CTF Report

Full Name: AVIREDDY RUSHIKESH
Program: HCS - Penetration Testing 1-Month Internship
Date: 10/03/2025

Category: Web 2.0 {Lock Web} – Points: 100

Description:

Web 2.0 challenges focus on modern web applications, often involving JavaScript, APIs, authentication mechanisms, and client-side security flaws.

Challenge Overview:

The challenge presents a locked web interface requiring a numeric PIN for access. By analyzing client-side code and server responses, the goal is to bypass or extract the correct PIN to proceed.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - **Use Burp Suite Proxy to intercept requests and analyze website behavior.**
    - **Check for hidden parameters, input fields, and API endpoints.**
2. **Input Validation Testing:**
    - **Intercept requests and modify inputs in Repeater to test for vulnerabilities.**
3. **Brute Force & Enumeration:**
    - **Use Intruder to brute-force numeric PINs, directories, or credentials.**

- **Analyze response differences (status, length, or content changes).**

4. **Exploitation:**
    - **Modify successful responses in Repeater to manipulate the application's behavior.**
    - **Check session tokens, cookies, or redirects for access control flaws.**
5. **Flag Retrieval:**
    - **Identify the correct request/response revealing the flag.**
    - **Submit the captured flag from the response.**

Flag: flag{ V13w_r0b0t5.txt_c4n_b3_u53ful!!!}

Category: Web 2.0 {The World} – Points: 150

Challenge Description:

You've landed on a webpage that displays "Hello World!". While it looks simple, hidden paths exist within the site. The objective is to explore and uncover the flag.

Challenge Overview:

This challenge involves discovering hidden files and directories within a seemingly simple webpage. By leveraging web enumeration techniques, analyzing responses, and decoding hidden messages, the goal is to retrieve the flag.

Steps for Finding the Flag:

1. Initial Reconnaissance:

- Opened the website and inspected its structure.
- Viewed the page source (Ctrl+U) and checked for hidden elements.
- Used **Burp Suite Proxy** to capture network traffic and analyze requests.

2. Directory Enumeration:

- Ran dirb on the target URL to uncover hidden directories.
- Found **privacy.txt** and **secret.txt** files.

3. Decoding Hints:

- privacy.txt contained a Base64-encoded message:
  Encoded:
  Sm91cm5leSBvbndhcmRzIHVudGlsIHlvdSBmaW5hbGx5IGFycml2ZSBhdCB5b3VyIGRlc3RpbmF0aW9u **Decoded:** Journey onwards until you finally arrive at your destination
    - This hinted at further exploration of hidden paths.
- secret.txt contained another Base64-encoded string:
  Encoded: RkxBR3tZMHVfaGF2M180eHBsMHJlRF90aDNfVzByTGQhfQ==
  Decoded Flag: FLAG{Y0u_hav3_4xpl0reD_th3_W0rLd!}

Flag Retrieved:

FLAG{Y0u_hav3_4xpl0reD_th3_W0rLd!}

Category: Reverse Engg {Lost in the Past} – Points:150

Description:

This challenge involves reverse engineering an App Inventor project. By analyzing .scm and .bky files, identifying encoded text, and decoding it using ROT47, the flag is extracted from the app's logic.

Challenge Overview:

The challenge provides an App Inventor project that needs to be reverse-engineered. By inspecting the `.scm` and `.bky` files, identifying encoded text within the app logic, and decoding it using ROT47, the hidden flag can be retrieved.

Steps for Finding the Flag

1. **Extract and Analyze Project Files**
    - Identify key files: Screen1.scm, Scrum.bky, and project.properties.
    - Look for any text components that might display a flag.
2. **Inspect App Logic (Scrum.bky)**
    - Identify conditions triggering the flag display (e.g., specific switch combinations).
    - Locate encoded text inside the logic blocks (found in ROT47 format).
3. **Decode ROT47 Text**
    - Extract the encoded string from Scrum.bky:
    - 7=28LE__0>F490C6GbCD`?8N
    - Use a ROT47 decoder to decrypt it.
4. **Validate and Submit the Flag**
    - The decoded text is the flag. **FLAG: flag{t00_much_rev3rs1ng}**

Category: Reverse Engg {Decrypt Quest} -- Points: 200

Description:

Samarth's imaginary friend, Arjun, hands him a text file containing encrypted secret data. Arjun claims it's impossible to decode but promises a $1,000,000 reward if Samarth extracts the hidden information. However, the file is filled with irrelevant data to mislead him. The challenge requires analyzing the file, identifying useful clues, and decrypting the hidden message to obtain the flag.

Challenge Overview:

The challenge involves extracting a hidden flag from a text file filled with misleading data. By analyzing the file in CyberChef, a Java code snippet is revealed. This code contains a Google Drive link leading to an encrypted file. A hint suggests using Unix Epoch Time for decryption. Modifying the Java code to detect the 1970 timestamp helps retrieve the flag successfully.

Steps for Finding the Flag:

1. Download the file and analyze it using CyberChef to extract hidden data.

2. Identify the Java code in the extracted data, which contains a Google Drive link

3. Access the Drive link and decrypt its content using the hint: Unix Epoch Time.

4. Modify the Java code to simplify it and focus on finding the 1970 value (Epoch Time).

5. Run the corrected code, stopping at the Unix Epoch timestamp, which reveals the flag.

Flag: flag{hjwilj111970djs}