picoCTF Practice Challenges Report

Executive Summary

This report documents the solutions to 40 challenges from the picoCTF practice exercises, covering various cybersecurity topics, including cryptography, digital forensics, web exploitation and General skills.

Challenge Solutions

1. **Verify**

Description: Verify SHA-256 hash to ensure legitimacy.

Solution: Downloaded challenge files, connected via SSH, and verified hash using sha256sum command.

i. ls: to list the files in the directory (files, checksum.txt)
ii. sha256sum files/* > hashes.txt --- output the result of the sha256sum to hashes.txt
iii. diff hashes.txt checksum.txt --- to see the hash that matches the checksum hash
iv. grep -w 5848768e56185707f76c1d74f34f4e03fb0573ecc1ca7b11238007226654bcda hashes.txt --- to find the name of the file the hash belongs to

Flag: ./decrypt.sh files/8eee7195 --- gives the flag

2. **Scan Surprise**

Description: Retrieve flag from QR code image.

Solution: Scanned QR code using mobile phone to reveal flag.

3. **Binary Search**

Description: Find flag using binary search algorithm.

Solution: Guessed correct number (549) within 10 attempts using binary search.

4. **WebDecode**

Description: Decode coded text using web inspector.

Solution: Inspected webpage, decoded text using CyberChef.

5. **IntroToBurp**

Description: Bypass OTP verification using Burp Suite.

Solution: Intercepted OTP request, deleted incorrect OTP, and forwarded request.

6. **Inspect HTML**

Description: Find flag in HTML elements.

Solution: Inspected HTML page to discover flag.

7. **Bookmarklet**

Description: Use bookmarklet to reveal flag.

Solution: Copied bookmarklet, pasted in webpage inspect console.

8. **Includes**

Description: Find flag in script.js and style.css links.

Solution: Inspected webpage to discover flag.

9. **Information**

Description: Decode image metadata to reveal flag.

Solution: Viewed image metadata using jimpl.com, decoded license using Base64.

10. **Cookies**

Description: Manipulate cookies to reveal flag.

Solution: Inspected webpage, tried different cookie values to obtain flag.

11. **Super SSH**

Description: Access flag via SSH login.

Solution: Logged in using SSH, inputted password, and gained access.

12. Secret of the polygot

Description: The Network Operations Center (NOC) of your local institution picked up a suspicious file, they're getting conflicting information on what type of file it is. They've brought you in as an external expert to examine the file. Can you extract all the information from this strange file?

Solution: Opening the suspicious file with Mozilla web browser gave part of the flag while converting the suspicious file using kali gave the rest of the flag.

13. Can You See

Description: Decode image metadata to reveal flag.

Solution: Viewed image metadata using jimpl.com, decoded AttributionURL using Base64.

14. Hashing JobApp

Description: If you want to hash with the best, beat this test! nc saturn.picoctf.net 52468

Solution: I used https://md5.gromweb.com to convert the strings generated to hash.

15. PW Crack 1

Description: Can you crack the password to get the flag?

Solution: Executed the code and inputted the user_pw provided within the code for the flag password and that printed out the flag

16. PW Crack 2

Description: Can you crack the password to get the flag?

Solution: Used the print statement on the user_pw input and that printed out the password for the flag. Inputted the password and the flag was revealed.

17. Runme.py

Description: Run the runme.py script to get the flag. Download the script with your browser or with wget in the webshell.

Solution: Executed the code and the flag was revealed

18. Convertme.py

Description: Run the Python script and convert the given number from decimal to binary to get the flag.

Solution: Executed the script and flag was revealed.

19. Codebook

Description: Run the Python script code.py in the same directory as codebook.txt.

Solution: Executed the script and the flag was revealed.

20. Fixme1.py

Description: Fix the syntax error in this Python script to print the flag.

Solution: Corrected the indentation error, executed the script and the flag was revealed.

21. Fixme2.py

Description: Fix the syntax error in the Python script to print the flag.

Solution: Corrected the indentation error, executed the script and the flag was revealed.

22. Magikarp Ground Mission

Description: Do you know how to move between directories and read files in the shell? Start the container, `ssh` to it, and then `ls` once connected to begin.

Solution: Navigated through the directory using cd, ls to list files in the directory and cat to print the details in the txt file. Gathered the fragmented flag pieces from each directory visited.

23. Wave a Flag

Description: Can you invoke help flags for a tool or binary?

Solution: Accessed the program with picoCTF webshell, made the ./warm file executable with chmod +x warm. Used ./warm –h to reveal the flag

24. Scavenger Hunt

Description: There is some interesting information hidden around this site http://mercury.picoctf.net:5080/. Can you find it?

Solution: Inspected the webpage to find the flag.

25. Obedient Cat

Description: This file has a flag in plain sight (aka "in-the-clear").

Solution: Accessed the file with picoCTF webshell, ls to see the available files in the directory. Finding FLAG as one of the files, just cat FLAG to get the final flag.

26. Python Wrangling

Description: Python scripts are invoked kind of like programs in the Terminal... Can you run this Python script using this password to get the flag?

Solution: Accessed the python script with the picoCTF webshell and added the flag.txt.en file to the directory also. Used the command python ende.py -d flag.txt.en to execute the file, inputted the password and got the flag.

27. 2Warm

Description: Can you convert the number 42 (base 10) to binary (base 2)?

Solution: Did the conversion and got the flag.

28. Mod 26

Description: Cryptography can be easy, do you know what ROT13 is? cvpbPGS{arkg_gvzr_V'yy_gel_2_ebhaqf_bs_ebg13_uJdSftmh}

Solution: Used ROT13 to decode using cyberchef and the decoded message was the flag.

29. First Grep

Description: Can you find the flag in file? This would be really tedious to look through manually, something tells me there is a better way.

Solution: Accessed the file with picoCTF webshell, used the command cat file | grep pico to find the flag.

30. The Numbers

Description: The numbers... what do they mean?

Solution: Found the flag by using the numerical representation of each alphabets.

31. Bases

Description: What does this bDNhcm5fdGgzX3IwcDM1 mean? I think it has something to do with bases.

Solution: Used cyberchef to decode the message from base 64.

32. 13

Description: Cryptography can be easy; do you know what ROT13 is? cvpbPGS{abg_gbb_onq_bs_n_ceboyrz}

Solution: Used ROT13 to decode using cyberchef and the decoded message was the flag.

33. Insp3ct0r

Description: Kishor Balan tipped us off that the following code may need inspection: https://jupiter.challenges.picoctf.org/problem/9670/ (link) or http://jupiter.challenges.picoctf.org:9670

Solution: Inspected the webpage to find the flag.

34. Let's Warm Up

Description: If I told you a word started with 0x70 in hexadecimal, what would it start with in ASCII?

Solution: Converted the hex to ASCII using https://onlinetools.com/hex/convert-hex-to-ascii and got the flag.

35. Glory of the Garden

Description: This garden contains more than it seems.

Solution: Accessed the garden.jpg file using picoCTF and used this command "strings garden.jpg | grep pico" to get the flag

36. what's a net cat?

Description: Using netcat (nc) is going to be pretty important. Can you connect to jupiter.challenges.picoctf.org at port 41120 to get the flag?

Solution: Connected using picoCTF webshell and used the command"nc jupiter.challenges.picoctf.org 41120" to get the flag.

37. strings it

Description: Can you find the flag in file without running it?

Solution: Used the command “strings strings | grep pico” on the file strings to find the flag.

38. where are the robots

Description: Can you find the robots? https://jupiter.challenges.picoctf.org/problem/56830/ (link) or http://jupiter.challenges.picoctf.org:56830

Solution: Found the flag by adding robots.txt to the end of the link. After being directed to another page, I removed the “robots.txt” at the end of the link and added “1bb4c.html” from the page I was directed to and there was the flag .

39. Vault-door-training

Description: Your mission is to enter Dr. Evil's laboratory and retrieve the blueprints for his Doomsday Project. The laboratory is protected by a series of locked vault doors. Each door is controlled by a computer and requires a password to open. Unfortunately, our undercover agents have not been able to obtain the secret passwords for the vault doors, but one of our junior agents obtained the source code for each vault's computer! You will need to read the source code for each level to figure out what the password is for that vault door. As a warmup, we have created a replica vault in our training facility. The source code for the training vault is here: VaultDoorTraining.java.

Solution: Inspected the code with Vscode and found the flag in the source code.

40. Glitch Cat

Description: Our flag printing service has started glitching! $ nc saturn.picoctf.net 61546

Solution: Found the flag using this command “python -c "print('picoCTF{gl17ch_m3_n07_' + chr(0x62) + chr(0x64) + chr(0x61) + chr(0x36) + chr(0x38) + chr(0x66) + chr(0x37) + chr(0x35) + '}')""