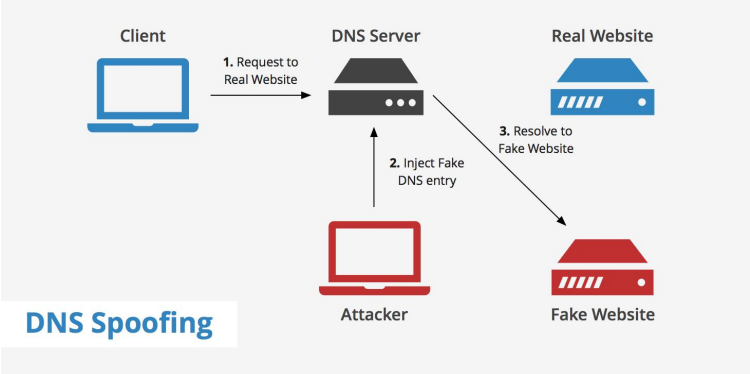
COSC 328 - Final project - 300311827 - YuanHu

Ethical hacking - part A: DNS poisoning



What is DNS poisoning?

DNS poisoning, also known as DNS spoofing, is a type of cyber attack that manipulates the Domain Name System (DNS) to redirect domain name resolutions to malicious websites. The DNS is responsible for translating human-readable domain names (like www.example.com) into the numerical IP addresses used by computers to communicate over the internet.

How does DNS poisoning work?

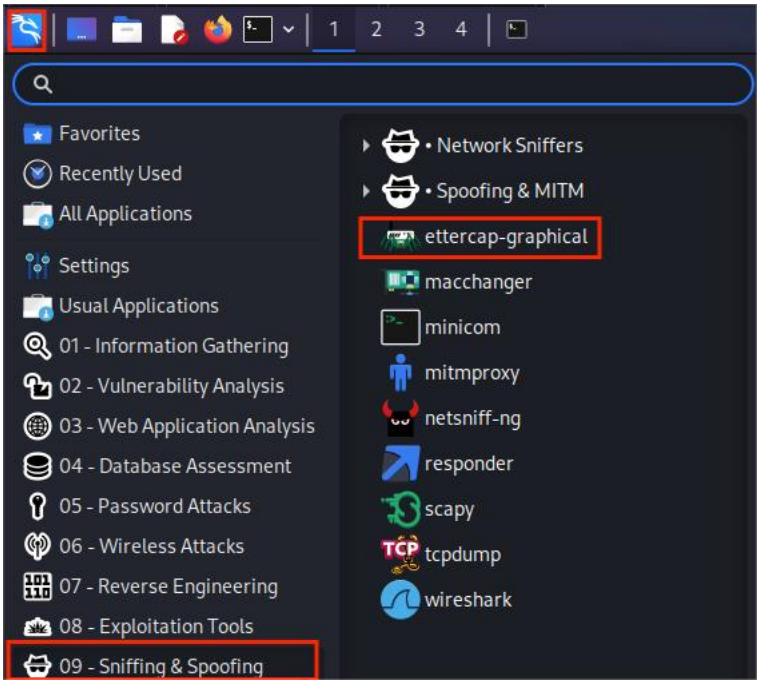
In a DNS poisoning attack, the attacker aims to corrupt the DNS cache of a DNS server or the DNS cache of a client machine. They do this by inserting false information into the cache, which causes the DNS server or client to redirect legitimate domain name queries to malicious IP addresses controlled by the attacker. This can lead users to unknowingly visit malicious websites, which may then attempt to steal sensitive information or install malware on their devices.

How to prevent DNS poisoning?

DNS poisoning attacks can be mitigated by implementing secure DNS protocols, such as DNS Security Extensions (DNSSEC), which use digital signatures to ensure the authenticity and integrity of DNS data. Additionally, regular monitoring and updating of DNS servers can help detect and prevent DNS poisoning attacks.

Replicate the Process of DNS poisoning:

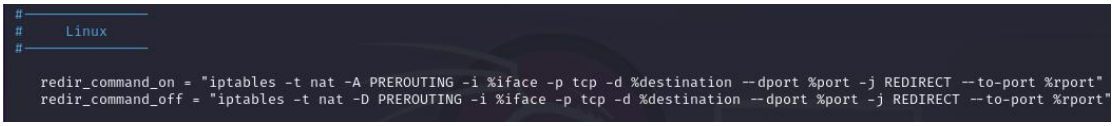
1. The tool will be used is called ***ettercap***, and it is preinstalled on Kali machine, navigate as follows:
   - Kali application -> 09.Sniffing and Spoofing -> ettercap-graphical



2. Modify the configuration file of ettercap

   - Opening the config file using the command line: *$ sudo nano /etc/ettercap/etter.conf*

   - Scroll down to the Linux/IPtables, and enable these two lines of code by removing the '#'



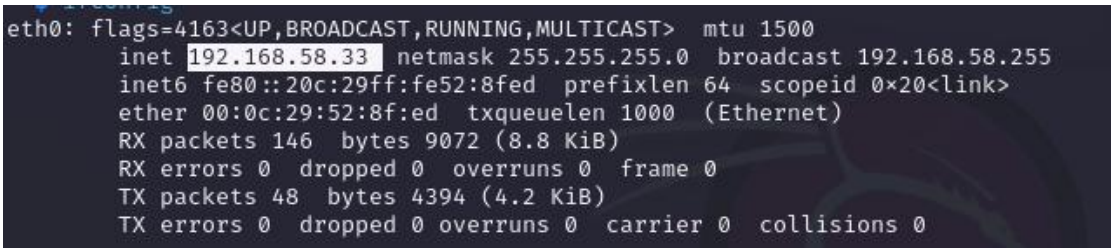
3. Modify the index.html as easy as *<h1>You got hacked1 </h1>*

4. Start local apache server with the following command:

   - *$ service apache2 start*

5. Find local ip address with *$ ifconfig*

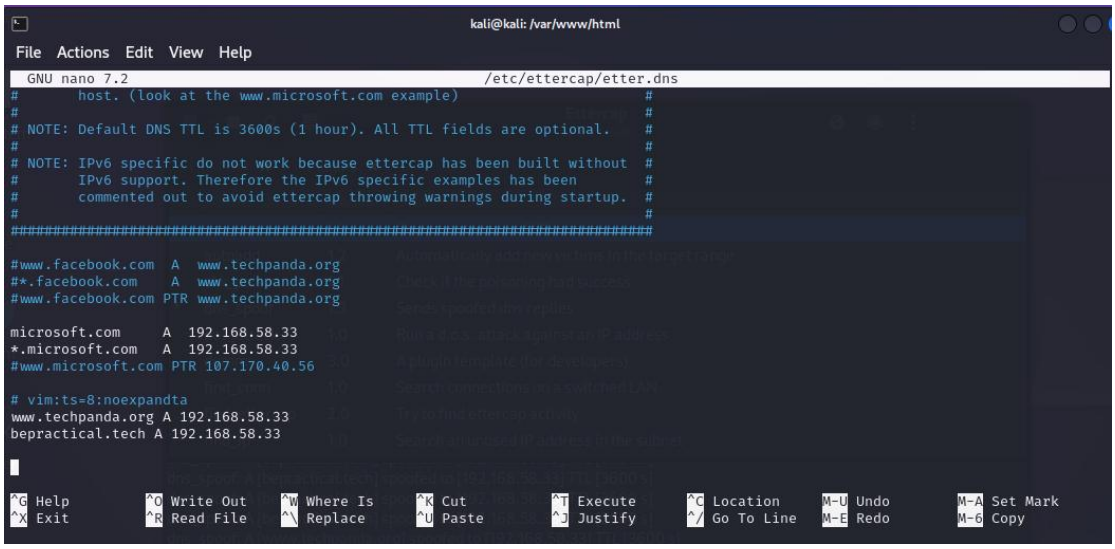
   - The result should look like this:



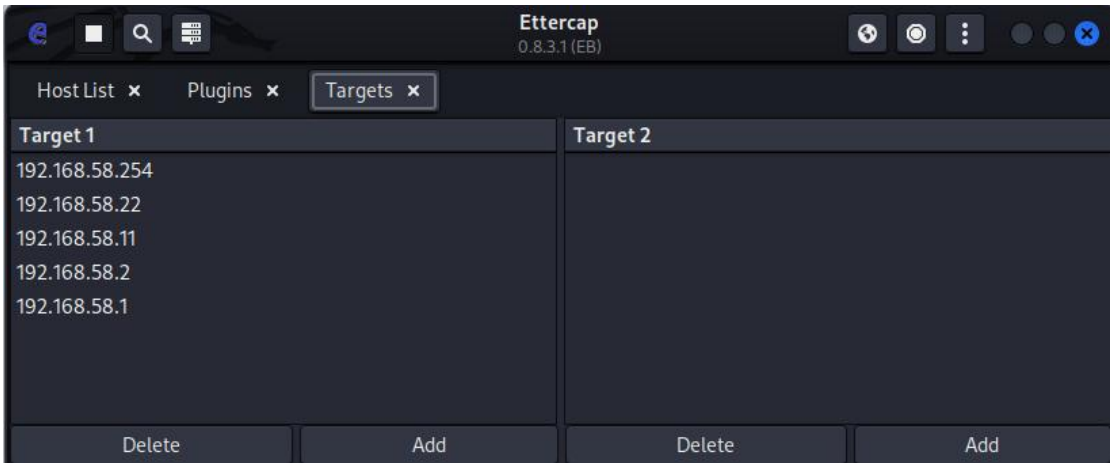
   - When opening localhost or 192.168.58.33 in a web browser, we should be able to see something like this:

6. Open the file ***etter.dns*** for more configuration using the command and add these lines of configuration:

   - *$sudo nano /etc/ettercap/etter.dns*

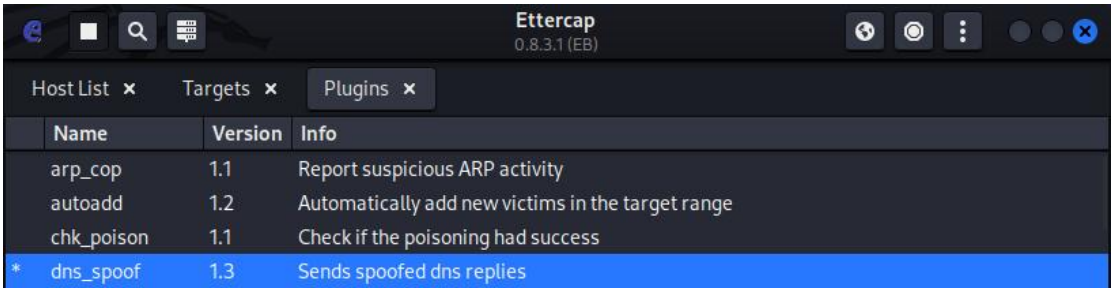


7. Open ***ettercap*** with the previous step and add the following configuration to the target and then start APR poisoning:

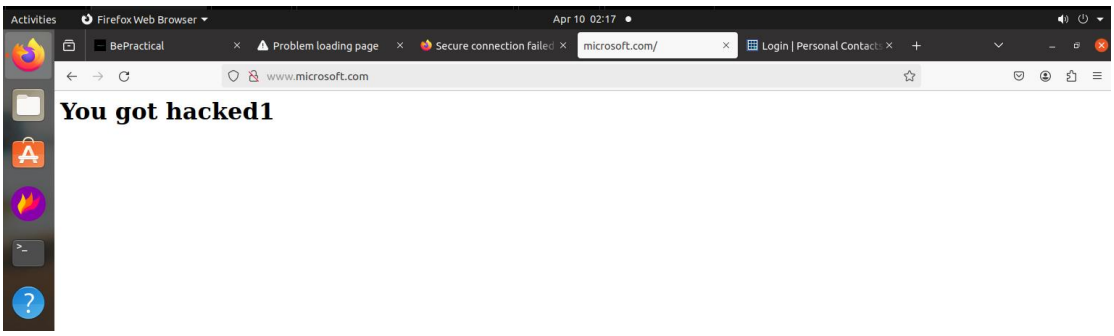


- Making sure APR poisoning is working functionally with a example website : *http://www.techpanda.org/dashboard.php*

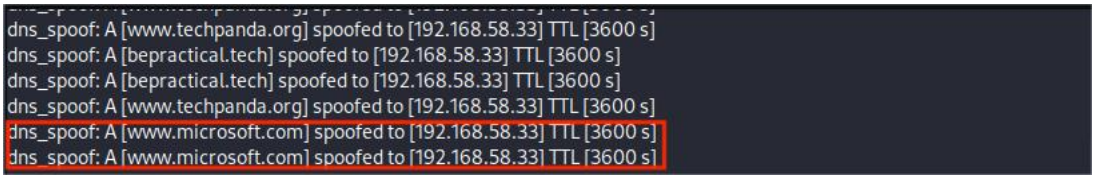
- Now start DNS spoofing by plugins/Manage plugins/dns_spoofing



- If you try to access the target page on Ubuntu machine, it will show the our attack page along with the console output like this:



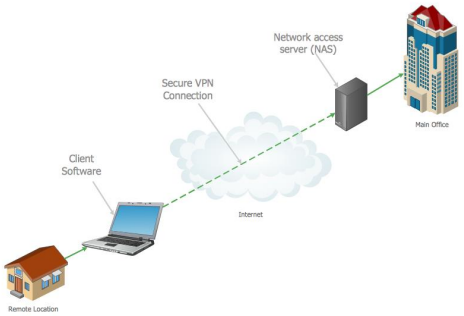
- Corresponding output shown on the kali machine ettercap console



Ethical hacking - Part B : Virtual Private Network

What is Virtual Private Network?

A VPN, or Virtual Private Network, is like a secure tunnel between your device and the internet. When you connect to a VPN, all your internet traffic goes through this tunnel, making it harder for others to see what you're doing online.



How does Virtual Private Network work?

1. **Establishing a Connection**: When you connect to a VPN service, your device (computer, smartphone, etc.) establishes a secure connection to a VPN server using a protocol (like OpenVPN, IKEv2, etc.).
2. **Encryption**: Once the connection is established, all data transmitted between your device and the VPN server is encrypted. This means that even if someone intercepts your data, they won't be able to understand it because it's scrambled.
3. Routing Your Traffic: Instead of your data traveling directly to its destination (like a website), it first goes to the VPN server. The VPN server then forwards your data to its intended destination on the internet.

4. **Masking Your IP Address**: When your data is sent from the VPN server to the internet, it appears as if it's coming from the VPN server's IP address, not your real IP address. This masks your true location and identity.
5. **Receiving Data**: When data is sent back to you from the internet, it goes to the VPN server first. The server then encrypts the data and sends it back to your device, where it's decrypted and you can access it.
6. **Privacy and Security**: By encrypting your data and masking your IP address, a VPN provides privacy and security, protecting your data from being intercepted by hackers, ISPs, or other third parties.

Set up the VPN on Ubuntu Virtual Machine

1. Update the system with the command lines:
   - $ sudo apt update
   - $ sudo apt upgrade
2. Download and install OpenVPN using the commands:
   - $ wget https://git.io/vpn -O openvpn-ubuntu-install.sh
   - $ chmod -v +x openvpn-ubuntu-install.sh
   - $ bash openvpn-ubuntu-install.sh

```
herrycooly@cosc328:~$ wget https://git.io/vpn -O openvpn-ubuntu-install.sh
--2024-04-10 03:26:46--  https://git.io/vpn
Resolving git.io (git.io)... 140.82.112.22
Connecting to git.io (git.io)|140.82.112.22|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://raw.github.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2024-04-10 03:26:46--  https://raw.github.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.github.com (raw.github.com)... 185.199.111.133, 185.199.108.133, 185.199.109.133, ...
Connecting to raw.github.com (raw.github.com)|185.199.111.133|:443... connected.
HTTP request sent, awaiting response... 301 Moved Permanently
Location: https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh [following]
--2024-04-10 03:26:46--  https://raw.githubusercontent.com/Nyr/openvpn-install/master/openvpn-install.sh
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 185.199.108.133, 185.199.109.133, 185.199.110.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|185.199.108.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 23714 (23K) [text/plain]
Saving to: ‘openvpn-ubuntu-install.sh’

openvpn-ubuntu-inst 100%[===================>]  23.16K  --.-KB/s    in 0.001s

2024-04-10 03:26:47 (23.1 MB/s) - ‘openvpn-ubuntu-install.sh’ saved [23714/23714]

herrycooly@cosc328:~$ chmod -v +x openvpn-ubuntu-install.sh
mode of 'openvpn-ubuntu-install.sh' changed from 0664 (rw-rw-r--) to 0775 (rwxrwxr-x)
```

   - After the complete the promoted configuration settings, you should see a similar output like this

```
Success
Created symlink /etc/systemd/system/multi-user.target.wants/openvpn-server@server.service → /lib/systemd/system/openvpn-server@.service.

Finished!

The client configuration is available in: /root/client.ovpn
New clients can be added by running this script again.
```

   - Check that the Openvpn service is running by:

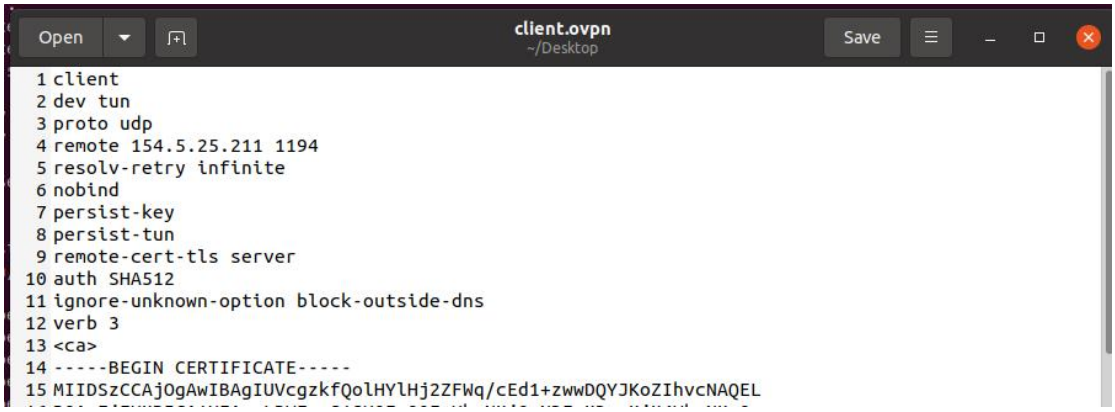
   $ sudo systemctl status openvpn-server@server.service

```
herrycooly@cosc328:~$ sudo systemctl status openvpn-server@server.service
[sudo] password for herrycooly:
● openvpn-server@server.service - OpenVPN service for server
     Loaded: loaded (/lib/systemd/system/openvpn-server@.service; enabled; vendor preset: enabled)
     Active: active (running) since Wed 2024-04-10 03:32:40 EST; 22min ago
       Docs: man:openvpn(8)
             https://community.openvpn.net/openvpn/wiki/Openvpn24ManPage
             https://community.openvpn.net/openvpn/wiki/HOWTO
   Main PID: 16623 (openvpn)
     Status: "Initialization Sequence Completed"
      Tasks: 1 (limit: 2217)
     Memory: 1.0M
     CGroup: /system.slice/system-openvpn\x2dserver.slice/openvpn-server@server.service
             └─16623 /usr/sbin/openvpn --status /run/openvpn-server/status-server.log --status-version 2 --suppress-timestamps --config server.conf

Apr 10 03:32:40 cosc328.okc openvpn[16623]: Could not determine IPv4/IPv6 protocol. Using AF_INET
Apr 10 03:32:40 cosc328.okc openvpn[16623]: Socket Buffers: R=[212992->212992] S=[212992->212992]
Apr 10 03:32:40 cosc328.okc openvpn[16623]: UDPv4 link local (bound): [AF_INET]192.168.58.22:1194
Apr 10 03:32:40 cosc328.okc openvpn[16623]: UDPv4 link remote: [AF_UNSPEC]
Apr 10 03:32:40 cosc328.okc openvpn[16623]: GID set to nogroup
Apr 10 03:32:40 cosc328.okc openvpn[16623]: UID set to nobody
Apr 10 03:32:40 cosc328.okc openvpn[16623]: MULTI: multi_init called, r=256 v=256
Apr 10 03:32:40 cosc328.okc openvpn[16623]: IFCONFIG POOL: base=10.8.0.2 size=252, ipv6=0
Apr 10 03:32:40 cosc328.okc openvpn[16623]: IFCONFIG POOL LIST
Apr 10 03:32:40 cosc328.okc openvpn[16623]: Initialization Sequence Completed
```

3. Download the client side of Openvpn on client machine via https://openvpn.net/client/

   - Copy the config .ovpn file generated by the host machine to the client machine, it should look like the following picture and paste it to the client machine



4. Connect using the client machine with the .ovpn file

   - $ sudo openvpn --config /etc/openvpn/client.conf

```
herrycooly@cosc328:~$ sudo openvpn --config /etc/openvpn/client.conf
Wed Apr 10 04:09:56 2024 OpenVPN 2.4.12 x86_64-pc-linux-gnu [SSL (OpenSSL)] [LZO] [LZ4] [EPOLL] [PKCS11] [MH/PKTINFO] [AEAD] built
on Aug 21 2023
Wed Apr 10 04:09:56 2024 library versions: OpenSSL 1.1.1f  31 Mar 2020, LZO 2.10
Wed Apr 10 04:09:56 2024 Outgoing Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
Wed Apr 10 04:09:56 2024 Outgoing Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authentication
Wed Apr 10 04:09:56 2024 Incoming Control Channel Encryption: Cipher 'AES-256-CTR' initialized with 256 bit key
Wed Apr 10 04:09:56 2024 Incoming Control Channel Encryption: Using 256 bit message hash 'SHA256' for HMAC authentication
Wed Apr 10 04:09:56 2024 TCP/UDP: Preserving recently used remote address: [AF_INET]154.5.25.211:1194
Wed Apr 10 04:09:56 2024 Socket Buffers: R=[212992->212992] S=[212992->212992]
Wed Apr 10 04:09:56 2024 UDP link local: (not bound)
Wed Apr 10 04:09:56 2024 UDP link remote: [AF_INET]154.5.25.211:1194
```

5. Verify the connection with the command line on the service machine

   - $ ip a show tun0

```
herrycooly@cosc328:~$ ip a show tun0
5: tun0: <POINTOPOINT,MULTICAST,NOARP,UP,LOWER_UP> mtu 1500 qdisc fq_codel state UNKNOWN group default qlen 100
    link/none
    inet 10.8.0.1/24 brd 10.8.0.255 scope global tun0
       valid_lft forever preferred_lft forever
    inet6 fe80::552c:77b2:f9f7:efae/64 scope link stable-privacy
       valid_lft forever preferred_lft forever
```

   This shows a successful connection between the client and the vpn

   - We can also capture the data on wireshark confirming it is using openVPN protocol