GM#3 CHALLENGES

https://20241008.gm.neu-ctf.club/

1. (Reverse) Hop
    a. Description
        i. It’s time to train! My coach gave me this program to quiz me on the exact rules of soccer, but I think if I read the source code, I can just skip through it…
    b. Solution
        i. Open python code, read the answers to each question, and then answer correctly to get the flag (it’s a bunch of questions about official sections of the rulebook)
    c. Flag
        i. CTF{3A$Y_QU1Z}
    d. Dependency files
        i. Hop.py

2. (OSINT) Map the Court
    a. Description
        i. In the Ownlympics world, we need to know our court well! Before the big match, take a check and see the lowest port number open by the IP 75.75.75.75.
           Don’t forget to wrap it in CTF{} !
    b. Solution
        i. Use nmap: `nmap 75.75.75.75`
    c. Flag
        i. CTF{53}

3. (Crypto) 16 Laps
    a. Description
        i. I’ve been running so many laps… back and forth… back and forth…
    b. Solution
        i. Decrypt hexadecimal, it gradually reveals a message. keep copying the result of decryption and pasting it as the new ciphertext until you get the flag
    c. Flag
        i. CTF{U_Is_Hex_Master}
    d. Dependency Files
        i. 16laps.txt

1. (CRYPTO) Manager’s Number
    a. Description
        i. My coach gave me his manager’s number to call him once I finished my first week of Ownlympics training, but it doesn’t look quite right..?
        222 8 333 7777 2 777 33 7777 88 7 33 777 55 666 666 555
        Hint -- don't forget to add CTF{} !
    b. Solution
        i. Telephone cipher multi tap. Dcode.fr has a decoder
    c. Flag
        i. CTF{CTFSARESUPERKOOL}

2. (Reverse) Tools of the Trade
    a. Description
        i. I hope you like Ghidra! In Ownlympics, every self-respecting runner needs it! Practice with this executable and find the flag!
        NOTE - if you are on Mac M series, you should follow this guide to install Ghidra:
        https://lachy.io/articles/properly-installing-ghidra-on-an-m1-mac
    b. Solution
        i. Use Ghidra to find a hex-encoded flag
    c. Flag
        i. CTF{NSA_DRAGON}
    d. Dependency files
        i. Ghidra_me

3. (Reverse) Gym Class
    a. Description
        i. I want to get into the gym after hours to train some more, but I need to give it the flag to unlock the door!
    b. Solution
        i. Decompile the class file (can use an online one) and figure out that it xors the flag with a 2-byte key "\x42\x24", base64-encodes the result & compares it to a static string. base64-decode the static string and xor it with "\x42\x24" to get the flag.
        https://gchq.github.io/CyberChef/#recipe=From_Base64('A-Za-z0-9%2B/%3D',true,false)XOR(%7B'option':'Hex','string':'4224'%7D,'Standard',false)&input=QVhBRVh6WldkaFVzZTNaSUxuc21FRHRa&oeol=VT
    c. Flag
        i. CTF{tr41n_4ll_d4y}
    d. Dependencies
        i. Gym.class

4. (Reverse) Skip
    a. Description
        i. Well, I passed through the rules quiz, but my coach was suspicious since I got through it so fast. He gave me another quiz! I bet I can outsmart him on this one, though…
    b. Solution
        i. Open python code, see that ‘debug mode’ is enabled if you enter “DEBUG base64(DEBUG)” in the first question (where base64(DEBUG) is the base64 encoding of the word “DEBUG”)
    c. Flag
        i. CTF{3A$Y_QU1Z_G0T_3A$13R}
    d. Dependency files
        i. Skip.py

1. (Forensic) Big Interception
    a. Description
        i. Intercepted this communication between athletes when I was snooping to see if they were cheating.. What were they saying?
           Remember, the flag is CTF{...}
    b. Solution
        i. Follow tcp stream and see communication that describes an openssl command to run to decrypt file.des3. Look at tcp stream 2 to find a datastream starting with "Salted.....", right click and convert the data to its raw format and save as ‘file.des3’. Run the command noted in the communication, and the flag is in ‘flag.txt’
    c. Flag
        i. CTF{nc_73115_411_dd54ab67}
    d. Dependencies
        i. capture.flag.pcap

2. (Web) Jump
    a. Description
        i. There’s a landing page that’s come online for an upcoming competition! See if there are any secrets hidden around to help us get an advantage.
    b. Solution
        i. Visit the site, see .js loaded from a 3rd-party subdomain, navigate to the .js file, navigate to the subdomain root and see the directory listing. Use this to determine that cflag.txt is of interest, open in, B64 decode and see a cipher is likely in use. This is an affine ciphertext with parameters A=5, B=12. Bruteforce (many online resources). Done!
    c. Flag
        i. CTF{sn34ky_l1nksssss}
    d. Artifacts
        i. https://devin.dog/ctf/animation.html
        ii. Source (subdomain fs snapshot, animation.html)

3. (Reversing) Warm-Up Laps
    a. Description
        i. It’s a mini race to get you warmed up for the real deal! I heard that when you finish you get a flag and not a medal? Weird… anyways, they left this funny-looking string at the starting line: 923dd55c2161a96c73cc48957b85c48b.
           Make sure to wrap the flag you get in CTF{}
    b. Solution
        i. Reverse the binary with Ghidra and essentially reverse all the functions (the original is MD5 hash, add 0xf, XOR 0xa,

so you reverse by doing XOR 0xa, subtract 0xf and then cracking the hash)

   c. Flag
      i. CTF{falsestart}
   d. Dependencies
      i. warmup-race

Last Event: RELAY RACE! Four challenges connected to each other -> solving one leads to the next: https://20241008.gm.neu-ctf.club/relay

- Chally1: Right click, download image, run ‘exiftool’, and see a link in the user comment
- Chally2: View source code on html page, and in the comments there is javascript on how it obfuscated the string. If you decode / reverse it, the url should be “/bigger-secret”
- Chally3: The url downloads a file and gives this hint: “Like a car, you must perform a rev; remember the magic to be a good dev!”. Reverse the file with the ‘rev’ command and base64 decode: ‘rev flag-maybe.png | base64 -d > test.png’
- Chally4: image has the wrong magic bytez and is base64ed and reversed, fix them to have the beginning hex of a png file, open the file, and then it shows flag

FLAG : CTF{DUST3D}