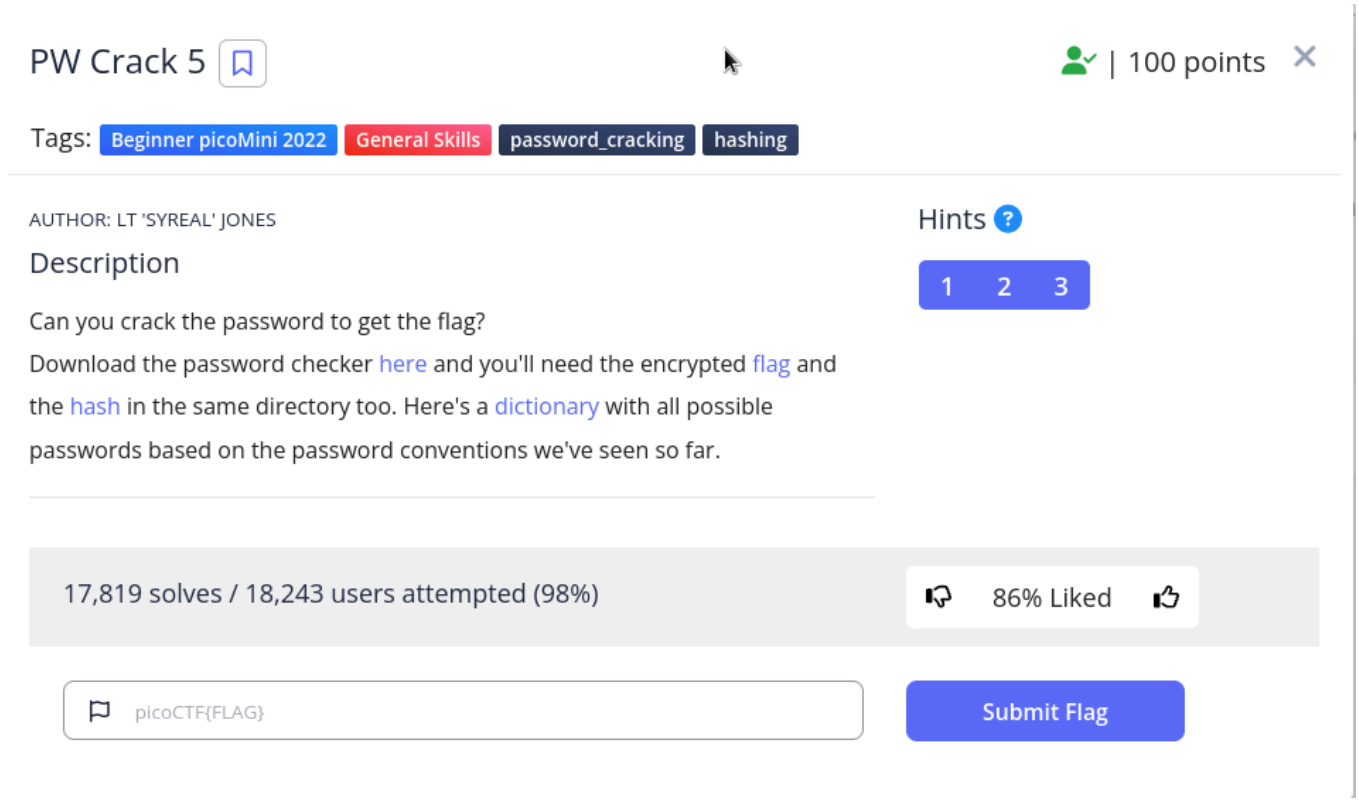
PW Crack 5

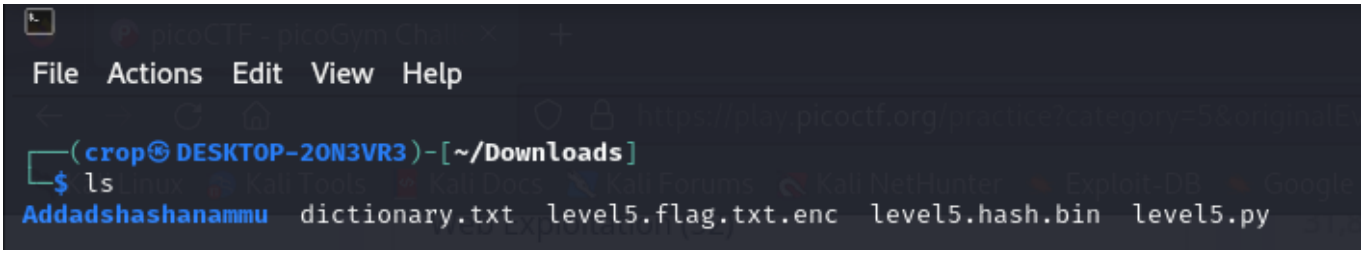
Linux/Python

Dictionary Attack



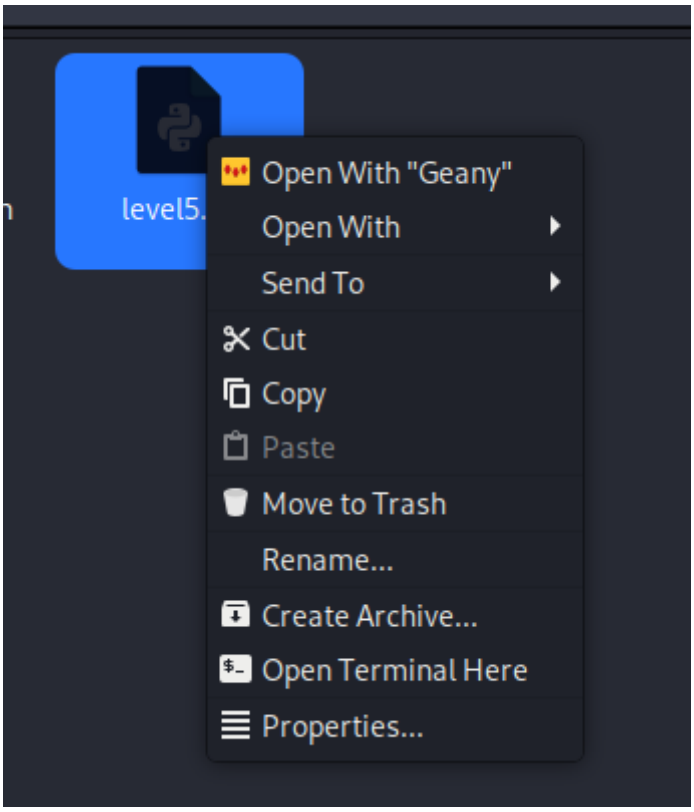
This is the prompt for the CTF and all of the download links

To Start I am going to download all of these files



Using the ls command I can verify all the files are downloaded and we can begin searching for the flag

The main file im interested in here is the encrypted Flag file which I know I can only decrypt with the level5.py script. So for me the natural starting point is viewing this python script



For this Ill be using a program called Geany as my text editor

level5.py

```python
import hashlib

### THIS FUNCTION WILL NOT HELP YOU FIND THE FLAG --LT ########################
def str_xor(secret, key):
    #extend key to secret length
    new_key = key
    i = 0
    while len(new_key) < len(secret):
        new_key = new_key + key[i]
        i = (i + 1) % len(key)
    return "".join([chr(ord(secret_c) ^ ord(new_key_c)) for (secret_c,new_key_c) in zip(secret,new_key)])
#################################################################################

flag_enc = open('level5.flag.txt.enc', 'rb').read()
correct_pw_hash = open('level5.hash.bin', 'rb').read()


def hash_pw(pw_str):
    pw_bytes = bytearray()
    pw_bytes.extend(pw_str.encode())
    m = hashlib.md5()
    m.update(pw_bytes)
    return m.digest()


def level_5_pw_check():
    user_pw = input("Please enter correct password for flag: ")
    user_pw_hash = hash_pw(user_pw)

    if( user_pw_hash == correct_pw_hash ):
        print("Welcome back... your flag, user:")
        decryption = str_xor(flag_enc.decode(), user_pw)
        print(decryption)
        return
    print("That password is incorrect")



level_5_pw_check()
```

This is what the code looks like. Its important to always review code to understand exactly how it works and what you can exploit. To make this easier to understand I have added comments to all of the code.

```python

# Read the encrypted flag content from 'level5.flag.txt.enc'
flag_enc = open('level5.flag.txt.enc', 'rb').read()

# Read the correct password hash from 'level5.hash.bin'
correct_pw_hash = open('level5.hash.bin', 'rb').read()

# Define a function to hash a password string using MD5
def hash_pw(pw_str):
    pw_bytes = bytearray()
    pw_bytes.extend(pw_str.encode())
    m = hashlib.md5()
    m.update(pw_bytes)
    return m.digest()

# Define a function to check the password for level 5
def level_5_pw_check():
    # Prompt the user for a password input
    user_pw = input("Please enter the correct password for the flag: ")
    user_pw_hash = hash_pw(user_pw)  # Hash the user-provided password

    # Compare the user's password hash with the correct password hash
    if user_pw_hash == correct_pw_hash:
        print("Welcome back... your flag, user:")
        # Decrypt the flag using XOR with the user's password
        decryption = str_xor(flag_enc.decode(), user_pw)
        print(decryption)
        return
    print("That password is incorrect")

# Call the function to check the password for level 5
level_5_pw_check()

```

Next lets take a look at our dictionary. We can infer from the prompt that the users password is somewhere in this dictionary.txt so we want to setup a dictionary attack. This should be easy to integrate with our current level5.py script

level5.py × dictionary.txt ×

```
0000
0001
0002
0003
0004
0005
0006
0007
0008
0009
000a
000b
000c
000d
000e
000f
0010
0011
0012
0013
0014
0015
0016
0017
0018
0019
001a
001b
001c
001d
001e
001f
0020
0021
0022
0023
0024
0025
0026
0027
0028
0029
002a
```

this dictionary has 65,536 different passwords in it.

```
    # user_pw = input("Please enter the correct password for the flag: ")
    with open('dictionary.txt') as f: #Open dictionary and assign it to variable f
        lines = f.readlines() #Performing the readlines function
        for line in lines: #Iterate through lines/passwords
            user_pw = line.strip() #assign each password to the user_pw variable and strip the excess
            user_pw_hash = hash_pw(user_pw)  # Hash the user-provided password

                    # Compare the user's password hash with the correct password hash
            if user_pw_hash == correct_pw_hash:
                print("Welcome back... your flag, user:")
                # Decrypt the flag using XOR with the user's password
                decryption = str_xor(flag_enc.decode(), user_pw)
                print(decryption)
                return
            pass # instead of returning 60,000 print messages for every incorrect password, the script will now pass over incrorrect passwords
           # print("That password is incorrect")

# Call the function to check the password for level 5
level_5_pw_check()
```

I have now added the dictionary attack code and commented all of my new lines to make them easy to understand. To sum up our dictionary attack we first start by opening the dictionary.txt and iterating through every line. The code will test each dictionary line against the user-pw value and pass over any incorrect passwords. When it finds the correct password it should print the decrypted flag file. Lets run our code !

```
┌──(crop㉿DESKTOP-2ON3VR3)-[~/Downloads]
└─$ python3 level5.py
Welcome back... your flag, user:
picoCTF{h45h_sl1ng1ng_40f26f81}

┌──(crop㉿DESKTOP-2ON3VR3)-[~/Downloads]
└─$
```

Perfect we have our flag ! And have successfully cracked the password. To know what the password is we can add a print line to our script

```
            # Compare the user's password hash with the correct password hash
        if user_pw_hash == correct_pw_hash:
            print("Welcome back... your flag, user:")
            # Decrypt the flag using XOR with the user's password
            decryption = str_xor(flag_enc.decode(), user_pw)
            print(decryption)
            print(user_pw)
            return
        pass # instead of returning 60,000 print messages for every incorrect password, th
```

This will print our password after it prints the decrypted flag

```
┌──(crop㉿DESKTOP-2ON3VR3)-[~/Downloads]
└─$ python3 level5.py
Welcome back... your flag, user:
picoCTF{h45h_sl1ng1ng_40f26f81}
7e5f
```

Our users password is 7e5f and this challenge is complete.