Network Traffic Analysis and Security Monitoring Project

Prepared by: Manojsai Bangaru

Masters in Computer Science

University of North Texas

Project Overview

This project involves setting up a virtual environment with Ubuntu and Kali Linux machines to simulate real-world network interactions. Using Wireshark, it captures normal traffic (like DNS and HTTP requests) and malicious activities (such as ARP and DNS spoofing). The project also analyzes SSL/TLS encrypted traffic to evaluate secure communication protocols. By identifying key indicators of attacks and secure protocols, the project demonstrates how network monitoring can be used to detect vulnerabilities and enhance overall security.

Table of Contents

Introduction

In the digital age, network traffic analysis is a vital skill for identifying and addressing security threats. By examining patterns within data flow, cybersecurity professionals can detect vulnerabilities, intrusions, and signs of attacks. This project uses Wireshark to explore network monitoring techniques in a controlled virtual environment, analyzing both normal network behaviors and simulated attack traffic. Through capturing and assessing various traffic types, the project highlights how traffic analysis helps protect networks from common security threats.

Objectives

- Set Up Virtual Environment: Configure Ubuntu and Kali Linux VMs to simulate real-world network traffic.
- Analyze Normal Traffic: Capture DNS, HTTP, and HTTPS traffic to establish a baseline of typical network behavior.
- Simulate ARP and DNS Spoofing Attacks: Conduct spoofing attacks and analyze indicators of compromise in Wireshark.

- Examine SSL/TLS Encryption: Review SSL/TLS handshakes and encryption details to assess secure communication.
- Highlight Security Insights: Identify vulnerabilities in spoofing attacks and assess secure protocols to demonstrate the value of network traffic analysis.

Step 1: Setting Up the Virtual Environment

In this step, you will set up two virtual machines using **VirtualBox**—one with **Ubuntu** and the other with **Kali Linux**. The Ubuntu VM will act as the victim machine, and the Kali Linux VM will act as the attacker machine.

Step 1.1: Install VirtualBox

If you haven't installed VirtualBox yet, follow these steps:

1. **Download VirtualBox**: Go to the VirtualBox website, download the latest version, and install it.
2. **Install VirtualBox**: Follow the on-screen instructions to complete the installation.

Step 1.2: Download Ubuntu and Kali Linux ISOs

1. **Ubuntu ISO**: Download the latest version of Ubuntu.
2. **Kali Linux ISO**: Download the latest version of Kali Linux.

Step 1.3: Set Up the Ubuntu Virtual Machine (VM)

1. **Create a New VM** in VirtualBox:
    - Open VirtualBox and click **New**.
    - Name your VM (e.g., "Ubuntu VM").
    - Set **Type** to **Linux** and **Version** to **Ubuntu (64-bit)**.
    - Click **Next**.
2. **Allocate Memory**: Allocate around **2 GB (2048 MB)** of RAM to the VM (or more if your system has sufficient memory).
3. **Create a Virtual Hard Disk**:
    - Select **Create a virtual hard disk now** and click **Create**.
    - Use the default settings (VDI, Dynamically Allocated), and set at least **20 GB** of space.

4. **Configure Ubuntu ISO**:
    - Go to **Settings → Storage**.
    - Under **Controller: IDE**, click **Empty** and then click the **CD icon** on the right to choose a disk file.
    - Select the **Ubuntu ISO** you downloaded earlier.
5. **Start the VM**:
    - Click **Start** to boot the Ubuntu VM.
    - Follow the on-screen instructions to install **Ubuntu** on the VM (selecting the default settings should be fine).

Step 1.4: Set Up the Kali Linux VM

1. **Create a New VM** for Kali Linux:
    - In VirtualBox, click **New**.
    - Name it (e.g., "Kali Linux VM").
    - Set **Type** to **Linux** and **Version** to **Debian (64-bit)** (Kali is based on Debian).
    - Click **Next**.
2. **Allocate Memory**: Allocate **2 GB (2048 MB)** of RAM to the Kali Linux VM.
3. **Create a Virtual Hard Disk**:
    - Select **Create a virtual hard disk now** and click **Create**.
    - Use the default settings (VDI, Dynamically Allocated), and allocate at least **20 GB** of storage.
4. **Configure Kali ISO**:
    - Go to **Settings → Storage**.
    - Under **Controller: IDE**, click **Empty** and then click the **CD icon** to choose the Kali Linux ISO file.
5. **Start the VM**:
    - Click **Start** to boot the Kali Linux VM.
    - Follow the on-screen prompts to install Kali Linux.

Step 1.5: Configure Network Settings

To ensure your Ubuntu and Kali Linux VMs can communicate with each other and simulate network traffic, configure the **network settings**.

1. **NAT Network**:
    - In VirtualBox, go to **File → Preferences → Network → NAT Networks**.
    - Create a new NAT network, if not already available, by clicking the + button.
2. **Attach the VMs to the NAT Network**:
    - For each VM (Ubuntu and Kali), go to **Settings → Network**.
    - In the **Adapter 1** tab, select **Attached to: NAT Network**, and choose the newly created NAT network.

This setup allows both VMs to communicate with each other and access the internet.

Step 1.6: Test the Network

1. **Start both VMs (Ubuntu and Kali)** and log into them.
2. In **Ubuntu**, open a terminal and find its IP address:

```bash
ifconfig
```

Look for the IP address under the **eth0** or **enp0s3** interface.

3. In **Kali Linux**, also find the IP address with:

```bash
ifconfig
```

4. **Ping from Ubuntu to Kali**: In Ubuntu, ping the IP address of the Kali Linux VM to ensure the two VMs can communicate:

```bash
ping [Kali_IP_Address]
```

5. **Ping from Kali to Ubuntu**: In Kali, ping the Ubuntu VM's IP address:

```bash
ping [Ubuntu_IP_Address]
```

If both VMs can successfully ping each other, your network is set up correctly!

Step 1.7: Update the System

1. In both **Ubuntu & Kali,** open a terminal and run the following command:

```bash
sudo apt update && sudo apt upgrade -y
```

2. **Reboot Ubuntu**: After your system is up to date, reboot your system to apply the changes:

```bash
sudo reboot
```

Step 2: Install and Use of Wireshark on Ubuntu

Step 2.1: Update Your System:

First, ensure that your system is up to date by running the following command in the **Ubuntu VM**:

```bash
sudo apt update && sudo apt upgrade -y
```

Step 2.2: Install Wireshark:

Install Wireshark using the following command:

```bash
sudo apt install wireshark -y
```

Step 2.3: Configure Wireshark for Non-root Users:

During installation, you'll be prompted with the question: **"Should non-superusers be able to capture packets?"**. Select **Yes** to allow non-root users to capture packets.

After the installation is complete, add your user to the `wireshark` group to use Wireshark without root privileges:

```bash
sudo usermod -aG wireshark $(whoami)
```

Step 2.4: Reboot Ubuntu:

After adding your user to the `wireshark` group, reboot your system to apply the changes:

```bash
sudo reboot
```

Complete the installation by following the on-screen instructions.

Step 2.5: launch Wireshark:

- In Ubuntu, click on the Activities menu in the top left corner.
- Type Wireshark and click on the Wireshark icon to open it.

Step 2.6: Select the Network Interface:

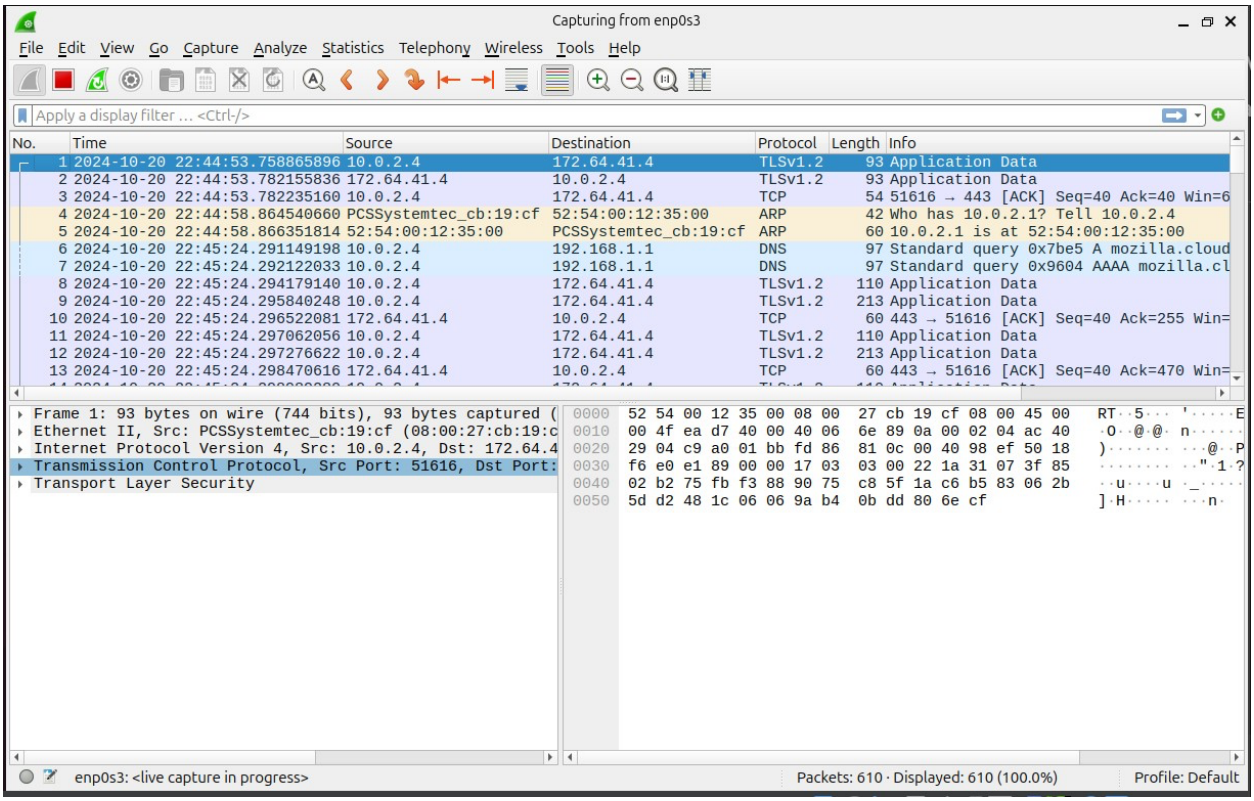
Once Wireshark is open, you need to select the correct network interface to capture traffic. You will see a list of network interfaces (usually labeled eth0, enp0s3, or something similar). These represent your VM's network connections.

- Look for the interface that's actively sending and receiving data (usually the one with a little graph showing activity).
- Click on the interface to select it.

Step 2.7: Start Capturing Traffic:

- Click the Start Button. Once you've selected the network interface, click the blue Start button (the shark fin icon) at the top of the screen to begin capturing traffic.

- Wireshark will now start capturing all the network traffic passing through your selected interface. You'll see the captured packets appear in real-time.



Step 4: Stimulating and Analyzing Normal Traffic:

Step 4.1: Stimulate Normal Traffic:

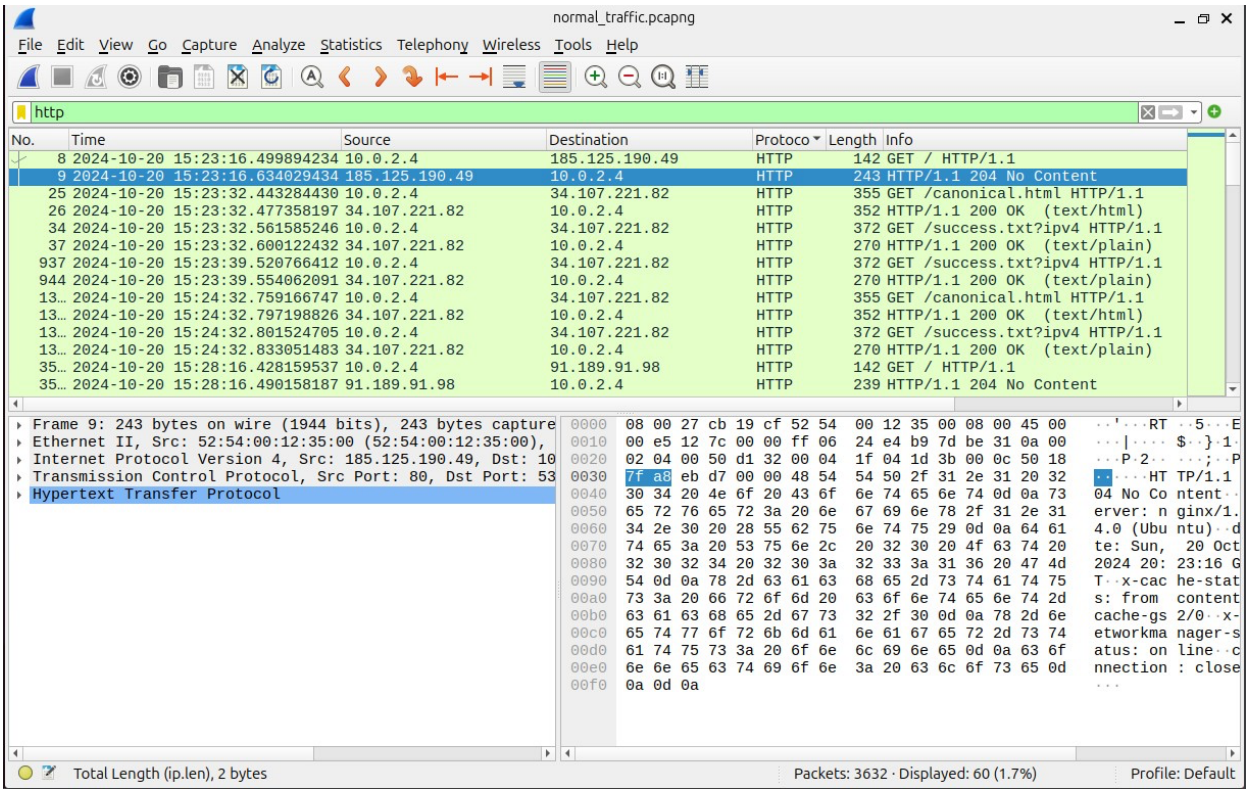
In this part, you will generate typical network traffic between your Ubuntu and Kali Linux VMs. This simulates regular user behavior like browsing the web, downloading files, or sending emails.

Step 1: Browsing Websites on Ubuntu

- First, we'll generate simple web traffic by browsing a few websites on your Ubuntu VM.
  1. **Open a Web Browser**:
     - In your **Ubuntu VM**, open the default web browser (usually Firefox).
     - **Visit Websites**:
       - Go to basic websites like:
         1. https://www.example.com

2. https://www.wikipedia.org

- When you do this, your Ubuntu VM sends **HTTP/HTTPS requests** over the network, which Wireshark can capture and analyze.
- Every time you load a webpage, it generates network requests and responses, which Wireshark will capture.



Step 2: Analyzing Normal Network Traffic:

- **DNS Query**: In Wireshark, you should see the **victim (Ubuntu)** sending a **DNS query** asking for the IP address of the websites you visited (e.g., www.example.com or www.wikipedia.org).
- **HTTP Request and Response**: You'll see the **GET request** sent from the Ubuntu VM to the website's server and the **200 OK response** from the server, delivering the webpage content.
- The **HTTP request** should show the **GET method** used to request a web page, along with details such as the URL and host (www.example.com).
- The **HTTP response** should show a **200 OK status**, indicating the successful retrieval of the webpage, along with content such as HTML, images, or files.

Step 3: Stop Traffic Capture

- **Stop browsing & capturing traffic** on **Ubuntu**.
- Save the captured traffic for analysis:
    - In **Wireshark**, go to **File → Save As** and save the capture as arpspoofattack_traffic.pcap.

Step 4.2: Stimulating and Analyzing ARP Spoofing Attack:

Step 1: Prepare Your Environment

- We will use **Kali Linux** to launch the attack and **Ubuntu** to capture the network traffic. Before proceeding, make sure:

    1. You have both **Ubuntu** and **Kali Linux** VMs running in **VirtualBox**.
    2. Both VMs are connected to the same network (for example, using the NAT network you set up earlier).

Step 2: Enable IP Forwarding on Kali Linux

- To successfully perform an ARP spoofing attack, we need to ensure that Kali Linux can forward traffic between the victim and the router.

    1. Open the terminal in **Kali Linux**.
    2. Enable IP forwarding by running:

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

Step 3: Identify IP Addresses

- You need to find the IP addresses of the **target machine** (Ubuntu VM) and the **default gateway** (router).

    1. On the **Ubuntu VM**, open a terminal and find its IP address:

```bash
ifconfig
```

Look for the IP under `eth0` or `enp0s3` (depending on your network interface).

2. On the **Kali Linux VM**, find the default gateway (router) IP address:

You should now have the following:

```bash
ip route | grep default
```

- **Ubuntu VM IP address** (target)
- **Default Gateway IP address** (router)

Step 4: Install arpspoof in Kali Linux

- On Kali Linux, you will use the arpspoof tool to conduct the ARP spoofing attack.

1. Install dsniff, which includes arpspoof:

```bash
sudo apt-get install dsniff
```

Step 5: Launch the ARP Spoofing Attack

Now, you will spoof both the Ubuntu VM and the router to make them believe that your Kali Linux machine is the other device.

1. **Spoof the target (Ubuntu VM) to believe you are the router**:

```bash
sudo arpspoof -i eth0 -t [Ubuntu_VM_IP] [Gateway_IP]
```

At this point, your Kali Linux VM is intercepting traffic between the Ubuntu VM and the router, allowing it to act as a middleman.

Step 6: Capture ARP Spoofing Traffic with Wireshark

1. **On the Ubuntu VM**, open **Wireshark**.

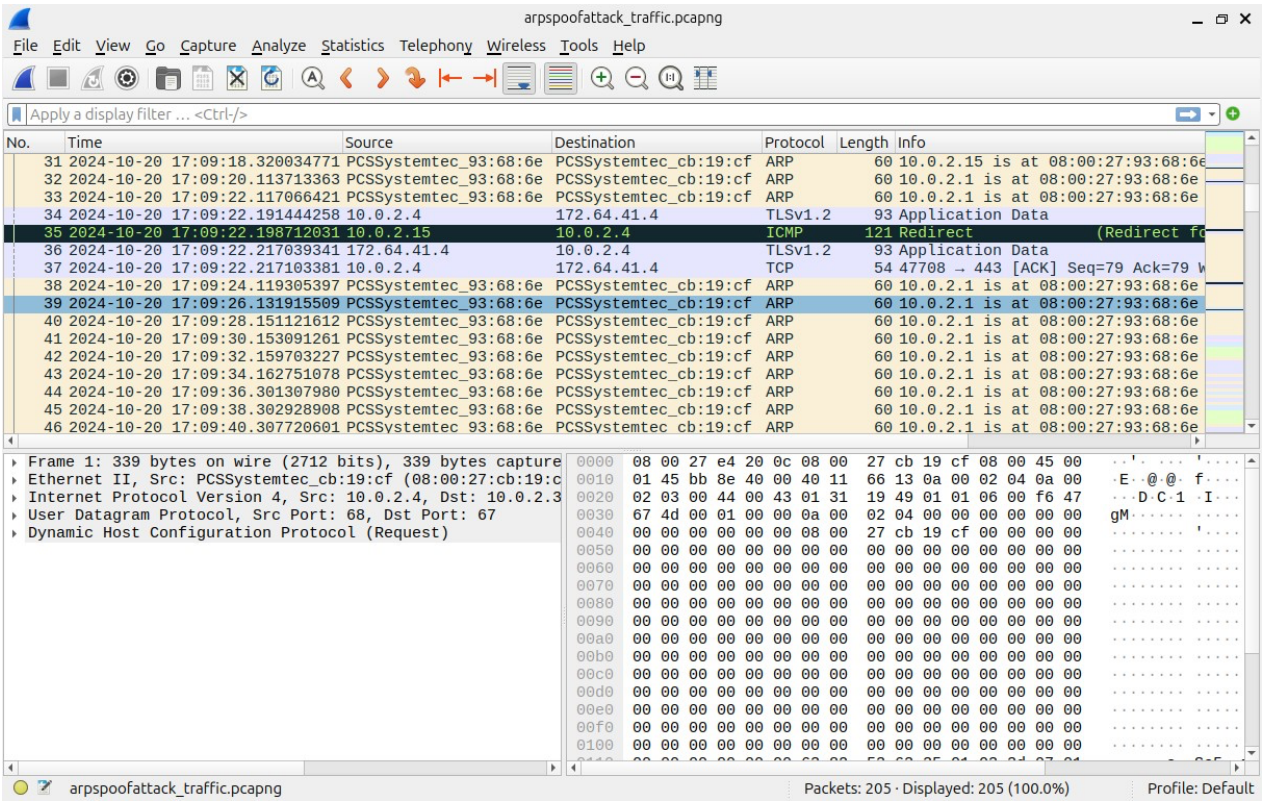
2. Start capturing traffic on the active network interface (likely eth0 or enp0s3).
3. Apply the following filter to see only **ARP packets**:

```bash
arp
```

4. While the attack is ongoing, Wireshark should capture **duplicate ARP replies**, where the same IP address (for either the router or Ubuntu) is being mapped to different MAC addresses — one being Kali Linux.

Step 7: Analyzing ARP Spoofing Attack:

- **ARP Request**: In Wireshark, you should see the **victim (Ubuntu)** sending an ARP request asking for the **MAC address** of the **router's IP** (192.168.1.1).



- **Fake ARP Reply**: You'll see a forged ARP reply from **Kali Linux**, telling the victim that the **router's IP** resolves to **Kali Linux's MAC address**.

Example ARP Reply:

- The ARP reply should show that the **MAC address** for the **router's IP (192.168.1.1)** is the **attacker's MAC address** (Kali Linux), rather than the real router MAC address.

Step 8: Stop the Attack

1. In the **Kali Linux terminal**, press Ctrl + C to stop the **arpspoof** command.
2. **Stop capturing traffic** in Wireshark on **Ubuntu**.

Step 9: Save the Capture

Save the captured traffic for analysis:

1. In **Wireshark**, go to **File → Save As** and save the capture as arpspoofattack_traffic.pcap.

Step 4.2: Stimulating and Analyzing DNS Spoofing Attack:

Step 1: Set Up the Environment

Ensure that both **Kali Linux** and **Ubuntu** are on the same network, as previously set up during the ARP spoofing attack.

- **Kali Linux interface**: eth0
- **Ubuntu interface**: enp0s3
- Both VMs should be connected using NAT or Bridged Adapter mode in VirtualBox.

Step 2: Enable IP Forwarding on Kali Linux

As with ARP spoofing, you need to enable **IP forwarding** on Kali Linux to allow it to act as a middleman and forward the victim's traffic:

1. Open the terminal in **Kali Linux** and run:

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

Step 3: Configure /etc/hosts on Kali Linux

In this step, we'll modify the /etc/hosts file on Kali Linux to direct the victim to a fake IP when they try to visit a specific domain (e.g., www.neverssl.com).

1. Open the /etc/hosts file on Kali Linux:

```bash
sudo nano /etc/hosts
```

2. Add the following line to the file:

```csharp
[Kali_IP] www.example.com
```

- Replace [Kali_IP] with the IP address of your **Kali Linux** machine.

3. Save and close the file by pressing Ctrl + X, then Y, and Enter.

This tells Kali Linux to send DNS requests for www.neverssl.com to the attacker's IP instead of the real server.

Step 4: Launch the DNS Spoofing Attack with dnsspoof

We will use the **dnsspoof** tool, which is part of the **dsniff** package, to intercept DNS requests from the victim and send forged responses.

1. Open a terminal in **Kali Linux** and run:

```bash
sudo dnsspoof -i eth0
```

This command will start **dnsspoof**, which listens for DNS requests on the network and responds with fake IP addresses based on the /etc/hosts file.

Step 5: Generate DNS Requests on Ubuntu (Victim Machine)

1. On **Ubuntu**, open a web browser.
2. In the address bar, type www.example.com and press Enter.

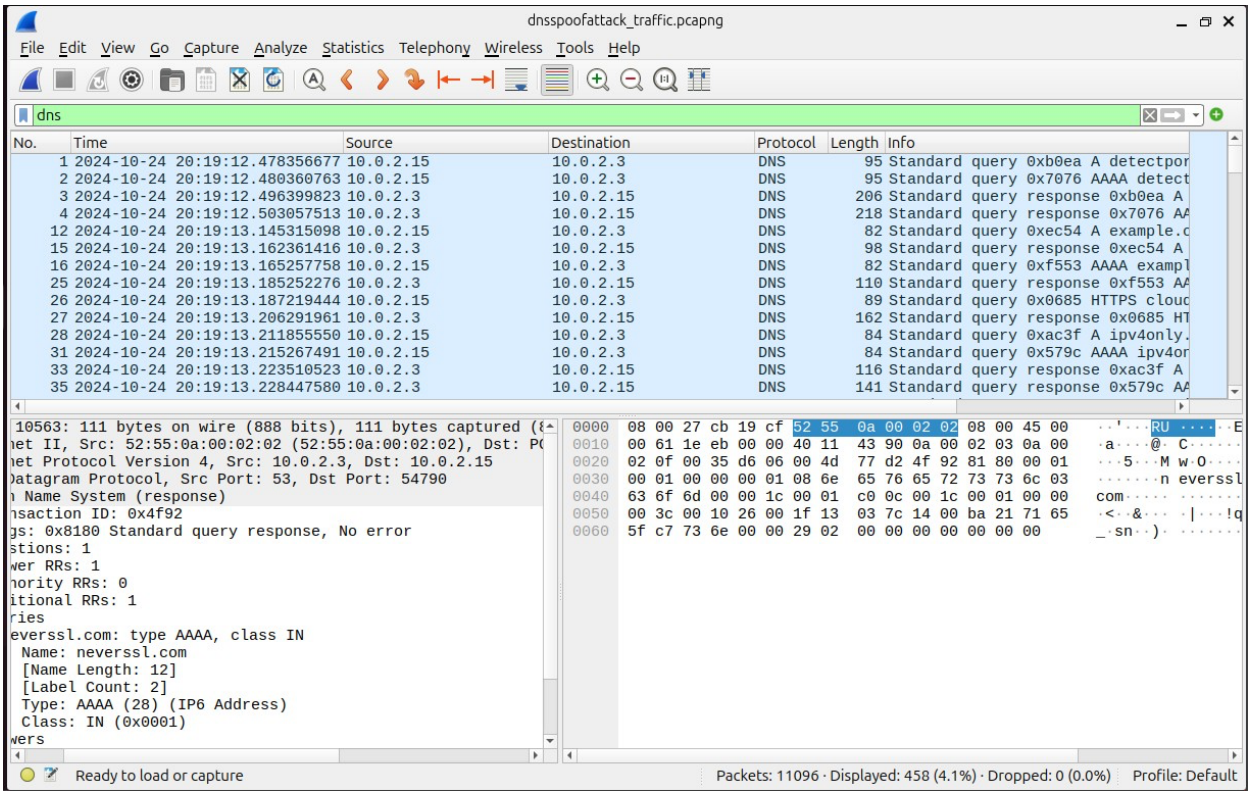
3. Normally, this would direct you to the legitimate neverssl.com website, but because of the DNS spoofing attack, you should be redirected to the IP address you specified in Kali's /etc/hosts file (the attacker's IP).

Step 6: Capture the Traffic in Wireshark on the Ubuntu:

1. **Start Wireshark** and begin capturing on the enp0s3 interface.
2. In the **filter bar**, type:

```bash
dns
```

This will filter for DNS traffic.



Step 7: Analyzing DNS Spoof Attack Traffic:

- **DNS Query**: In Wireshark, you should see the **victim (Ubuntu)** sending a DNS query asking for the IP address of www.neverssl.com.
- **Fake DNS Response**: You'll see a response from **Kali Linux**, telling the victim that www.neverssl.com resolves to the **attacker's IP address**.

Example DNS Response:

- The **DNS response** should show that the IP for www.neverssl.com is the **Kali Linux IP address**, as specified in the /etc/hosts file.

Step 8: Stop the Attack

1. In the **Kali Linux** terminal, press Ctrl + C to stop dnsspoof.
2. Stop capturing traffic in **Wireshark** on **Ubuntu**.

Step 9: Save the Capture

Save the captured traffic for analysis:

1. In **Wireshark**, go to **File → Save As** and save the capture as dnsspoofattack_traffic.pcap.

Step 6: Simulating and Analyzing SSL/TLS Traffic

Step 1: Generating SSL/TLS Traffic

Objective: Create encrypted traffic by visiting secure websites over HTTPS, which initiates an SSL/TLS handshake and session.

1. **Open a Web Browser on the Ubuntu VM**:
   - Launch Firefox (or any web browser installed on your Ubuntu VM).
2. **Visit HTTPS Websites**:
   - In the browser, navigate to secure websites, such as:
     - https://www.google.com
     - https://www.wikipedia.org
   - These sites use **HTTPS**, which means they will automatically initiate an **SSL/TLS handshake** to establish a secure, encrypted session.
   - Every time you load a webpage, it generates network requests and responses, which Wireshark will capture.

Step 2: Capturing SSL/TLS Traffic in Wireshark

Objective: Capture the SSL/TLS handshake and data exchange using Wireshark on the Ubuntu VM.

1. **Open Wireshark on Ubuntu**:
   - Start Wireshark on the **Ubuntu VM** and select the active network interface (e.g., enp0s3) to begin capturing traffic.

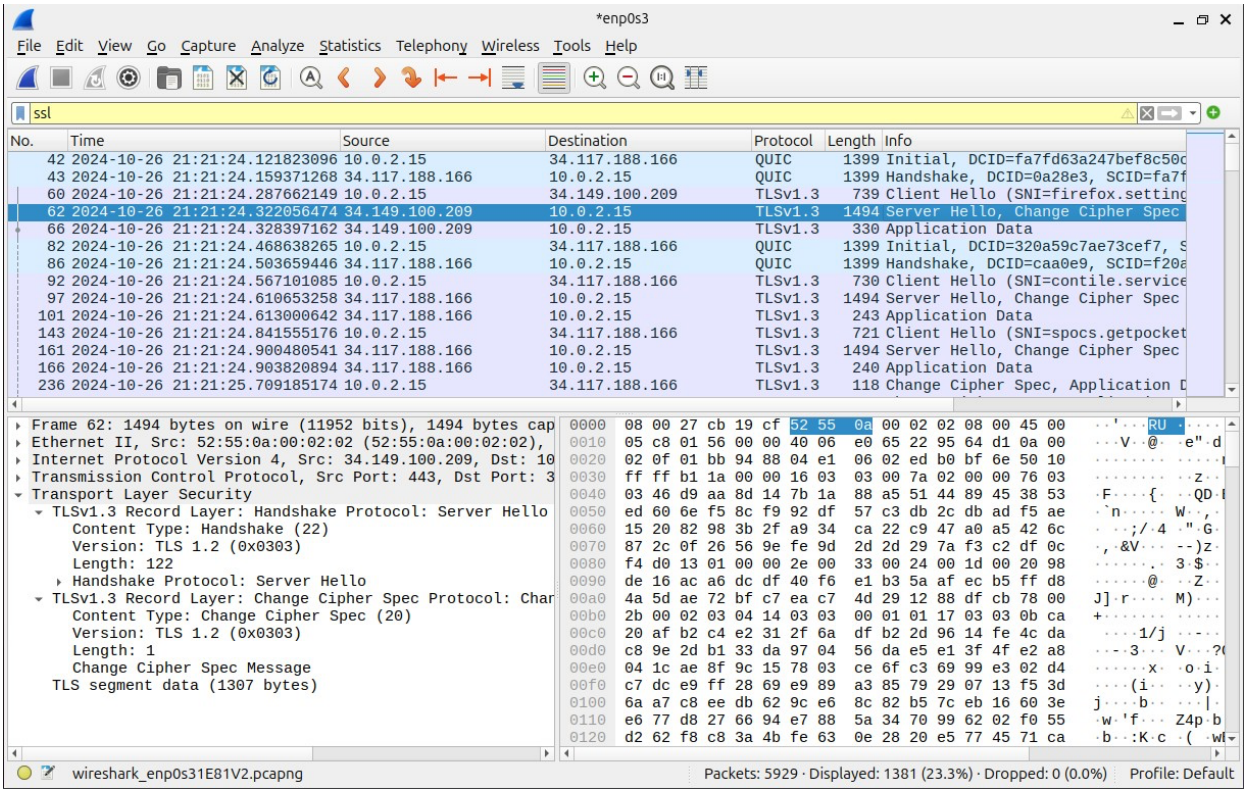
2. **Filter for SSL/TLS Traffic**:

    - In the Wireshark filter bar, enter:

```
plaintext                                                    Copy code

ssl or tls
```

    - This filter will display only **SSL/TLS packets**, making it easier to locate and analyze the handshake and encrypted traffic.
    - Continue capturing until you see a variety of SSL/TLS packets in Wireshark, including **Client Hello** and **Server Hello** messages.



Step 3: Analyzing SSL/TLS Traffic in Wireshark

Objective: Understand the SSL/TLS handshake, identify the protocol version and cipher suite used, and verify the secure connection details.

1. **Locate the SSL/TLS Handshake in Wireshark**:

    - In your Wireshark capture, scroll through the packets and look for **Client Hello** and **Server Hello** messages. These packets are usually

marked as **TLSv1.2** or **TLSv1.3** depending on the SSL/TLS version used by the server.

- **Tip**: You can filter for the handshake packets specifically by typing tls.handshake in the filter bar.

2. **Inspect the Client Hello Packet**:
    - Click on the **Client Hello** packet to view its details in the middle pane.
    - Expand the **Transport Layer Security** section to explore the following information:
        - **Version**: The SSL/TLS version requested by the client (e.g., TLS 1.2, TLS 1.3).
        - **Cipher Suites**: A list of encryption algorithms the client supports. This is important for establishing the encryption strength of the session.
        - **Session ID**: Used for identifying an SSL/TLS session, which can enable session reuse.
3. **Inspect the Server Hello Packet**:
    - Now, find and click on the **Server Hello** packet. The server responds with its selected SSL/TLS version and cipher suite.
    - In the details, expand **Transport Layer Security** and look for:
        - **Selected Version**: The SSL/TLS version the server has agreed to use.
        - **Selected Cipher Suite**: This shows the specific encryption algorithm that will be used for the session. Strong cipher suites (e.g., AES-GCM with 256-bit keys) indicate a secure connection.
4. **Review the Certificate Packet**:
    - Following the Server Hello, look for a packet labeled **Certificate**. This packet contains the server's SSL/TLS certificate, which is used to authenticate the server's identity.
    - In the **Certificate** packet:
        - **Issuer**: The entity that issued the certificate (e.g., Let's Encrypt, DigiCert).

- **Validity Period**: The date range during which the certificate is valid.
- **Public Key**: Part of the server's certificate, used for the encryption process.

Step 4: Stop the Capture

1. In the **Kali Linux** terminal, press Ctrl + C to stop dnsspoof.
2. Stop capturing traffic in **Wireshark** on **Ubuntu**.

Step 5: Save the Capture

Save the captured traffic for analysis:

5. In **Wireshark**, go to **File → Save As** and save the capture as ssl/tls_traffic.pcap.

Step 7: Conclusion

This project demonstrated the importance of network traffic analysis in identifying and mitigating security threats. By using Wireshark to capture and examine various types of network traffic, we established a baseline of normal behaviors and detected key indicators of network vulnerabilities through ARP and DNS spoofing simulations. The analysis of SSL/TLS traffic provided insight into secure communication protocols and reinforced the role of encryption in protecting data.

Through hands-on simulations, this project highlighted how network monitoring can reveal patterns associated with potential attacks and secure connections, emphasizing its role in enhancing network defenses. The skills gained in detecting anomalies and understanding traffic behaviors are essential for maintaining a secure network environment, showcasing the value of continuous traffic analysis in cybersecurity.