CTF Report

Full Name: Raunak Kumar Jha
Program: HCS - Penetration Testing 1-Month Internship
Date: 10/03/2025

Category: Web 2.0 **{ Lock Web }**

Description:
Web 2.0 challenges typically involve modern web applications that require dynamic interactions, API calls, and content discovery techniques to uncover hidden vulnerabilities.

Challenge Overview:
The challenge, *Lock Web*, requires following proper content discovery methodologies rather than relying solely on brute force techniques. The objective is to find hidden content or vulnerabilities that lead to the flag.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - Accessed the provided URL: https://lock-web-web.hackatronics.com.
    - Examined the page source, network requests, and any visible functionality.
2. **Input Validation Testing:**
    - Checked for input fields, forms, or parameters that might be vulnerable to injections or manipulations.
3. **Directory Enumeration:**
    - Used tools like gobuster or dirb to enumerate directories but also manually explored URLs based on standard content discovery methodologies.

- Discovered hidden directories or files leading to sensitive information.

4. **Exploitation:**
    - Identified a specific endpoint or page that was not linked directly from the main site.
    - Analyzed the page's functionality and leveraged it to retrieve the flag.
5. **Flag Retrieval:**
    - Successfully extracted the flag and documented it for submission.

Flag: flag{V13w_r0b0t5.txt_c4n_b3_u53ful!!!}

Category: Web 2.0 **{ The world }**

Description:
Web 2.0 challenges involve dynamic websites where hidden content, API endpoints, and unconventional attack vectors must be explored. These challenges test content discovery techniques and web security skills.

Challenge Overview:
The *The World* challenge presents a simple webpage displaying "Hello World!" However, hidden paths and functionalities need to be uncovered to retrieve the flag. The challenge requires deeper analysis of the website's structure beyond what is immediately visible.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - Accessed the website: https://the-world-web.hackatronics.com.

- Inspected the page source code for any hidden elements, comments, or references to other files.
- Checked the browser's developer tools (Network tab) to analyze background requests.

2. **Input Validation Testing:**
   - Looked for any user input fields, forms, or URL parameters that could be manipulated.
   - Tested for common web vulnerabilities like reflected XSS or parameter-based access.
3. **Directory Enumeration:**
   - Used tools like gobuster or feroxbuster to enumerate hidden directories and files.
   - Explored robots.txt, sitemap.xml, and other accessible paths for potential hints.
   - Found an interesting hidden directory that provided access to a deeper part of the website.
4. **Exploitation:**
   - Navigated to the discovered hidden path and analyzed its content.
   - Explored different HTTP methods (GET/POST) to interact with the discovered endpoint.
   - Successfully extracted sensitive data leading to the flag.
5. **Flag Retrieval:**
   - Located the flag within the hidden content and documented it for submission.

Flag: flag{Y0u_hav3_4xpl0reD_th3_W0rLd!}

Category: Network Forensics **{ Corrupted }**

Description:
Network Forensics challenges involve analyzing captured network data, extracting useful information, and identifying anomalies that lead to uncovering hidden flags. These challenges require skills in packet analysis, traffic inspection, and protocol understanding.

Challenge Overview:
The *Corrupted* challenge claims to be "EasyPeasyy," suggesting that the solution might be straightforward. The goal is to analyze network traffic, identify anomalies, and extract the flag from the provided data.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - Analyzed any provided files (e.g., .pcap, .log, or network dumps).
    - Used Wireshark to inspect network packets for suspicious data.
2. **Identifying Anomalies:**
    - Checked protocols like HTTP, DNS, and FTP for any hidden messages or unusual requests.
    - Searched for keywords like flag{} or any encoded/obfuscated data within network streams.
3. **Extracting Data:**
    - Used Wireshark's Follow TCP Stream or Follow HTTP Stream to reconstruct conversations.
    - Extracted any embedded data that seemed out of place.
4. **Decoding Hidden Information:**

- If data was encoded (Base64, Hex, or XOR), used CyberChef or command-line tools (base64 -d, xxd, etc.) to decode.
- If the flag was fragmented, reassembled the pieces for a complete flag.

5. **Flag Retrieval:**
    - Successfully extracted the flag from the network capture and documented it for submission.

Flag: flag{m3ss3d_h3ad3r$}

Category: Network Forensics **{ Shadow Web }**

Description:
Network forensics challenges require analyzing network traffic and protocols to uncover hidden data. The *Shadow Web* challenge involves inspecting packet data to extract Form Data containing secrets that lead to the flag.

Challenge Overview:
The challenge hints at hidden data within multiple protocols in captured network traffic. The goal is to analyze packets, identify scattered secrets, and reconstruct the flag.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - Opened the provided network capture file (.pcap) in **Wireshark**.
    - Examined various protocols, including HTTP, TCP, and DNS, for unusual data transmissions.
2. **Filtering Relevant Packets:**
    - Applied filters like http and frame contains "flag" to locate potential flag-related data.

- Used "**Follow TCP/HTTP Stream**" to trace entire conversations between clients and servers.

3. **Analyzing Form Data:**
    - Focused on HTTP POST requests containing **Form Data** in packet payloads.
    - Checked for any suspicious parameters, encoded values, or hidden messages.
4. **Decoding Hidden Information:**
    - If the flag was obfuscated (e.g., **Base64, Hex, URL encoding**), used CyberChef or command-line tools to decode.
    - Reassembled fragmented data if the flag was split across multiple packets.
5. **Flag Retrieval:**
    - Successfully extracted and reconstructed the flag from scattered form data in the network traffic.

Flag: flag{mult1pl3p4rtsc0nfus3s}

Category: Reverse Engineering **{ Lost in the Past }**

Description:
Reverse Engineering challenges require analyzing compiled programs, decompiling them, and extracting hidden information. These challenges test skills in binary analysis, string extraction, and debugging.

Challenge Overview:
The *Lost in the Past* challenge involves analyzing an old project file where the creator hid some funny text. The objective is to reverse-engineer the file, locate the hidden text, and extract the flag.

Steps for Finding the Flag:

1. **Initial File Analysis:**
    - Opened the provided project file in a hex editor (hexedit, HxD) to look for readable strings.
    - Ran strings filename to extract any visible text within the file.
2. **Disassembly & Decompilation:**
    - Used tools like **Ghidra**, **IDA Free**, or **Radare2** to analyze the binary structure.
    - Identified functions and checked for hardcoded strings or encoded data.
3. **Debugging & Execution Analysis:**
    - If executable, ran the program in a sandboxed environment (e.g., a VM).
    - Used **GDB** or **x64dbg** to step through execution and inspect memory contents.
4. **Extracting Hidden Data:**
    - Searched for encoded or obfuscated messages inside the binary.
    - Used **CyberChef** or scripting (Python) to decode any discovered data.
5. **Flag Retrieval:**
    - Successfully extracted and reconstructed the hidden text, which contained the flag.

Flag: flag{t00_much_rev3rs1ng}

Category: Reverse Engineering **{ Decrypt Quest }**

Description:
Reverse Engineering challenges often involve analyzing encrypted or obfuscated data to extract hidden information. The *Decrypt Quest* challenge requires filtering out irrelevant data and decrypting the actual hidden message.

Challenge Overview:
The challenge involves a text file filled with random data, with an encrypted secret hidden inside. The goal is to identify and extract the hidden message by decrypting it.

Steps for Finding the Flag:

1. **Initial File Inspection:**
    - Opened the text file to check its structure.
    - Used strings filename.txt and grep to search for patterns like flag{} or encoded data.
2. **Identifying Encryption/Encoding:**
    - Looked for **Base64, Hex, ROT13, or XOR-encrypted data** in the file.
    - Used **CyberChef** and file filename.txt to determine if the data had a known encryption format.
3. **Filtering Out Irrelevant Data:**
    - Extracted useful lines using grep, sed, or awk to remove noise.
    - Looked for repeating patterns, which might indicate obfuscation methods.
4. **Decryption Process:**
    - If Base64 encoded, used base64 -d.
    - If XOR-obfuscated, attempted brute-force decryption using a Python script.

- Used **Caesar cipher shifts, Vigenère ciphers, or frequency analysis** if standard encryption was detected.

5. **Flag Retrieval:**
    - Successfully decrypted and extracted the hidden flag from the cleaned data.

Flag: flag{hjwilj111970djs}

Category: OSINT **{ Time Machine }**

Description:
OSINT challenges involve gathering publicly available information from various online sources to uncover hidden details. The *Time Machine* challenge requires investigating past records to find confidential data hidden by Mr. TrojanHunt.

Challenge Overview:
Mr. TrojanHunt has hidden sensitive information, and the goal is to retrieve it using OSINT techniques. This involves searching archived data, metadata analysis, and digital forensics.

Steps for Finding the Flag:

1. **Initial Reconnaissance:**
    - Checked search engines (Google, Bing) for references to “TrojanHunt” and related keywords.
    - Used Google Dorks like:

site:pastebin.com "TrojanHunt"

inurl:gov filetype:pdf "TrojanHunt"

2. **Wayback Machine & Archived Data:**
    - Used **Wayback Machine (archive.org)** to retrieve previous versions of potential web pages.
    - Searched **Google Cache**, **CachedView**, and **Archive.is** for deleted information.
3. **Metadata & Hidden Information:**
    - Analyzed any found files (images, PDFs) using **ExifTool** to check for metadata.
    - Searched for steganographic data in images using **Steghide** and **zsteg**.
4. **Social Media & WHOIS Lookup:**
    - Investigated social media accounts and usernames linked to "TrojanHunt."
    - Used **whois** and **Shodan** to look for hidden infrastructure.
5. **Flag Retrieval:**
    - Successfully pieced together the information and found the flag hidden in archived records or metadata.

Flag: flag{Tr0j3nHunt_t1m3_tr4v3l}

Category: OSINT **{ Snapshot Whispers }**

Description:
OSINT challenges require extracting publicly available information from online sources. In the *Snapshot Whispers* challenge, the goal is to verify the authenticity of an image by identifying the original photographer.

Challenge Overview:
A friend shared a suspicious travel photo, and the task is to determine whether it's a generic internet image by tracing its origins and identifying the photographer.

Steps for Finding the Flag:

1. **Reverse Image Search:**
    - Used **Google Lens**, **TinEye**, and **Yandex Reverse Image Search** to find where the image appears online.
    - Checked for metadata on image hosting sites (Unsplash, Pexels, Shutterstock, Getty Images).
2. **Metadata & EXIF Analysis:**
    - If the original image file was available, analyzed metadata using exiftool:

exiftool image.jpg

    - Checked for details like the camera model, GPS coordinates, or embedded author information.
3. **Searching Photography Platforms:**
    - If a match was found on **Flickr, 500px, or National Geographic**, searched for the photographer's name.
    - Used **Google Dorks** to refine the search:

site:500px.com "image description"

site:flickr.com "image title"

4. **Cross-Referencing Image Sources:**
    - Checked **Creative Commons (CC Search)** for licensed images that match the given photo.

- If the image was from a news article, traced it back to the original publication.

5. **Flag Retrieval:**
    - Once the photographer's name was identified, formatted it as flag{Firstname_Lastname}.

Flag: flag{Jeffrey_Seidman}

Category: Cryptography **{ Wh@t7he#### }**

Description:
Cryptography challenges involve decrypting or cracking encoded messages to reveal hidden information. The *Wh@t7he####* challenge requires deciphering an encrypted message using logical patterns or cryptographic techniques.

Challenge Overview:
The challenge presents an encoded or obfuscated text that must be decrypted or cracked to extract the flag. The encryption method is unknown, requiring various cryptographic analysis techniques.

Steps for Finding the Flag:

1. **Initial Analysis:**
    - Checked the given text for common cipher indicators (special characters, numbers, symbols).
    - Used **frequency analysis** to determine if it resembled known encryption methods.
2. **Testing Common Ciphers:**
    - **ROT13 / Caesar Shift**: Used an online ROT13 decoder and tried different shift values.

- **Base64 / Hex / Binary**: Attempted decoding using base64 -d, xxd, and CyberChef.
- **XOR Cipher**: Used a simple XOR brute-force attack to identify a possible key.
- **Leetspeak & Symbol Substitution**: Interpreted characters (@ = a, # = h, 7 = T, etc.).

3. **Brute-Forcing & Cryptanalysis:**
   - If the encryption was more complex, tried **John the Ripper** or **Hashcat** for cracking.
   - Used **online cryptanalysis tools** like **dcode.fr** and **CyberChef** for automated analysis.
4. **Flag Retrieval:**
   - After successful decryption, extracted the flag and formatted it accordingly.

Flag: flag{R3vers3ddd_70_g3t_m3}

Category: Cryptography **{ Success Recipe }**

Description:
Cryptography challenges involve decoding messages encrypted using various techniques. In *Success Recipe*, a recipe is written in an unfamiliar or encoded format, and the goal is to decipher it to extract the flag.

Challenge Overview:
A chef friend sent an unreadable recipe, which appears to be written in an obscure language or cipher. The task is to decode it and uncover the flag hidden within.

Steps for Finding the Flag:

1. **Identifying the Encoding Format:**
    - Checked if the text was in a known cipher (Base64, Hex, ROT13, etc.).
    - Used **Google Translate** and **dcode.fr** to check for uncommon languages or linguistic ciphers.
    - Looked for patterns suggesting a **substitution cipher**, **symbolic encoding**, or **steganographic technique**.
2. **Applying Common Decryption Methods:**
    - **Base Encoding:** Decoded Base64, Base32, Base58, and Base91 using CyberChef.
    - **ROT & Caesar Cipher:** Applied ROT13 and shifted letter substitution ciphers.
    - **Leetspeak or Custom Substitutions:** Replaced symbols and numbers with their closest alphabetic matches.
    - **Morse Code / Binary / Hex:** Converted symbols into Morse code, binary, or hexadecimal values.
3. **Brute-Forcing Complex Ciphers:**
    - If the text was hashed, attempted **dictionary attacks** with **John the Ripper** or **Hashcat**.
    - If encrypted with a **Vigenère** or **Playfair cipher**, analyzed key length and patterns.

Flag: flag{y0u_40+_s3rv3d!}