Saturday, June 16, 2025 - DOD Cyber Sentinel Skills Challenge (CTF write-up)

I competed in the DOD Cyber Sentinel Skills Challenge. It was a Capture the Flag-style competition sponsored by the US Department of Defense to raise awareness for the DOD and the need for cybersecurity professionals in the military and the US government. There were cash prizes for the top 10 winners, ranging from $5000 to $500, and 10 $300 randomly selected raffle winners.

This was my first time competing in a CTF competition. So I wasn’t expecting to do very well. I approached this as an educational opportunity to learn, gain experience, try new things, and get out of my comfort zone. I wanted to see how I would perform in a competition setting. I honestly haven't seriously competed in anything since 2016.

The week leading up to the competition, I was nervous as I am only in my 2nd year of my Cybersecurity degree at Unt. I haven’t had any formal experience with Kali Linux or its cybersec tools. The Saturday before the competition, I spent the day refactoring an old computer I found on a shelf in my home office and installing Kali Linux.

The competition rules included a long list of “suggested” tools to use in the competition to help us solve the challenges and capture the flag. I found that the Kali Linux OS included the majority of those tools. Having it installed before the competition really saved me so much time and confusion in the long run.

I spent the first couple of days that week on [picoctf.org](http://picoctf.org) in an attempt to practice capturing the flag and familiarizing myself with a few of the tools, such as Burpsuite and Wireshark, and using the tools through my command line. Although my expectation going into this was purely educational, I still wanted to be as competitive as possible on competition day, to put my best foot forward and set a marker for myself.

One competition day, I placed 502nd of 2155 competitors. Which puts me in the upper 25%. I received a total of 1175 points and captured a total of 14 flags. The competition began at 10 am CST and ended at 6 pm CST. I locked in and spent about 90% of the time actively working on challenges. Running only on a cup of coffee and a can of Vienna sausages, the entire day. I didn’t even notice I had barely eaten until the end of the competition, and the adrenaline came down.

I had so much fun in my first CTF. I really enjoyed working through the challenges, using new tools, pushing myself to think outside of the box, and building my confidence. I wish I had taken more pictures to put into my post-competition write-up. But I will do my best to provide as much context as I possibly can.

I hadn’t intended to write up this competition because I didn’t think I was going to do as well as I did, but I decided I would like to memorialize my first successful CTF and a couple of my favorite/most memorable flags captured.

1. Cafe Confidential
   1. Given: 2 Picture files
   2. Category: OSINT
   3. Level/Points: Very Easy - 75pts
   4. Goal: Locate the exact location where the pictures were taken.
   5. 1st photo:
      1. Used Google reverse image search
      2. Harrods Department Store
      3. Address: 87-135 Brompton Rd, London SW1X 7XL, United Kingdom
   6. 2nd photo:
      1. Used Google reverse image search
      2. Cake from a cafe
      3. Leicester Square, London, UK (very close to Harrods)
   7. I recognized the cake from an Instagram reel I saw recently that referred to it as the “Matilda Cake”. So I googled the “Matilda Cake London”, and discovered that it was made at Parker's Jumeirah Lowndes Hotel.
   8. The flag was to be C1{name of the cafe\_street location}, which meant that I had captured the flag C1{Parker’s\_Lowndes}
2. Listening Post
   1. Given: Audio files
   2. Category: Forensics
   3. Level/Points: Easy - 150 pts
   4. Challenge Prompt: “We've intercepted a radio broadcast being bounced off a satellite likely intended for the North Torbian cells located around the world. Do you think you can unravel what they are transmitting?”
   5. Goal: Analyze a .wav and .mp3 file to find hidden messages or decode a transmission using Kali Linux tools.
   6. Exiftool -
      1. The metadata line encoder: LAME3.100.�. tells us that the MP3 file was encoded using LAME version 3.100, which is a common open-source MP3 encoder. That weird symbol (�) suggests there's possibly:

* Corruption or hidden data in the metadata area
* Or the tag is misencoded, possibly hiding a clue or steganographic payload
  1. Audacity -
     1. Spectrogram mode: View → Spectrogram → Look for visual patterns (Morse code, QR codes, text in frequency bands)
        1. Showed white horizontal lines
     2. The audio was silent, but there was definitely something playing. White horizontal lines in the Audacity spectrogram almost always indicate sound at consistent frequencies, which is a huge clue in a CTF. Depending on the spacing and pattern of the lines, this could be:
        1. Morse Code
        2. DTMF tones
        3. Binary Tones or Encoded Characters
     3. I used <https://morsecode.world/international/decoder/audio-decoder-input.html> to be able to finally hear the audio, decode/inspect what I thought may be Morse code. However, it ended up not translating to actual words. But sounded like the audio of a dial tone from a phone from 90’s.
     4. I suspected it to be a DTMF so I:
        1. Installed multimon-ng in Kali Linux (CLI DTMF decoder):
           1. sudo apt install multimon-ng sox
        2. Converted my .wav audio file
           1. sox intercepted.mp3 -r 22050 -e signed -b 16 -c 1 dtmf.raw
        3. Decoded the DTMF
           1. multimon-ng -t raw -a DTMF dtmf.raw
           2. It showed each of the DTMF tones as either a 1 or a 0, indicating some kind of Binary code.
        4. Exported the output of the dtmf.raw file into a .txt file to extract the binary code.
           1. multimon-ng -t raw -a DTMF dtmf.raw > dtmf\_output.txt
           2. grep "DTMF:" dtmf\_output.txt | awk '{print $2}' | tr -d '\n' > binary.txt
           3. cat binary.txt
           4. Received: 0100001100110001011110110111001000110100011001000011000101101111010111110110101100110001011011000110110000110011011001000101111101110100011010000011001101011111011101000011000001110010011000100011000101100001010111110111001101110100001101000111001001111101
        5. Then used Chaptgpt to decode the Binary string for me and captured my flag.
           1. C1{r4d1o\_k1ll3d\_th3\_t0rb1a\_st4r}

1. Hoasted Toasted
   1. Given: <https://not-torbian.ethtrader-ai.com/>
   2. Category: Recon
   3. Level/Points: Easy - 150
   4. Challenge Prompt: “We have discovered what we believe is a North Torbian public website and have suspicions that there is a secret internal-only site hidden there as well. Figure out how to connect to the hidden site and find the flag! The site is at <https://not-torbian.ethtrader-ai.com/>”
   5. Goal: Access the internal-only site and find the flag
   6. I used dig to passively check the DNS records for the website we were given
      1. dig [not-torbian.ethtrader-ai.com](http://not-torbian.ethtrader-ai.com)
      2. Nothing out of the ordinary came back
   7. Tried an “internal” host header
      1. curl -k -H "Host: internal.not-torbian.ethtrader-ai.com" <https://not-torbian.ethtrader-ai.com/>
      2. I successfully uncovered the hidden internal site by using a custom Host header
      3. That confirms this site is vhost-gated — it only shows its internal version when accessed with the correct virtual host.
      4. The response:
         1. <!DOCTYPE html> <html lang="en"> <head> <meta charset="UTF-8"> <meta name="viewport" content="width=device-width, initial-scale=1.0"> <title>Official Statement</title> <style> body { font-family: 'Segoe UI', Tahoma, Geneva, Verdana, sans-serif; background-color: #f0f0f0; color: #333; display: flex; justify-content: center; align-items: center; min-height: 100vh; text-align: center; margin: 0; padding: 20px; box-sizing: border-box; } .container { background-color: #fff; padding: 40px; border-radius: 10px; box-shadow: 0 4px 8px rgba(0,0,0,0.1); max-width: 600px; } h1 { color: #d9534f; /\* Red color \*/ border-bottom: 2px solid #d9534f; padding-bottom: 10px; } p { font-size: 1.1em; line-height: 1.6; } strong { color: #337ab7; /\* Blue color \*/ } .emphasis { font-weight: bold; font-size: 1.2em; margin-top: 20px; display: block; color: #5cb85c; /\* Green color \*/ } .footer { margin-top: 30px; font-size: 0.9em; color: #777; } </style> </head> <body> <div class="container"> <h1>Official Communiqué</h1> <p>Greetings Citizen,</p> <p>It has come to the attention of the <strong>Ministry of Truth</strong> that certain rumors may be circulating regarding the nature of this web server.</p> <p>Let it be known, definitively and unequivocally:</p> <p class="emphasis">This is NOT an official website of the Glorious Democratic Republic of North Torbia!</p> <p>Any resemblance to North Torbian infrastructure, digital or otherwise, is purely coincidental and likely the result of subversive western propaganda.</p> <p>Please disregard any unusual digital signatures you might encounter. They are certainly not related to any internal governmental systems.</p> <p class="emphasis">Thank you for your attention to this matter. Your cooperation is greatly appreciated.</p> <div class="footer"> Issued by: The Committee for Public Clarity (Definitely Not North Torbian) </div> </div> </body> </html>
         2. The HTML looks polished, but it’s loaded with satirical political language and sarcasm — classic CTF red-flag bait.
            1. Let’s break down key clues from the message:

"Please disregard any unusual digital signatures you might encounter."

This line screams: "Look at the source code or SSL details."

* 1. I used torsocks to connect to the internal website to look for hidden text, base64 strings, or comments in the HTML source code
     1. torsocks curl -k -H "Host: internal.not-torbian.ethtrader-ai.com" https://not-torbian.ethtrader-ai.com/ | grep -i flag
     2. The output showed that:
        1. I successfully connected to the hidden site using the internal Host header.
        2. The full page downloaded — 2487 bytes were received — but:
        3. No line matched the word “flag” when piped through grep -i flag.
  2. I then spent about an hour attempting to use Gobuster to use brute force to find the DNS for internal or hidden directories using well-known words like /admin, /secret, /internal, etc.
     1. gobuster dir \

-u https://not-torbian.ethtrader-ai.com/ \

-H "Host: internal.not-torbian.ethtrader-ai.com" \

-k \

-w /usr/share/seclists/Discovery/Web-Content/common.txt

===============================================================

Gobuster v3.6

by OJ Reeves (@TheColonial) & Christian Mehlmauer (@firefart)

===============================================================

[+] Url: https://not-torbian.ethtrader-ai.com/

[+] Method: GET

[+] Threads: 10

[+] Wordlist: /usr/share/seclists/Discovery/Web-Content/common.txt

[+] Negative Status codes: 404

[+] User Agent: gobuster/3.6

[+] Timeout: 10s

===============================================================

Starting gobuster in directory enumeration mode

===============================================================

/index.html (Status: 200) [Size: 2487]

Progress: 4746 / 4747 (99.98%)

===============================================================

Finished

===============================================================

* + 1. The response told me that:
* Gobuster successfully connected to the internal site using the correct Host header
* /index.html is the only path that returned a valid response (HTTP 200)
* No other directories or files were discovered in the common wordlist (common.txt)
  1. After a bit of frustration and running around in circles, I consulted the #need-help Slack channel for a bit of direction using gobuster. Turns out, the only thing an admin mentioned about Gobuster was that it would not help in today’s competition. So I had to adjust my approach.
  2. Sometimes internal hostnames are listed in the SSL cert, so I used:
     1. echo | openssl s\_client -connect not-torbian.ethtrader-ai.com:443 | openssl x509 -text | grep DNS
        1. The response:
           1. $ echo | openssl s\_client -connect not-torbian.ethtrader-ai.com:443 | openssl x509 -noout -text | grep -i torbia Connecting to 34.86.60.228 depth=0 C=NT, ST=GloriousState, L=CapitalCity, O=Ministry of Truth, OU=Web Operations, CN=not-torbian.ethtrader-ai.com verify error:num=18:self-signed certificate verify return:1 depth=0 C=NT, ST=GloriousState, L=CapitalCity, O=Ministry of Truth, OU=Web Operations, CN=not-torbian.ethtrader-ai.com verify return:1 DONE Issuer: C=NT, ST=GloriousState, L=CapitalCity, O=Ministry of Truth, OU=Web Operations, CN=not-torbian.ethtrader-ai.com Subject: C=NT, ST=GloriousState, L=CapitalCity, O=Ministry of Truth, OU=Web Operations, CN=not-torbian.ethtrader-ai.com DNS:not-torbian.ethtrader-ai.com, DNS:definitelynotaflag.north.torbia
           2. That SSL certificate just gave away a **huge clue** — a new internal hostname to investigate.

DNS:[not-torbian.ethtrader-ai.com](http://not-torbian.ethtrader-ai.com) — The main domain I was already hitting.

DNS:definitelynotaflag.north.torbia — This was clearly suspicious. This is very likely a custom virtual host or a flag domain in disguise

* + 1. I then tried connecting to it via Host Header Spoofing
       1. curl -k -H "Host: definitelynotaflag.north.torbia" https://not-torbian.ethtrader-ai.com/
    2. And my flag was revealed.

I learned so much about cybersecurity through this experience. I am so grateful to have participated in this competition. It helped me gain the confidence to continue competing in these challenges and the practice/experience to know what to study for the future. I can’t wait for the next CTF and eventually a prosperous career in Cybersecurity.