:... 2405:200:800::1 DNS 111 Standard query 0x4f6a A settings-win.data.microsoft.com
1786 37.184520 2405:204:8585:f1ce:... 2405:200:800::1 DNS 111 Standard query 0xc816 AAAA settings-win.data.microsoft.com
1787 37.215936 192.168.43.126 192.168.43.1 DNS 91 Standard query 0x4f6a A settings-win.data.microsoft.com
1788 37.215937 192.168.43.126 192.168.43.1 DNS 91 Standard query 0xc816 AAAA settings-win.data.microsoft.com
Frame 335: 257 bytes on wire (2056 bits), 257 bytes captured (2056 bits) on interface 0
Interface id: 0 (\Device\NPF_{248F7303-0D69-477D-8D48-45E48B5CF477})
Encapsulation type: Ethernet (1)
Arrival Time: May 1, 2019 01:51:23.990632000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1556655683.990632000 seconds
[Time delta from previous captured frame: 0.004155000 seconds]
[Time delta from previous displayed frame: 0.028039000 seconds]
[Time since reference or first frame: 7.623783000 seconds]
Frame Number: 335
Frame Length: 257 bytes (2056 bits)
0000 78 45 61 eb 52 b9 18 f0 e4 a3 af 5d 08 00 45 00
0010 00 f3 32 9e 40 00 40 11 2f 8c c0 a8 2b 01 c0 a8
0020 2b 7e 00 35 cf 43 00 df 3e fb ba b6 81 80 00 01
0030 00 04 00 01 00 00 08 61 63 74 69 76 69 74 79 07
0040 77 69 6e 64 6f 77 73 03 63 6f 6d 00 00 1c 00 01
0050 c0 0c 00 05 00 01 00 00 0b 8a 00 28 06 67 6c 6f
0060 62 61 6c 08 61 63 74 69 76 69 74 79 07 77 69 6e
0070 64 6f 77 73 03 63 6f 6d 06 61 6b 61 64 6e 73 03
0080 6e 65 74 00 c0 32 00 05 00 01 00 00 00 35 00 08
0090 05 61 70 61 63 32 c0 39 c0 66 00 05 00 01 00 00
00a0 00 35 00 06 03 68 6b 32 c0 39 c0 7a 00 05 00 01
00b0 00 00 00 35 00 0c 09 61 66 73 31 61 70 68 6b 32
00c0 c0 39 c0 4e 00 06 00 01 00 00 00 5f 00 33 08 69
wireshark_WiFi_20190501015116_a03192.pcapng
Packets: 11315 · Displayed: 22 (0.2%) · Dropped: 0 (0.0%)
Profile: Default
Type here to search
ENG 1:54 AM 01-May-19

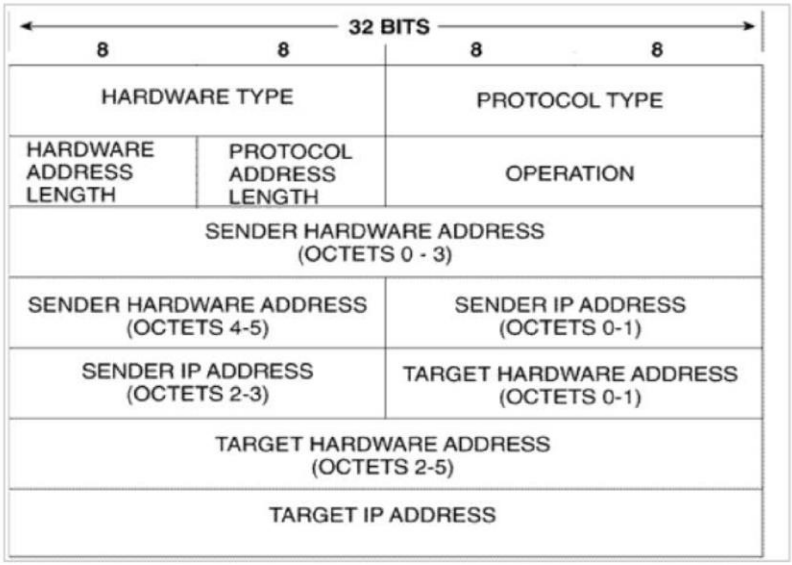
HTTP


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
http
Expression...
No. Time Source Destination Protocol Length Info
3607 20.991741 157.140.2.32 192.168.43.126 HTTP 767 HTTP/1.1 200 OK (PNG)
3627 21.001045 157.140.2.32 192.168.43.126 HTTP 1240 HTTP/1.1 200 OK (PNG)
3628 21.001047 157.140.2.32 192.168.43.126 HTTP 770 HTTP/1.1 200 OK (PNG)
3633 21.008835 192.168.43.126 157.140.2.32 HTTP 361 GET /sites/scratchpads.eu/files/scratchy_favicon.ico HTTP/1.1
3640 21.074808 192.168.43.126 157.140.2.32 HTTP 60 POST /modules/statistics/statistics.php HTTP/1.1 (application/x-www-form-urlencoded)
3641 21.079335 192.168.43.126 157.140.2.32 HTTP 531 GET /sites/all/themes/scratchpads_eu/images/weevil-202px.png HTTP/1.1
3688 21.320857 157.140.2.32 192.168.43.126 HTTP 685 HTTP/1.1 200 OK (image/vnd.microsoft.icon)
3731 21.426460 157.140.2.32 192.168.43.126 HTTP 376 HTTP/1.1 200 OK
3828 21.600051 157.140.2.32 192.168.43.126 HTTP 1357 HTTP/1.1 200 OK (PNG)
3903 21.982697 157.140.2.32 192.168.43.126 HTTP 529 HTTP/1.1 200 OK (PNG)
Frame 2719: 505 bytes on wire (4040 bits), 505 bytes captured (4040 bits) on interface 0
Interface id: 0 (\Device\NPF_{248F7303-0D69-477D-8D48-45E48B5CF477})
Encapsulation type: Ethernet (1)
Arrival Time: May 1, 2019 01:56:00.448831000
[Time shift for this packet: 0.000000000 seconds]
Epoch Time: 1556655960.448831000 seconds
[Time delta from previous captured frame: 0.000133000 seconds]
[Time delta from previous displayed frame: 0.000000000 seconds]
[Time since reference or first frame: 17.296200000 seconds]
Frame Number: 2719
Frame Length: 505 bytes (4040 bits)
0000 18 f0 e4 a3 af 5d 78 45 61 eb 52 b9 08 00 45 00 ·····]xE a·R···E·
0010 01 eb 70 ca 40 00 80 06 fc 6f c0 a8 2b 7e 9d 8c ··p·@··· ·o··+~··
0020 02 20 f5 f7 00 50 28 f7 49 d1 00 4b 0b 3c 50 18 · ···P(· I··K·<P·
0030 04 00 f1 9e 00 00 47 45 54 20 2f 65 78 70 6c 6f ·····GE T /explo
0040 72 65 2f 73 69 74 65 73 2d 6c 69 73 74 20 48 54 re/sites -list HT
0050 54 50 2f 31 2e 31 0d 0a 52 65 66 65 72 65 72 3a TP/1.1·· Referer:
0060 20 68 74 74 70 73 3a 2f 2f 77 77 77 2e 67 6f 6f https:/ /www.goo
0070 67 6c 65 2e 63 6f 6d 2f 0d 0a 41 63 63 65 70 74 gle.com/ ··Accept
0080 3a 20 74 65 78 74 2f 68 74 6d 6c 2c 61 70 70 6c : text/h tml,appl
0090 69 63 61 74 69 6f 6e 2f 78 68 74 6d 6c 2b 78 6d ication/ xhtml+xm
00a0 6c 2c 61 70 70 6c 69 63 61 74 69 6f 6e 2f 78 6d l,applic ation/xm
00b0 6c 3b 71 3d 30 2e 39 2c 2a 2f 2a 3b 71 3d 30 2e l;q=0.9, */*;q=0.
00c0 38 0d 0a 41 63 63 65 70 74 2d 4c 61 6e 67 75 61 8··Accep t-Langua
wireshark_WiFi_20190501015543_a11956.pcapng
Packets: 4070 · Displayed: 62 (1.5%)
Profile: Def
Type here to search
ENG
1:56 AM
01-May-19

Lab 7:
Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: Ethernet and ARP

Ans: **ARP:-**
Address Resolution Protocol (ARP) is a predominant protocol for finding a host's hardware address when only its network layer address is known. This protocol operates below the network layer as a part of the interface between the OSI network and OSI link layer. The format of ARP packet can be discussed as follows:-

ARP PACKET FORMAT



HARDWARE TYPE: - Each data link layer protocol is assigned a number used in this field. For Ethernet it is 1.

PROTOCOL TYPE: - Each protocol is assigned a number used in this field. For example, IPv4 is 0x0800.

HARDWARE LENGTH: - Length in bytes of a hardware address. Ethernet addresses are 6 bytes long.

PROTOCOL LENGTH: - Length in bytes of a logical address. IPv4 addresses are 4 bytes long.

OPERATION: - It specifies the operation the sender is performing: 1 for request, and 2 for reply. There are actually four types of ARP messages that may be sent by the ARP protocol. These are identified by four values in the "operation" field of an ARP message. The types of message are:-
1. ARP request
2. ARP reply
3. RARP request
4. RARP reply

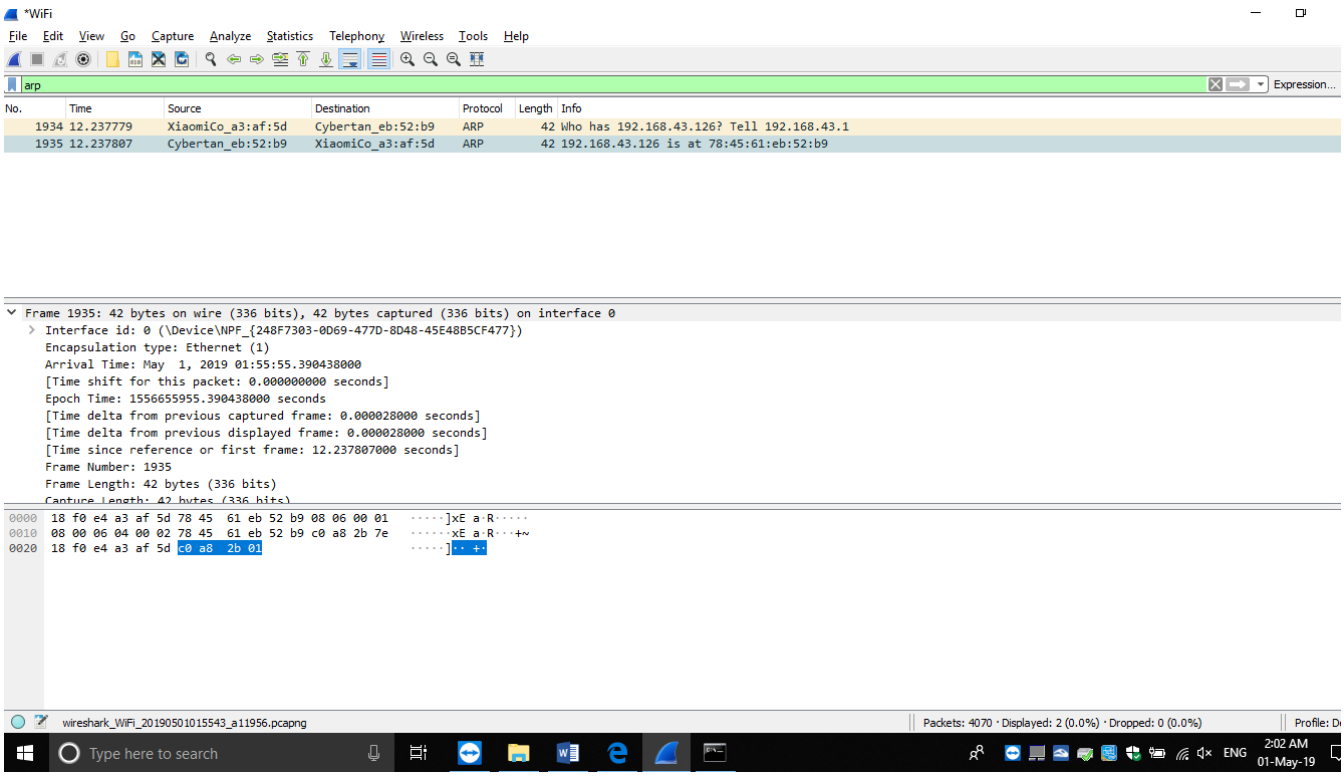
SENDER HARDWARE ADDRESS: - Hardware address of the sender.

SENDER PROTOCOL ADDRESS: - Protocol address of the sender.

TARGET HARDWARE ADDRESS: - Hardware address of the intended receiver. This field is zero on request.

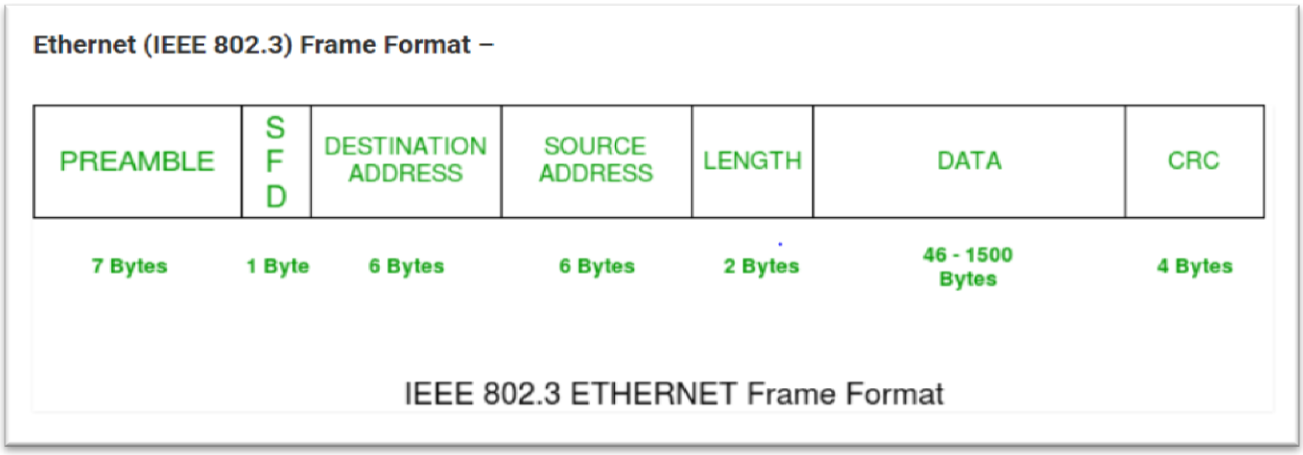
TARGET PROTOCOL ADDRESS: - Protocol address of the intended receiver.

Output:-



Ethernet:-

An Ethernet frame is made up of five fields: the Destination and the Source MAC address field, the Ether type field that contains some control information, a Payload field, and a trailing Frame Check Sequence field that holds a checksum for the frame. The Ethernet frame starts with preamble and start frame delimiter, followed by an Ethernet header containing MAC address of source and destination devices. The middle section of the frame consists of payload data including headers for other protocols such as internet protocol, carried in the frame. The end part of the frame consist a 32-bit cyclic redundancy check which detects the errors in transmission. The standard data frame has a maximum length of 1518 bytes, and contains a Payload field of at least 46 and up to 1500 bytes.



Preamble: – Ethernet frame starts with 7-Bytes Preamble. This is pattern of alternative 0's and 1's which indicates starting of the frame and allow sender and receiver to establish bit synchronization. Initially, PRE (Preamble) was introduced to allow for the loss of few bits due to signal delays. But todays highspeed Ethernet don't need Preamble to protect the frame bits. PRE (Preamble) indicates the receiver that frame is coming and allow the receiver to lock onto the data stream before the actual frame begins.

Start of frame delimiter (SFD): – This is a 1-Byte field which is always set to 10101011. SFD indicates that upcoming bits are starting of frame, which is destination address. Sometimes SFD is considered the part of PRE, this is the reason Preamble is described as 8 Bytes in many places.

Destination Address: – This is 6-Byte field which contains the MAC address of machine for which data is destined.

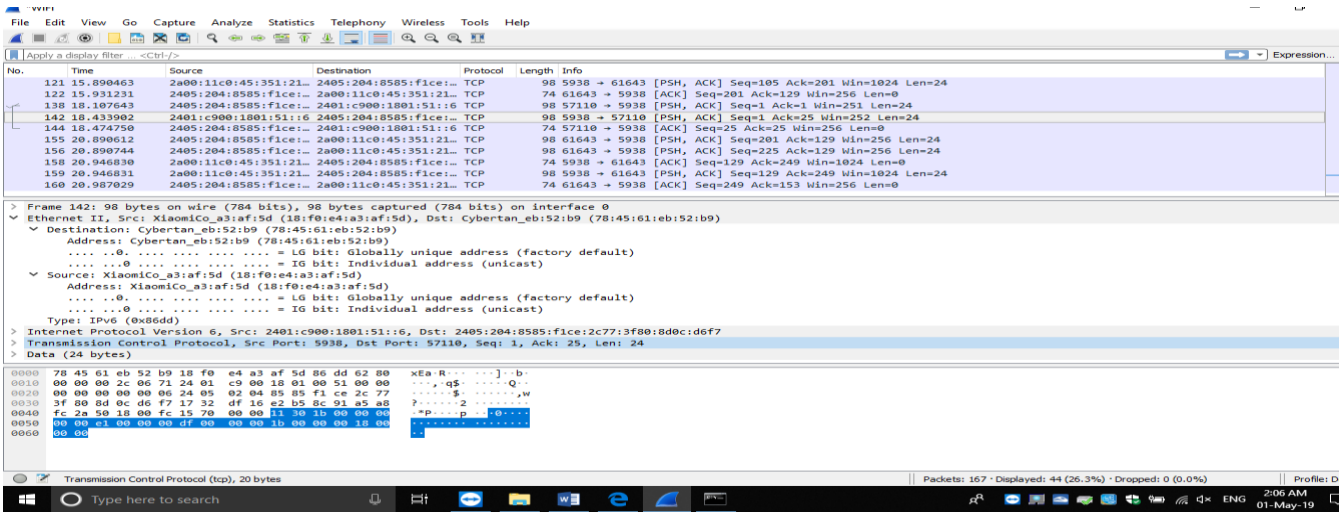
Source Address: – This is a 6-Byte field which contains the MAC address of source machine. As Source Address is always an individual address (Unicast), the least significant bit of first byte is always 0.

Length: – Length is a 2-Byte field, which indicates the length of entire Ethernet frame. This 16-bit field can hold the length value between 0 and 65534, but length cannot be larger than 1500 because of some own limitations of Ethernet.

Data: – This is the place where actual data is inserted, also known as **Payload**. Both IP header and data will be inserted here, if Internet Protocol is used over Ethernet. The maximum data present may be as long as 1500 Bytes. In case data length is less than minimum length i.e. 46 bytes, then padding 0's is added to meet the minimum possible length.

Cyclic Redundancy Check (CRC): – CRC is 4 Byte field. This field contains 32-bits hash code of data, which is generated over Destination Address, Source Address, Length and Data field. If the checksum computed by destination is not same as sent checksum value, data received is corrupted.

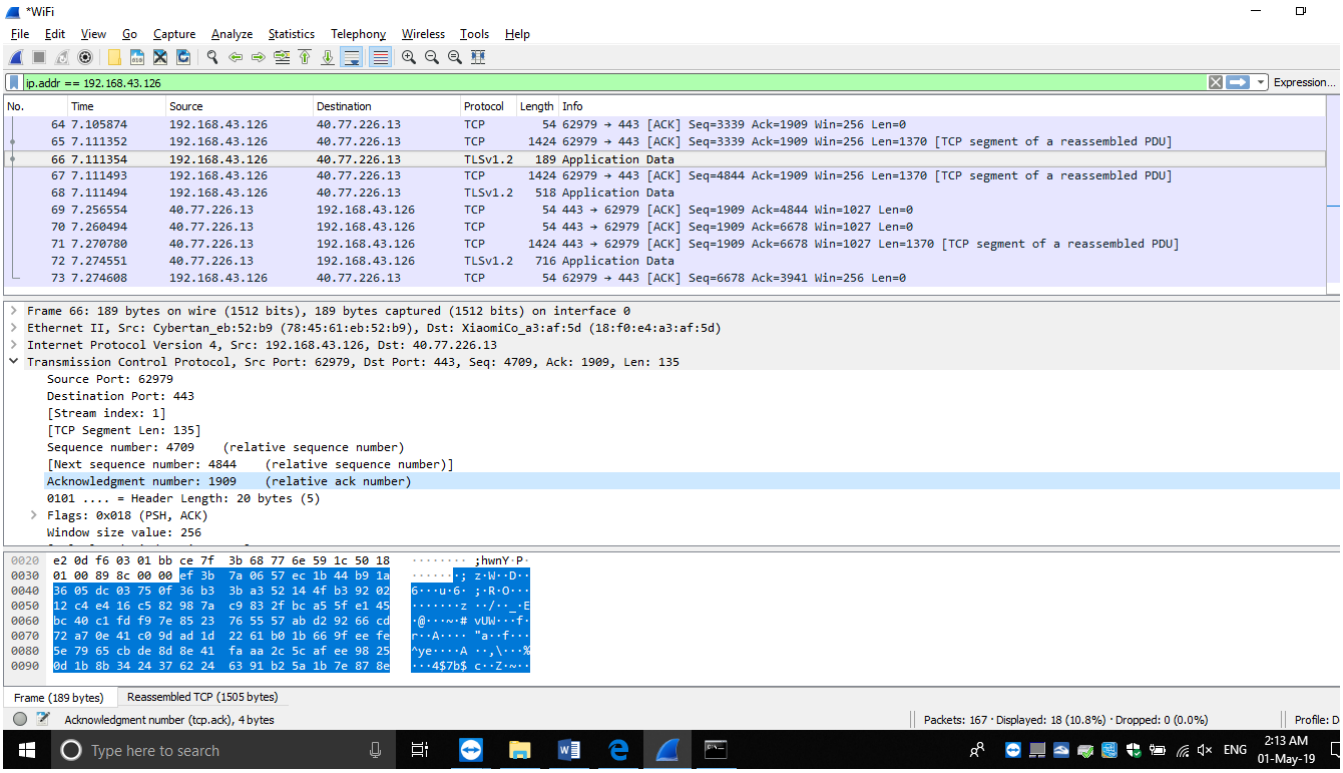
Output:-



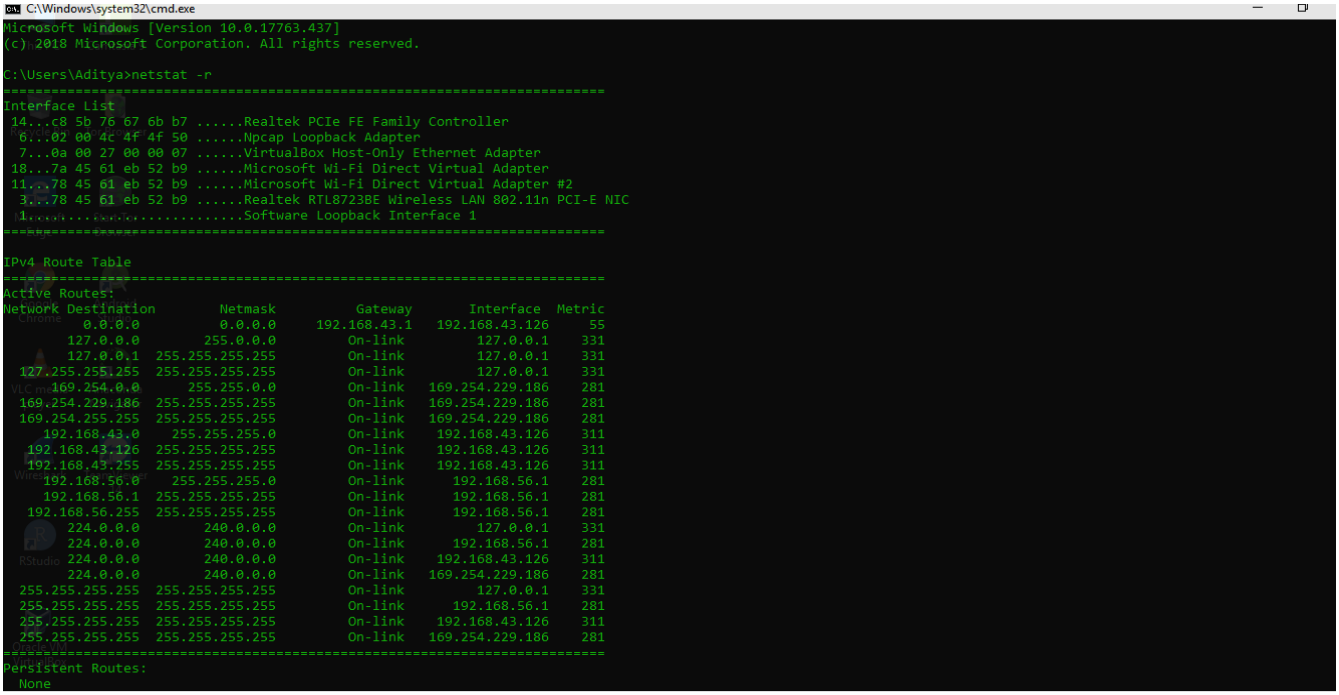
Lab 8:
Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: IP and ICMP

ANS:-

IP



IP traffic using command prompt



```
IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  3    311 ::/0                    fe80::5a8:30b9:f5b:cfc5
  1    331 ::1/128                 On-link
  3    311 2405:204:8585:f1ce::/64 On-link
  3    311 2405:204:8585:f1ce:4:6adb:611f:c5c3/128
                                   On-link
  3    311 2405:204:8585:f1ce:2c77:3f80:8d0c:d6f7/128
                                   On-link
  7    281 fe80::/64               On-link
  3    311 fe80::/64               On-link
  6    281 fe80::/64               On-link
  3    311 fe80::4:6adb:611f:c5c3/128
                                   On-link
  6    281 fe80::30db:8113:f467:e5ba/128
                                   On-link
  7    281 fe80::3d3c:4d14:4a35:54a5/128
                                   On-link
  1    331 ff00::/8                On-link
  7    281 ff00::/8                On-link
  3    311 ff00::/8                On-link
  6    281 ff00::/8                On-link
===========================================================================
Persistent Routes:
  None

C:\Users\Aditya>
```

ICMP

```
C:\Windows\system32\cmd.exe
   Default Gateway . . . . . . . . . : fe80::8d7f:c201:3905:8194%3
                                       192.168.43.1

C:\Users\Aditya>ping 192.168.43.1

Pinging 192.168.43.1 with 32 bytes of data:
Reply from 192.168.43.1: bytes=32 time=3ms TTL=64
Reply from 192.168.43.1: bytes=32 time=3ms TTL=64
Reply from 192.168.43.1: bytes=32 time=3ms TTL=64
Reply from 192.168.43.1: bytes=32 time=3ms TTL=64

Ping statistics for 192.168.43.1:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 3ms, Maximum = 3ms, Average = 3ms

C:\Users\Aditya>
```

Capturing from WiFi

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

icmp

No.	Time	Source	Destination	Protocol	Length	Info
79	57.429996	192.168.43.126	192.168.43.1	ICMP	74	Echo (ping) request id=0x0001, seq=25/6400, ttl=128 (reply in 80)
80	57.433511	192.168.43.1	192.168.43.126	ICMP	74	Echo (ping) reply id=0x0001, seq=25/6400, ttl=64 (request in 79)
82	58.450687	192.168.43.126	192.168.43.1	ICMP	74	Echo (ping) request id=0x0001, seq=26/6656, ttl=128 (reply in 83)
83	58.454265	192.168.43.1	192.168.43.126	ICMP	74	Echo (ping) reply id=0x0001, seq=26/6656, ttl=64 (request in 82)
85	59.466232	192.168.43.126	192.168.43.1	ICMP	74	Echo (ping) request id=0x0001, seq=27/6912, ttl=128 (reply in 86)
86	59.469827	192.168.43.1	192.168.43.126	ICMP	74	Echo (ping) reply id=0x0001, seq=27/6912, ttl=64 (request in 85)
87	60.481795	192.168.43.126	192.168.43.1	ICMP	74	Echo (ping) request id=0x0001, seq=28/7168, ttl=128 (reply in 88)
88	60.485400	192.168.43.1	192.168.43.126	ICMP	74	Echo (ping) reply id=0x0001, seq=28/7168, ttl=64 (request in 87)

```
> Frame 79: 74 bytes on wire (592 bits), 74 bytes captured (592 bits) on interface 0
> Ethernet II, Src: Cybertan_eb:52:b9 (78:45:61:eb:52:b9), Dst: XiaomiCo_a3:af:5d (18:f0:e4:a3:af:5d)
v Internet Protocol Version 4, Src: 192.168.43.126, Dst: 192.168.43.1
    0100 .... = Version: 4
    .... 0101 = Header Length: 20 bytes (5)
  > Differentiated Services Field: 0x00 (DSCP: CS0, ECN: Not-ECT)
    Total Length: 60
    Identification: 0xa4af (42159)
  > Flags: 0x0000
    Time to live: 128
    Protocol: ICMP (1)
    Header checksum: 0xbe41 [validation disabled]
    [Header checksum status: Unverified]
    Source: 192.168.43.126
    Destination: 192.168.43.1
> Internet Control Message Protocol
```

```
0000  18 f0 e4 a3 af 5d 78 45  61 eb 52 b9 08 00 45 00   .....]xE a·R···E·
```

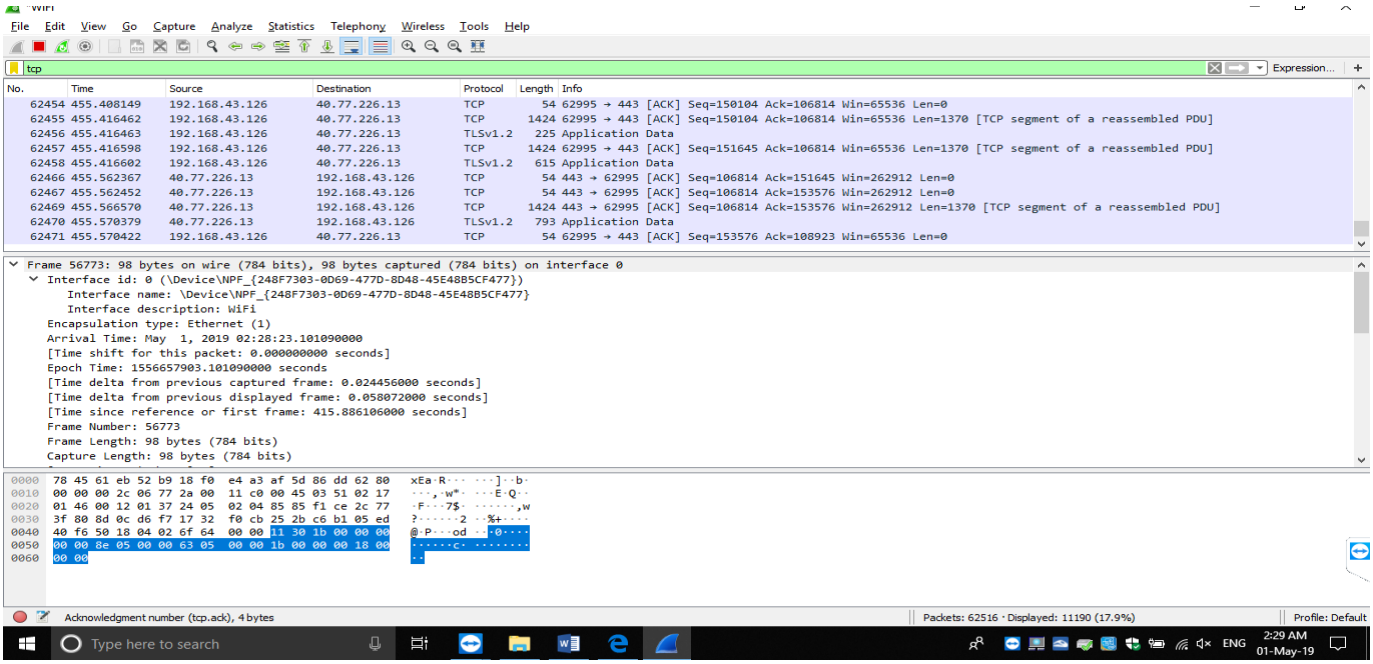
WiFi: <live capture in progress> Packets: 353 · Displayed: 8 (2.3%) Profile: Default

Lab 9:

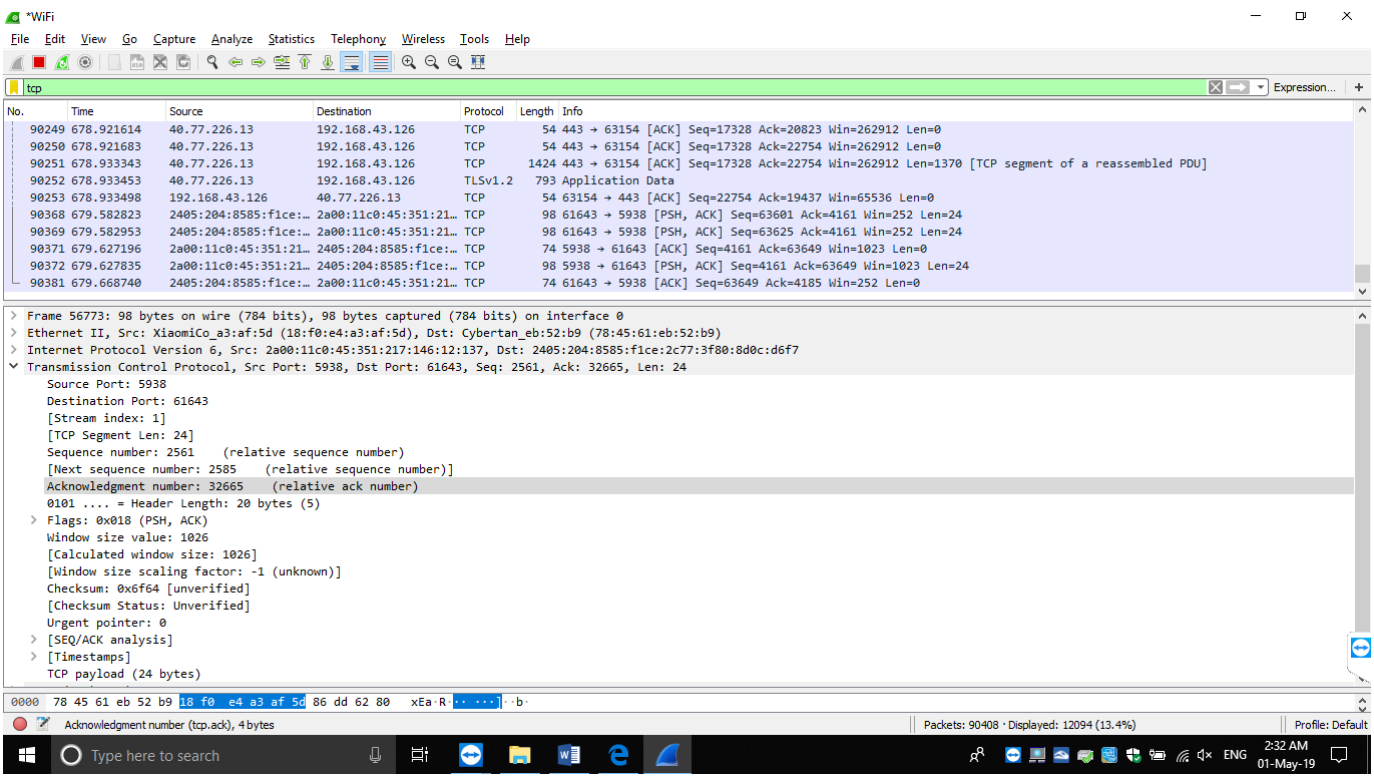
Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: TCP and UDP

TCP

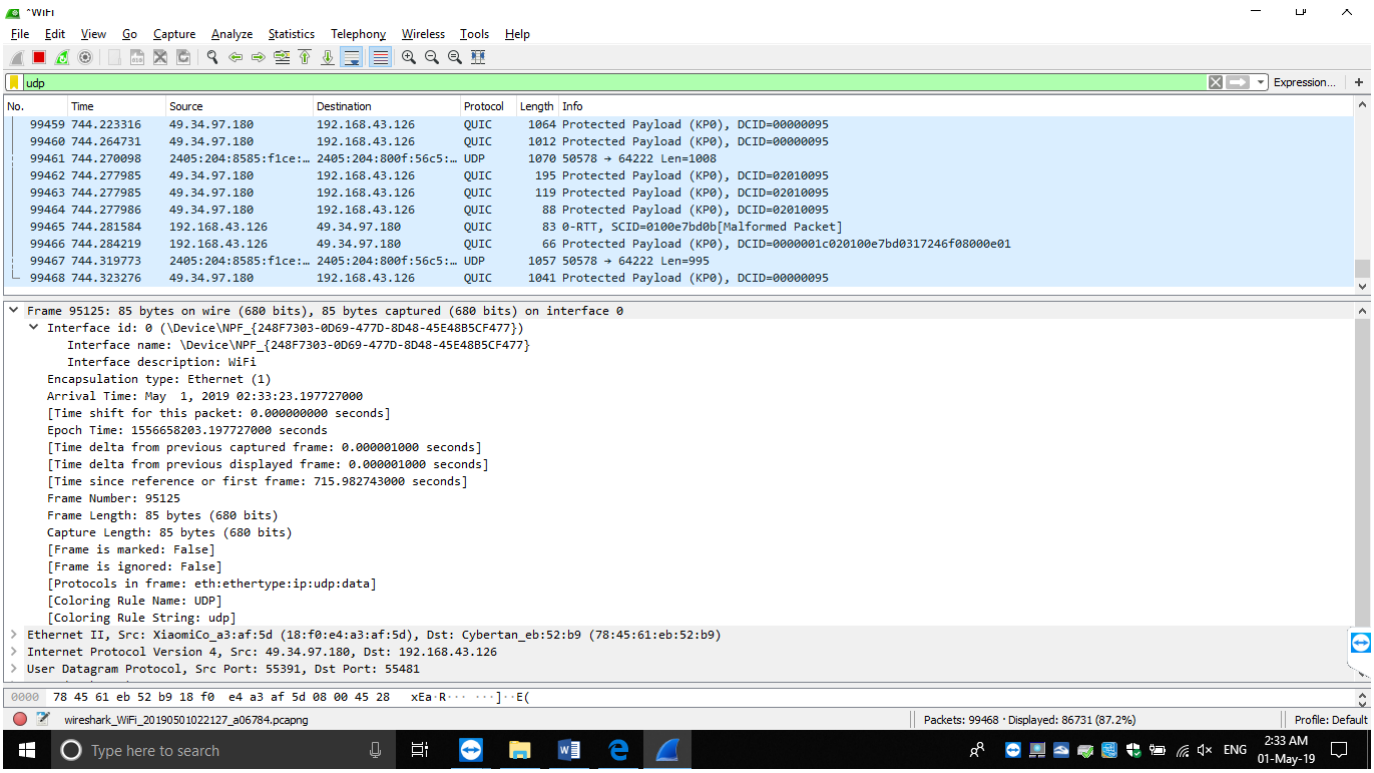
Header



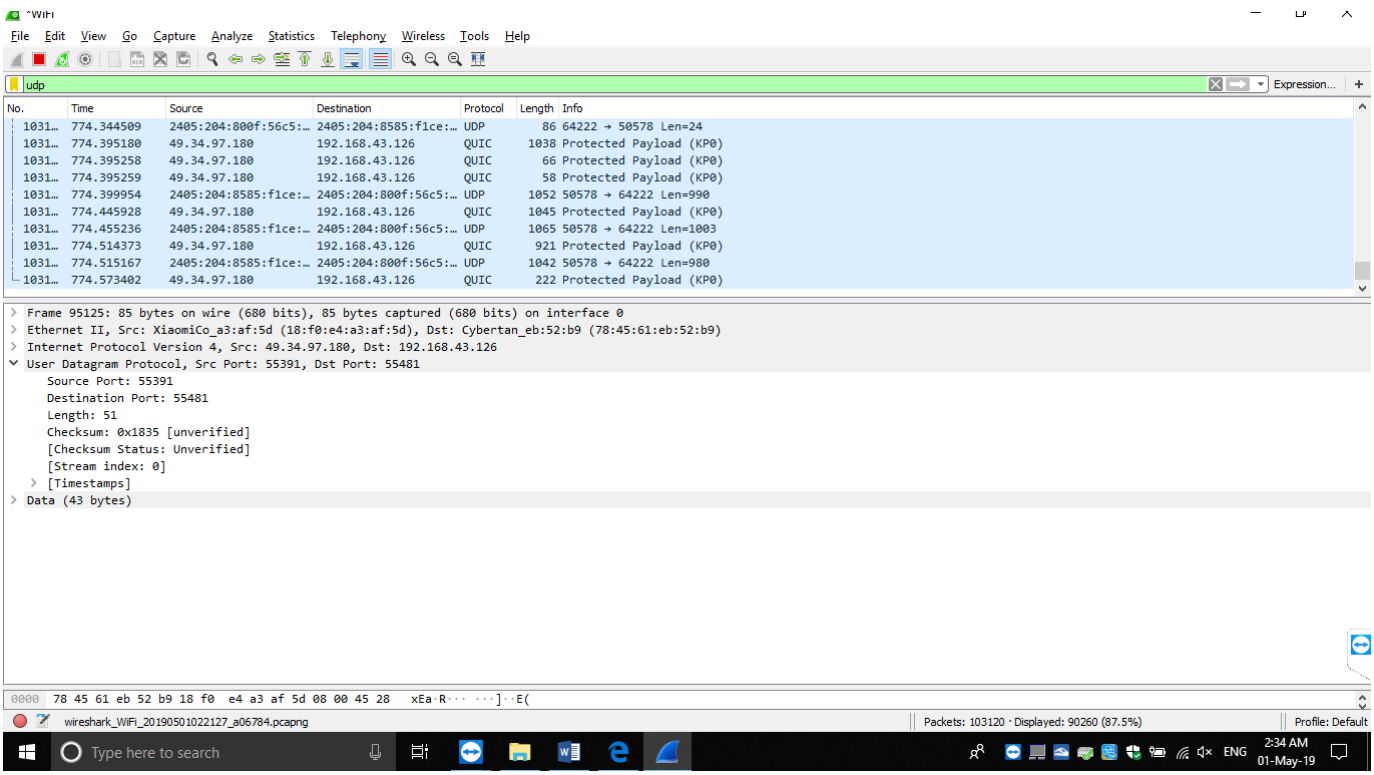
PayLoad



UDP : Header



PayLoad



Lab 10:

Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: HTTP and DNS.

Ans: **HTTP**:-

The Hypertext Transfer Protocol (HTTP) is a wildly successful protocol. However, the way HTTP/1.1 uses the underlying transport has several characteristics that have a negative overall effect on application performance today. All frames begin with a fixed 9-octet header followed by a variable-length payload. Furthermore, HTTP header fields are often repetitive and verbose, causing unnecessary network traffic as well as causing the initial *TCP* congestion window to quickly fill. This can result in excessive latency when multiple requests are made on a new TCP connection.

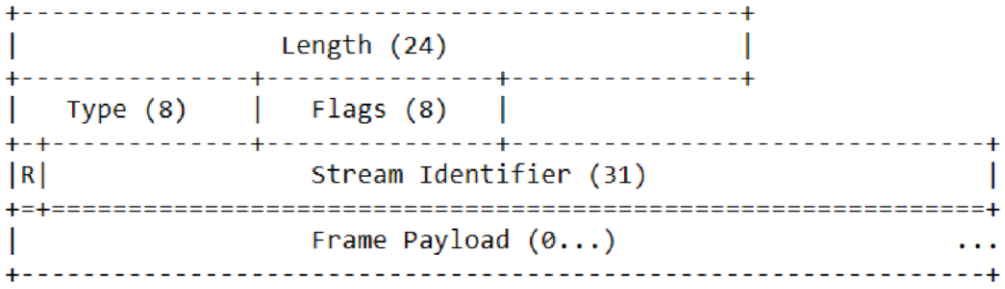


Figure 1: Frame Layout

The fields of the frame header are defined as: -

Length: - The length of the frame payload expressed as an unsigned 24-bit integer. Values greater than 2^{14} (16,384) MUST NOT be sent unless the receiver has set a larger value for SETTINGS_MAX_FRAME_SIZE.

The 9 octets of the frame header are not included in this value.

Type: - The 8-bit type of the frame. The frame type determines the format and semantics of the frame. Implementations MUST ignore and discard any frame that has a type that is unknown.

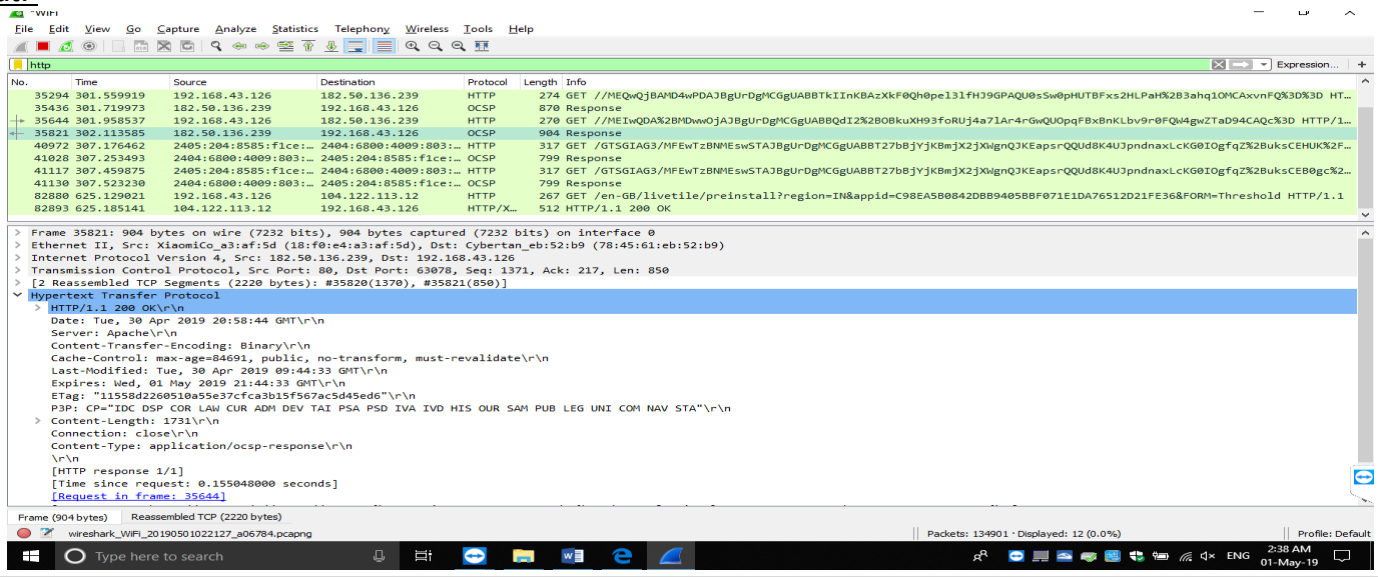
Flags: - An 8-bit field reserved for Boolean flags specific to the frame type. Flags are assigned semantics specific to the indicated frame type. Flags that have no defined semantics for a particular frame type MUST be ignored and MUST be left unset (0x0) when sending.

R: - A reserved 1-bit field. The semantics of this bit are undefined, and the bit MUST remain unset (0x0) when sending and MUST be ignored when receiving.

Stream Identifier: - A stream identifier (see Section 5.1.1) expressed as an unsigned 31-bit integer. The value 0x0 is reserved for frames that are associated with the connection as a whole as opposed to an individual stream.

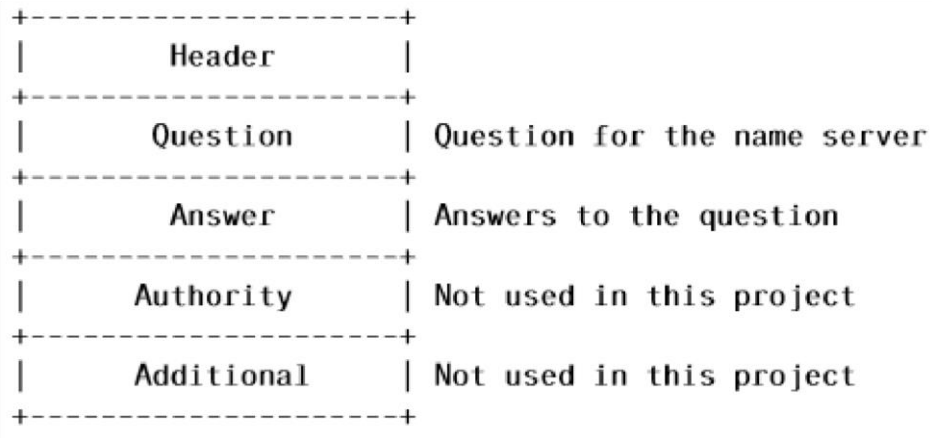
The structure and content of the frame payload is dependent entirely on the frame type

Output:-

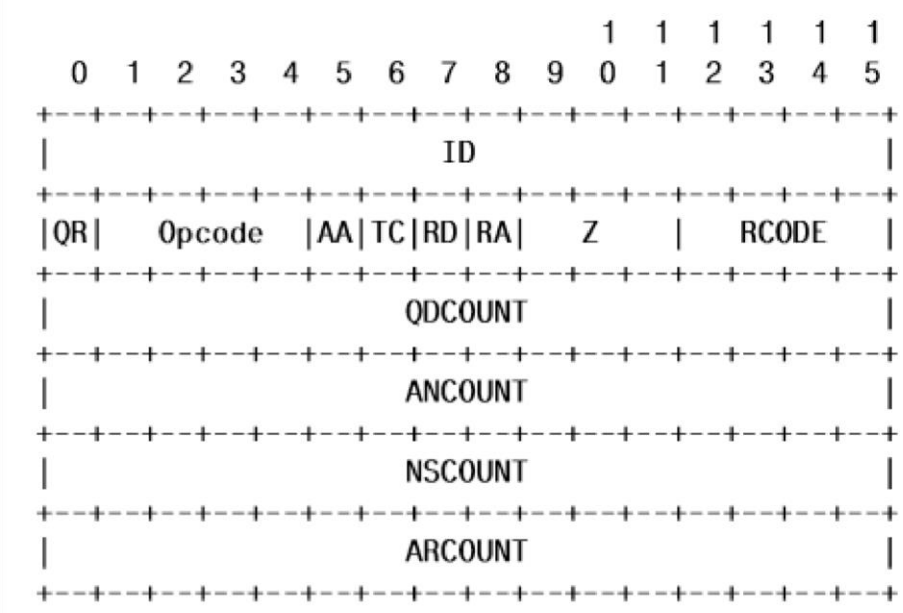


DNS:-

The client/server information exchange in DNS is facilitated using query/response messaging. Both queries and responses have the same general format, containing up to five individual sections carrying information. All DNS packets have a structure that is:-



The header describes the type of packet and which fields are contained in the packet. Following the header are a number of questions, answers, authority records, and additional records. For this project, we will be ignoring the authority and additional fields - your client program must accept packets with such fields, but must ignore them. DNS packets have a header that is shown below. Note that requests and replies follow the same header format.



Where each of these fields is as described below:-

ID: - A 16 bit identifier assigned by the program that generates any kind of query. This identifier is copied the corresponding reply and can be used by the requester to match up replies to outstanding queries. You should always use 1337 for this field.

QR: - A one bit field that specifies whether this message is a query (0), or a response (1). Obviously, you should use 0 for your requests, and expect to see a 1 in the response you receive.

OPCODE: - A four bit field that specifies kind of query in this message. You should use 0, representing a standard query.

AA Authoritative Answer: - This bit is only meaningful in responses, and specifies that the responding name server is an authority for the domain name in question section. You should use this bit to report whether or not the response you receive is authoritative.

TC Truncation: - specifies that this message was truncated. Forth is project, you must exit and return an error if you receive a response that is truncated.

RD Recursion Desired: - this bit directs the name server to pursue the query recursively. You should use 1, representing that you desire recursion.

RA Recursion Available: - this be is set or cleared in a response, and denotes whether recursive query support is available in the name server. Recursive query support is optional. You must exit and return an error if you receive a response that indicates the server does not support recursion.

Z: - Reserved for future use. You must set this field to 0.

RCODE Response code: - this 4 bit field is set as part of responses. The values have the following interpretation:-

0 No error condition
1 Format error - The name server was unable to interpret the query.
2 Server failure - The name server was unable to process this query due to a problem with the name server.
3 Name Error - Meaningful only for responses from an authoritative name server, this code signifies that the domain name referenced in the query does not exist.
4 Not Implemented - The name server does not support the requested kind of query.
5 Refused - The name server refuses to perform the specified operation for policy reasons.

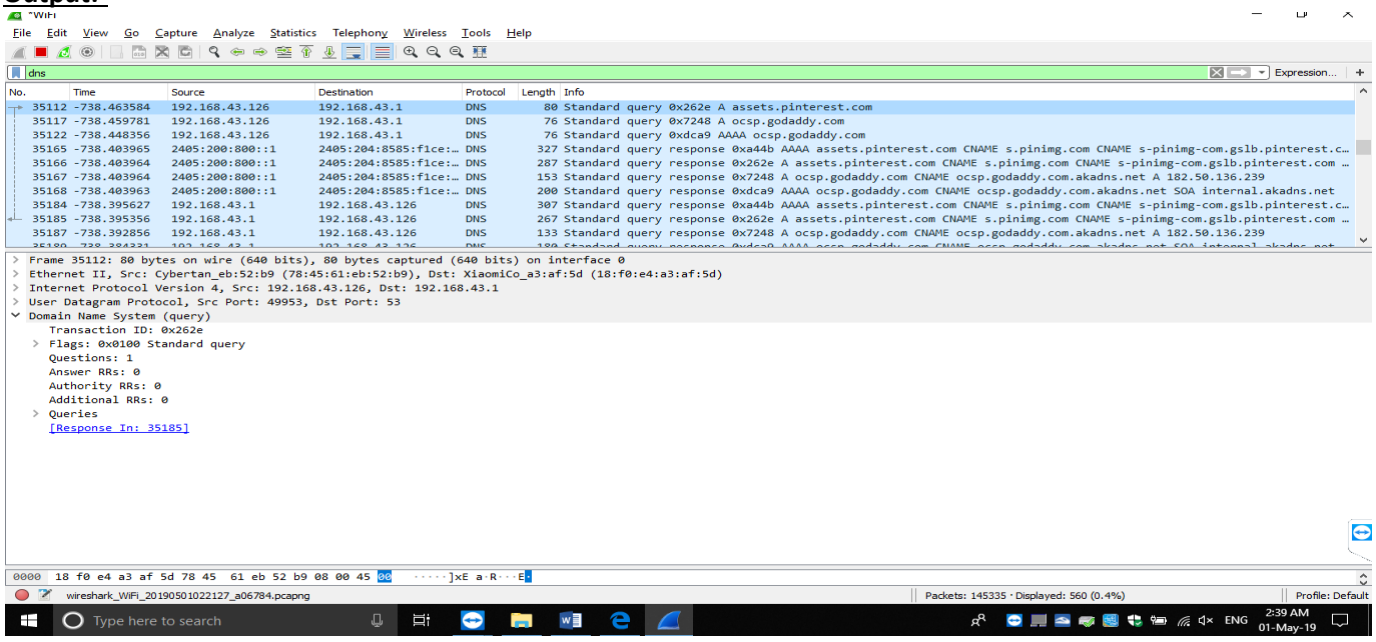
QDCOUNT: - an unsigned 16 bit integer specifying the number of entries in the question section. You should set this field to 1, indicating you have one question.

ANCOUNT: - an unsigned 16 bit integer specifying the number of resource records in the answer section. You should set this field to 0, indicating you are not providing any answers.

NSCOUNT: - an unsigned 16 bit integer specifying the number of name server resource records in the authority records section. You should set this field to 0, and should ignore any response entries in this section.

ARCOUNT: - an unsigned 16 bit integer specifying the number of resource records in the additional records section. You should set this field to 0, and should ignore any response entries in this section.

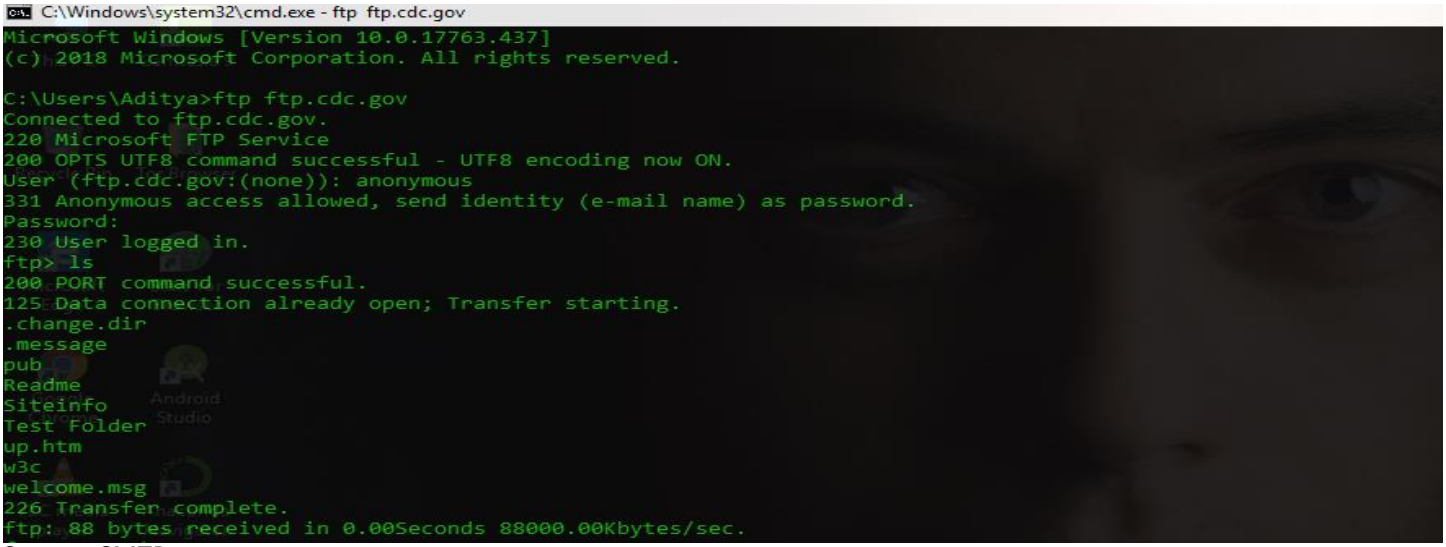
Output:-



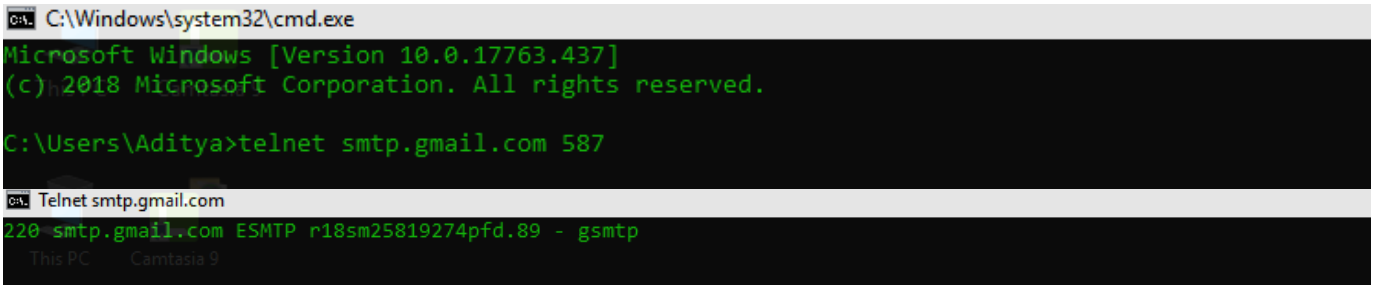
Lab 11:

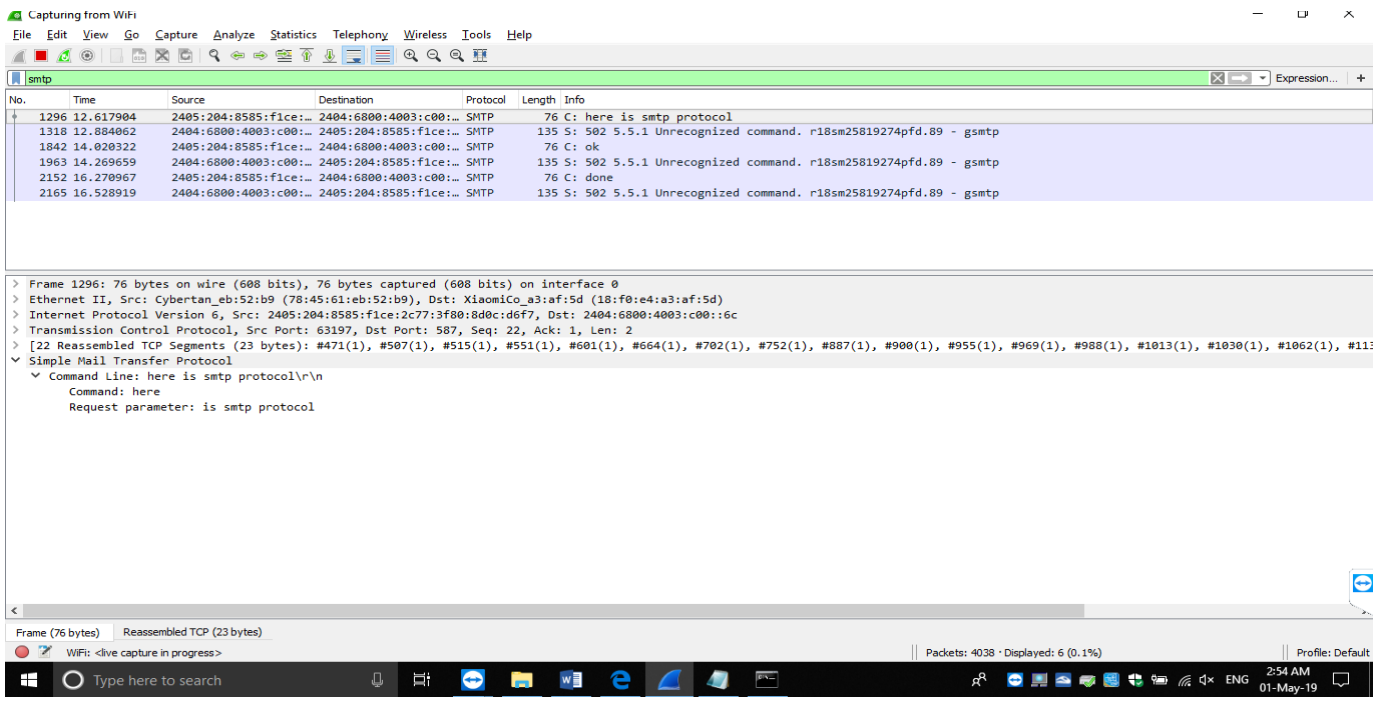
Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: FTP, SMTP and Telnet.

1. FTP

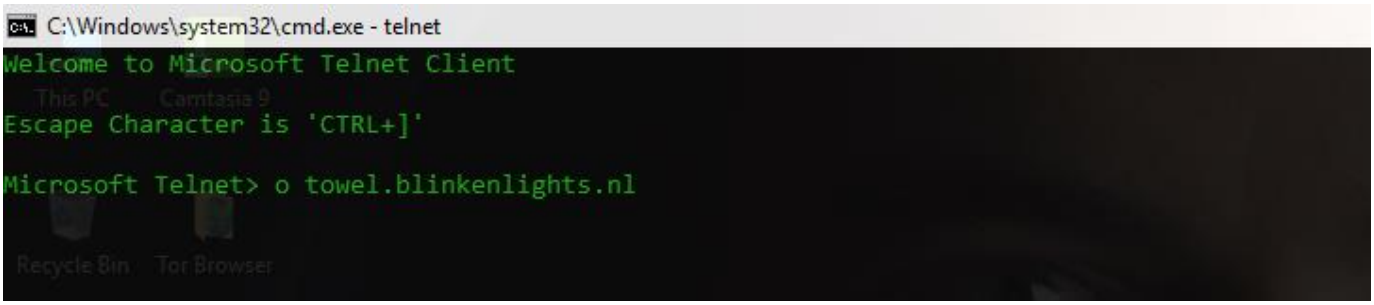


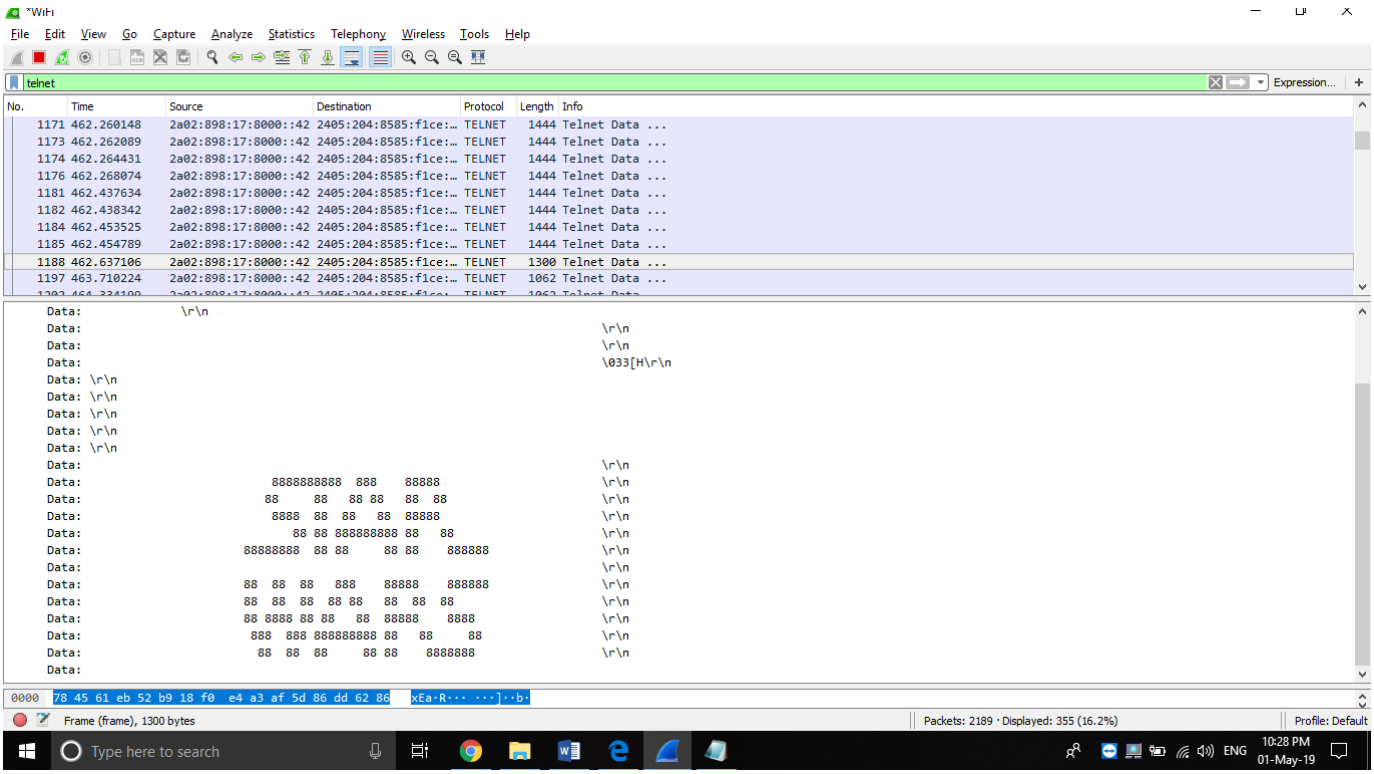
2. SMTP





3. Telnet


C:\Windows\system32\cmd.exe - telnet
Welcome to Microsoft Telnet Client
Escape Character is 'CTRL+]'
Microsoft Telnet> o towel.blinkenlights.nl


File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help
telnet
No. Time Source Destination Protocol Length Info
1171 462.260148 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1173 462.262089 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1174 462.264431 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1176 462.268074 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1181 462.437634 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1182 462.438342 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1184 462.453525 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1185 462.454789 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1444 Telnet Data ...
1188 462.637106 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1300 Telnet Data ...
1197 463.710224 2a02:898:17:8000::42 2405:204:8585:f1ce:… TELNET 1062 Telnet Data ...
Frame (frame), 1300 bytes
Packets: 2189 · Displayed: 355 (16.2%)
Profile: Default
Type here to search
ENG
10:28 PM
01-May-19

Lab 12:

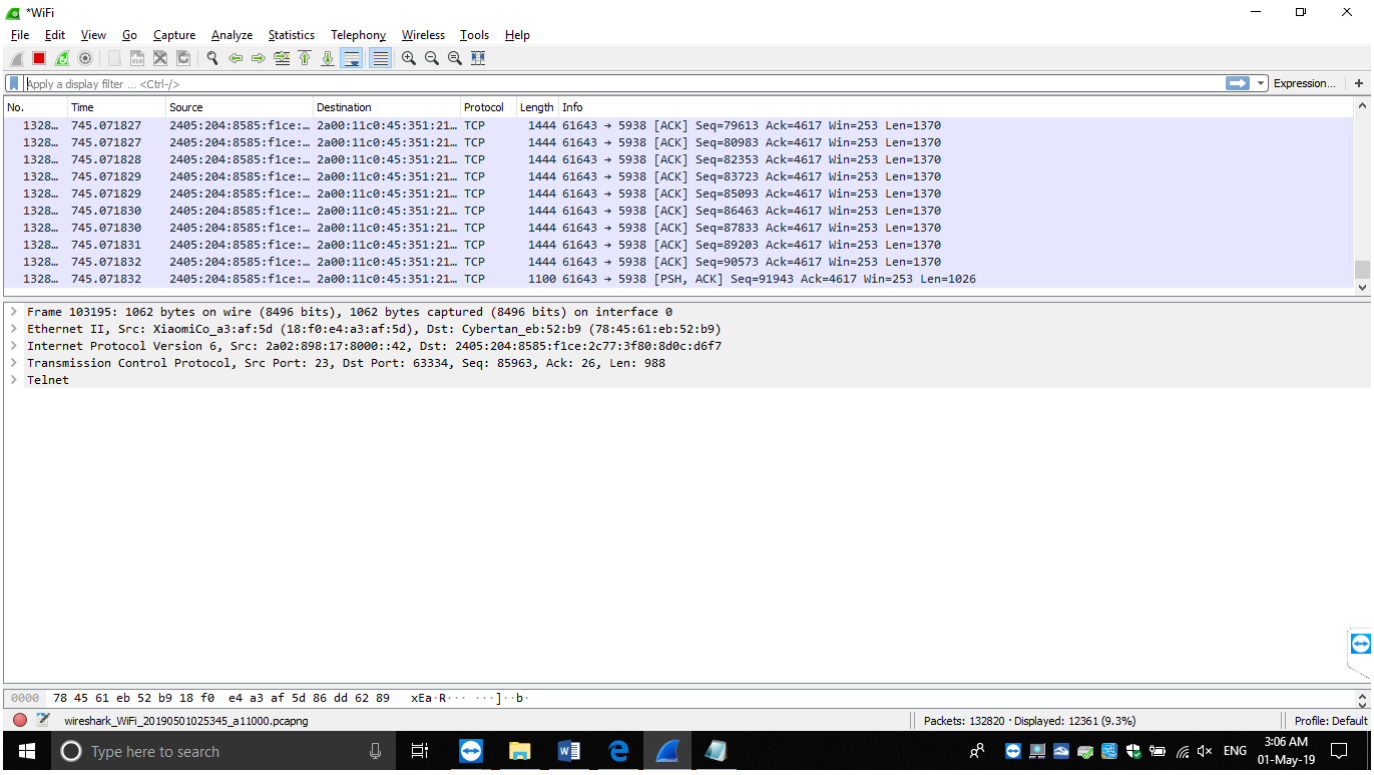
Objective: Capture the following traffic types and Interpret/ Analyze the corresponding header and payload: Ethernet and ARP

Ans: - Same as Lab-7.

Lab 13:

Objective: Capture Wi-FI and Bluetooth Traffic and Interpret/ Analyze the corresponding header and payload using Wireless Traffic Sniffing tools like WireShark-USB/AirCrackng/Kismet, etc.

1.wifi



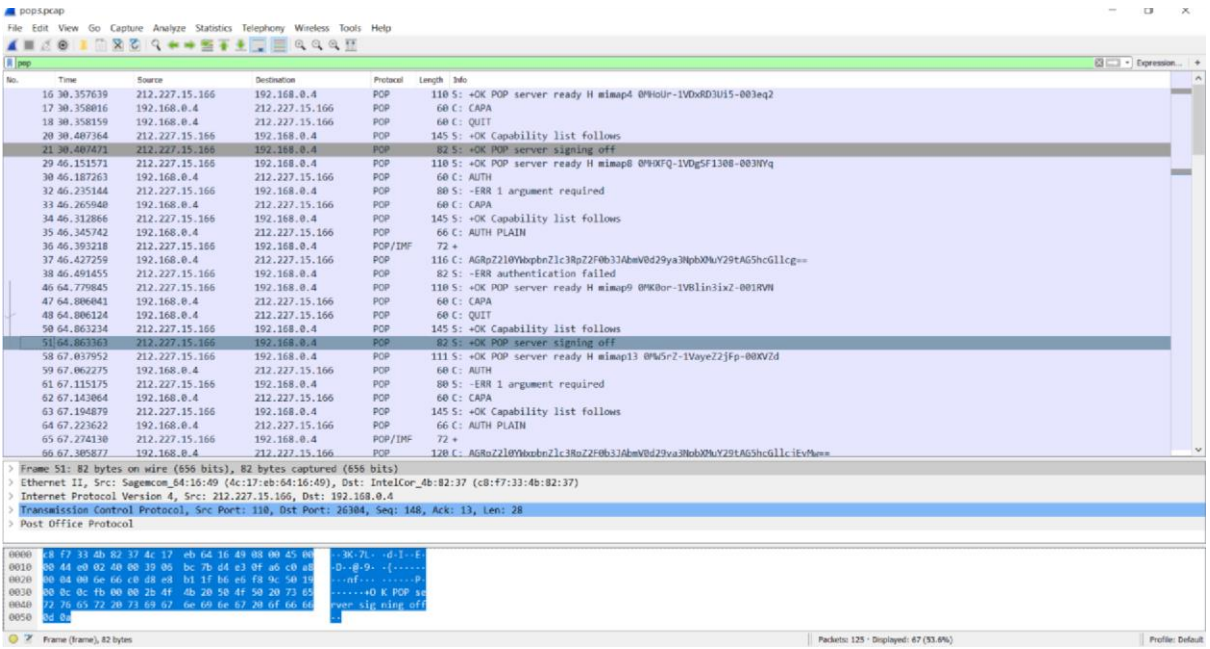
2.Bluetooth

Lab 14:

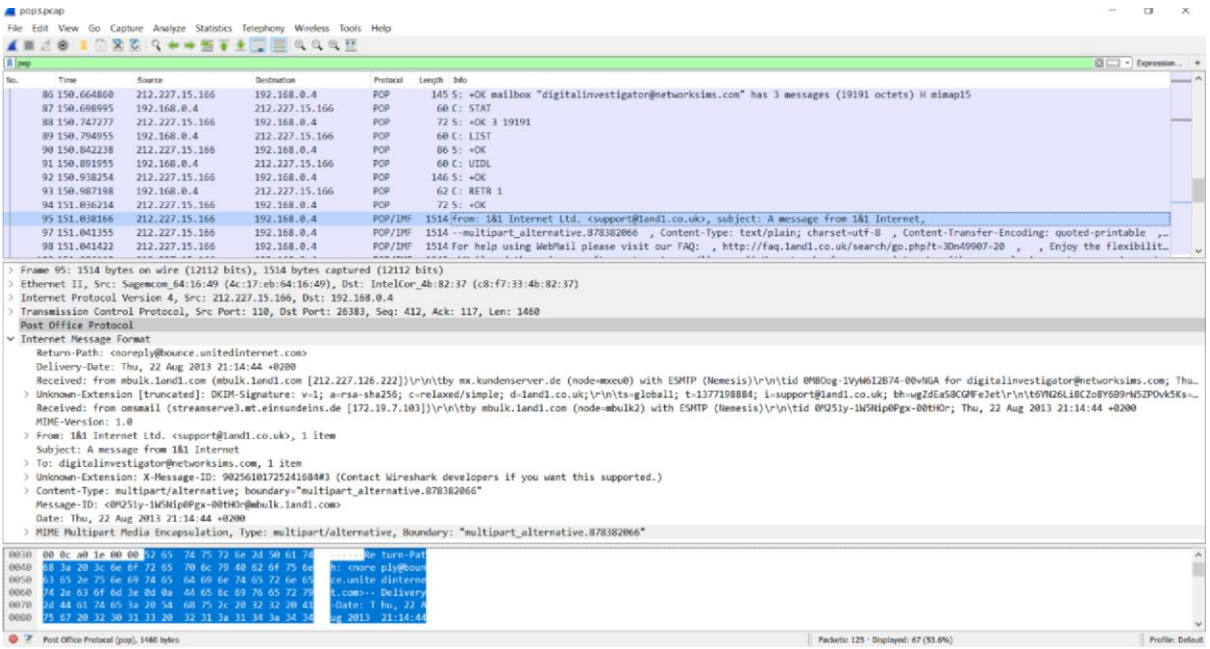
Objective: Analyze Email Traffic: Normal POP Communications, POP Problems, Dissect the POP Packet Structure, Filter on POP Traffic, Normal SMTP Communications, SMTP Problems, Dissect the SMTP Packet Structure, Filter on SMTP Traffic.

POP: Post Office Protocol (POP) is an application layer protocol used by email systems to retrieve mail from email servers. The email client uses POP commands such as LOGIN, LIST, RETR, DELE, QUIT to access and manipulate (retrieve or delete) the email from the server. POP3 uses TCP port 110 and wipes the mail from the server once it is downloaded to the local client.

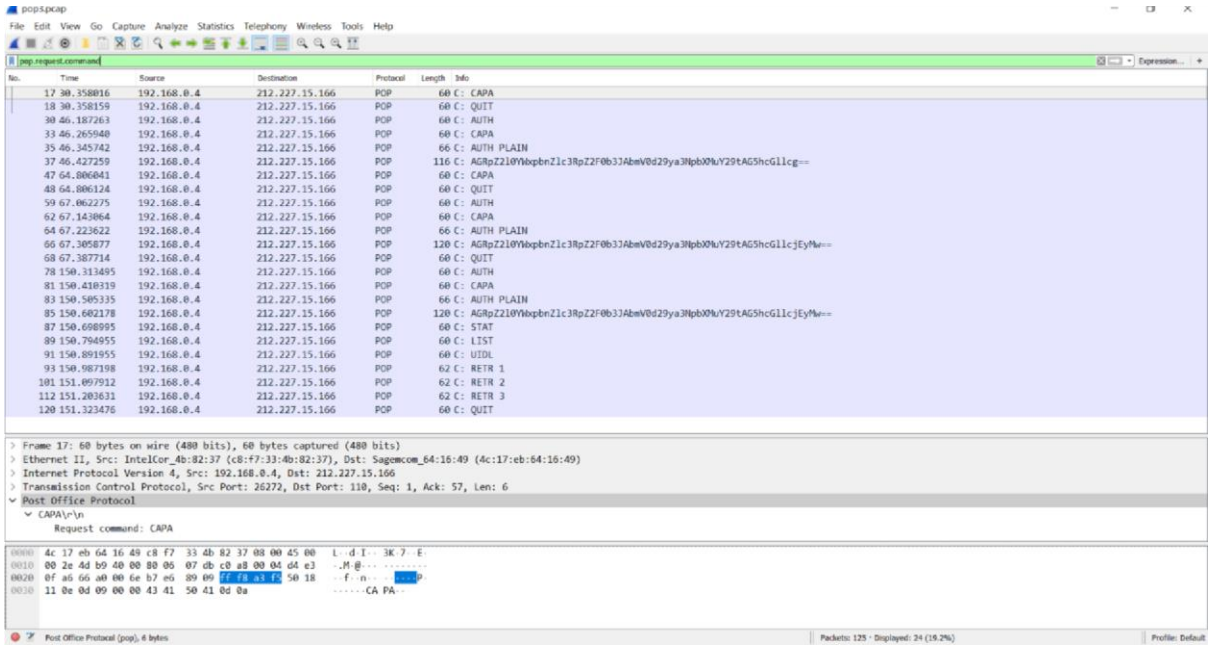
POP communications



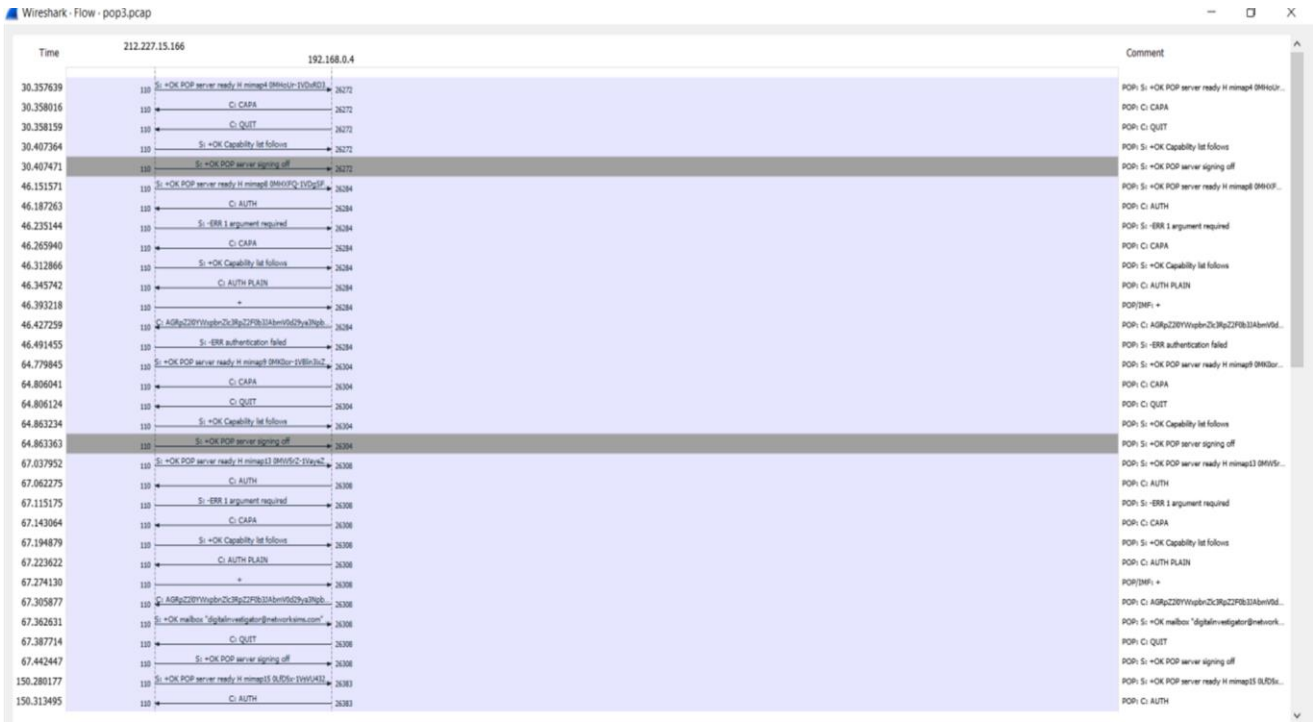
POP packet structure



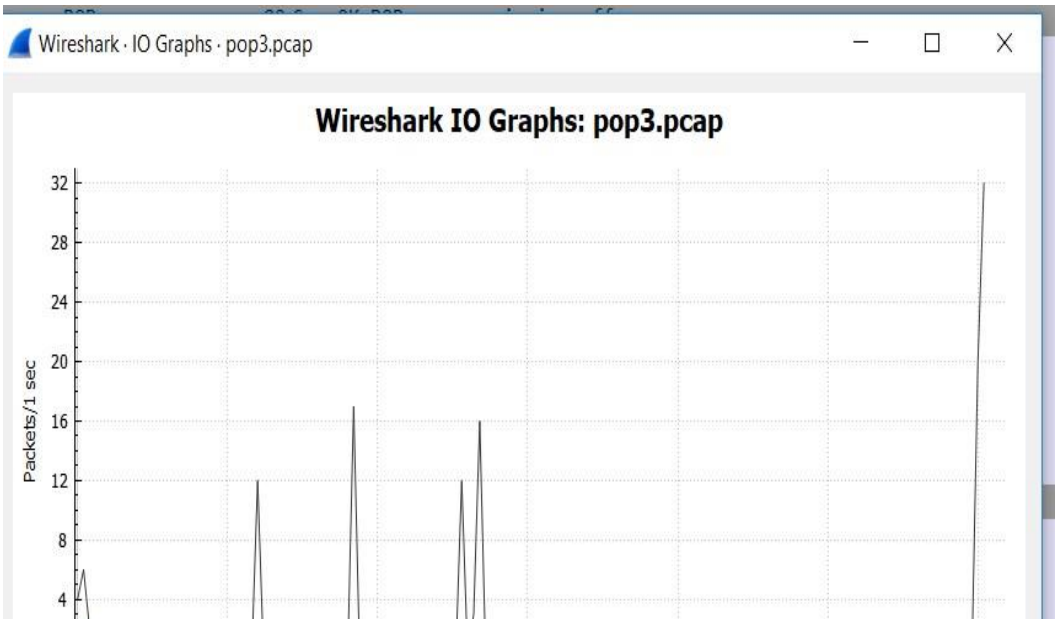
Filter on POP Traffic



Flow graph of POP

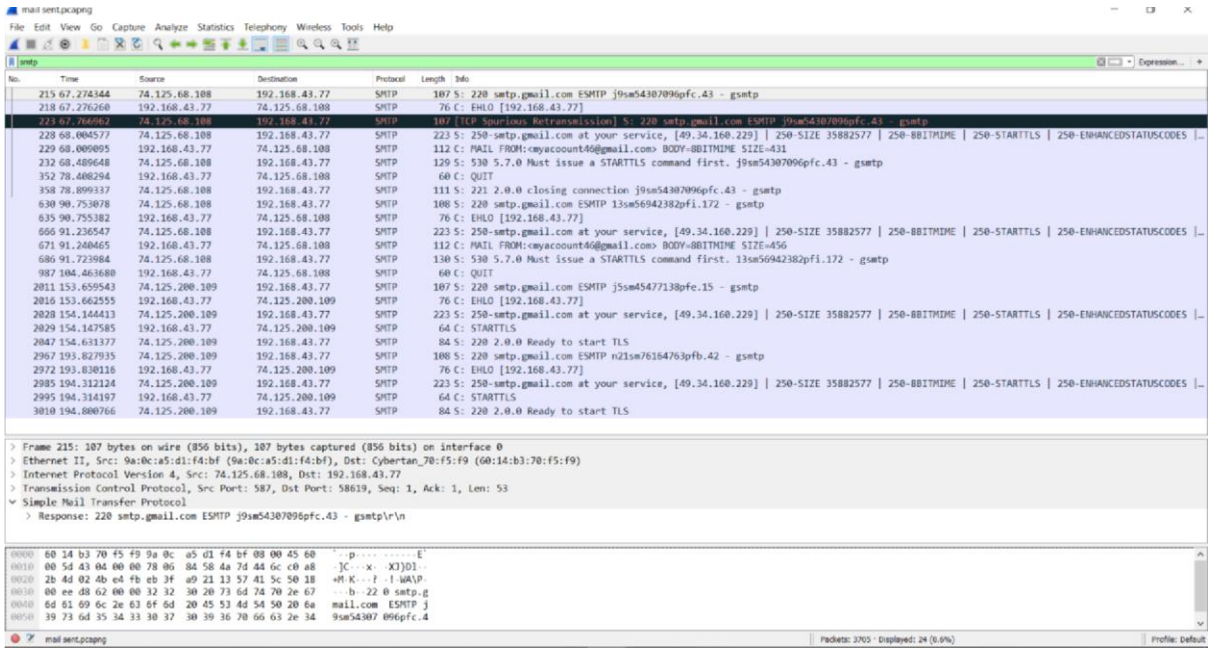


IO graph of POP

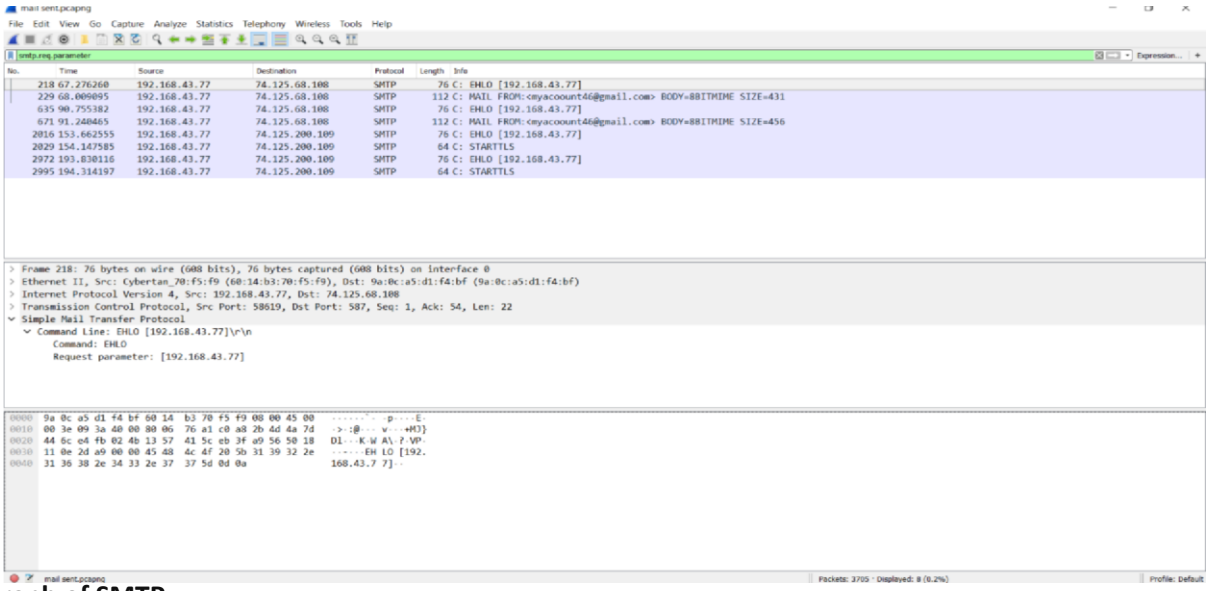


SMTP: **Simple Mail Transfer Protocol** (**SMTP**) is an application layer protocol that is used to send email from the client to the mail server. When the sender and receiver are in different email domains, SMTP helps to exchange the mail between servers in different domains. It uses TCP port 25:

SMTP Communication

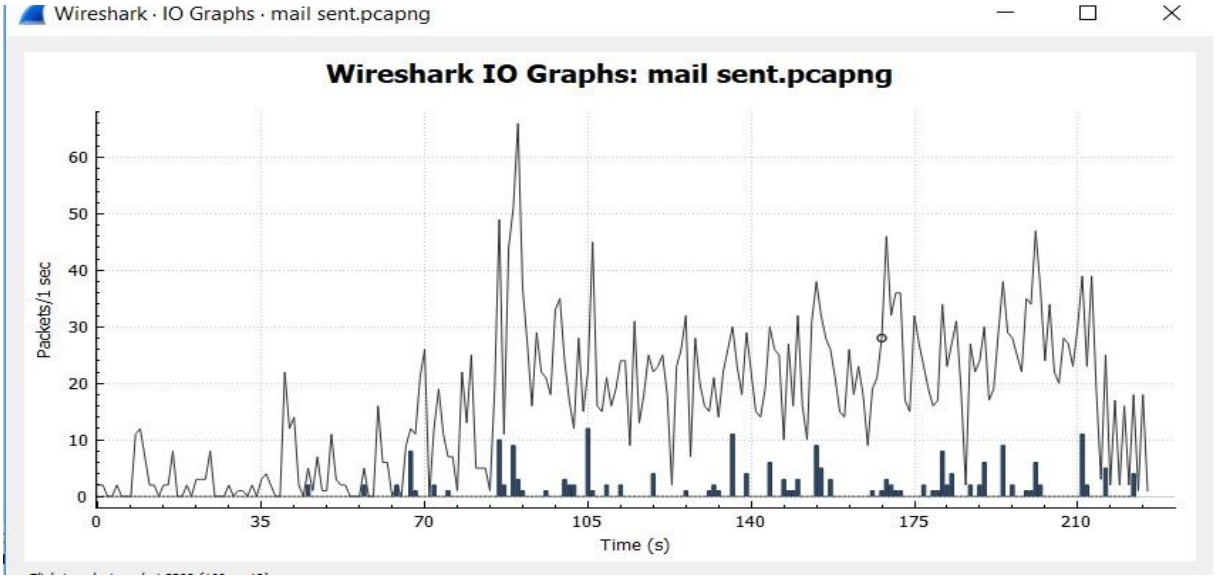


Filter on SMTP Traffic



Flow graph of SMTP

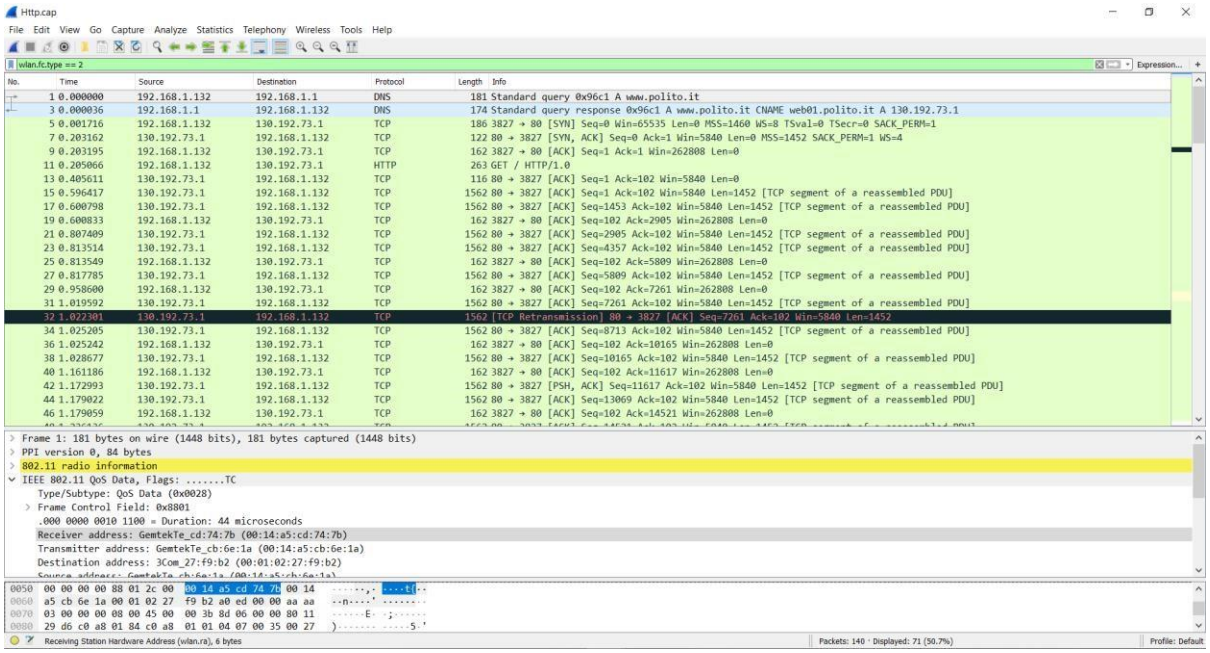
IO graph of SMTP



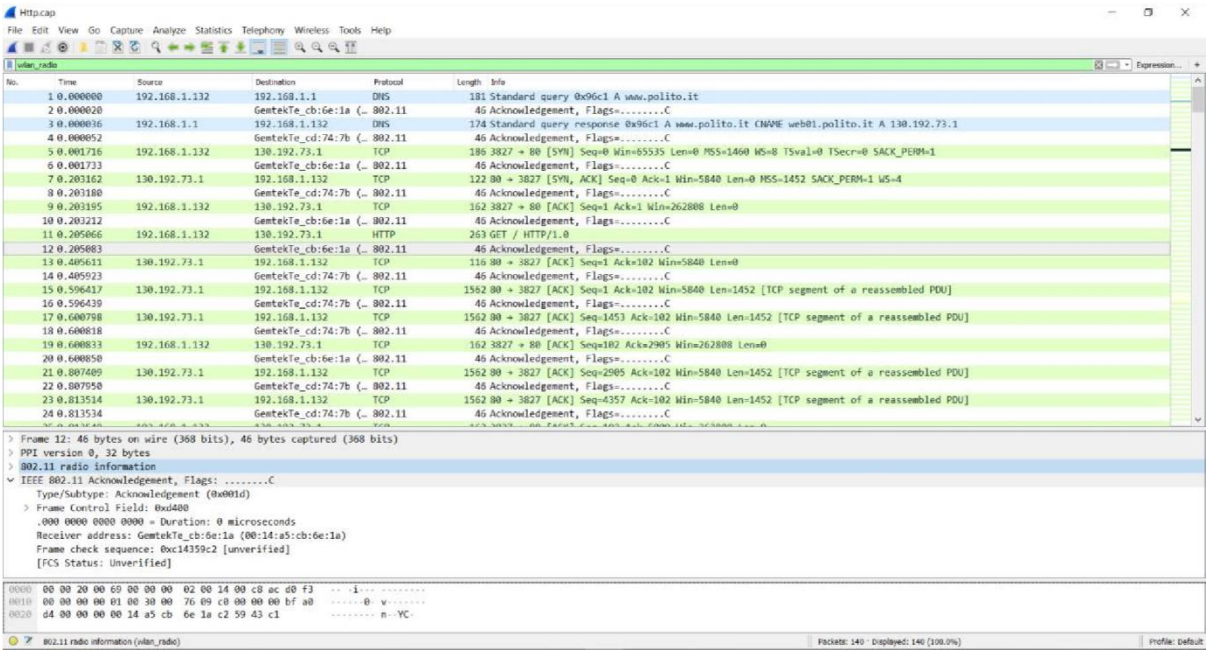
Lab 15:

Objective: Analyze IEEE 802.11 (WLAN): Wireless LANs (WLANs) Traffic, Signal Strength and Interference, Capture WLAN Traffic, 802.11 Traffic Basics like Data Frame, Normal 802.11 Communications

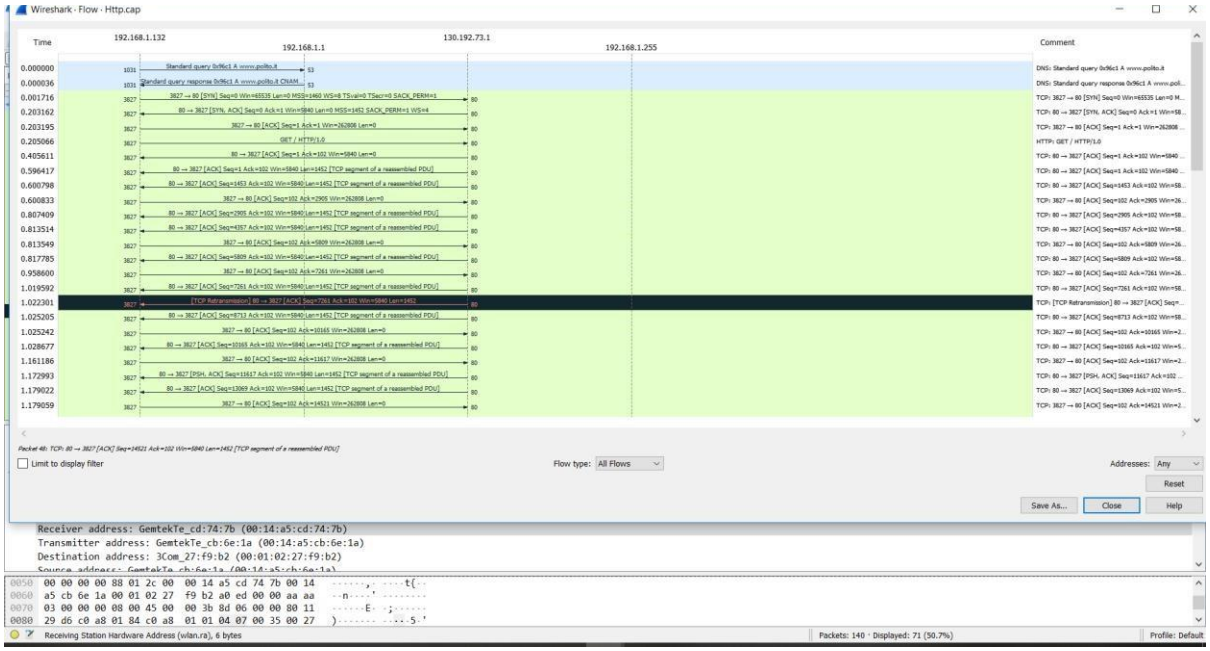
802.11 Communication



Filter on 802.11 Traffic



Flow graph of 802.11



IO graph of 802.11