
WIRESHARK

Lab - Network Sniffing with Wireshark

Objectives

In this lab, you will use the Linux utility **tcpdump** to capture and save network traffic. You will then use Wireshark to investigate the traffic capture.

- Prepare the host to capture network traffic.
- Capture and save network traffic.
- View and Analyze the Packet capture.

Background / Scenario

Wireshark is a network packet capture utility that can be used by network administrators to troubleshoot network problems. It can also be used to eavesdrop on network communications to passively collect information about users and services. Wireshark is considered a passive tool because it does not create traffic on the network.

Required Resources

- Kali VM customized for Ethical Hacker course
- Internet access

Instructions

Part 1: Prepare the Host to Capture Network Traffic.

Step 1: Start the virtual machine and log in.

a. Start the Kali workstation virtual machine. Use the following user credentials:

b. Start a terminal session by clicking on the terminal icon on the menu bar.

Step 2: Verify the environment.

a. Verify the user directory that will be used to store the captured traffic. To display the current directory, use the **pwd** command to display the full path to the current working directory.

b. Determine the IP address of the Kali Ethernet interface using the **ifconfig** command. The ethernet interface is usually named **eth0**.

```
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 14  bytes 2274 (2.2 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

r-internal: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.6.6.1  netmask 255.255.255.0  broadcast 10.6.6.255
        inet6 fe80::42:6eff:fea5:9f72  prefixlen 64  scopeid 0x20<link>
        ether 02:42:6e:a5:9f:72  txqueuelen 0  (Ethernet)
        RX packets 48  bytes 2756 (2.6 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 20  bytes 2738 (2.6 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

ocker0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 172.17.0.1  netmask 255.255.0.0  broadcast 172.17.255.255
        inet6 fe80::42:79ff:fed9:c053  prefixlen 64  scopeid 0x20<link>
        ether 02:42:79:d9:c0:53  txqueuelen 0  (Ethernet)
        RX packets 50  bytes 6079 (5.9 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 15  bytes 2517 (2.4 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

th0: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet 10.0.2.15  netmask 255.255.255.0  broadcast 10.0.2.255
        inet6 fe80::a00:27ff:fe4a:f36e  prefixlen 64  scopeid 0x20<link>
        inet6 fd17:625c:f037:2:a00:27ff:fe4a:f36e  prefixlen 64  scopeid 0x0<global>
        inet6 fd17:625c:f037:2:ca4b:8b9e:badc:a0b1  prefixlen 64  scopeid 0x0<global>
        ether 08:00:27:4a:f3:6e  txqueuelen 1000  (Ethernet)
        RX packets 14  bytes 4260 (4.1 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 35  bytes 5498 (5.3 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

o: flags=73<UP,LOOPBACK,RUNNING>  mtu 65536
        inet 127.0.0.1  netmask 255.0.0.0
        inet6 ::1  prefixlen 128  scopeid 0x10<host>
        loop  txqueuelen 1000  (Local Loopback)
        RX packets 4  bytes 240 (240.0 B)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 4  bytes 240 (240.0 B)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth2573665: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::941f:a0ff:fe3a:e031  prefixlen 64  scopeid 0x20<link>
        ether 96:1f:a0:3a:e0:31  txqueuelen 0  (Ethernet)
        RX packets 27  bytes 1898 (1.8 KiB)
        RX errors 0  dropped 0  overruns 0  frame 0
        TX packets 27  bytes 3799 (3.7 KiB)
        TX errors 0  dropped 0 overruns 0  carrier 0  collisions 0

eth0ee77be: flags=4163<UP,BROADCAST,RUNNING,MULTICAST>  mtu 1500
        inet6 fe80::6cfe:17ff:fe67:1cff  prefixlen 64  scopeid 0x20<link>
        ether 6e:fe:17:67:1c:ff  txqueuelen 0  (Ethernet)
```

c. Determine the default gateway assigned to the Kali host using the **ip route** command.

```
┌──(kali㉿Kali)-[~]
└─$ ip route
default via 10.0.2.2 dev eth0 proto dhcp src 10.0.2.15 metric 100
10.0.2.0/24 dev eth0 proto kernel scope link src 10.0.2.15 metric 100
10.5.5.0/24 dev br-339414195aeb proto kernel scope link src 10.5.5.1
10.6.6.0/24 dev br-internal proto kernel scope link src 10.6.6.1
172.17.0.0/16 dev docker0 proto kernel scope link src 172.17.0.1
192.168.0.0/24 dev br-355ee7945a88 proto kernel scope link src 192.168.0.1

┌──(kali㉿Kali)-[~]
└─$
```

d. Determine the address of the configured default DNS server by displaying the contents of the **/etc/resolv.conf** file. You can view the file using the **cat <file name>** command.

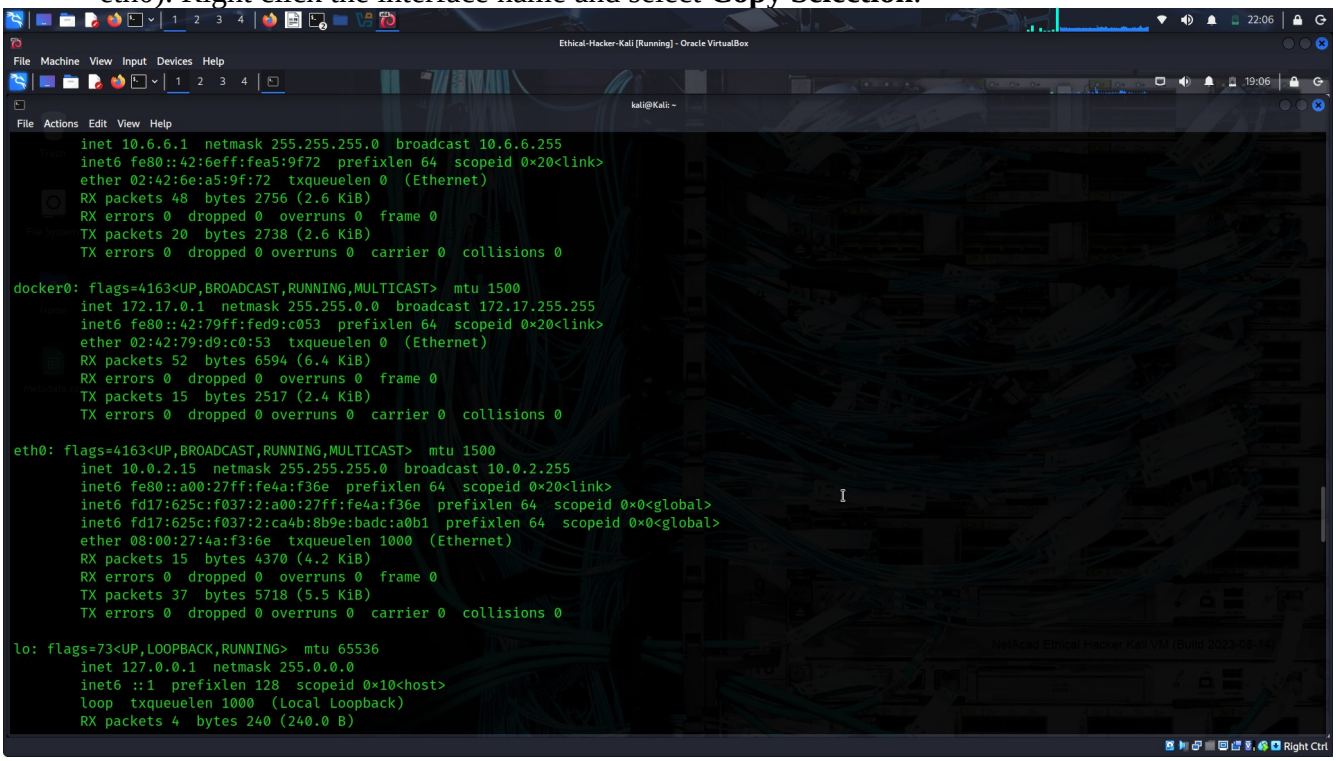
```
┌──(kali㉿Kali)-[~]
└─$ cat /etc/resolv.conf
# Generated by NetworkManager
nameserver 10.188.244.90

┌──(kali㉿Kali)-[~]
```

Part 2: Capture and Save Network Traffic.

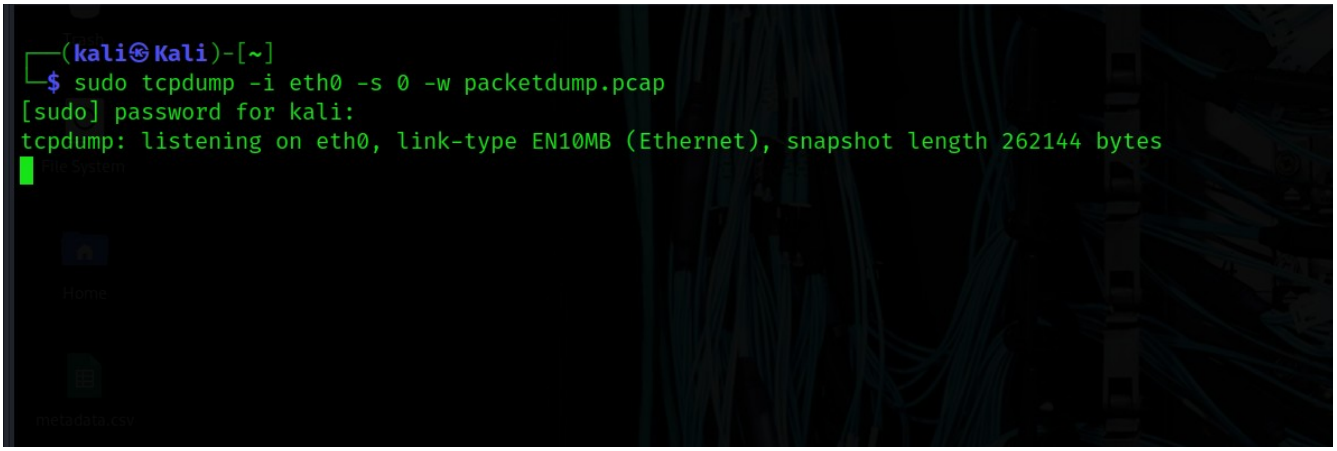
In this part, you will use **tcpdump** from the CLI to capture traffic. You will use command options to save the traffic to a packet capture (pcap) file. These records can then be analyzed using different applications that read pcap files, including Wireshark.

Step 1: Open a terminal and start tcpdump.

a) Open a terminal application and enter the command **ifconfig**.
b) In the **ifconfig** output, find the interface name that corresponds to the Ethernet adapter (usually eth0). Right click the interface name and select **Copy Selection**.



c) Enter the **sudo tcpdump** command as shown. Replace the ***<interface>*** text with the interface name that you copied in the previous step. This command requires root user access, so enter **kali** as the password if prompted

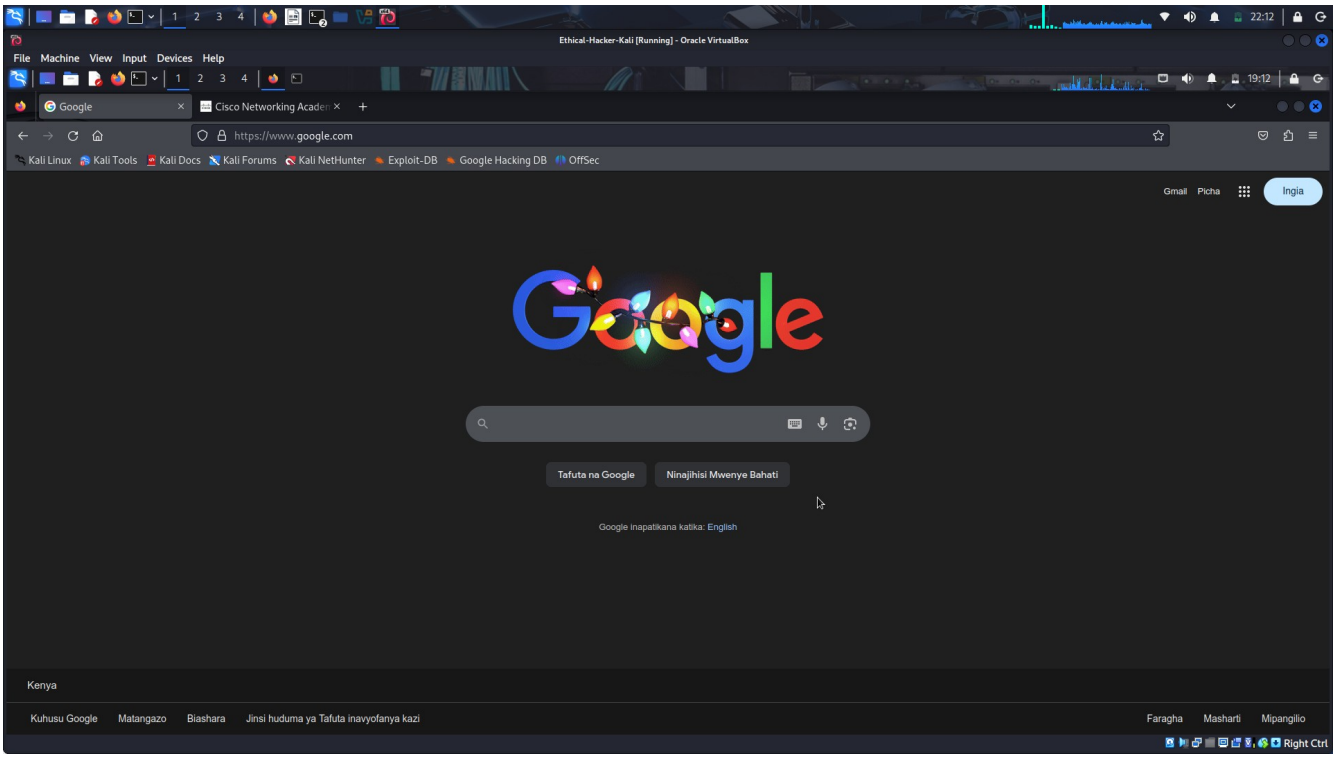


The **-i** command option allows you to specify the interface. If not specified, the tcpdump will capture all traffic on all interfaces.
The **-s** command option specifies the length of the snapshot for each packet. Setting this option to 0 sets it to the default of 262144.

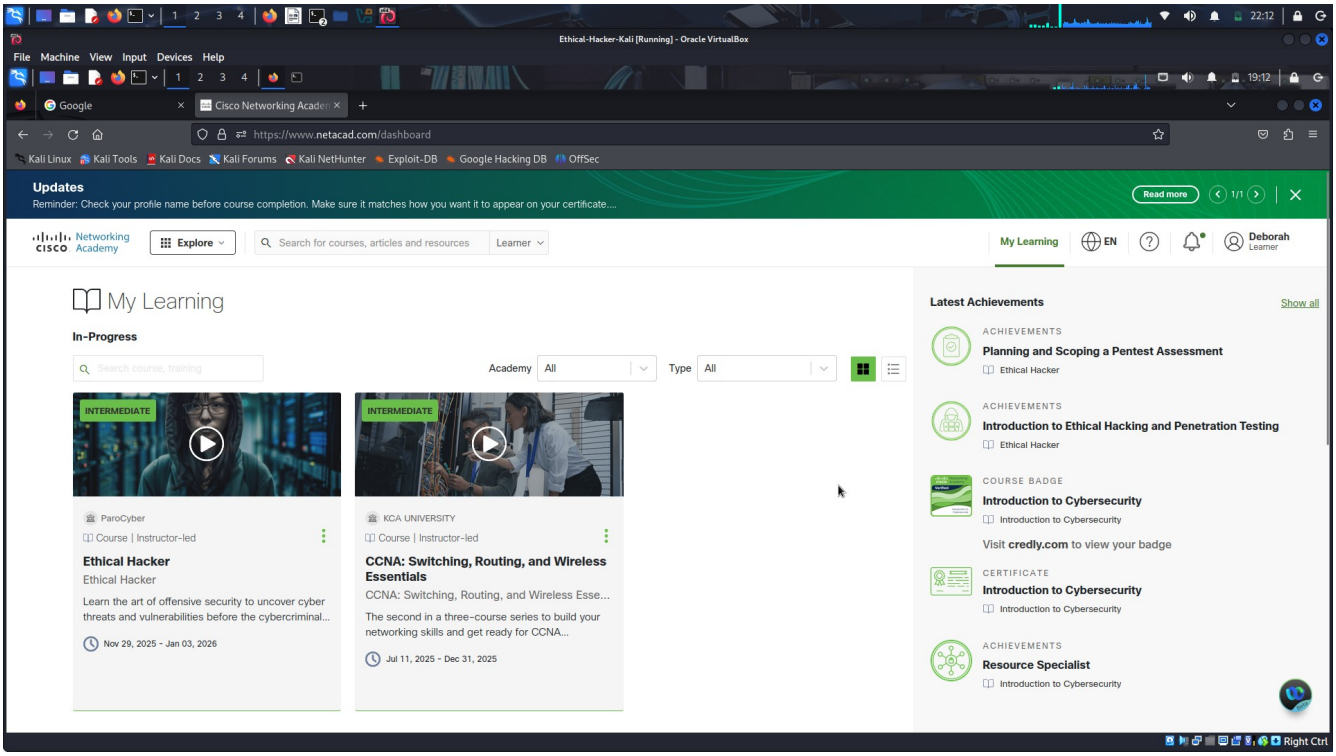
The **-w** command option is used to write the result of the **tcpdump** command to a file. Adding the extension **.pcap** ensures that operating systems and applications will be able to read the file. All recorded traffic will be printed to the file **packetdump.pcap** in the home directory of the user.

Step 2: Generate network traffic using a web browser

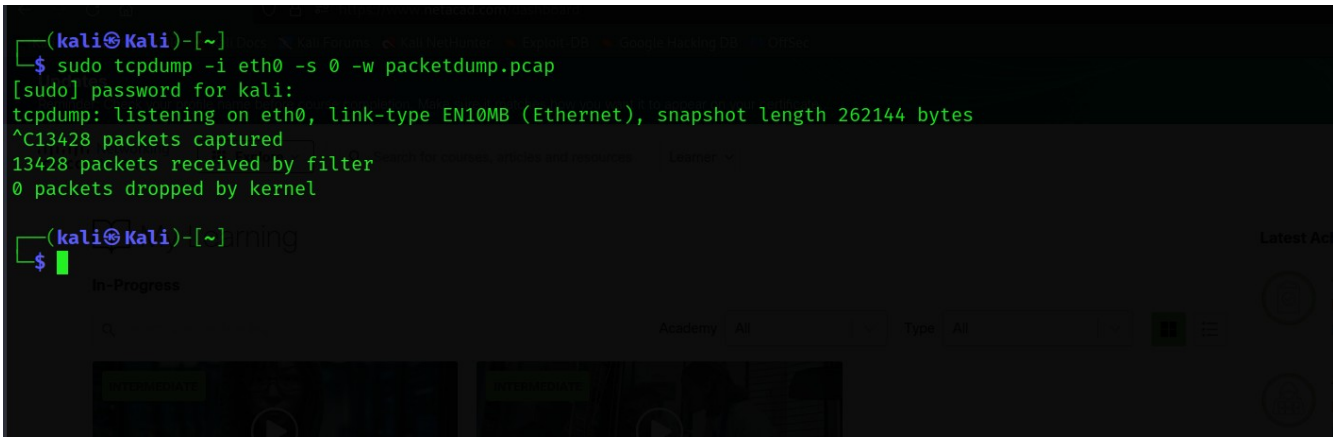
a. To capture an HTTP request and reply, open a web browser in Kali desktop. Navigate to **Google.com**. Do not login or search.



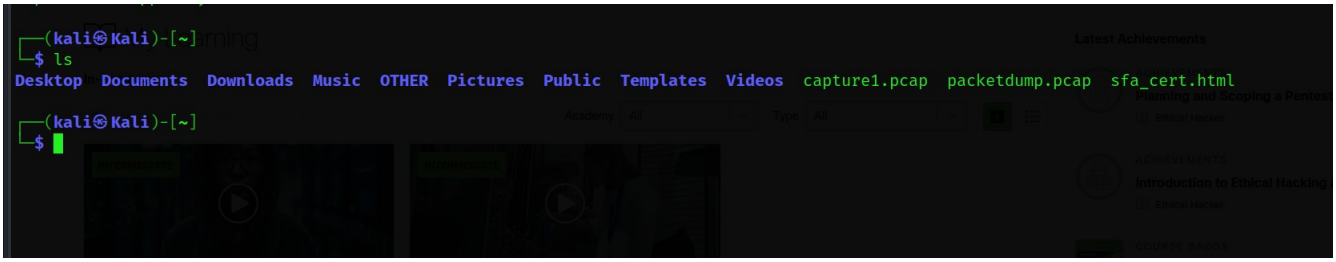
b. Open a second tab in the browser, enter **netacad.com** on the launch bar. Once the page appears, click the user icon at the top right of the page. Log in with your Skills for All login information.



c. Return to the terminal window that is running the **tcpdump** utility and enter **CTRL-C** to complete the packet capture.



d. The tcpdump utility saved the output to a file named **packetdump.pcap**. This file should be saved in the default home directory. Verify that the file exists in the directory using the **ls** command.

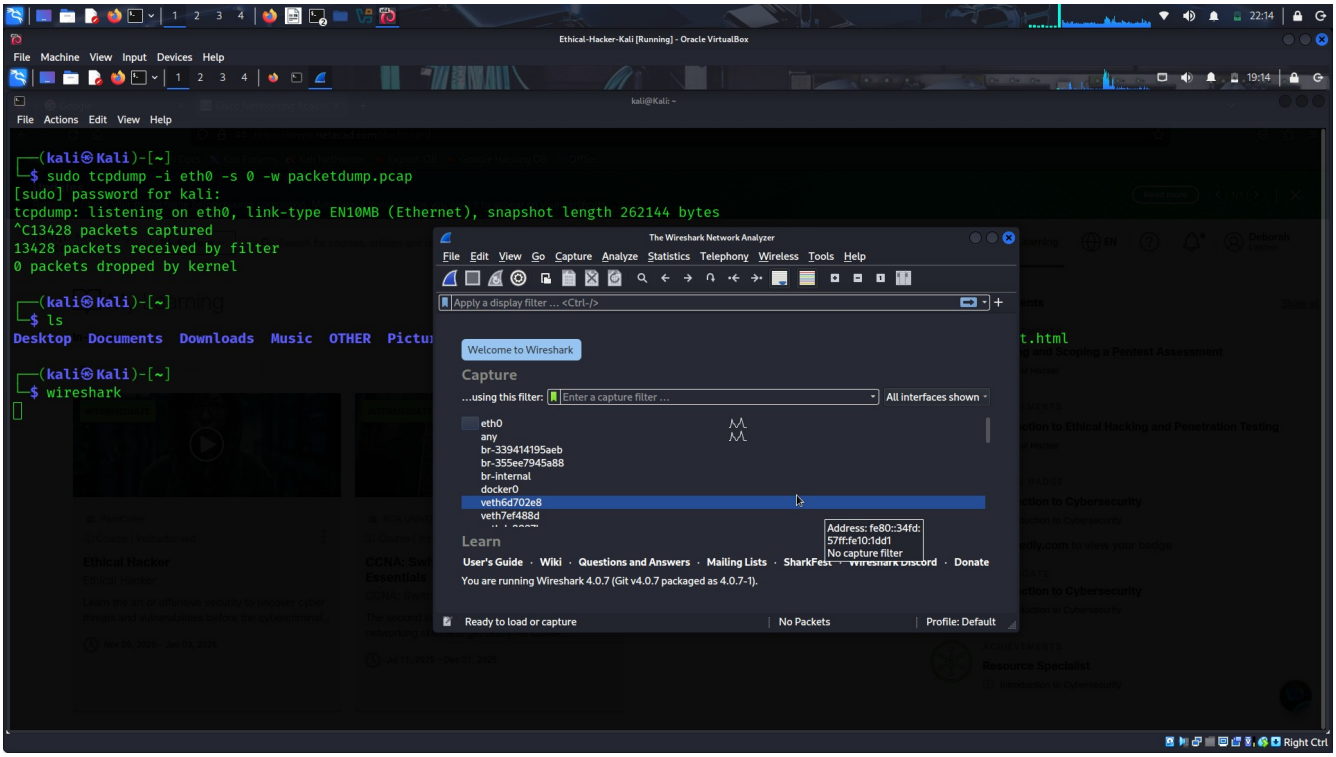


Part 3: View and Analyze the Packet capture.

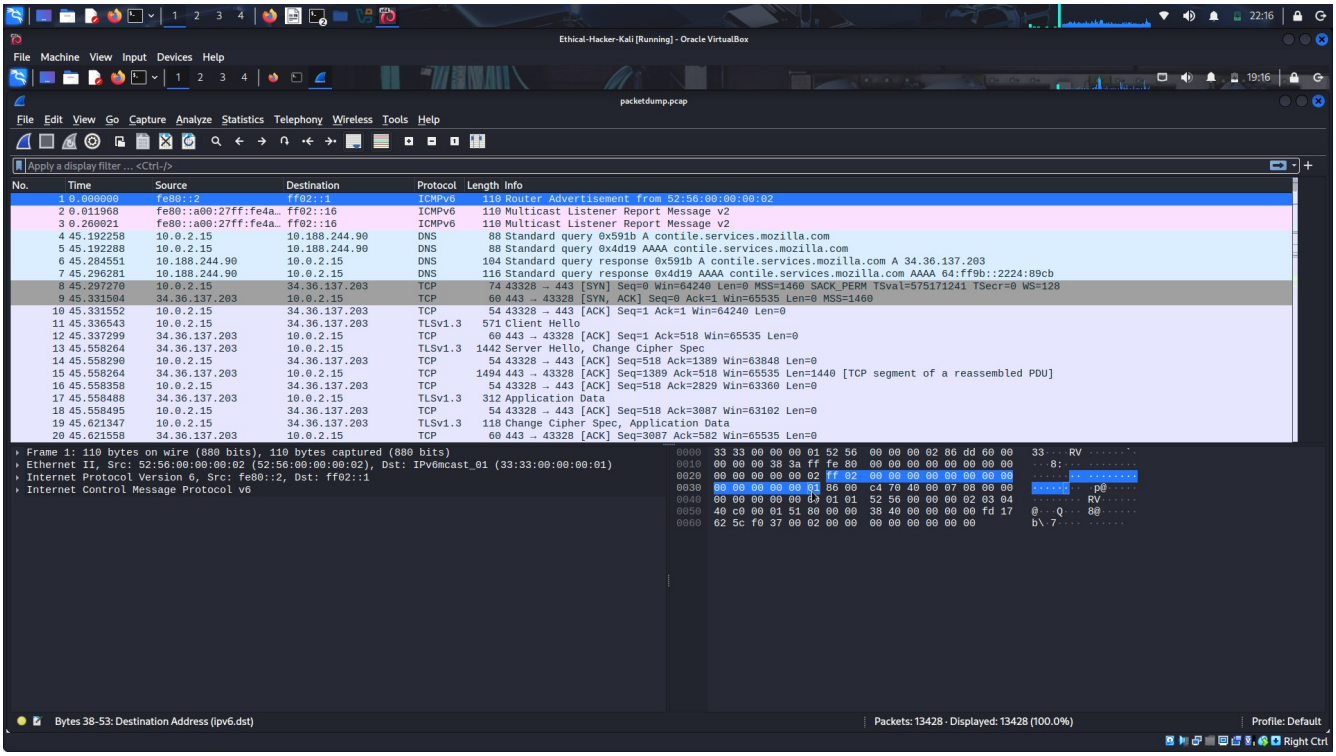
In this part, you will use Wireshark to analyze the packet capture file that you created in Part 2 of this lab.

Step 1: Open the Wireshark application to view the packet capture.

a. Use Wireshark to view the captured packets. Launch the graphical Wireshark application by typing **wireshark** at the CLI prompt.



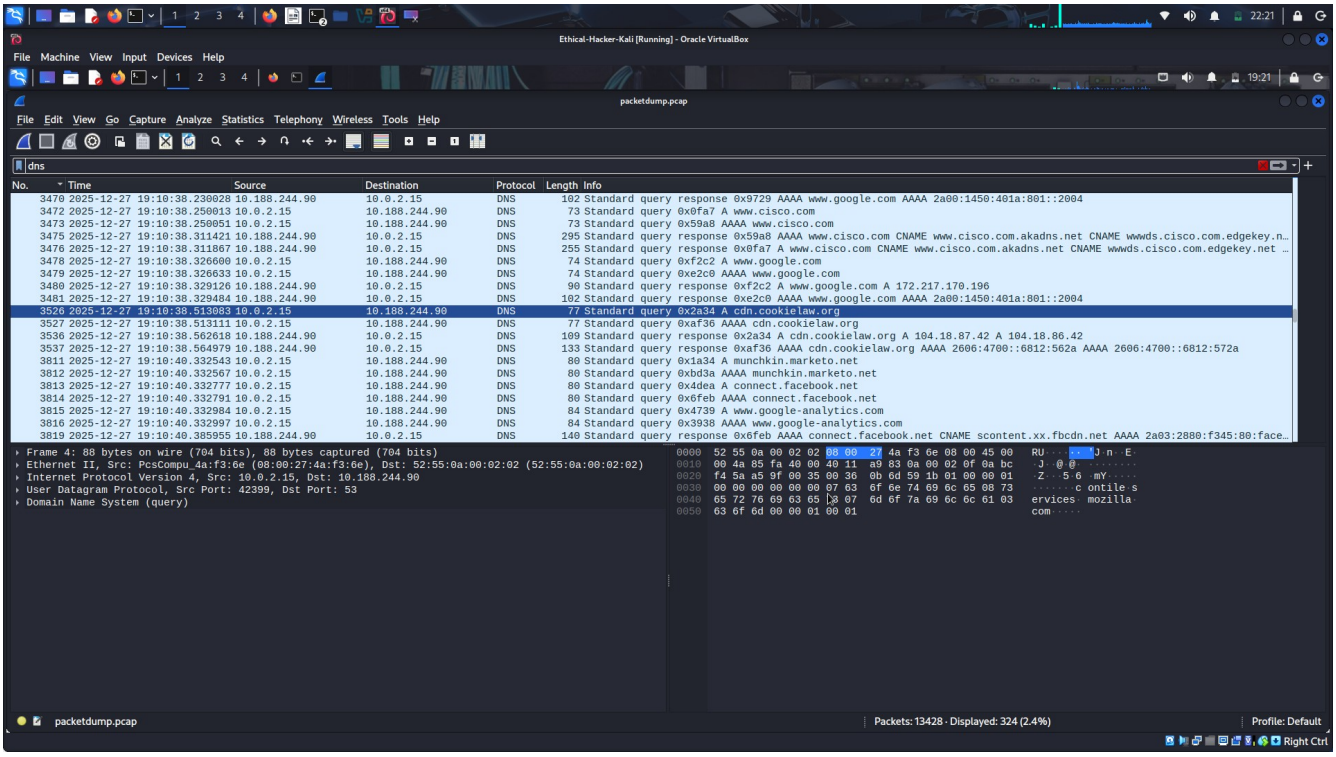
b. Use the **File -> Open** menu option and enter **packetdump.pcap** in the File Name field. Click **Open**. A screen should open displaying the contents of the **packetdump.pcap** file.



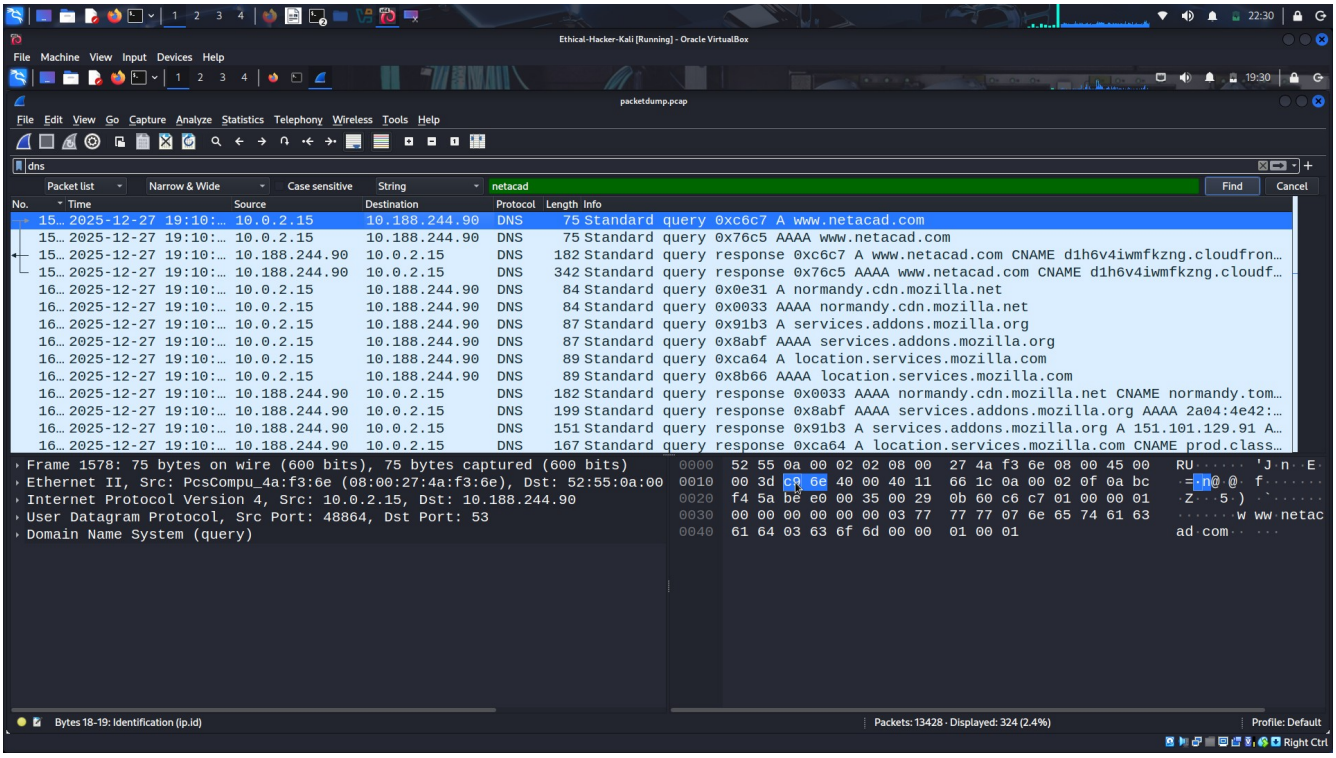
Step 2: Analyze DNS traffic.

When you type a website URL into your browser, your PC performs a DNS query to the DNS server's IP address. Observing DNS queries and responses provides the names (URLs) and IP addresses of sites a user visits. Knowing websites that are commonly visited by users can be valuable when formulating social engineering attacks.

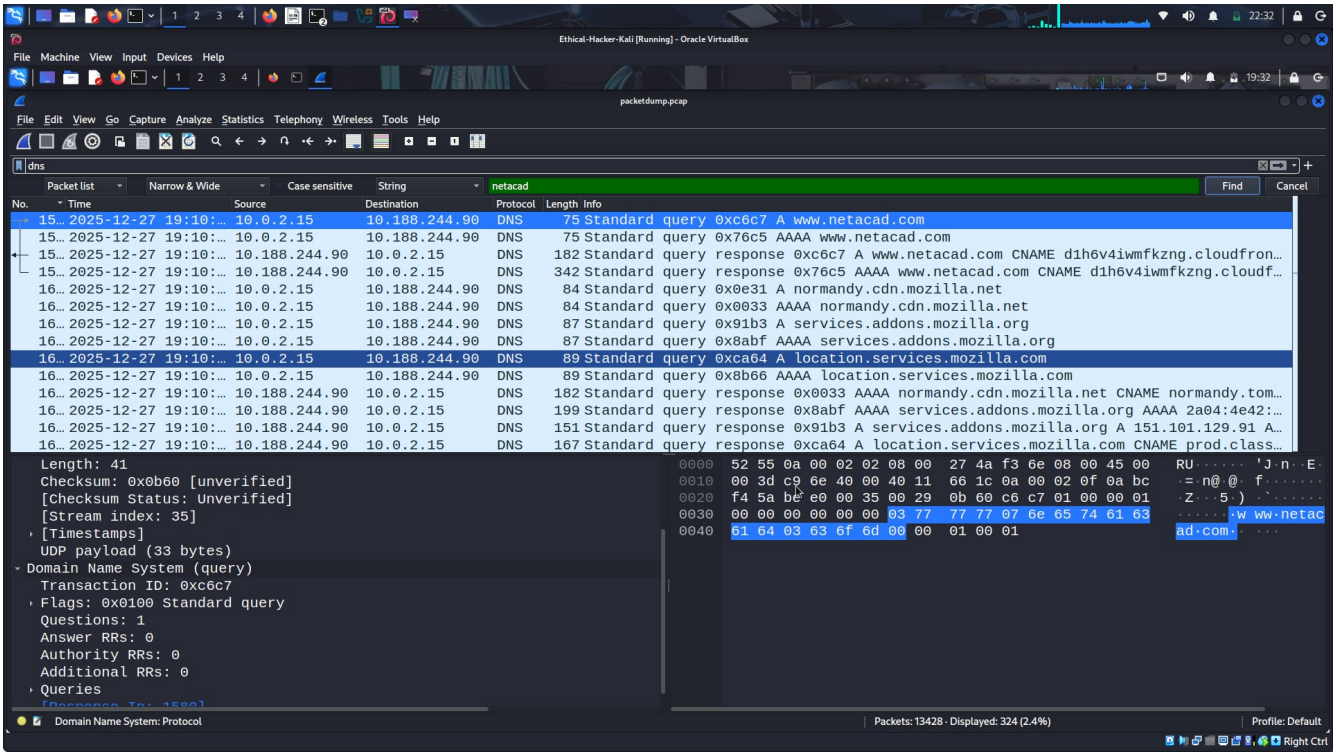
a. Filter the captured traffic to only display DNS queries and responses. Enter **dns** in the Filter Field on the Wireshark main screen. You will notice that in addition to the **Skills for All** website that you requested, other DNS lookups are shown. These correspond to links contained within the Skills For All and Google homepages.



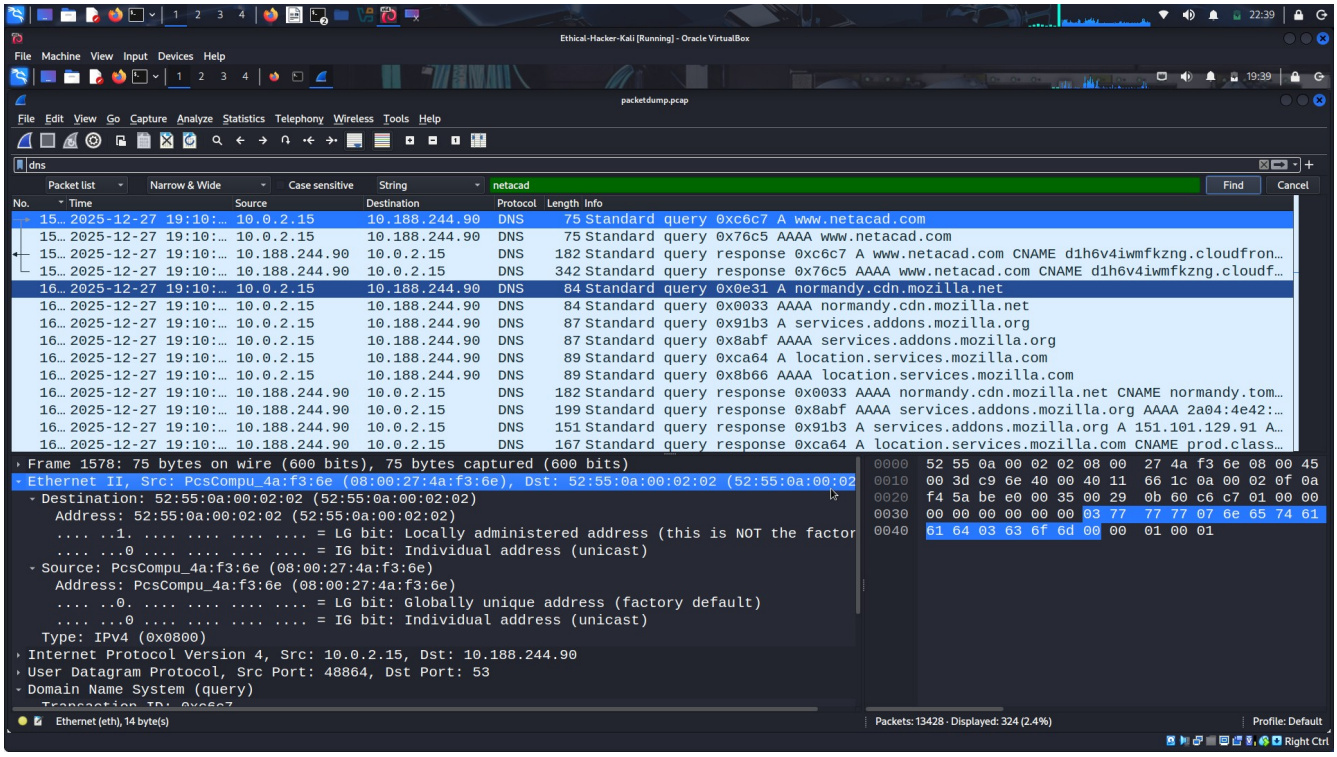
b. Click the search icon and search for the **netacad**.com hostname. Select **String** in the dropdown box and enter **netacad** in the search box.



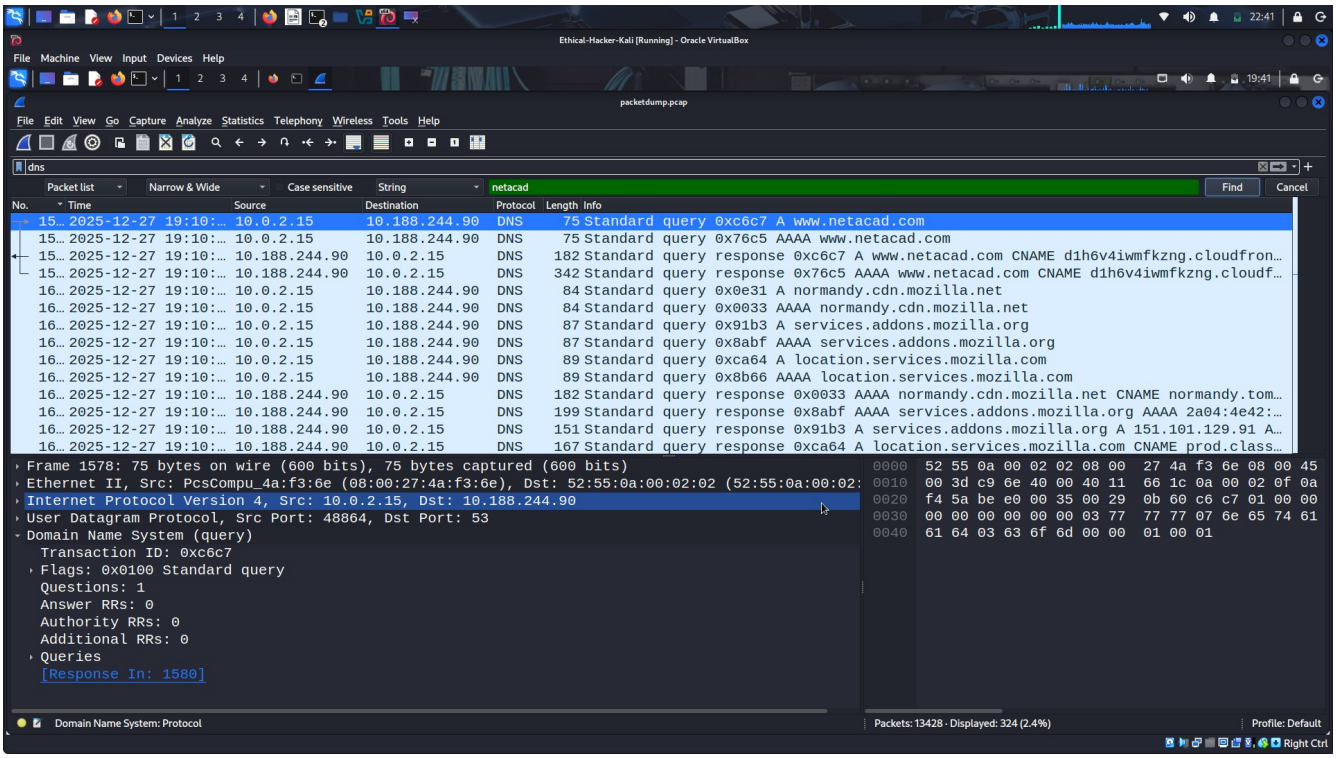
c. Select the first Standard query for the **netacad.com** website. Expand the query details pane below the packet list to view the contents of the query packet.



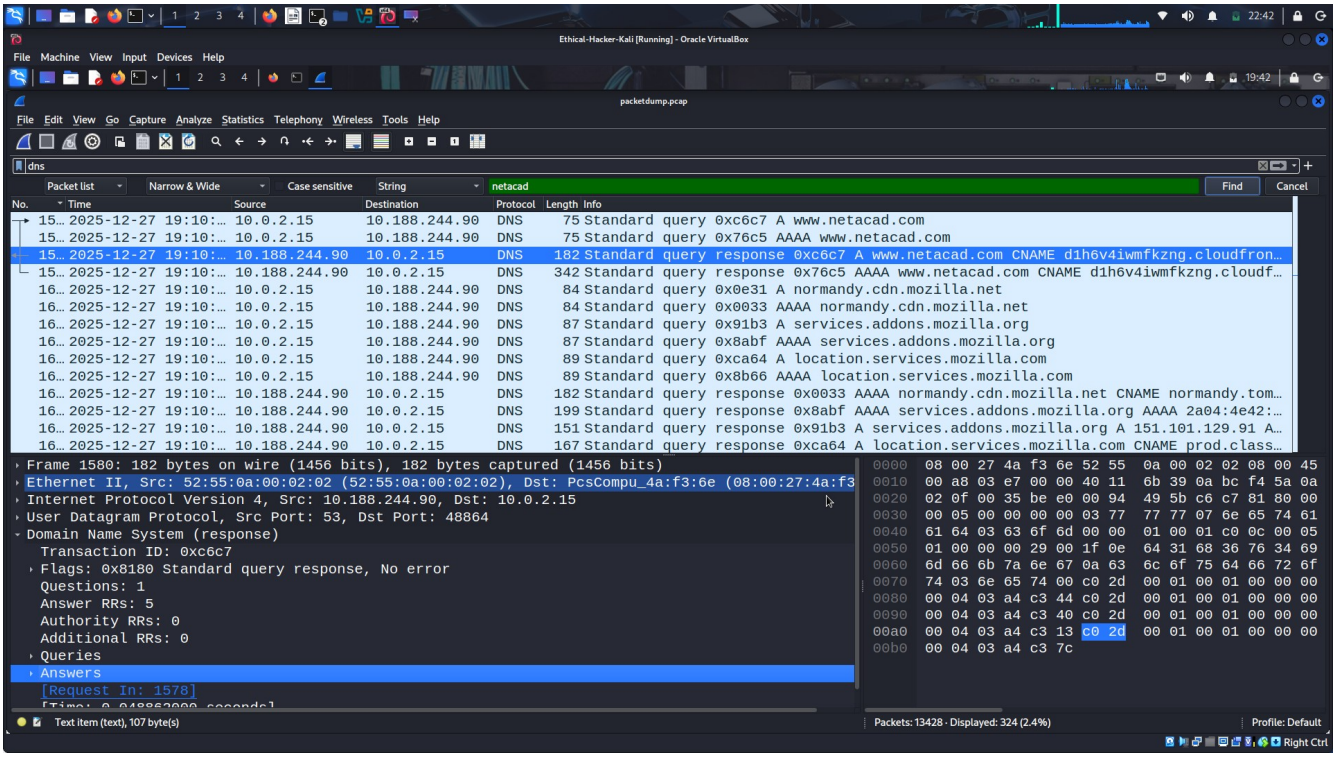
d. Expand the **Ethernet II** information to display the Layer 2 header data contained in the packet. The source MAC address is the MAC of the sending device interface, in this case the Kali VM, and the destination MAC address is the MAC of the default gateway since the DNS server is not on the same Layer 2 network.



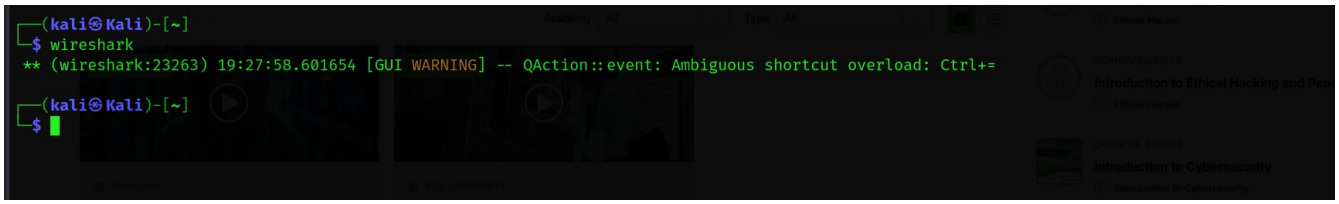
Expand the **Domain Name System (query)** section to see the details of what is being sent to the DNS server.



It also indicates the line that contains the reply packet that was received in response to the query. Double-click the link to the response. The details of the Standard query response packet are shown.



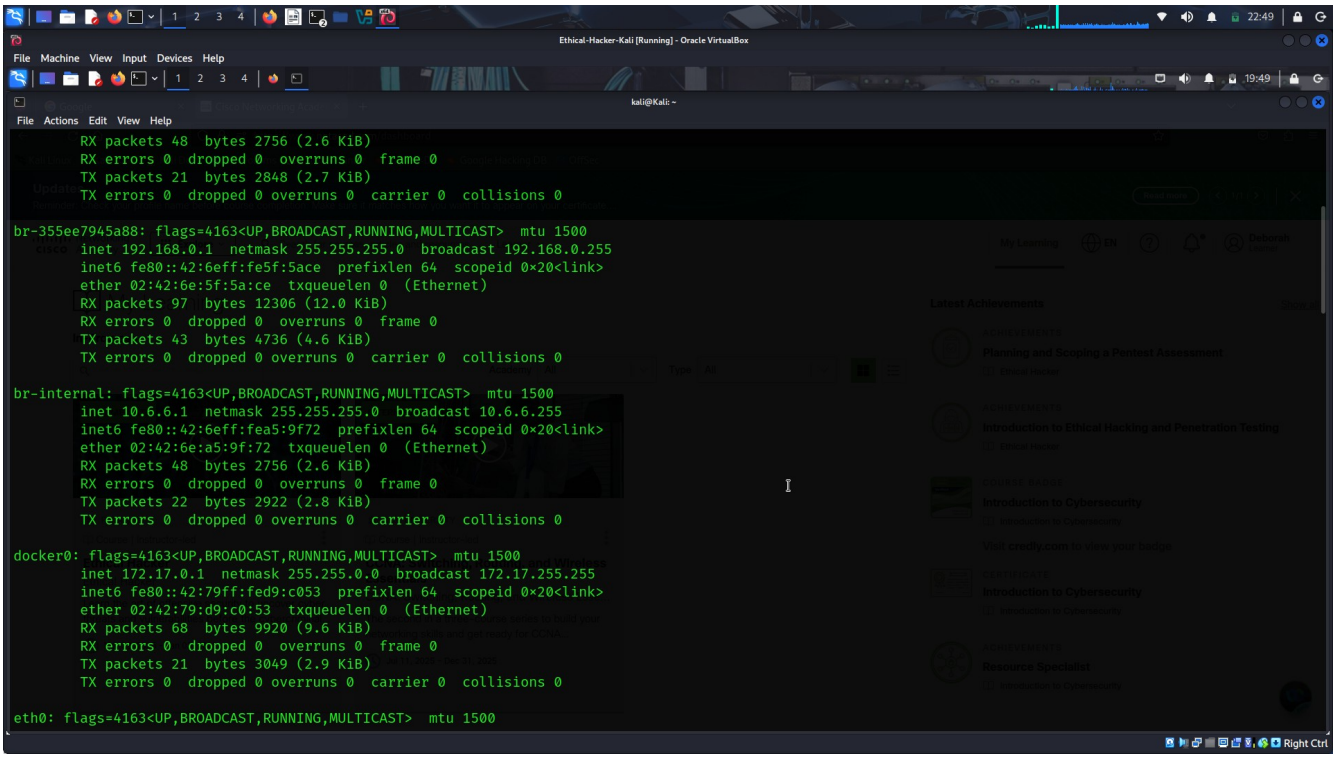
Close Wireshark to return to the CLI prompt.



Step 3: Analyze an HTTP Session

In this step, you will capture and analyze a web request and response. You will use Wireshark to capture the traffic and to analyze the messages exchanged between the web server and the client. The website server is a VM server running in a Docker container on the Kali Linux host.

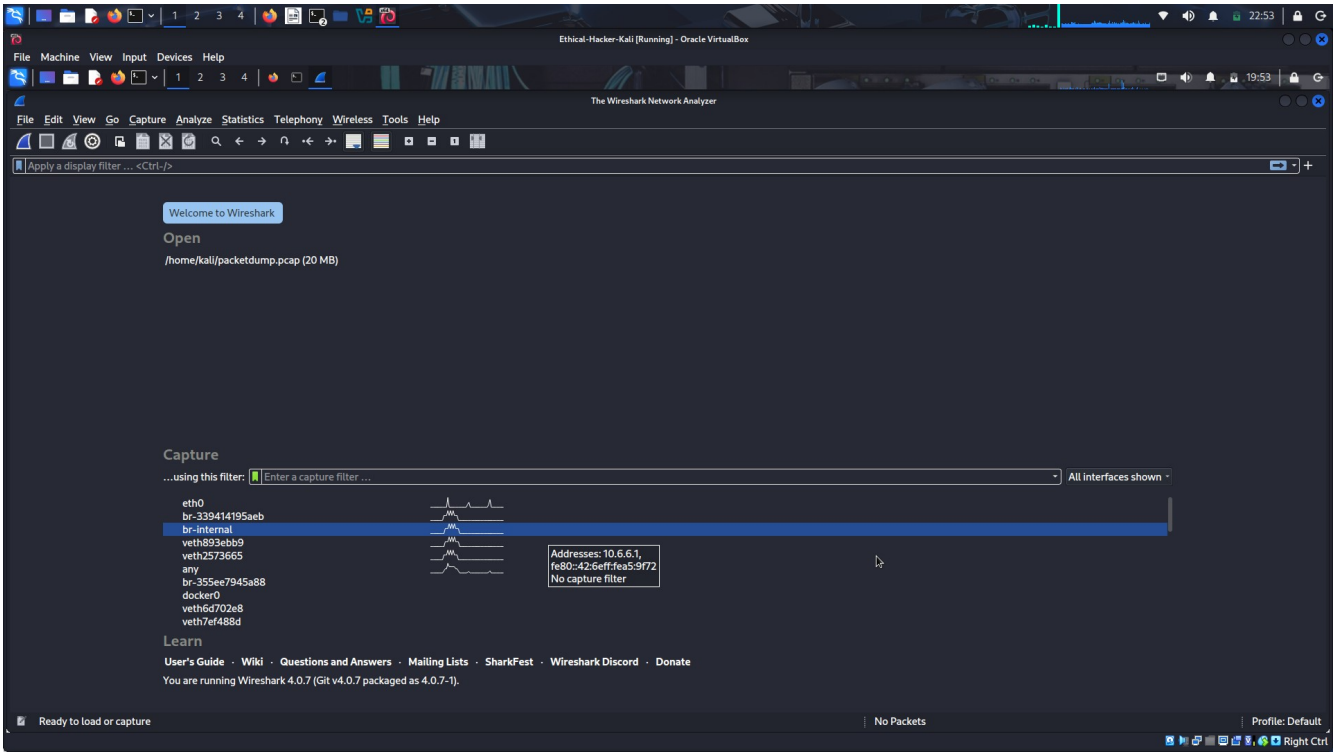
a. Use **ifconfig** to determine which interface on the Kali Linux VM is configured in the 10.6.6.0/24 network.



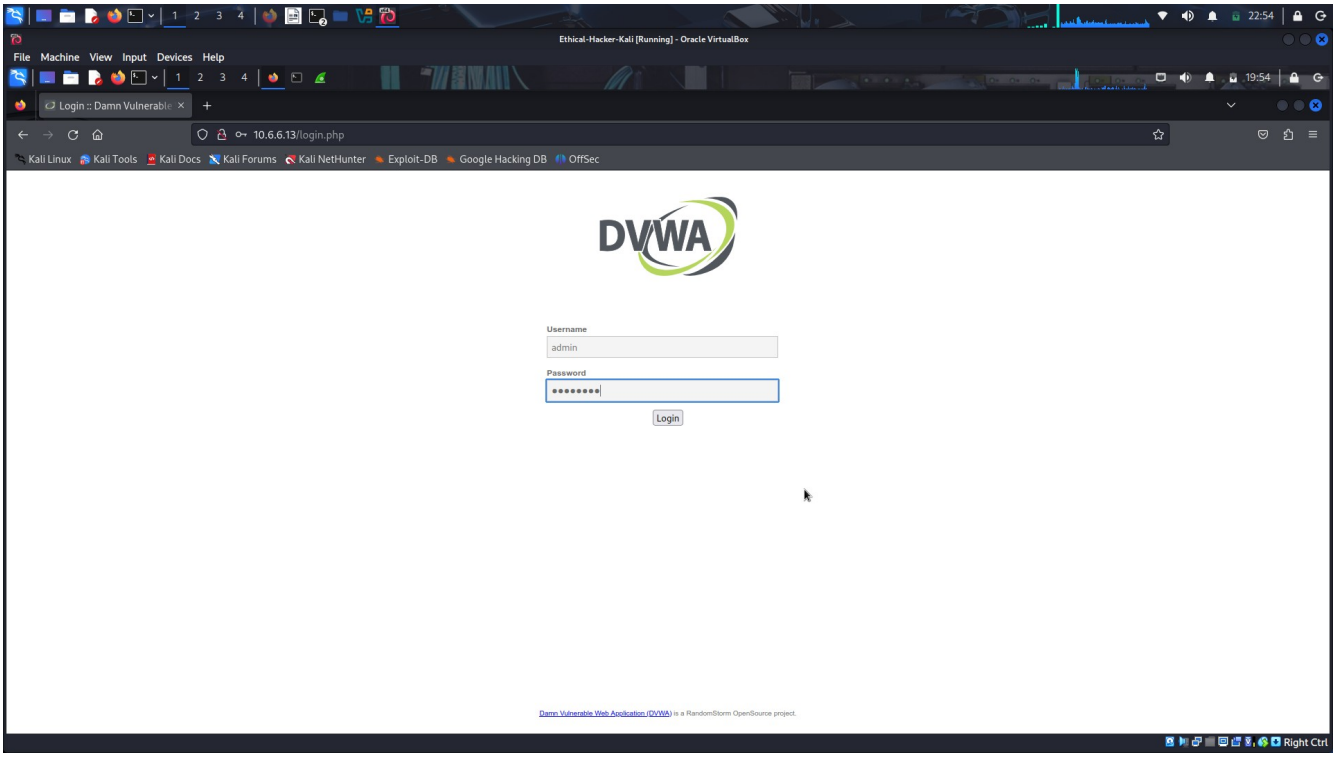
What is the name of the interface connected to the 10.6.6.0/24 network? **Br-internal**

What is the IP address assigned to that interface? **10.6.6.1**

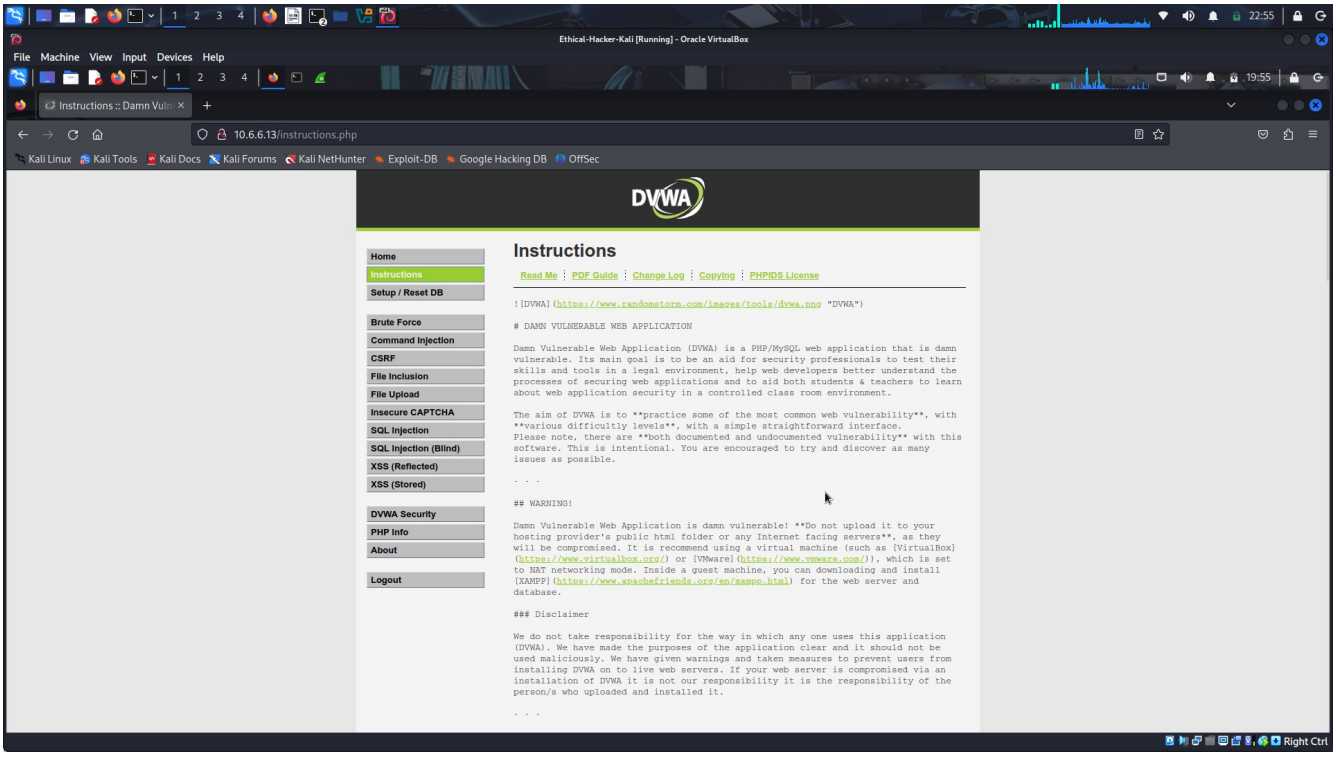
b. Open Wireshark by typing **wireshark** at the command prompt. Wireshark will open in a new window, expand the window to full screen. At the center of the main Wireshark screen there will be a list of interface names to choose to capture traffic. Double click the interface name that matches the interface name you discovered in step 2a.



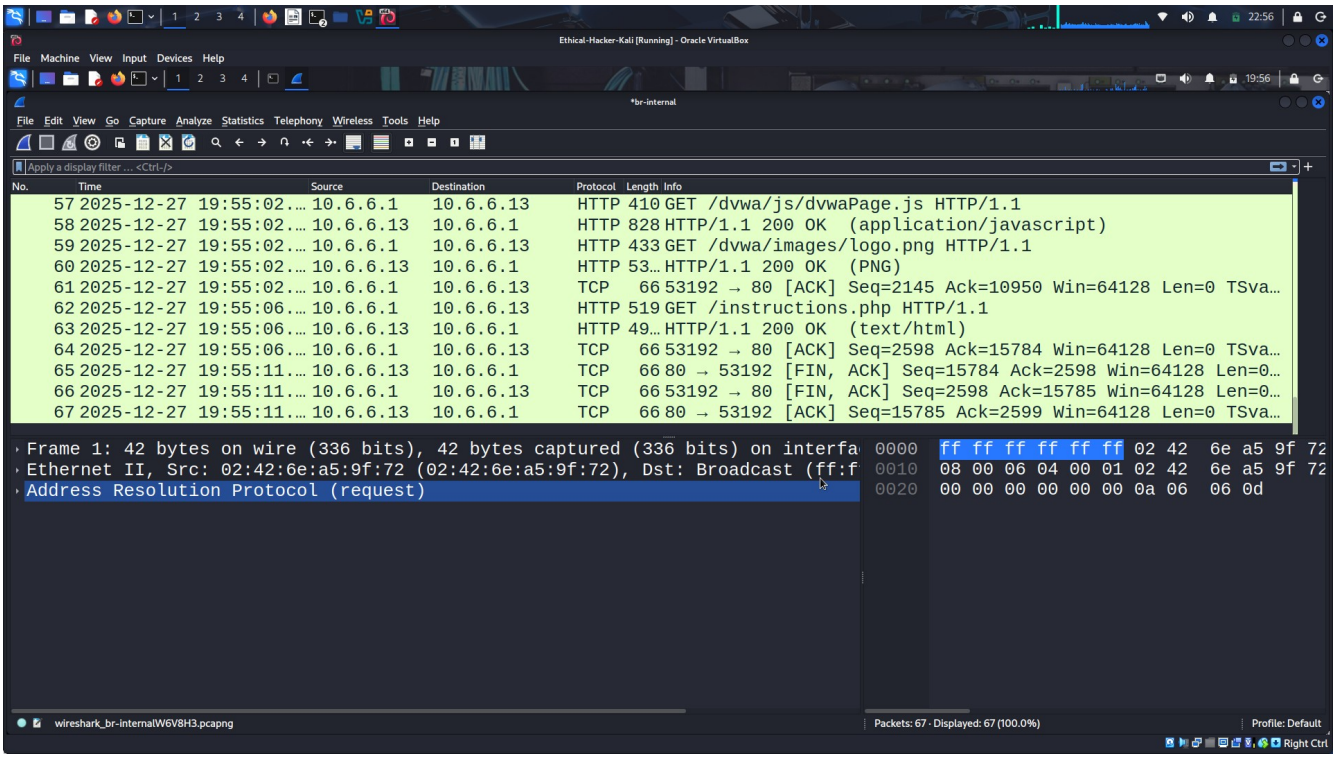
- Open a browser window and enter the IP address 10.6.6.13 on the launch bar. A login screen for the DVWA web server appears. Enter **admin** as the username and **password** as the password.



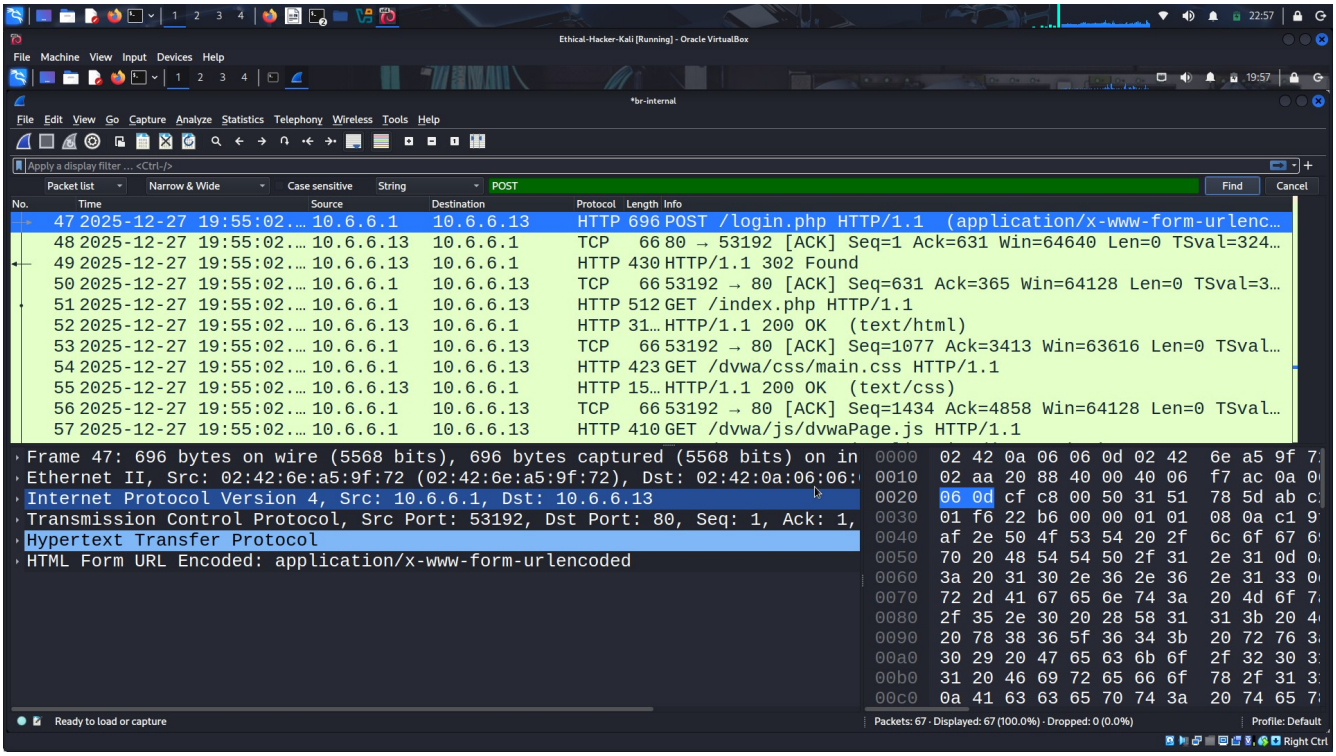
When the main DVWA page appears, click the **Instructions** button at the top of the menu on the left side of the screen. When the instructions page appears close the browser window.



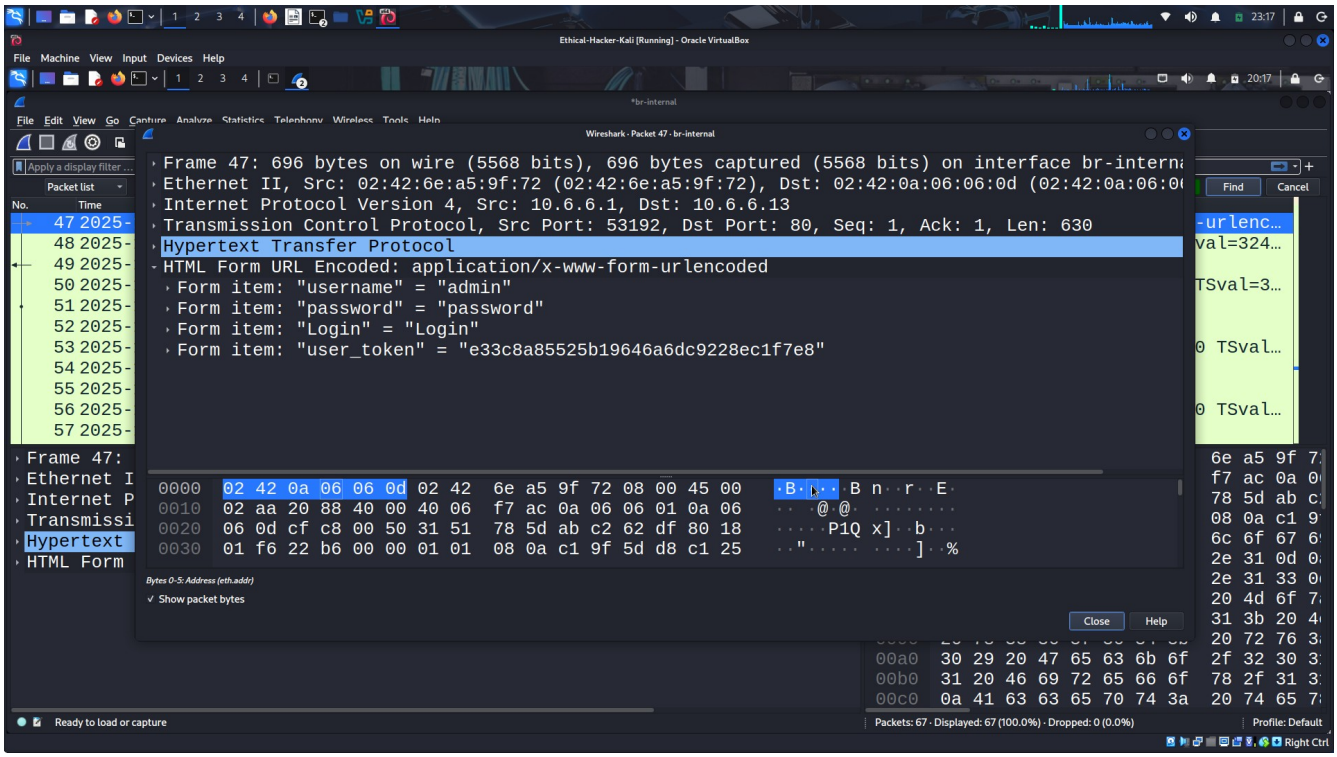
Return to the Wireshark window. Stop the capture using the **red square icon** on the menu bar.



The web server DVWA is using HTTP, not HTTPs. Use the **search icon** to find the string **POST** in the captured packets. **POST** messages transfer form data from the client to the server, in this case the login information.

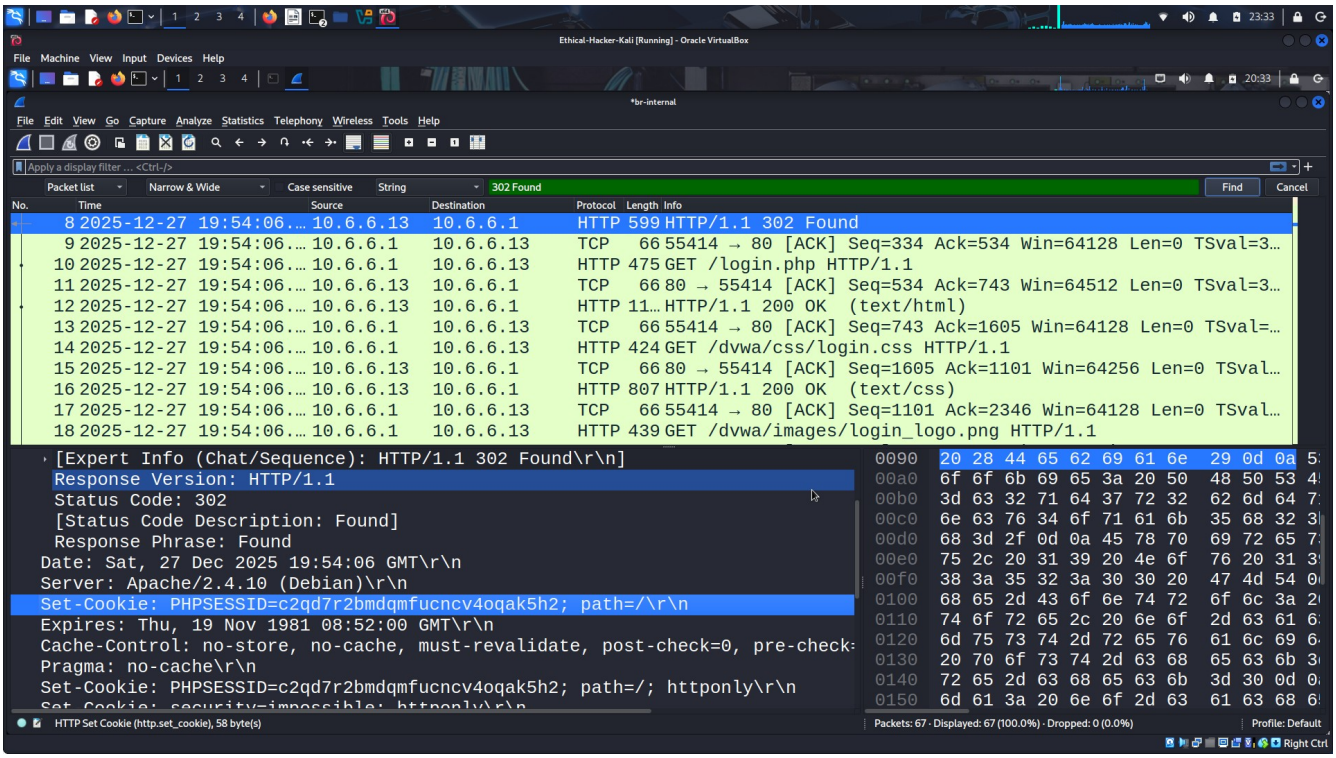


Double click the first POST packet to view the detail in a separate window. Expand the section titled **HTML Form URL Encoded**: You will see the username, password ans user token

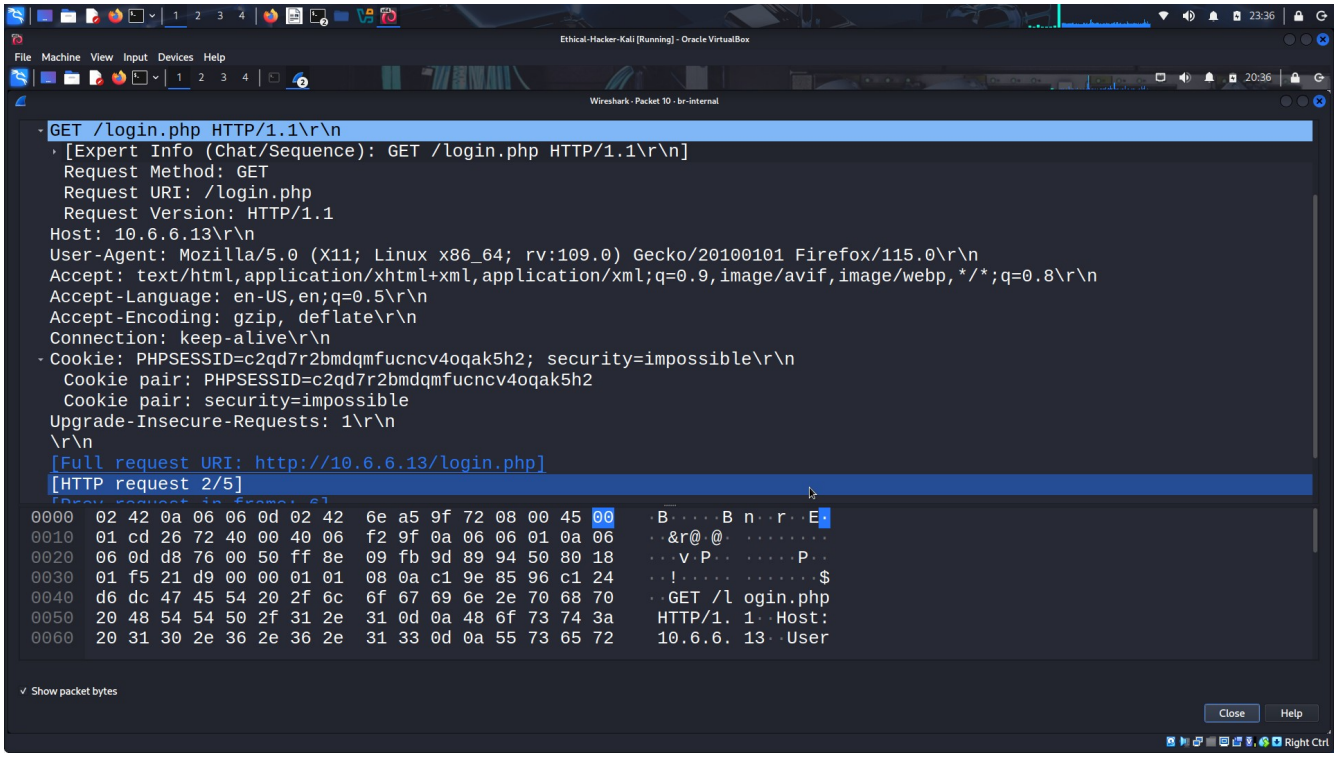


Cookies are used for various purposes. Most frequently, they are used to save information about a user's session. Cookies can be hijacked and used in session hijacking attacks. The initial cookie for a session is sent from the web server to the client with the **Set-Cookie** value in a HTTP response. Use the search icon to find the string **302 Found** in the packet pane. Double click the first packet that was

found and expand the **Hypertext Transport Protocol** section. What value is being set in the cookie being sent from the web server to the Kali client? **PHPSESSID**



Examine the next **GET** packet being sent from the Kali client browser after receiving the cookie information. Expand the **Hypertext Transfer Protocol** section. Look for the Cookie values being sent in the packet. **PHPSESSID**



Does the PHPSESSID being sent back to the server in the GET request the same as the one sent from server in the earlier reply? **YES**

Close Wireshark. You will have the option to save the .pcap file containing the capture or to quit without saving. The .pcap file will be saved in the current working directory unless otherwise specified.

Conclusion

In this lab, i had the opportunity to become familiar with capturing packets in both the tcpdump utility and the Wireshark application.

The benefits of using packet capture utilities when performing passive reconnaissance on a potential target include:

Being able to monitor and collect traffic without being detected.

Packet captures can be saved and analyzed at a later date.

Information about target systems, including websites visited and cookie values can be gathered without direct interaction with the systems.

Information that can be gathered using packet captures:

IP Addresses

Protocols

MAC addresses

DNS servers

Data sent to and from various applications

Router and switch traffic

Gateway identification