# National Network

## Network Sniffing Solution Guides

### Table of Contents

# LAB 1: Filtering Traffic in Wireshark (Follow Telnet Stream)

In this challenge, students will use Wireshark to monitor a network and identify a Telnet stream. By following the stream, they will discover the first flag and Telnet login credentials. Using these credentials, they can log in to the Telnet server to locate the second flag.

---

**Domain Name:**
*myrupholdings.com*

**Static Flags:**

- **Flag 1:** Found in the Telnet stream. – Video Demonstration Included
- **Flag 2:** Found on the Telnet server after login.

**Proposed Difficulty:**

- **Follow Telnet Stream:** 10 points
- **Telnet Login:** 10 points

---

**Prerequisites:**

- Knowledge of Wireshark and how to use it to capture network traffic.
- Understanding of traffic types, particularly unencrypted protocols like Telnet.

---

**Outcome:**

- Learn how to use Wireshark filters to isolate Telnet traffic.
- Gain experience following a network stream in Wireshark.
- Understand the risks of transmitting sensitive information over unencrypted protocols.

---

**Solution:**

## Flag 1: Follow Telnet Stream – Video Demonstration Included

1. **Open Wireshark:**
   - Start Wireshark and apply the following filter to isolate Telnet traffic:

     ```
     telnet
     ```

2. **Capture Network Traffic:**
   - Observe the live traffic and identify Telnet packets.
3. **Follow the Telnet Stream:**
   - Right-click on a Telnet packet and select:

     ```
     Follow > TCP Stream
     ```

   - The stream will reveal the first flag and Telnet login credentials.

---

## Flag 2: Telnet Login

1. **Login to the Telnet Server:**
   - Use the credentials discovered in the Telnet stream to log in:

     ```
     telnet myrupholdings.com
     ```

2. **Explore the Server:**
   - Once logged in, list the available files on the server using:

     ```
     ls
     ```

3. **Retrieve the Second Flag:**
   - Open and read the file containing the second flag using:

     ```
     cat <filename>
     ```

---

# LAB 2: MITM – Video Demonstration Included

This challenge involves intercepting traffic between a client and a server using a man-in-the-middle (MITM) attack to demonstrate the importance of using HTTPS everywhere. Participants will use Ettercap to downgrade HTTPS to HTTP and capture the flag from plaintext traffic.

---

**Domain Name:**
*secureshare.com*

**Proposed Difficulty:**
50 points

---

**Prerequisites:**

- Basic knowledge of networking, including ARP, HTTP, and TLS.
- Understanding of MITM attacks and how to execute them using tools like Ettercap.

---

**Outcome:**

- Learn how to perform a man-in-the-middle attack using ARP poisoning.
- Understand how to downgrade secure HTTPS traffic to insecure HTTP.
- Recognize the importance of using HTTPS everywhere.

---

**Solution:**

1. **Identify the Client and Server:**
    - Use Nmap and/or Wireshark to determine the IP addresses of the client and the server:
        - Example:

          ```
          Client IP: 192.168.0.10
          Server IP: 192.168.0.20
          ```

2. **Configure Ettercap:**
   - Open `/etc/ettercap/etter.conf` as a superuser:

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

   - Uncomment all lines starting with *redir_command_on* or *redir_command_off* under the Linux banner near the bottom of the file.
3. **Start Ettercap:**
   - Launch Ettercap and click the checkmark to begin.
4. **Scan for Hosts:**
   - Click the magnifying glass icon to scan the network for connected hosts.
5. **Set Targets:**
   - Open the host list menu (next to the scan button).
   - Set the client IP as **Target 1** and the server IP as **Target 2**.
6. **Enable ARP Poisoning:**
   - Click the **MITM menu** (globe icon) and select **ARP poisoning**.
   - Start the ARP poisoning process.
7. **Verify ARP Poisoning (Optional):**
   - Go to the plugin menu (under the vertical "..." icon).
   - Double-click the *chk_poison* plugin to confirm that ARP poisoning is working properly.
8. **Downgrade HTTPS to HTTP:**
   - Double-click the *sslstrip* plugin in the plugin menu to strip HTTPS traffic and force HTTP connections.
9. **Intercept HTTP Traffic:**
   - Open Wireshark and filter for HTTP POST requests to the server IP:

     ```
     ip.addr == 192.168.0.20 and http.request.method ==
     POST
     ```

   - Look for HTTP POST packets and examine their content.
10. **Retrieve the Flag:**
    - The flag will be visible in one of the intercepted HTTP POST packets.

---

# Extra LABs

## Extra LAB 1: meowTTP

Analyze the provided *.pcap* file to explore HTTP errors and locate an encoded flag in the *key.txt* file. Decode the flag using Base64 decoding to complete the challenge.

---

**Nextcloud Link:**
https://nextcloud.ntp-event.dk:8443/s/yGStb2ZdPiKxCPf

**Flag:**
*HKN{y0u_h4v3_b33n_m30wn3d}*

**Proposed Difficulty:**
10 points

---

**Prerequisites:**

- Basic understanding of Wireshark.
- Knowledge of common HTTP errors (e.g., 404, 405, 301, 502).
- Familiarity with Base64 encoding and decoding.

---

**Outcome:**

- Gain experience analyzing .pcap files with Wireshark.
- Understand and recognize various HTTP error codes.
- Learn simple Base64 decoding techniques.

---

**Solution**

1. **Open the *.pcap* File:**
   - Use Wireshark to open the provided file.

2. **Export HTTP Objects:**
   - Navigate to:

     ```
     File > Export Objects > HTTP
     ```

   - This will display a list of all HTTP objects within the *.pcap* file.
3. **Locate the Encoded Flag:**
   - Look for a file named *key.txt* in the exported objects.
   - Save the file locally and open it to find the Base64-encoded flag.
4. **Decode the Flag:**
   - Copy the encoded text from key.txt and use any online Base64 decoder (e.g., [https://www.base64decode.org](https://www.base64decode.org)) to decode it.
   - The decoded text will reveal the flag:
     - *HKN{y0u_h4v3_b33n_m30wn3d}*

---

## Extra LAB 2: RockingSSH

Use Hydra or other brute-forcing tools to identify the password for user *john* on the SSH server. This exercise demonstrates the importance of securing systems against brute-force attacks.

---

**Domain Name:**
*rockingssh.com*

**Proposed Difficulty:**
25 points

---

**Prerequisites:**

- Basic understanding of Hydra (a tool available in Kali Linux).
- Knowledge of brute-forcing techniques and how to apply them to a live system.

---

**Outcome:**

- Practical experience in using Hydra to brute-force credentials for SSH.
- Understanding how weak credentials can be exploited.

---

**Solution:**

1. **Identify the Protocol:**
   - Determine the protocol to target for brute-forcing: SSH.

2. **Analyze the Hint:**
   - Username is given as *john*.
   - Use the *rockyou.txt* password list to brute-force the password.
3. **Run Hydra to Brute-Force:**
   - Use Hydra to find the password for the SSH server:

     ```
     hydra -l john -P rockyou.txt ssh://localhost:2222
     ```

4. **Access the SSH Server:**
    - Once the correct credentials are identified, log in to the SSH server:

      ```
      ssh john@localhost -p 2222
      ```

5. **Retrieve the Flag:**
    - After logging in, navigate through the directories and find the flag file.

---

## Extra LAB 3: HTTP Everywhere

Analyze the provided .pcap file to decrypt localhost TLS traffic and retrieve the flag. The challenge highlights the importance of using TLS and provides hands-on experience with Wireshark and static trace analysis.

---

**Domain Name:**
*everywhere.hkn*

**Static Flag:**
*HKN{fj2-pS-KIo7Z}*

**Proposed Difficulty:**
20 points

---

**Prerequisites:**

- Basic understanding of networking and how to analyze *.pcap* files.
- Familiarity with Wireshark or willingness to learn from documentation.

---

**Outcome:**

- Understand the importance of using TLS to secure traffic over HTTP.
- Learn how to perform static trace analysis in Wireshark.
- Gain insights into how to identify and leverage key elements in a trace file (e.g., unusual protocols or localhost traffic).

---

**Solution:**

1. **Open the Trace File in Wireshark:**
    - Download the *.pcapng* file from [http://everywhere.hkn/trace.pcapng](http://everywhere.hkn/trace.pcapng) or replace the domain with localhost if running locally.
    - Open the file in Wireshark.

2. **Filter for Localhost Traffic:**
    - Use the following filter to focus on localhost traffic:

      ```
      ip.addr == 127.0.0.1
      ```

    - You will see mostly TLSv1.2 traffic. The goal is to decrypt this traffic to reveal the HTTP contents.
3. **Analyze the Protocol Hierarchy:**
    - Click **Statistics** > **Protocol Hierarchy** to review the distribution of traffic in the trace file.
    - Note that the traffic is dominated by HTTP, but there is some SMTP traffic that stands out.
4. **Inspect SMTP Traffic:**
    - Use the following filter to isolate SMTP traffic:

      ```
      smtp
      ```

    - Right-click on one of the SMTP frames, select **Follow** > **TCP Stream**.
5. **Extract the Private Key:**
    - Within the SMTP conversation, you'll find a message containing a certificate and private key. Save the private key to a text file on your local machine.
6. **Configure Wireshark with the Private Key:**
    - Open **Preferences** (CTRL + SHIFT + P).
    - Navigate to **Protocols** > **TLS** > **RSA keys** and add the saved private key file.
7. **Decrypt TLS Traffic:**
    - Reload the trace file by clicking the reload icon (circular arrow) or using CTRL + R.
    - Use the following filter to find decrypted traffic:

      ```
      ip.addr == 127.0.0.1 and tls
      ```

    - The TLS frames should now be replaced with HTTP traffic.
8. **Follow the HTTP Stream:**
    - Right-click on one of the HTTP frames, select **Follow** > **TLS Stream**.
    - The unencrypted HTTP traffic will contain the flag:
        - *HKN{fj2-pS-KIo7Z}*

---

## Extra LAB 4: monitor mode. ON

*You are basically Mr. Robot with your heavy Lenovo running Kali Linux and your trusty Alfa card. The monitor mode is set to ON, and you are blasting away. You know exactly what to do from here!*

This challenge involves analyzing a *.pcap* file captured in monitor mode, cracking a WPA2 WiFi password, and decrypting the network traffic to locate an HTTP request containing the flag. Participants will learn about WiFi handshakes, decryption, and HTTP traffic analysis.

---

**Flag:**
*HKN{wp42_1s_m4yb3_1ns3cur3}*

**Download Link:**
Capture File

**Proposed Difficulty:**
20 points

---

**Prerequisites:**

- Basic understanding of WiFi and how it works.
- Knowledge of using Wireshark for packet analysis.
- Familiarity with tools like aircrack-ng.

---

**Outcome:**

- Learn how to analyze monitor mode captures and identify WPA2 handshakes.
- Understand how to crack WiFi passwords and decrypt network traffic.
- Gain experience with HTTP request analysis to extract sensitive information.

---

**Solution:**

1. **Download and Extract the File:**
    - Download the capture file from the provided link and extract it to find a *.pcap* file.

2. **Analyze the Capture File in Wireshark:**
   - Open the *.pcap* file in Wireshark.
   - You will notice that the data is encrypted because it is a monitor mode capture.
3. **Identify the WPA2 Handshake:**
   - Use the following filter to locate EAPOL traffic, which indicates a WPA2 handshake:

     ```
     eapol
     ```

   - Look for a WPA2-protected network with a captured handshake.
4. **Crack the WPA2 Password:**
   - Use aircrack-ng to crack the WPA2 password using the rockyou.txt wordlist:

     ```
     aircrack-ng -w /usr/share/wordlists/rockyou.txt
     capture.pcap
     ```

   - Select the network with the handshake (e.g., fuckevilcorp) and crack the password.
   - The cracked password will be:

     ```
     teddybear
     ```

5. **Generate the PSK:**
   - Use the Wireshark tool to generate the WPA-PSK:
     - ESSID: fuckevilcorp
     - Password: teddybear
   - The generated PSK will be:

     ```
     12d302f07adb9ed74c0f67deac4535a4c82878702e10fbe52cb71f
     149126fc4e
     ```

6. **Set Up Wireshark for Decryption:**
    - Open **Preferences** in Wireshark (CTRL + SHIFT + P).
    - Navigate to **Protocols** > **IEEE 802.11**.
    - Configure the following settings:
        - **Reassemble Fragmented 802.11 diagrams:** ON
        - **Ignore vendor-specific HT elements:** OFF
        - **Call subdissector for retransmitted 802.11 frames:** ON
        - **Assume packets have FCS:** ON
        - **Validate checksum if possible:** ON
    - Add the WPA-PSK in **Decryption Keys**:

    ```
    wpa-psk:
    12d302f07adb9ed74c0f67deac4535a4c82878702e10fbe52cb71f
    149126fc4e
    ```

7. **Decrypt the Traffic:**
    - Reload the *.pcap* file in Wireshark.
    - The previously encrypted packets should now be decrypted.
8. **Search for HTTP Requests:**
    - Use the filter to locate HTTP traffic:

    ```
    http
    ```

    - Find the HTTP request containing the flag in the payload:
        - *HKNwp42_1s_m4yb3_1ns3cur3*
9. **Add the Flag Format:**
    - Wrap the extracted flag in {} to get:
        - *HKN{wp42_1s_m4yb3_1ns3cur3}*

---