



Practical Wireshark

Theory ⇒

What is Wireshark ?

- Wireshark is **network traffic analyzer** tool
- Very essential tool for any security professional or system admins
- A network packet analyzer will try to **capture network packets** and tries to **display that packet data** as details as possible
- Best tool for **troubleshooting issues in your network**

Wireshark Features ⇒

- Capture live packet data from a **network interface or more**
- Open **files containing packet data** captured with tcpdump/WinDUMP, and many other packet capture programs
- Display Packets with **very detailed protocol information**
- **Filer** and **Search** for packets is very useful
- Colorization of Packet is most useful feature of Wireshark



Remember ⇒

Wireshark is not tool for **Intrusion Detection System [IDS]**

Also, Wireshark will not **manipulate things on the network**

Wireshark is only use to **measure packets**

Wireshark doesn't send packets on the network

Who Use Wireshark ?

- Network Administrator
 - use it to **troubleshoot network problems, monitoring**
- QA Engineers
 - use it to **verify network application**
- Network Security Engineers
 - use it to **examine security problems**
- Developers
 - use it to **debug protocol implementations**

So, what actually Wireshark do ?

Wireshark intercepts traffic and converts that binary traffic into human-readable format.

This makes it easy to identify what traffic is crossing your network, how much of it, how frequently, how much latency there is between certain hops, and so forth.

What things you know from Wireshark as important concept ?

- Analyze IP Packets
- What is Packet ?
- Protocols

A Packet is a single message from any network protocol (i.e., TCP, DNS, etc.)

How to Download wireshark ?

Downloading and installing Wireshark is easy.

Step one is to check the official Wireshark Download page for the operating system you need.

The basic version of Wireshark is free.

The current stable release of Wireshark is 3.4.5. It supersedes all previous releases.

Stable Release (3.4.5)

- Windows Installer (64-bit)
- Windows Installer (32-bit)
- Windows PortableApps® (32-bit)
- macOS Intel 64-bit .dmg
- Source Code

Old Stable Release (3.2.13)

Documentation

Go Beyond with Riverbed Technology

Riverbed is Wireshark's primary sponsor and provides our funding. They also make great products that fully integrate with Wireshark.

I have a lot of traffic...

ANSWER: SteelCentral™ AppResponse 11

- Full stack analysis – from packets to pages
- Rich performance metrics & pre-defined insights for fast problem identification/resolution
- Modular, flexible solution for deeply-analyzing network & application performance

[Learn More](#)

Download wireshark according to your operating system

You can download wireshark for windows and mac os from there

Wireshark for Linux

Installing Wireshark on Linux can be a little different depending on the Linux distribution.

Common steps in linux to download wireshark and configure

```
sudo apt install wireshark
```

```
sudo dpkg-reconfigure wireshark-common
```

```
sudo adduser $USER wireshark
```



For Kali Linux ⇒

Wireshark is probably already installed! It's part of the basic package.

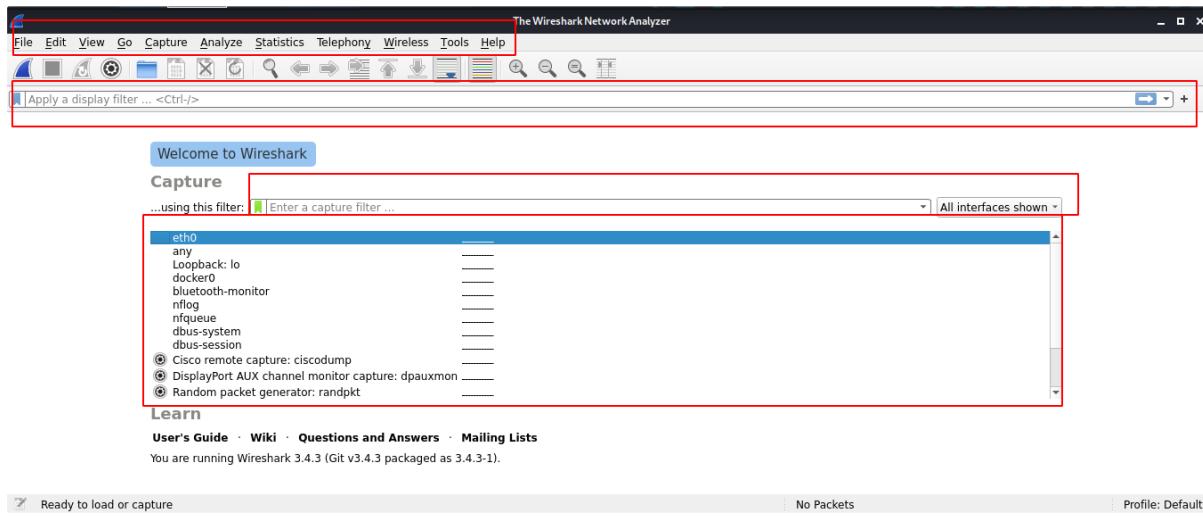
Check your menu to verify.

It's under the menu option "Sniffing & Spoofing."

You can run wireshark in terminal also ⇒ by just typing `wireshark` in terminal console

Start Wireshark from Kali Linux

- Type `sudo wireshark` in terminal of kali linux



- Highlighted things is important for us to know

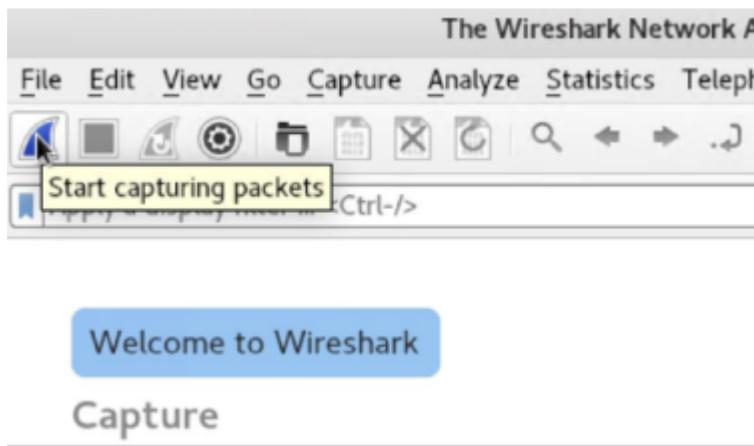


Remember ⇒

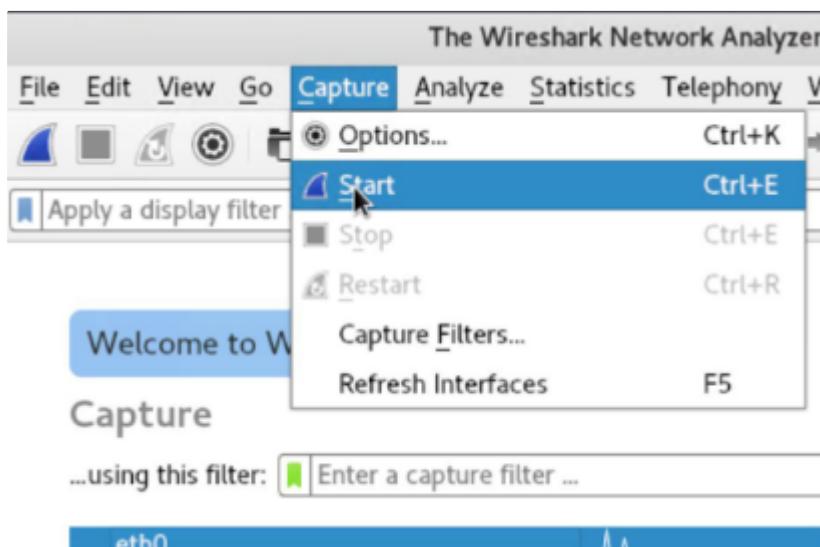
In virtual machine , you will not find **inbuilt wireless adapter (wlan0)**
Because virtual machines only have **virtual (indirect) access to your hardware**

So, in virtual machine you will always find a wired network adapter ⇒
[eth0]

- You can select one or more of the network interfaces using "shift left-click."
- Once you have the network interface selected, you can start the capture, and there are several ways to do that
- Click the first button on the toolbar, titled "Start Capturing Packets."



- You can select the menu item Capture -> Start.



- Or you could use the keystroke Control – E.
- During the capture, Wireshark will show you the packets that it captures in real-time.

Apply a display filter ... <Ctrl-/>

No.	Time	Source	Destination	Protocol	Length	Info
8	61.440392100	192.168.0.3	192.168.0.1	TCP	66	52060 → 445 [ACK]
9	66.559903000	Microsoft_00:15:5d:00:15:5d	Microsoft_00:15:5d:00:15:5d	ARP	42	Who has 192.168.0.1?
10	66.561858700	Microsoft_00:15:5d:00:15:5d	Microsoft_00:15:5d:00:15:5d	ARP	42	192.168.0.1 is at
11	83.533524600	fe80::2c14:87e5:857...	ff02::1:2	DHCPv6	164	Solicit XID: 0xcd5
12	84.545422700	fe80::2c14:87e5:857...	ff02::1:2	DHCPv6	164	Solicit XID: 0xcd5
13	86.549466300	fe80::2c14:87e5:857...	ff02::1:2	DHCPv6	164	Solicit XID: 0xcd5
14	90.565378200	fe80::2c14:87e5:857...	ff02::1:2	DHCPv6	164	Solicit XID: 0xcd5

```

Frame 1: 138 bytes on wire (1104 bits), 138 bytes captured (1104 bits) on interface 0
Ethernet II, Src: Microsoft_00:15:5d (00:15:5d:00:15:5d), Dst: Microsoft_00:15:5d (00:15:5d:00:15:5d)
Internet Protocol Version 4, Src: 192.168.0.3, Dst: 192.168.0.1
Transmission Control Protocol, Src Port: 52060, Dst Port: 445, Seq: 1, Ack: 1, Len: 72
NetBIOS Session Service
SMB2 (Server Message Block Protocol version 2)

```

Hex	Dec
0000 00 15 5d d0 8b 01 00 15	5d d0 8b 00 00 45 00
0010 00 7c 55 e5 40 00 40 06	63 42 c0 a8 00 03 c0 a8
0020 00 01 cb 5c 01 bd a6 a7	f5 0b 10 a1 ac 33 80 18
0030 04 04 01 02 00 00 01 04	00 00 31 00 00 00 00 04

- Once you have captured all the packets you need, you use the same buttons or menu options to stop the capture.

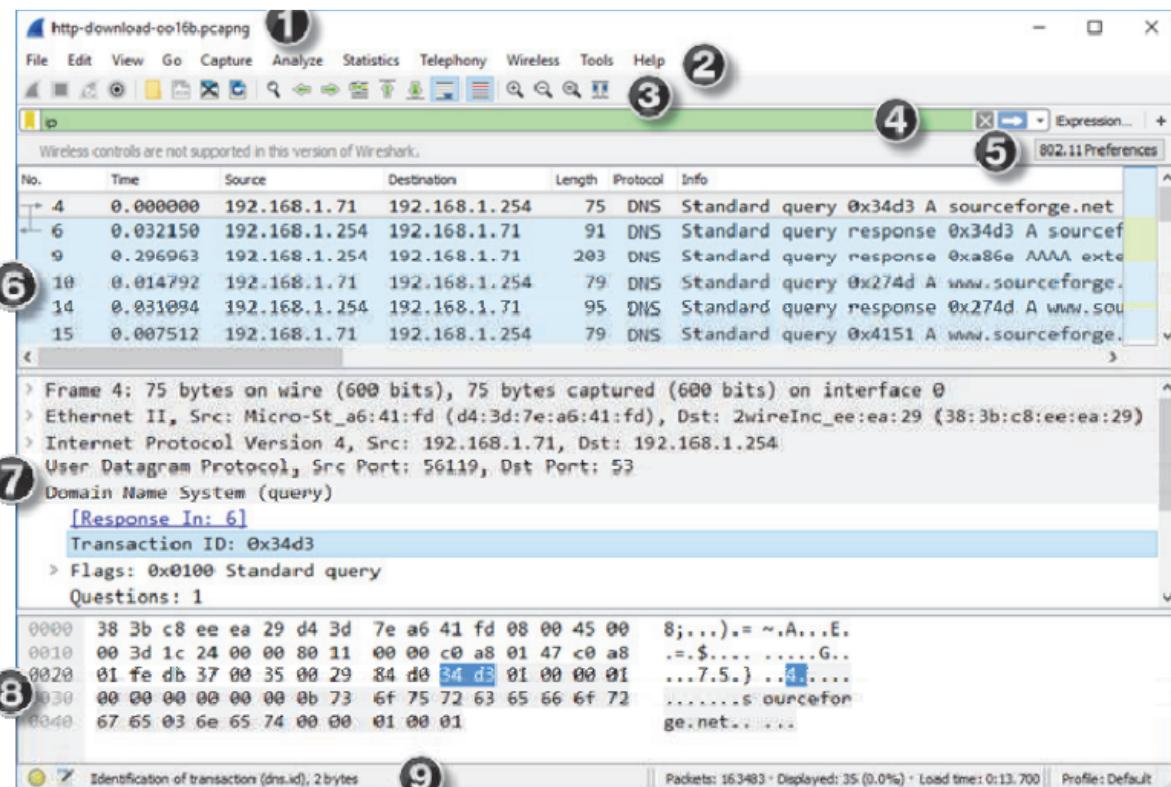


Best practice says that you should stop Wireshark packet capture before you do analysis.

Before analyzing packet , you must know some basic things

- Wireshark shows you three different panes for inspecting packet data
- The Packet List, the top pane, is a list of all the packets in the capture
- When you click on a packet, the other two panes change to show you the details about the selected packet
- You can also tell if the packet is part of a conversation.

Graphical Interface of wireshark ⇒



- (1) **Title bar** – trace file name, capture source, or “The Wireshark Network Analyzer”
- (2) **Main menu** – standard menu
- (3) **Main toolbar** – learn to use this set of icon buttons!
- (4) **Display Filter and Filter Expressions area** – focus on specific traffic
- (5) **Wireless toolbar** – define 802.11 settings
- (6) **Packet List pane** – packet relationship indicator and summary of each frame
- (7) **Packet Details pane** – dissected frames
- (8) **Packet Bytes pane** – hex and ASCII details
- (9) **Status Bar** – access to the Expert, annotations, packet counts, and profiles

You will confuse about their icons for what meaning and purpose, So ⇒

Toolbar Icon	Toolbar Item	Description
	Start	Starts capturing packets with the same options as the last capture or the default options if none were set
	Stop	Stops the currently running capture
	Restart	Restarts the current capture session
	Options	Opens the “Capture Options” dialog box
	Open	Opens the file open dialog box, which allows you to load a capture file for viewing
	Save As	Save the current capture file to whatever file you would like
	Close	Closes the current capture. If you have not saved the capture, you will be asked to save it first
	Reload	Reloads the current capture file
	Find packet	Find a packet based on different criteria
	Go back	Jump back in the packet history. Hold down the Alt key (Option key on macOS) to go forward in the selection history

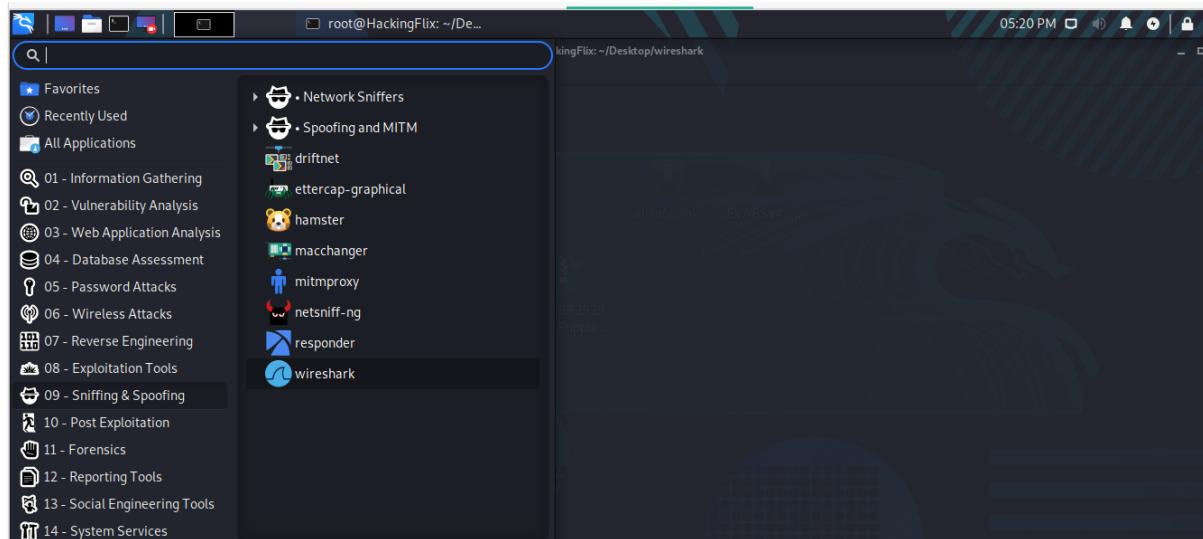
Toolbar Icon	Toolbar Item	Description
	Go Forward	Jump forward in the packet history. Hold down the Alt key (Option on macOS) to go forward in the selection history.
	Go To Packet	Go to a specific packet
	Go To First Packet	Jump to the last packet of the capture file
	Go To Last Packet	Jump to the last packet of the capture file
	Auto Scroll in Live capture	Auto scroll packet list while doing a live capture (or not)
	Colorize	Colorize the packet list (or not)
	Zoom In	Zoom out of the packet data (increase the font size)
	Zoom Out	Zoom out of the packet data (decrease the font size)
	Normal Size	Set zoom level back to 100%
	Resize Columns	Resize columns, so the content fits into them.

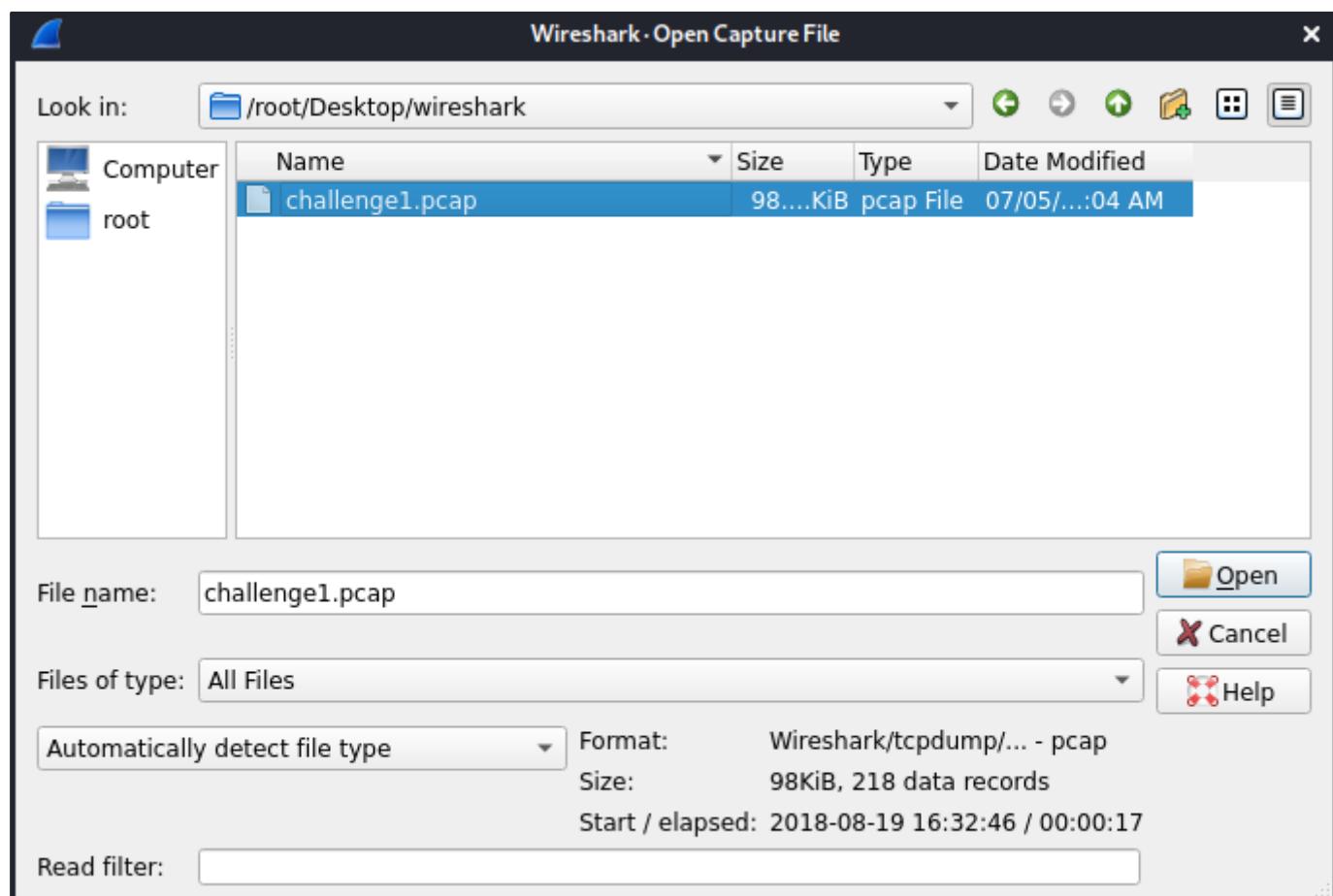
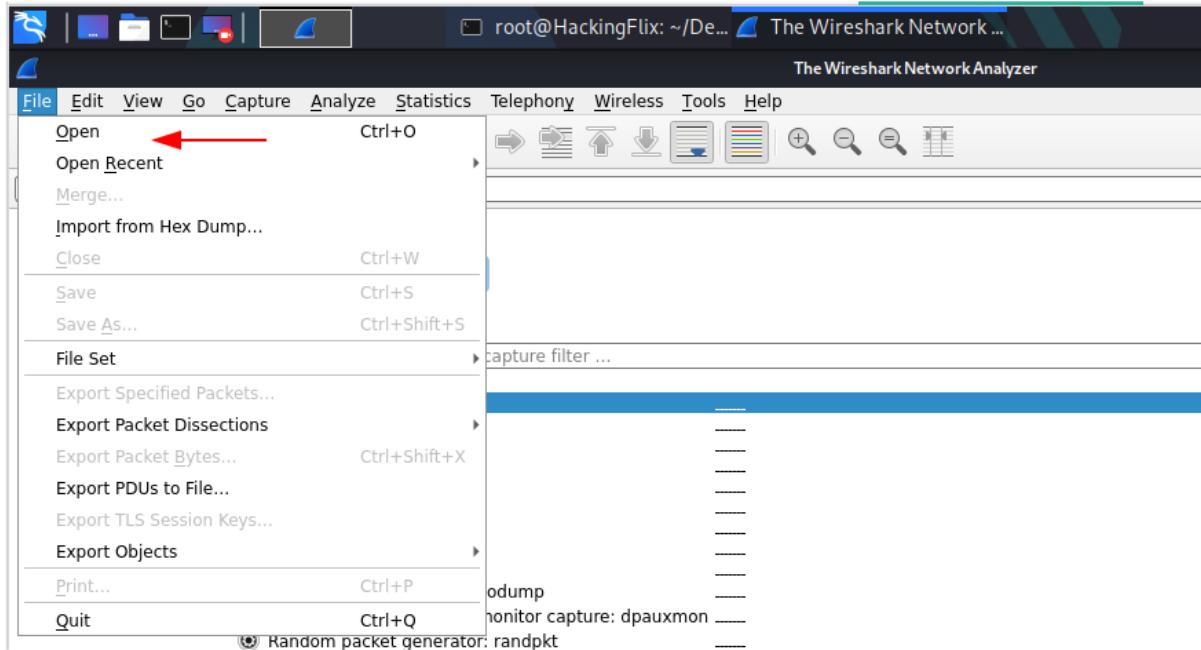
Toolbar Icon	Name	Description
	Bookmarks	Manage or select save filters.
	Filter input	The area is provided to enter or edit a display string. A syntax check of your filter string while you are typing such as the background will turn red if you enter an incomplete or invalid string and will become green when you enter a valid string. Apply a display filter ..
	Clear	Reset the current display filter and clear the edit area.
	Apply	Apply the current value in the edit area as the new display filter.
	Recent	Select from a list of recently applied filters.
	Add Button	Add a new filter button.

Practical Time

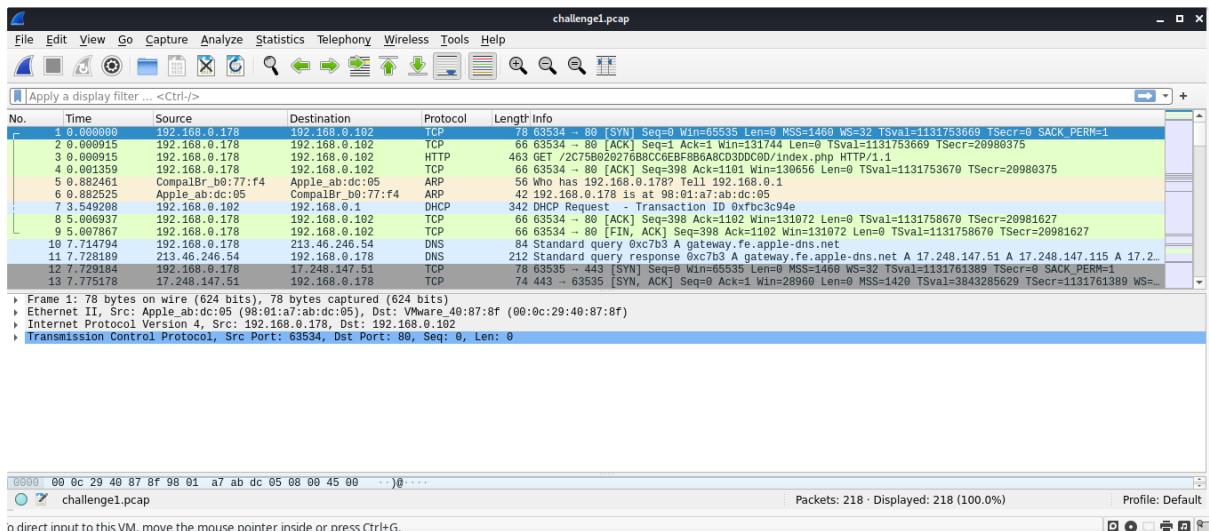
Theory is enough for starting, Now time to learn wireshark practically, from various pcap files to analyze them and find useful information from there

challenge1.pcap





Open challenge1.pcap file



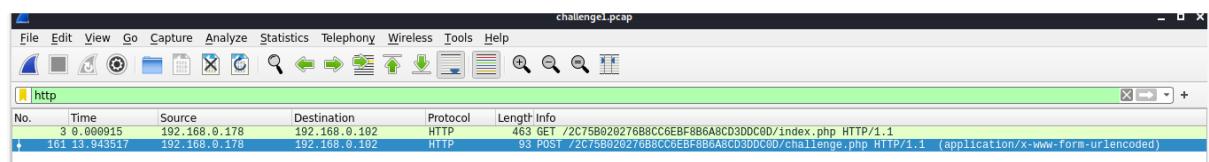
Now, we will do packet analyzes task by task

Task 1

Find out what HOST is using in pcap file ?

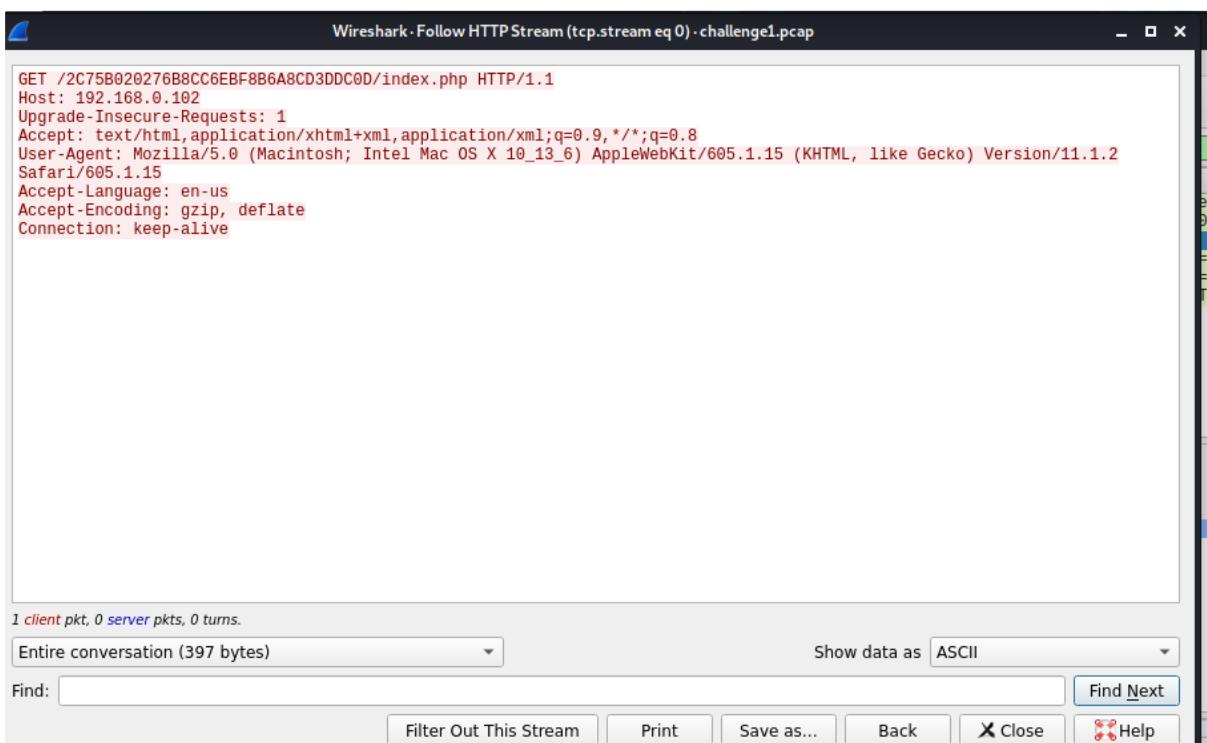
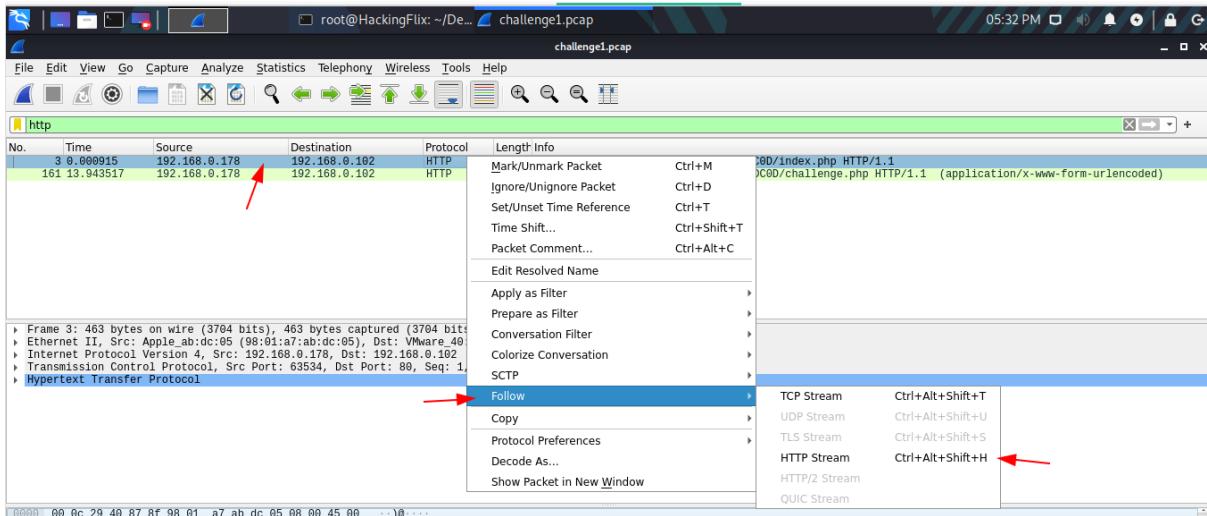
ANS ⇒

We know **host** is related to **http** part. So in filter type **http**



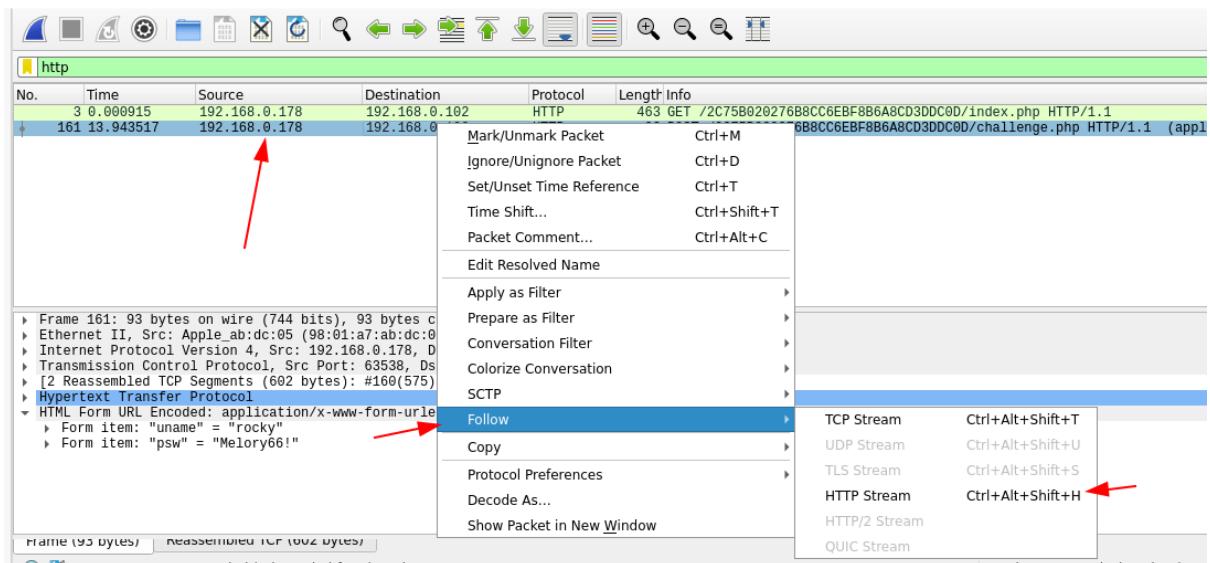
You will see 2 http results.

Check 1st one



We got Host: 192.168.0.102

Now, check 2nd Http result



```
Wireshark - Follow HTTP Stream (tcp.stream eq 6) · challenge1.pcap
```

```
POST /2C75B020276B8CC6EBF8B6A8CD3DDC0D/challenge.php HTTP/1.1
Host: 192.168.0.102
Content-Type: application/x-www-form-urlencoded
Origin: http://192.168.0.102
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/11.1.2
Safari/605.1.15
Referer: http://192.168.0.102/2C75B020276B8CC6EBF8B6A8CD3DDC0D/index.php
Content-Length: 27
Accept-Language: en-us

uname=rocky&psw=MeIory66%21
```

1 client pkt, 0 server pkts, 0 turns.

Entire conversation (602 bytes) Show data as ASCII

Find: Filter Out This Stream Print Save as... Back Close Help

Ok, Now we got our 2nd Host ⇒ Host: 192.168.0.102

We, can see that we have got traffic from same HOST

Task 2

What other information we can get from same challenge pcap file about Host 192.168.0.102

```
Wireshark - Follow HTTP Stream (tcp.stream eq 6) · challenge1.pcap

POST /2C75B020276B8CC6EBF8B6A8CD3DDC0D/challenge.php HTTP/1.1
Host: 192.168.0.102
Content-Type: application/x-www-form-urlencoded
Origin: http://192.168.0.102
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/11.1.2
Safari/605.1.15
Referer: http://192.168.0.102/2C75B020276B8CC6EBF8B6A8CD3DDC0D/index.php
Content-Length: 27
Accept-Language: en-us

uname=rocky&psw=Melory66%21

1 client pkt, 0 server pkts, 0 turns.
Entire conversation (602 bytes) Show data as ASCII Find Next
Find: Filter Out This Stream Print Save as... Back Close Help
```

```
Wireshark - Follow HTTP Stream (tcp.stream eq 6) · challenge1.pcap

POST /2C75B020276B8CC6EBF8B6A8CD3DDC0D/challenge.php HTTP/1.1
Host: 192.168.0.102
Content-Type: application/x-www-form-urlencoded
Origin: http://192.168.0.102
Accept-Encoding: gzip, deflate
Connection: keep-alive
Upgrade-Insecure-Requests: 1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozilla/5.0 (Macintosh; Intel Mac OS X 10_13_6) AppleWebKit/605.1.15 (KHTML, like Gecko) Version/11.1.2
Safari/605.1.15
Referer: http://192.168.0.102/2C75B020276B8CC6EBF8B6A8CD3DDC0D/index.php
Content-Length: 27
Accept-Language: en-us

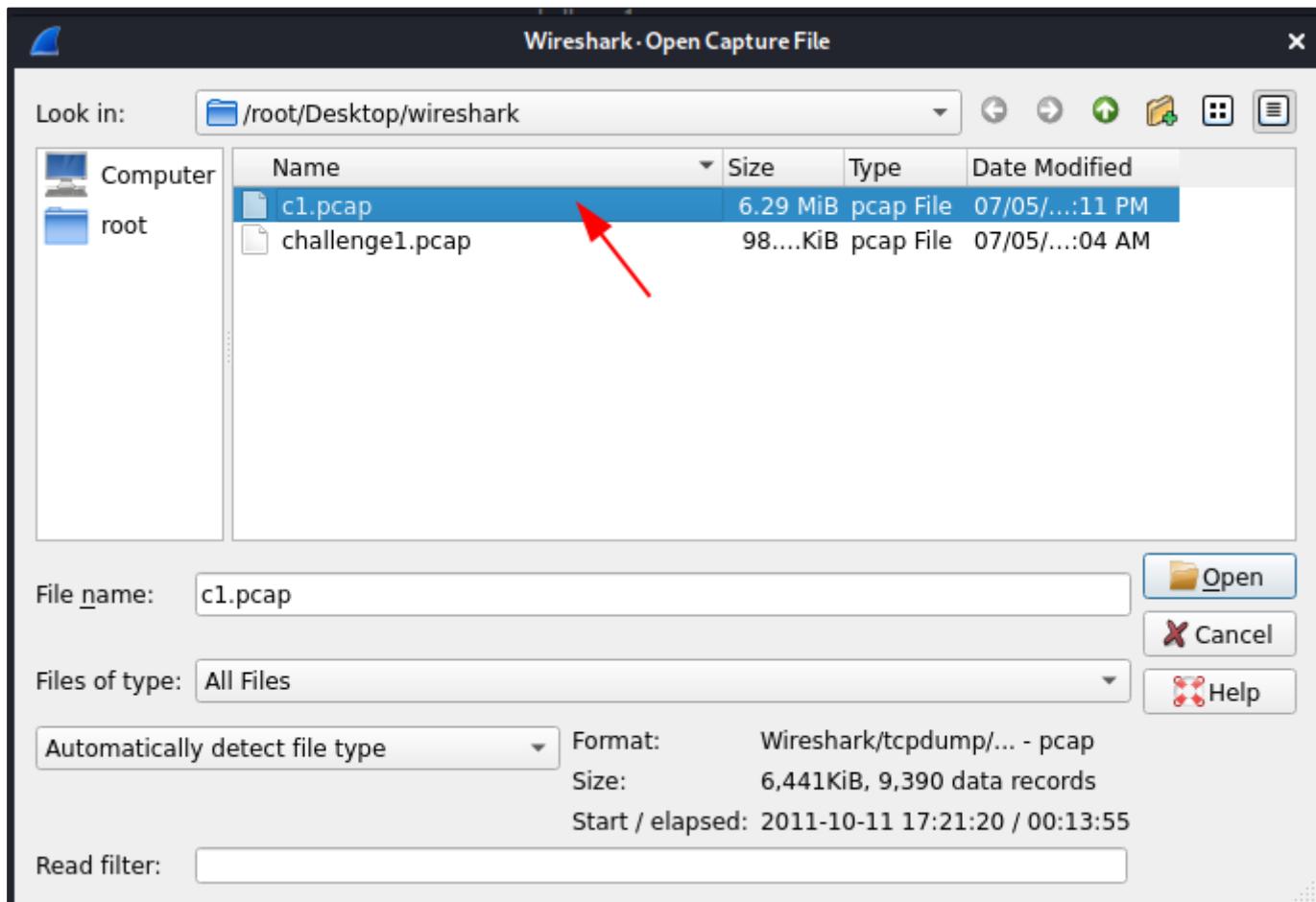
uname=rocky&psw=Melory66%21

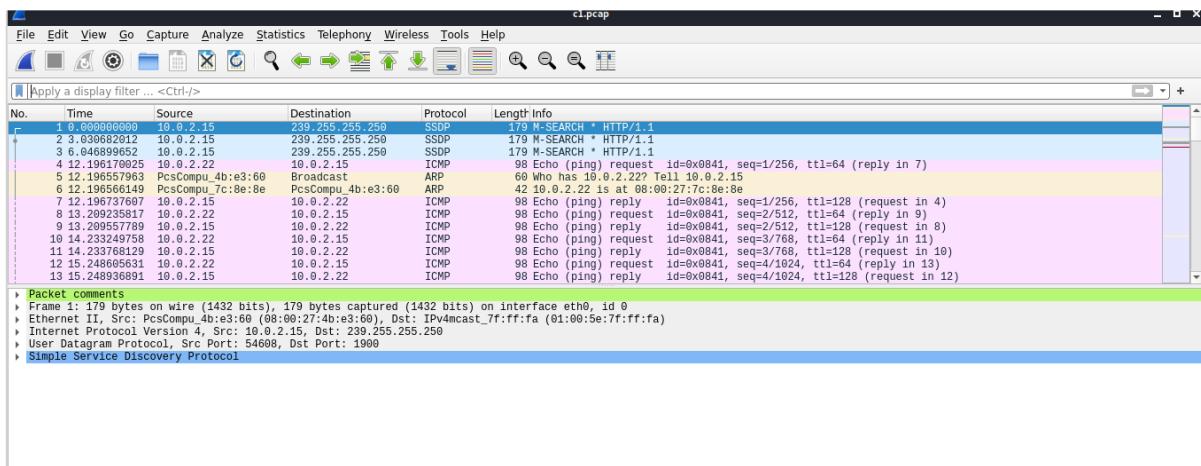
1 client pkt, 0 server pkts, 0 turns.
Entire conversation (602 bytes) Show data as ASCII Find Next
Find: Filter Out This Stream Print Save as... Back Close Help
```

We can see there is POST Request and username and password is in clear text mean no encoding on challenge.php file

Now, we will use another pcap file and do more tasks

c1.pcap





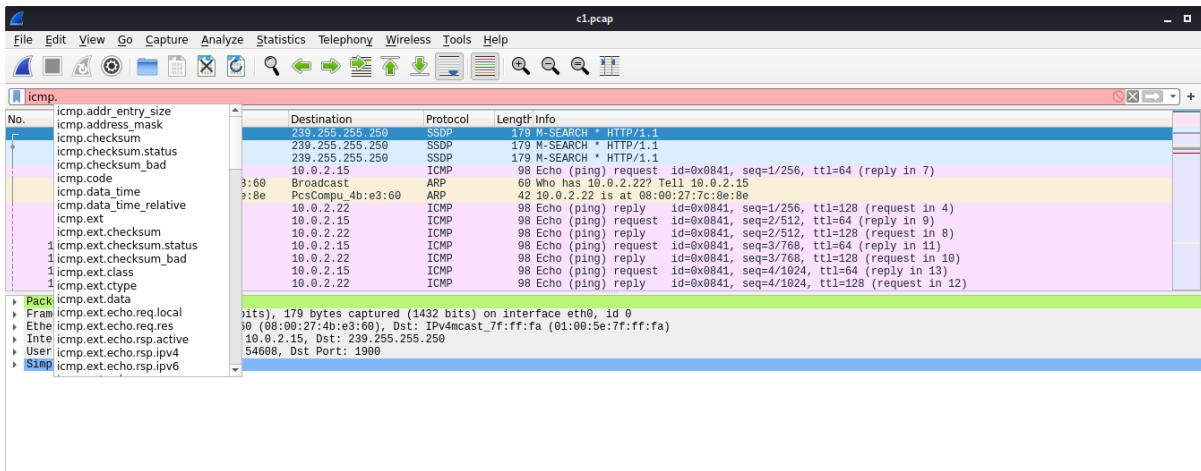
Task 1

How many ping requests were sent in the c1.pcap capture?

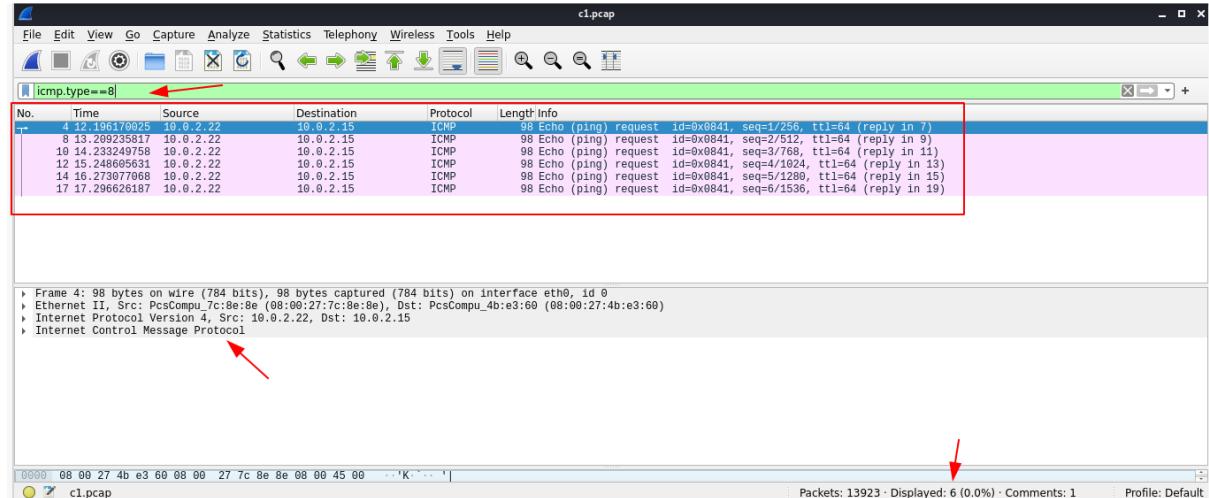
Ans :

Open the packet capture and apply the following filter: **icmp.type==8**

This filter will show all ICMP (aka ping) packets that are Type 8, which is an echo ping request



Above pic, in this you will see there are many options to use, so, this will let you know what you need to filter for specific result. For now let's use **icmp.type==8**



So, you see there are total 6 icmp packets so we got our ans

So, answer is there are 6 ping requests

Task 2

What is the IP address of the device associated with 08:00:27:4b:e3:60?

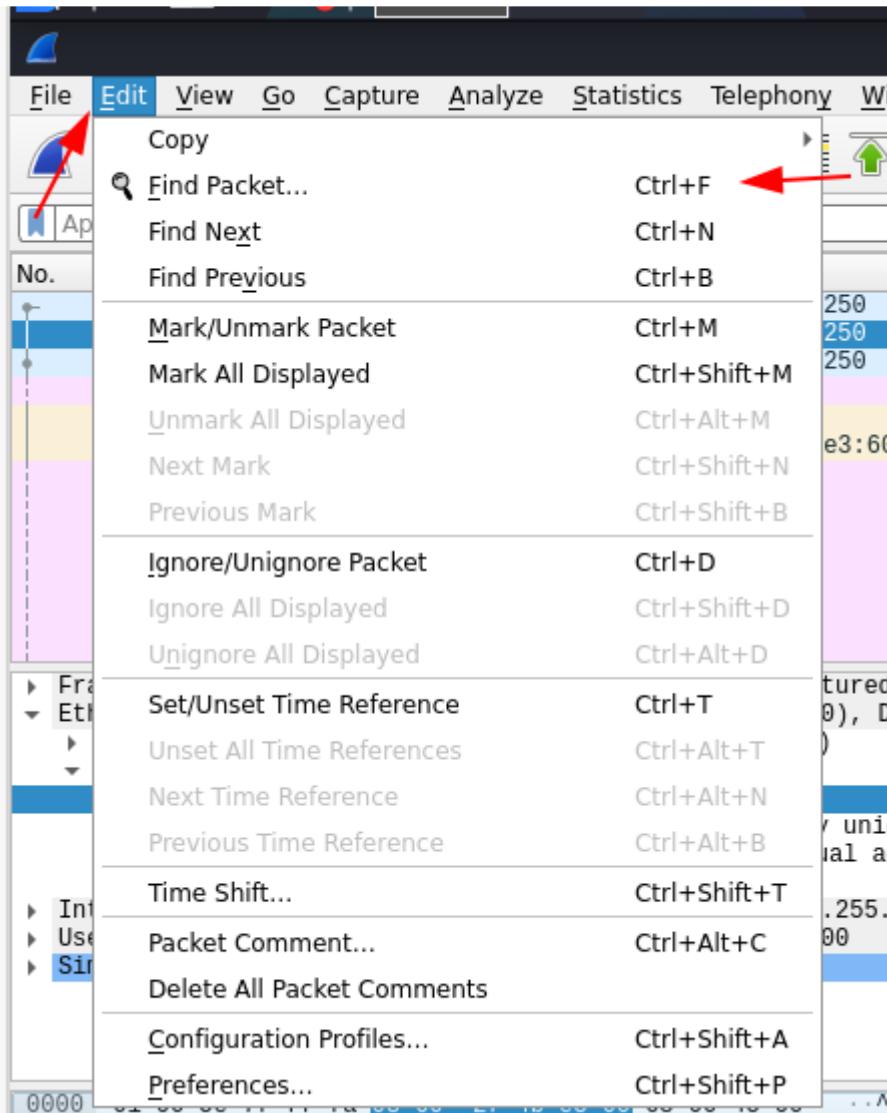
Ok, now we need to find IP for mac address 08:00:27:4b:e3:60

So, to find out IP , we need to find hexadecimal value for MAC Address

This ⇒ 08:00:27:4b:e3:60 ⇒ is a six-byte hexadecimal value separated by colon (:)

How to do this ?

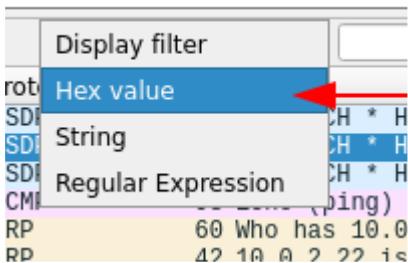
First ⇒



Second ⇒

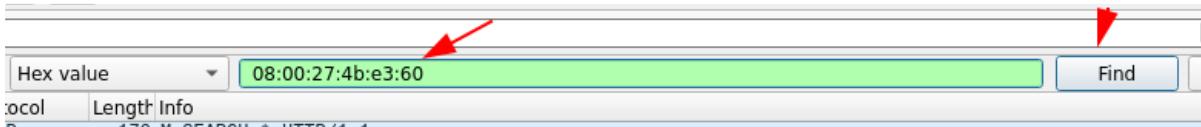
The screenshot shows the Wireshark interface with a packet list titled 'c1.pcap'. The list contains numerous network packets, each with details like Time, Source, Destination, Protocol, and Length. A red arrow points from the top towards the toolbar, and another red arrow points from the right towards the 'Display filter' dropdown menu.

Change Display Filter to Hex value



Third ⇒

Now enter mac address and click on find



No.	Time	Source	Destination	Protocol	Length/Info
1	0.000000000	10.0.2.15	239.255.255.250	SSDP	179 M-SEARCH * HTTP/1.1
2	0.00082012	10.0.2.15	239.255.255.250	SSDP	179 M-SEARCH * HTTP/1.1
3	0.04699952	10.0.2.15	239.255.255.250	SSDP	179 M-SEARCH * HTTP/1.1
4	12.196170025	10.0.2.22	10.0.2.15	ICMP	98 Echo (ping) request id=0x0841, seq=1/256, ttl=64 (reply in 7)
5	12.196557063	PcsCompu_4b:e3:60	Broadcast	ARP	60 Who has 10.0.2.22? Tell 10.0.2.15
6	12.196566149	PcsCompu_7c:8e:8e	PcsCompu_4b:e3:60	ARP	42 10.0.2.22 is at 08:00:27:7c:8e:8e
7	12.196737607	10.0.2.15	10.0.2.22	ICMP	98 Echo (ping) reply id=0x0841, seq=1/256, ttl=128 (request in 4)
8	13.209235817	10.0.2.22	10.0.2.15	ICMP	98 Echo (ping) request id=0x0841, seq=2/512, ttl=64 (reply in 9)
9	13.209557789	10.0.2.15	10.0.2.22	ICMP	98 Echo (ping) reply id=0x0841, seq=2/512, ttl=128 (request in 8)
10	14.233768129	10.0.2.22	10.0.2.15	ICMP	98 Echo (ping) request id=0x0841, seq=3/768, ttl=64 (reply in 11)
11	14.233768129	10.0.2.15	10.0.2.22	ICMP	98 Echo (ping) reply id=0x0841, seq=3/768, ttl=128 (request in 10)
12	15.248605631	10.0.2.22	10.0.2.15	ICMP	98 Echo (ping) request id=0x0841, seq=4/1024, ttl=64 (reply in 13)
13	15.248605631	10.0.2.15	10.0.2.22	TCP	00 Echo (ping) reply id=0x0841, seq=4/1024, ttl=128 (request in 12)

Frame 3: 179 bytes on wire (1432 bits), 179 bytes captured (1432 bits) on interface eth0, id 0
 Ethernet II, Src: PcsCompu_4b:e3:60 (08:00:27:4b:e3:60), Dst: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
 Destination: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
 Address: IPv4mcast_7f:ff:fa (01:00:5e:7f:ff:fa)
0..... = L6 bit: Globally unique address (factory default)
0..... = L6 bit: Group address (multicast/broadcast)
 Source: PcsCompu_4b:e3:60 (08:00:27:4b:e3:60)
 Address: PcsCompu_4b:e3:60 (08:00:27:4b:e3:60)
0..... = L6 bit: Globally unique address (factory default)
0..... = L6 bit: Individual address (unicast)
 Type: IPv4 (0x0800)
 Internet Protocol Version 4, Src: 10.0.2.15, Dst: 239.255.255.250
 User Datagram Protocol, Src Port: 54688, Dst Port: 1900
 ICMPv4, Type: Echo Request (8)
 01 00 5e 7f ff fa 08 00 27 4b e3 60 08 00 45 00
 Source or Destination Hardware Address (eth.addr), 6 bytes

Packets: 13923 · Displayed: 13923 (100.0%) · Comments: 1 · Profile: Default

So, we got our IP Address ⇒ 10.0.2.15

Task 3

What version of Internet Group Management Protocol is in use?

ANS :

Google IGMP



Display Filter

A complete list of IGMP display filter fields can be found in the [display filter reference](#)

Show only the IGMP based traffic:

```
igmp
```

Capture Filter

Capture only the IGMP based traffic:

```
igmp
```

External links

- [RFC 988 Host Extensions for IP Multicasting](#) - describes the obsolete "version 0" of IGMP

Whenever you in doubt, just google like above

Now ⇒ go to filter and type **igmp** ⇒ and in packet detail pane + packet list pane check for IGMP Version

So, IGMP Version is \Rightarrow 3

Task 4

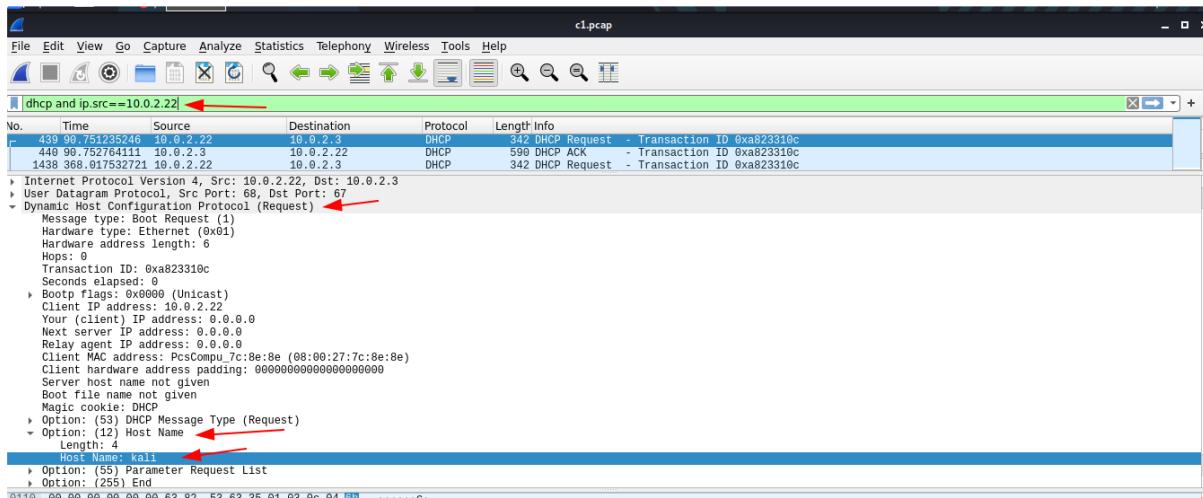
What is the name of the host located at 10.0.2.22?

ANS :

Host names can be identified in dynamic host configuration protocol (DHCP) traffic

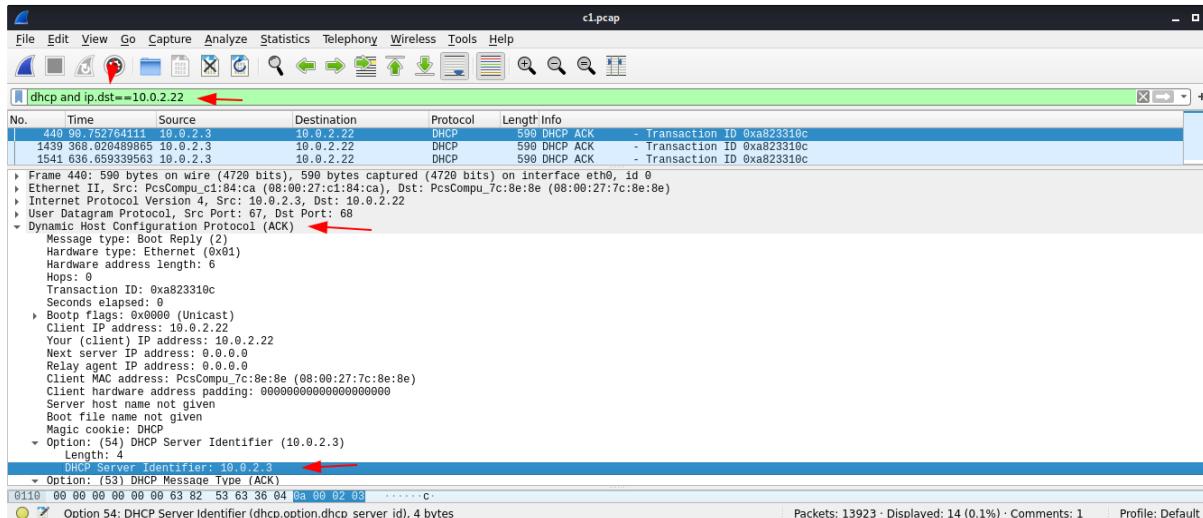
Using a filter ⇒ dhcp and ip.src == 10.0.2.22

where ip.src is source of IP



So, we got our hostname \Rightarrow Kali

Suppose now, we want to know IP Address of DHCP Server , then now change ip.src to ip.dst



and we will get out IP \Rightarrow 10.0.2.3

Task 5

What is the name of the host located at 10.0.2.15?

ANS:

In this case we are going to look for server message block (SMB) protocol activity, which features host announcements

Filter for this \Rightarrow **smb and ip.addr == 10.0.2.15**

The screenshot shows the Wireshark interface with a packet list and details panes. A red arrow points to the filter bar at the top, which contains the text "smb and ip.addr == 10.0.2.15". Another red arrow points to the "Host Name" field in the expanded details pane, which displays "MSEdgeWin10". The bottom status bar indicates "Packets: 13923 · Displayed: 17 (0.1%) · Comments: 1 · Profile: Default".

So, we got host \Rightarrow **MSEdgeWin10**

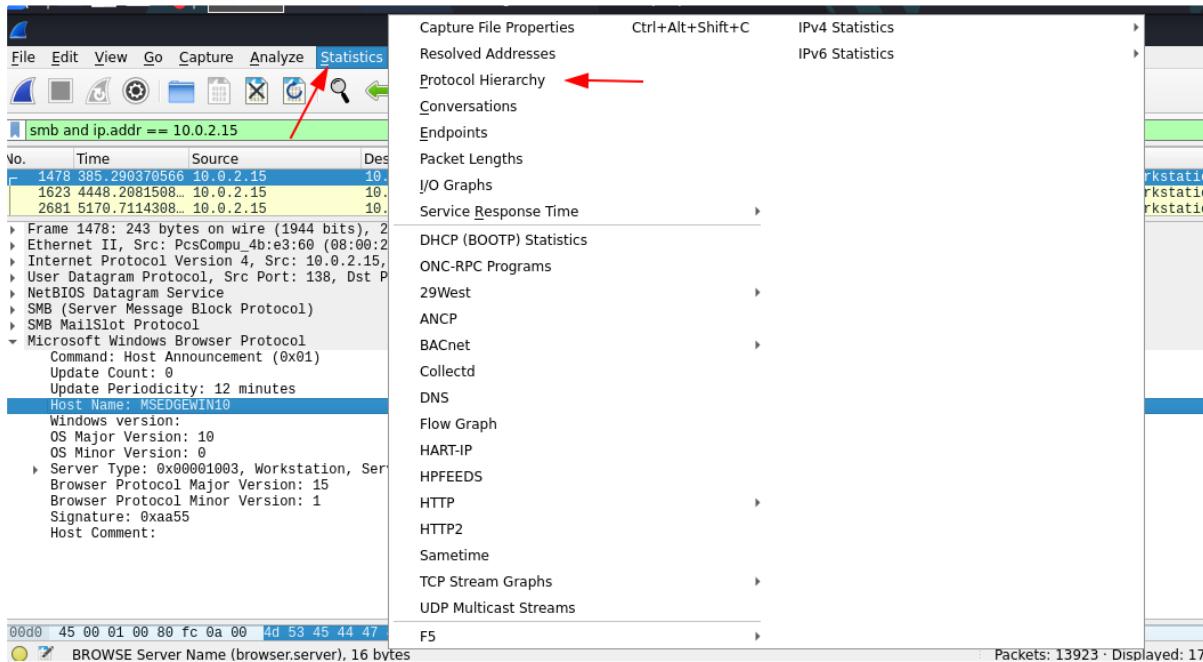
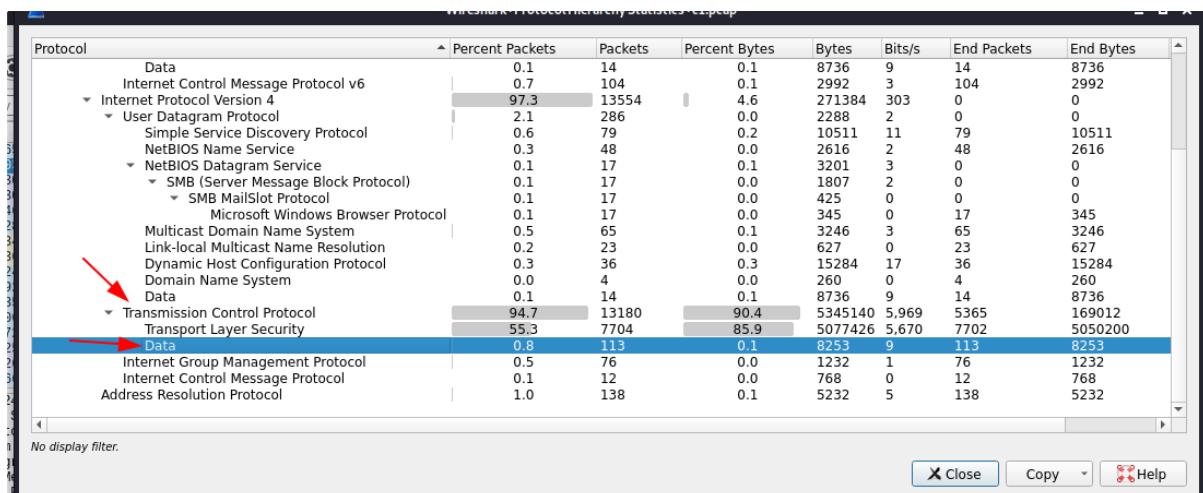
Task 6

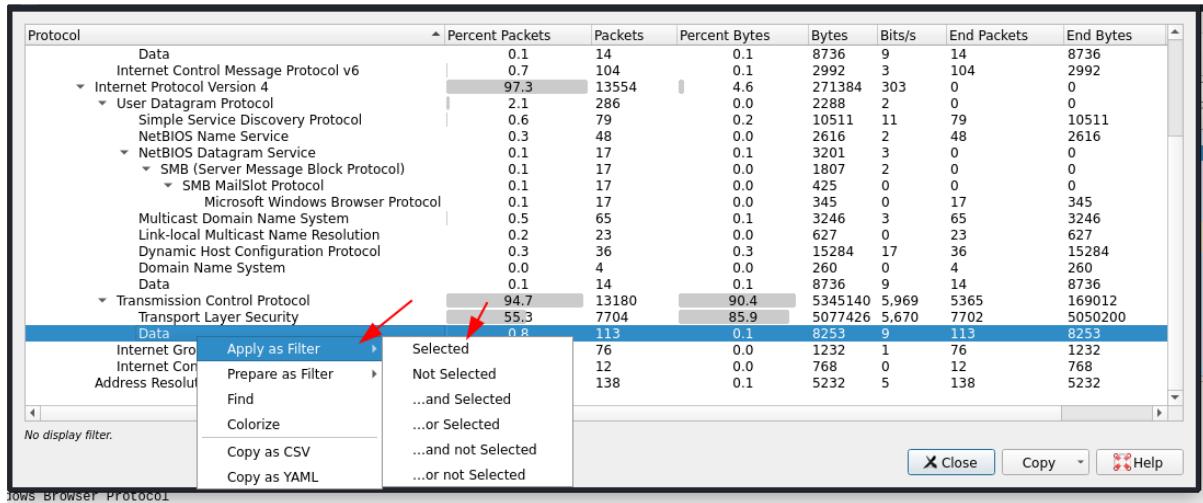
What is the IP address of the attacker?

Now, suppose you want to know the IP of attacker, and for this we need to check those IP Address which give lots of request to our server

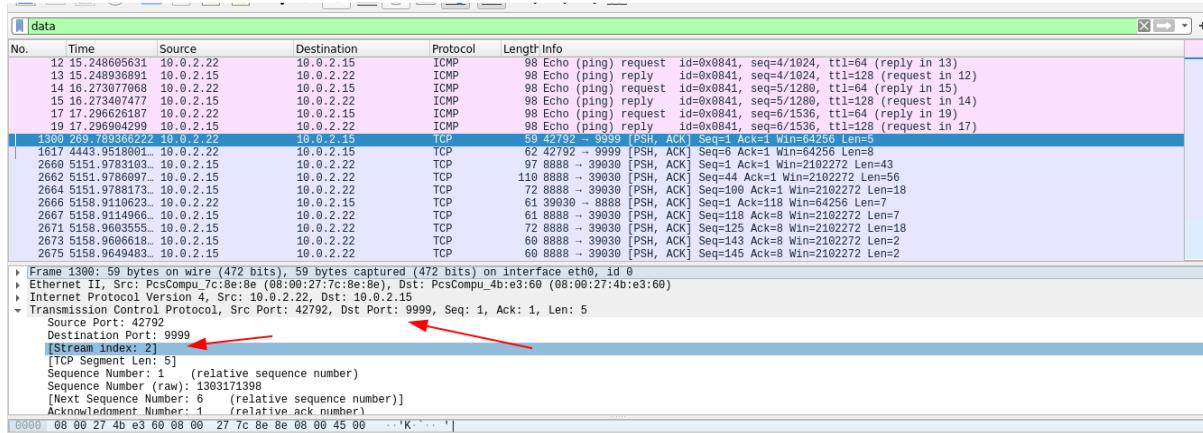
So, we need to identify the attack activity in the capture.

Let's do this

Step 1 ⇒**Step 2 ⇒****Step 3 ⇒**



Step 4 ⇒



Expand Subtrees
Collapse Subtrees
Expand All
Collapse All
Apply as Column Ctrl+Shift+I
Apply as Filter
Prepare as Filter
Conversation Filter
Colorize with Filter
Follow
Copy
Show Packet Bytes... Ctrl+Shift+O
Export Packet Bytes... Ctrl+Shift+X
Wiki Protocol Page
Filter Field Reference
Protocol Preferences
Decode As... Ctrl+Shift+U
Go to Linked Packet
Show Linked Packet in New Window

[TCP Segment Len: 5]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1303171398
[Next Sequence Number: 6 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)

No.	Time	Source	Destination	Protocol	Length	Stream index	Info
12	15.248605631	10.0.2.15	10.0.2.15	ICMP	98		Echo (ping) request id=0x0041 seq=4/1024 ttl=64 (reply in 13)
13	15.248936891	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=4/1024 ttl=128 (request in 12)
14	16.273077068	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) request id=0x0041 seq=5/1280 ttl=64 (reply in 15)
15	16.273407477	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=5/1280 ttl=128 (request in 14)
17	17.296626187	10.0.2.22	10.0.2.15	ICMP	98		Echo (ping) request id=0x0041 seq=6/1536 ttl=64 (reply in 19)
19	17.296904299	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=6/1536 ttl=128 (request in 17)
1300	269.789366222	10.0.2.15	10.0.2.15	TCP	59	2	42792 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=5
1617	4443.9518001	10.0.2.22	10.0.2.15	TCP	62		2 42792 → 9999 [PSH, ACK] Seq=6 Ack=1 Win=64256 Len=8
2660	5151.9783103	10.0.2.15	10.0.2.22	TCP	97		5 8888 → 39030 [PSH, ACK] Seq=3 Ack=1 Win=2102272 Len=43
2662	5151.9786097	10.0.2.15	10.0.2.22	TCP	110		5 8888 → 39030 [PSH, ACK] Seq=4 Ack=1 Win=2102272 Len=56
2664	5151.9788173	10.0.2.15	10.0.2.22	TCP	72		5 8888 → 39030 [PSH, ACK] Seq=100 Ack=1 Win=2102272 Len=18
2666	5158.9110623	10.0.2.22	10.0.2.15	TCP	61		5 39030 → 8888 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=7
2667	5158.9114966	10.0.2.15	10.0.2.22	TCP	61		5 8888 → 39030 [PSH, ACK] Seq=118 Ack=8 Win=2102272 Len=7
2671	5158.9603555	10.0.2.15	10.0.2.22	TCP	72		5 8888 → 39030 [PSH, ACK] Seq=125 Ack=8 Win=2102272 Len=18
2673	5158.9606618	10.0.2.15	10.0.2.22	TCP	60		5 8888 → 39030 [PSH, ACK] Seq=143 Ack=8 Win=2102272 Len=2
2675	5158.9649483	10.0.2.22	10.0.2.15	TCP	60		

Frame 1300: 59 bytes on wire (472 bits), 59 bytes captured (472 bits) on interface eth0, id 0
Ethernet II, Src: PcsCompu_7c:8e:8e (08:00:27:7c:8e:8e), Dst: PcsCompu_4b:e3:60 (08:00:27:4b:e3:60)
Internet Protocol Version 4, Src: 10.0.2.22, Dst: 10.0.2.15
Transmission Control Protocol, Src Port: 42792, Dst Port: 9999, Seq: 1, Ack: 1, Len: 5
Source Port: 42792
Destination Port: 9999
Stream index: 2

[Stream index: 2]

[TCP Segment Len: 5]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1303171398
[Next Sequence Number: 6 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)

0020 02 0f a7 28 27 0f 4d ac d1 46 bc 22 fb c0 50 18 ... (1.M..)

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

data

No.	Time	Source	Destination	Protocol	Length	Stream index	Info
12	15.248605631	10.0.2.22	10.0.2.15	ICMP	98		Echo (ping) request id=0x0041 seq=4/1024 ttl=64 (reply in 13)
13	15.248936891	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=4/1024 ttl=128 (request in 12)
14	16.273077068	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) request id=0x0041 seq=5/1280 ttl=64 (reply in 15)
15	16.273407477	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=5/1280 ttl=128 (request in 14)
17	17.296626187	10.0.2.22	10.0.2.15	ICMP	98		Echo (ping) request id=0x0041 seq=6/1536 ttl=64 (reply in 19)
19	17.296904299	10.0.2.15	10.0.2.22	ICMP	98		Echo (ping) reply id=0x0041 seq=6/1536 ttl=128 (request in 17)
1300	269.789366222	10.0.2.22	10.0.2.15	TCP	59	2	42792 → 9999 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=5
1617	4443.9518001	10.0.2.22	10.0.2.15	TCP	62		2 42792 → 9999 [PSH, ACK] Seq=6 Ack=1 Win=64256 Len=8
2660	5151.9783103	10.0.2.15	10.0.2.22	TCP	97		5 8888 → 39030 [PSH, ACK] Seq=3 Ack=1 Win=2102272 Len=43
2662	5151.9786097	10.0.2.15	10.0.2.22	TCP	110		5 8888 → 39030 [PSH, ACK] Seq=4 Ack=1 Win=2102272 Len=56
2664	5151.9788173	10.0.2.15	10.0.2.22	TCP	72		5 8888 → 39030 [PSH, ACK] Seq=100 Ack=1 Win=2102272 Len=18
2666	5158.9110623	10.0.2.22	10.0.2.15	TCP	61		5 39030 → 8888 [PSH, ACK] Seq=1 Ack=1 Win=64256 Len=7
2667	5158.9114966	10.0.2.15	10.0.2.22	TCP	61		5 8888 → 39030 [PSH, ACK] Seq=118 Ack=8 Win=2102272 Len=7
2671	5158.9603555	10.0.2.15	10.0.2.22	TCP	72		5 8888 → 39030 [PSH, ACK] Seq=125 Ack=8 Win=2102272 Len=18
2673	5158.9606618	10.0.2.15	10.0.2.22	TCP	60		5 8888 → 39030 [PSH, ACK] Seq=143 Ack=8 Win=2102272 Len=2

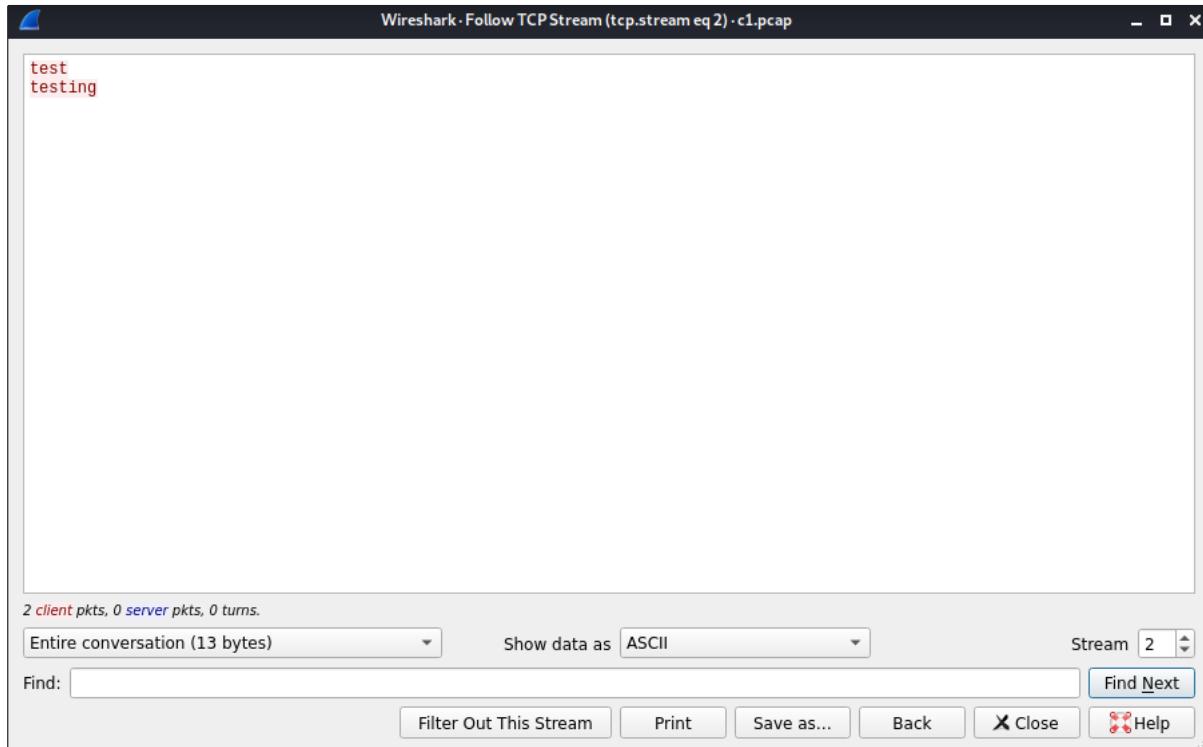
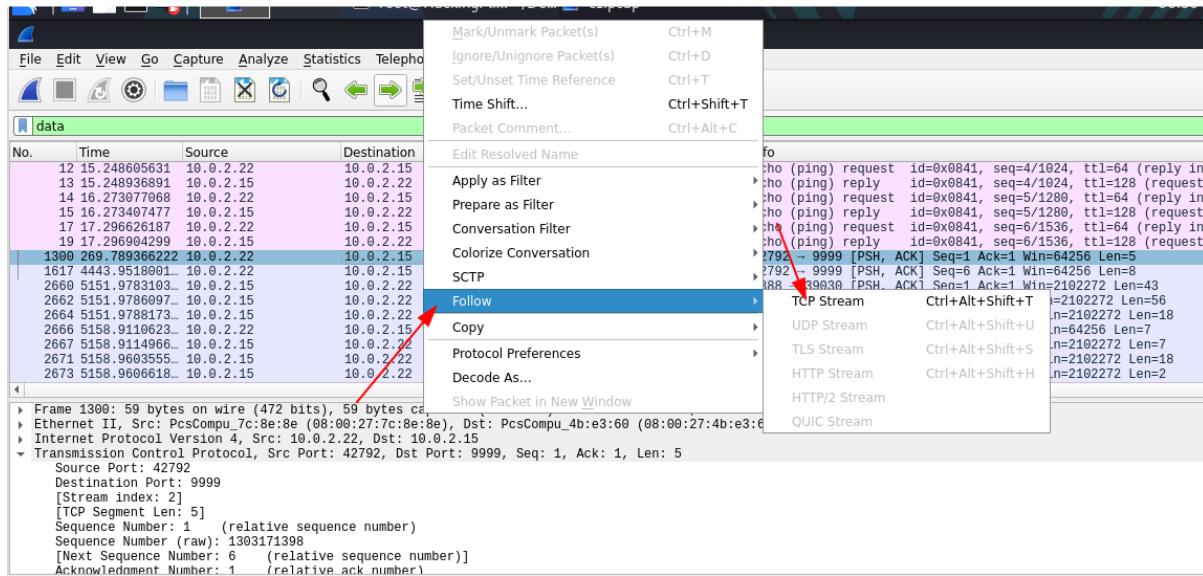
Frame 1300: 59 bytes on wire (472 bits), 59 bytes captured (472 bits) on interface eth0, id 0
Ethernet II, Src: PcsCompu_7c:8e:8e (08:00:27:7c:8e:8e), Dst: PcsCompu_4b:e3:60 (08:00:27:4b:e3:60)
Internet Protocol Version 4, Src: 10.0.2.22, Dst: 10.0.2.15
Transmission Control Protocol, Src Port: 42792, Dst Port: 9999, Seq: 1, Ack: 1, Len: 5
Source Port: 42792
Destination Port: 9999
Stream index: 2

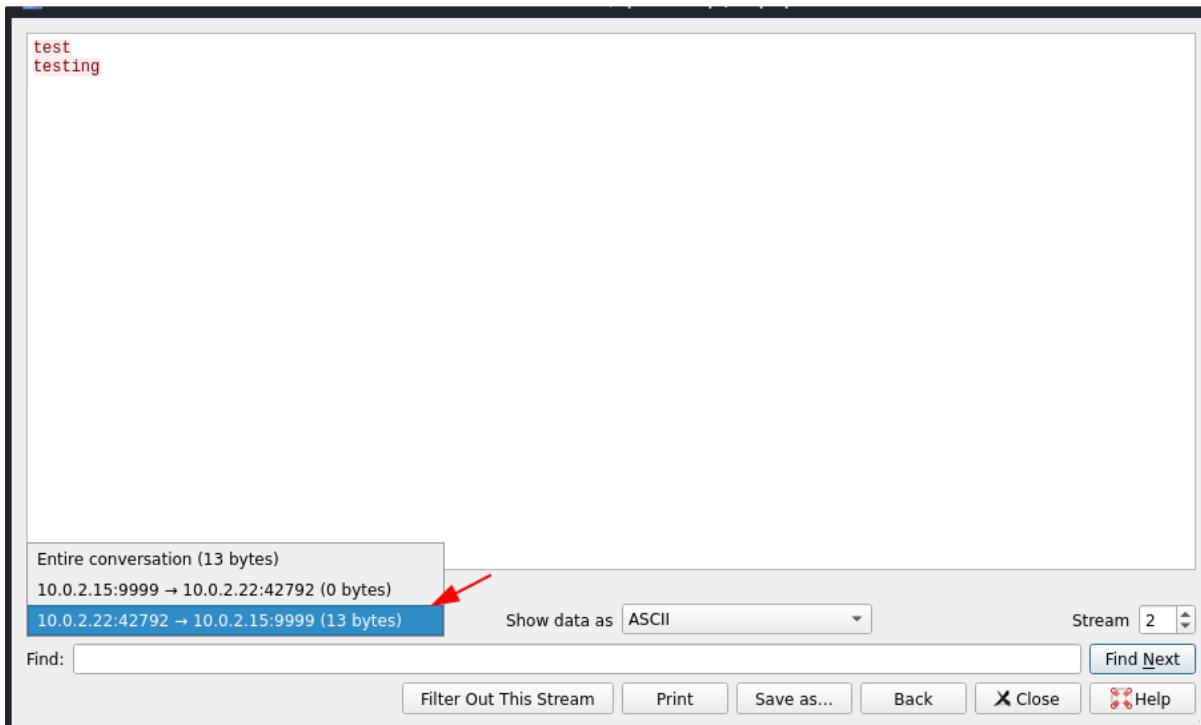
[Stream index: 2]

[TCP Segment Len: 5]
Sequence Number: 1 (relative sequence number)
Sequence Number (raw): 1303171398
[Next Sequence Number: 6 (relative sequence number)]
Acknowledgment Number: 1 (relative ack number)

0020 02 0f a7 28 27 0f 4d ac d1 46 bc 22 fb c0 50 18 ... (1.M..)

Step 5 ⇒

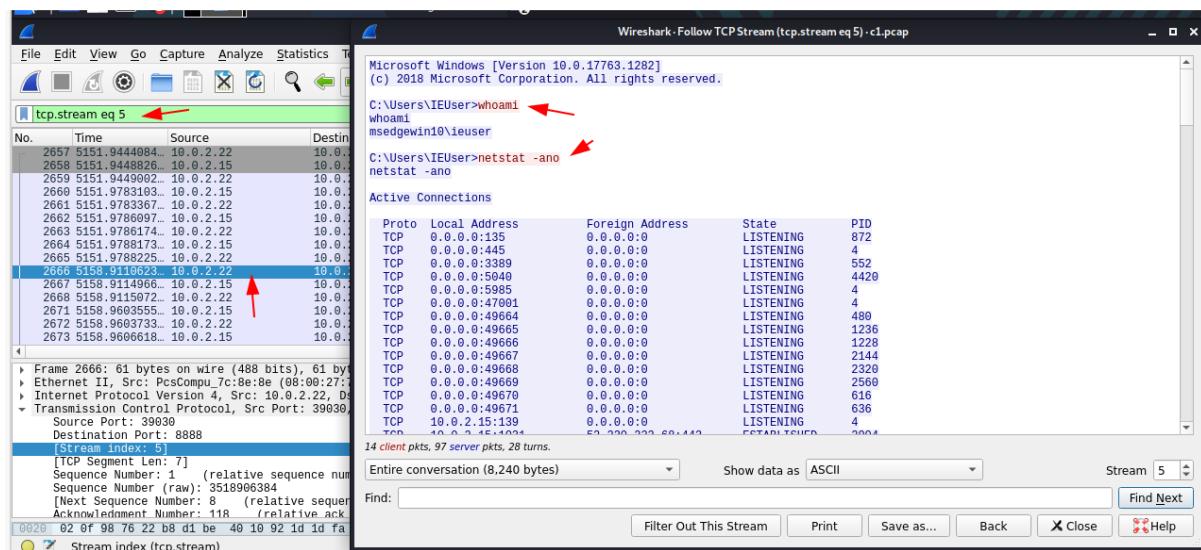




So, we got our IP \Rightarrow 10.0.2.22 \Rightarrow this is the likely attacker IP and the threat actor appears to have remote shell access to the victim box.

Now, if you want to know what actually attacker did ? or what command he used ? then \Rightarrow

Just follow any tcp stream index 5 and right click \Rightarrow follow \Rightarrow tcp stream and we see \Rightarrow **whoami** command is run by attacker



So, we have learned many new things in this challenge through tasks. like how to use filter for specific task

BONUS

Let's us see lists of important Wireshark Filters

English	C-like	Description and example
eq	==	Equal. ip.src==10.0.0.5
ne	!=	Not equal. ip.src!=10.0.0.5
gt	>	Greater than. frame.len > 10
lt	<	Less than. frame.len < 128
ge	>=	Greater than or equal to. frame.len ge
0x100		
le	<=	Less than or equal to. frame.len <= 0x20
contains		Protocol, field or slice contains a value. sip.To
contains "a1762"		
matches expression.	~	Protocol or text field match Perl regular
		http.host matches "acme\.(org com net)"
bitwise_and	&	Compare bit field value. tcp.flags & 0x02
and	&&	Logical AND. ip.src==10.0.0.5 and tcp.flags.fin
or		Logical OR. ip.src==10.0.0.5 or
ip.src==192.1.1.1		
xor	^^	Logical XOR. tr.dst[0:3] == 0.6.29 xor tr.src[0:3]
== 0.6.29		
not	!	Logical NOT. not llc
[...]		Slice Operator. eth.addr[0:3]==00:06:5B
in		Membership Operator. tcp.port in {80 443 8080}

Examples ⇒

- `tcp.port eq 80` ⇒
 - filter only that tcp protocol whose port is **80**
- `tcp.srcport==443` ⇒
 - filter only that tcp whose source port is **443**
- List

Filter for HTTP and HTTPS traffic:

```
tcp.port==443 or tcp.port==80
ssl or http
tcp.port in {80 443 8080}
tcp.port == 80 || tcp.port == 443 || tcp.port == 8080
```

- List

Filter for a protocol:

```
tcp
udp
dns
```

- Filter for IP Addresses ⇒
 - `ip.addr == 10.43.54.65`
 - `! (ip.addr == 10.43.54.65)`

You will have doubt what actually is **src** and **dst** ?

src ⇒ Source - where the packet came from

dst ⇒ Destination - where the packet is going