**CTF Challenge Report - StoutCTF**

**1. Cryptography Challenges**

* **Base64 Decode:**
  + **Description:** A simple Base64 decoding challenge.
  + **Tools Used:** dcode.fr for decoding.
  + **Steps Taken:**
    1. Used an online Base64 decoder at dcode.fr.
    2. Copied the Base64 encoded string and decoded it.
    3. Extracted the flag from the decrypted output.
* **Vigenère Cipher:**
  + **Description:** A Vigenère cipher encrypted message.
  + **Tools Used:** [dcode.fr Vigenère Cipher tool](https://www.dcode.fr/vigenere-cipher).
  + **Steps Taken:**
    1. Entered the encrypted message into the Vigenère cipher tool.
    2. Decrypted the message using the provided key.
    3. Extracted the flag from the decrypted output.
* **Whitespace Language Encoding:**
  + **Description:** The challenge asked for decoding whitespace encoded text.
  + **Tools Used:** [Whitespace Language Decoder](https://www.dcode.fr/whitespace-language).
  + **Steps Taken:**
    1. Copied the encoded text and pasted it into the whitespace decoder.
    2. Gzipped the decoded text and saved it in a text file.
    3. Used a Python script to replace 0 with space and 1 with tab, then decoded the binary to ASCII using an online binary to ASCII converter.(Attach the code inside zip)
* **Custom Cipher:**
  + **Description:** A custom cipher encryption challenge.
  + **Tools Used:** Custom Python script.(Attached with the Zip file)
  + **Steps Taken:**
    1. Analyzed the custom cipher.
    2. Wrote a Python script to decode the message based on the cipher logic.
    3. Retrieved the flag.
* **Huffman Encoding:**
  + **Description:** Huffman encoding decoding.
  + **Tools Used:** Custom Python script. (Attached with the Zip file)
  + **Steps Taken:**
    1. Wrote a Python script to decode the Huffman encoded message.
    2. Extracted the flag from the decoded text.

**2. Forensic Challenges**

* **Normal Image:**
  + **Description:** A steganography challenge embedded within an image.
  + **Tools Used:** Custom code for extracting hidden data.
  + **Steps Taken:**
    1. Analyzed the image using custom Python scripts.( (Attached with the Zip file))
    2. Retrieved hidden data from the image and discovered the flag.
* **RockYou:**
  + **Description:** A password cracking challenge using the RockYou wordlist.
  + **Tools Used:** fcrackzip, Kali RockYou wordlist.
  + **Steps Taken:**
    1. Used the command fcrackzip -v -u -D -p /usr/share/wordlists/rockyou.txt RockYou.zip.
    2. Cracked the zip file password and retrieved the flag.
* **The Orbs of Light:**
  + **Description:** A password hidden using a Caesar cipher.
  + **Tools Used:** Caesar cipher decryption tool.
  + **Steps Taken:**
    1. Identified the cipher as Caesar cipher with a shift of 3.
    2. Decrypted the password orb5ofL1ght using the Caesar cipher.
    3. Retrieved the flag.

**3. Scripting Challenges**

* **Hackers Keyboard:**
  + **Description:** Keystroke capture challenge from a USB device.
  + **Tools Used:** Wireshark, Tshark, custom Github Python script. (Attached with the Zip file)
  + **Steps Taken:**
    1. Captured the USB data by filtering Wireshark using usb.transfer\_type == URB\_INTERRUPT.
    2. Saved the capture as usb.pcapng and used Tshark to extract keystroke data.
    3. Used the custom Python script from [here](https://gist.github.com/MightyPork/6da26e382a7ad91b5496ee55fdc73db2) to parse the data and retrieve the flag.

**4. Miscellaneous Challenges**

* **Grass (Stereogram Solver):**
  + **Description:** Hidden message inside a stereogram.
  + **Tools Used:** [Stereogram Solver](https://piellardj.github.io/stereogram-solver/).
  + **Steps Taken:**
    1. Uploaded the stereogram to the solver.
    2. Retrieved the hidden message (flag).
* **Binary to ASCII:**
  + **Description:** Binary encoded message.
  + **Tools Used:** Binary to ASCII converter.
  + **Steps Taken:**
    1. Converted the binary data to ASCII using an online binary to ASCII converter.
    2. Retrieved the flag.
* **MakeAlanProud (Screenshot):**
  + **Description:** A hidden message in a screenshot.
  + **Tools Used:** Screenshot analysis tools.
  + **Steps Taken:**
    1. Analyzed the screenshot for hidden data or metadata.
    2. Retrieved the flag from the hidden data.
* **Dots & Dashes (Morse Code):**
  + **Description:** A Morse code challenge.
  + **Tools Used:** [Morse Code Translator](https://morsecodetranslator.com/).
  + **Steps Taken:**
    1. Converted the dots and dashes to text using the Morse code translator.
    2. Retrieved the flag.

**5. PHP File Upload Challenges**

* **File Upload Level 1:**
  + **Description:** Hiding a shell script inside a JPEG file.
  + **Tools Used:** Burp Suite, custom scripts.
  + **Steps Taken:**
    1. Used Burp Suite to intercept and upload the shell script inside a JPEG file.
    2. Used a command find / -name "flag\*" to search for the flag.
* **File Upload Level 4:**
  + **Description:** Uploading a PNG file with executable code.
  + **Tools Used:** .htaccess, PHP shell script.
  + **Steps Taken:**
    1. Modified .htaccess to allow PNG files to execute.
    2. Uploaded the file and retrieved the flag.
* **File Upload Level 5:**
  + **Description:** Uploading a PHP script to find a secret file.
  + **Tools Used:** .php script.
  + **Steps Taken:**
    1. Uploaded the PHP script and accessed the secret file secret.txt.
* **File Upload Level 6:**
  + **Description:** Uploading a file with a custom header to bypass security checks.
  + **Tools Used:** .htaccess, PHP script, echo command.
  + **Steps Taken:**
    1. Used echo -ne "\xFF\xD8\xFF" to add a JPEG header to the .htaccess file.
    2. Uploaded the .htaccess file with a .php extension and retrieved the flag.

**6. Web Challenges**

* **Nuclear Code (codes.php):**
  + **Description:** Code injection vulnerability.
  + **Steps Taken:**
    1. Exploited the vulnerability in codes.php to retrieve the flag.
* **PharmNet (SQL Injection):**
  + **Description:** SQL injection vulnerability.
  + **Steps Taken:**
    1. Performed an SQL injection to retrieve the flag.
* **Whois Levels 1-3:**
  + **Description:** Exploiting whois command for flag retrieval.
  + **Steps Taken:**
    1. Level 1: Executed ; ls -ls; cat flag.txt.
    2. Level 2: Executed || cat flag.txt.
    3. Level 3: Used dig option with echo $(cat flag.txt) to retrieve the flag.
* **The Bean (/admin):**
  + **Description:** Admin page access.
  + **Steps Taken:**
    1. Accessed the admin page /admin to retrieve the flag.