GM#5: Big Study

(Pwn) Library

- Description
  - My library is full of great reads! Ensure that you study up, but only on the ones I allow you to!
    nc gm.neu-ctf.club {ip}
- Solution
  - You can input any arbitrary file and it will read it. Use the LS option to list files, and then read flag.txt
- Flag
  - CTF{DONT_READ_ME_PLZ}
- Dependencies
  - Pwnme.py

(Pwn) RNG

- Description
  - I made a random number guessing game, it shouldn't take you too long to guess the number if you just keep trying!
- Solution
  - Observe in the source code the seeding of rng (srand) with a constant
  - Reimplement the two lines seeding and generating the number, print it
  - Type to server
- Flag
  - CTF{s33d1ng_w1th_c0nst4nts_1s_b4d}
- Dependencies
  - Redacted-src.c

(Crypto) Hexed Text

- Description
  - This flag is encrypted so thoroughly it's like magic :((Can you help me recover the original flag?

    43 54 46 7b 79 33 72 5f 61 5f 77 31 7a 61 72 64 5f 68 61 72 72 79 7d
- Solution
  - Hex to Plaintext. Cyberchef is a great tool that can help decrypt this
- Flag
  - CTF{y3r_a_w1zard_harry}

(Crypto) Get the flag

- Description
  - Get the flag!

    oh mah ihpixh pu mah ftlmhk wmcmhw, lt pqkhq mp upqb c bpqh ihquhvm ftlpt, hwmcnxlwa rfwmlvh, ltwfqh kpbhwmlv mqctyflxlmg, iqpelkh upq mah vpbbpt khuhtvh, iqpbpmh mah zhthqcx ohxucqh, ctk whvfqh mah nxhwwltzw pu xlnhqmg mp pfqwhxehw ctk pfq ipwmhqlmg, kp pqkclt ctk hwmcnxlwa malw vptwmlmfmlpt upq mah ftlmhk wmcmhw pu cbhqlvc. vmu{apihgpfklktmkpmalwngactk}
- Solution
  - The letters seem to have patterns similar to English. This should hint to you that this is a substitution cipher
  - Use quipquip.com because we know plaintext ctf{ and insert it in the ‘clues’ box on the website
  - Or you can write a script to brute force
- Flag
  - ctf{hopeyoudidntdothisbyhand}

(Forensics) (Bin)walk em down

- Description
  - You heard me! Walk ‘em down!
- Solution
  - Run `binwalk -e walk-em.jpg`
  - Open flag.png to see the flag
- Flag
  - CTF{r3g1ster_4_BIG}
- Dependencies
  - walk-em.jpg

(Forensics) Grr… More Hackers

- Description
  - I captured this traffic of someone trying to hack me. What tool and what version of that tool are they using to do it?
    Note: Wrap the answer in CTF{} ex. CTF{toolName_versionNumber}
- Solution
  - Open pcap in Wireshark
  - Traffic looks like something of a brute force attempt to see files
  - Look at pretty much any Packet, in the HTTP Request the User Agent will say Nikto 2.1.6
- Flag
  - CTF{Nikto_2.1.6}

- Dependencies
    - hackers.pcap

(OSINT) Stalking the Prez

- Description
    - What is the CTF Club President’s email? Wrap the first part of it (before the @) in CTF{} for the flag!
- Solution
    - Find it through outlook email or any other way!
- Flag
    - CTF{defloor.e}

(OSINT) Hijacked

- Description
    - Someone changed my desktop background to this image??? Can you find out who took this picture? Maybe it’ll help me track down whoever did this…

      Note: Wrap the full name in CTF{}, ex. CTF{First_Middle_Last}. If there are any accents, ignore them, ex replace á with a.
- Solution
    - Reverse-image search the picture of the anglerfish to find any article that mentions who took the picture. TinEye is a useful tool that can help do this.
- Flag
    - CTF{David_Jara_Boguna}
- Dependencies
    - new_desktop_background.jpg

(RE) Your average binary

- Description
    - Here’s a binary that does nothing and has a flag in it somewhere. It’s static analysis time!
- Solution
    - Running “strings normalbinary” prints a string containing “the flag is”
    - Take the part right after that, base64 decode it
    - All done!
- Flag
    - CTF{just_4_n0rm4l_b1n4ry}
- Dependencies
    - normalbinary

(RE) decrypt0r

- Description
  - This program has the ciphertext, along with the code to decrypt it into plaintext! Just doesn’t seem to print it anywhere…
- Solution
  - Statically analyze in a decompiler and identify the XOR of byte payload with 0xD1. Use CyberChef or reimplement the code in a language of your choice with a print statement.
  - Alternatively, use GDB and print the stack after decryption completes.
- Flag
  - CTF{x0r_x0r_x0r_x0r_x0r}
- Dependencies
  - decrypt0r

(RE) zigzag

- Description:
  - Your job here is to reverse a Zig crackme to get the flag. No funny business here.
- Solution:
  - Reverse the binary using a decompiler (or even a debugger), identify the XOR decryption, decrypt the provided data in the binary.
- Flag:
  - CTF{n0w_1sn7_7ha7_fun}
- Dependencies
  - zigzag

(Web) Where are the Robots?

- Description
  - Where the robots at?
- Solution
  - Navigate to /robots.txt
- Flag
  - ctf{1_f0und_7h3_r0b0ts!}

(Web) Rockets

- Description
  - Oops! I forgot my rocket launch code! Can you get it for me?
- Solution
  - Go to /static/global.css to see admin:password is the login
  - There are hints to access the site from CTFC2THEMOON, and hints about User Agents

    - Refresh the page with CTFC2THEMOON set as the user Agent. Do so in the browser (inspect -> more tools -> network conditions in chrome) or use Burp Suite to intercept the request and change the user agent
- Flag
    - CTF{0nly_Up!}

(Misc) SimString

- Description
    - I heard that if you take two really similar strings, their hashes will be completely different! I’m not so sure about that, though … how many characters are different in the MD5 hash between the two strings “Hello, Tro” and “Hello, Dro” ? Wrap the number in CTF{} for the flag!
- Solution
    - MD5 hash for Hello, Tro is cc111ef22aad5b3ee4f2de18c4e9634e, and MD5 hash for Hello, Dro is 860144fc3d0fb1277942fabb5243f6c0
    - Only 3 characters are the same, so we subtract 3 from 32
- Flag
    - CTF{29}
- Dependencies
    - None

(Misc) Port Probe

- Description
    - a = SSH + WHOIS b = Kerberos + HTTP c = FTP + HTTPS d = Finger + SMTP
    - (a + b) * (c - d) = x
    - Format the flag as CTF{x} where x is the value of x
- Solution
    - All of the things provided in the prompt are protocols. They also have ports associated with them which can be found
- Flag
    - CTF{83647}