

# PW Crack 5

Linux/Python

Dictionary Attack

PW Crack 5

100 points

Beginner picoMini 2022

General Skills

password\_cracking

hashing

AUTHOR: LT 'SYREAL' JONES

Description

Can you crack the password to get the flag?  
Download the password checker [here](#) and you'll need the encrypted [flag](#) and the [hash](#) in the same directory too. Here's a [dictionary](#) with all possible passwords based on the password conventions we've seen so far.

Hints ?

1 2 3

17,819 solves / 18,243 users attempted (98%)

86% Liked

picoCTF{FLAG}

Submit Flag

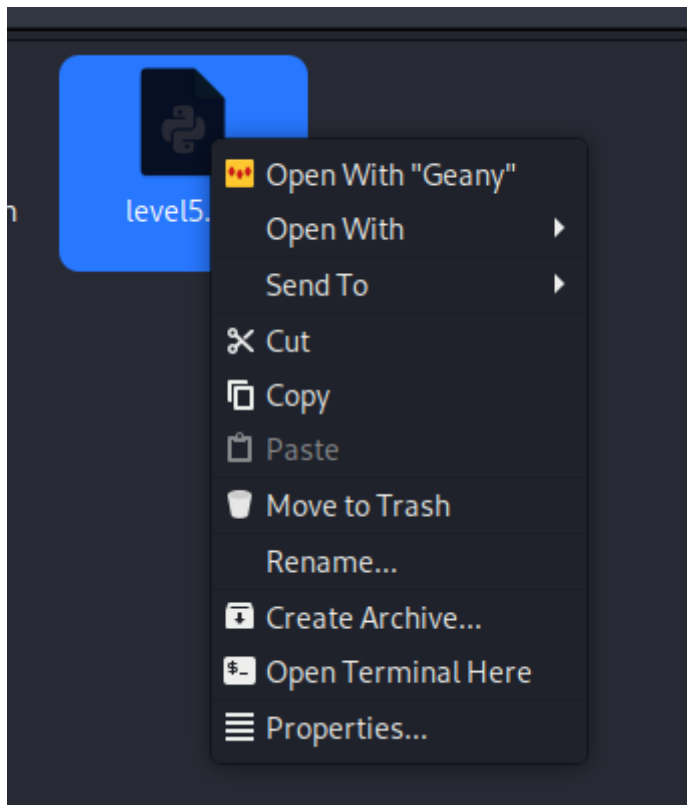
This is the prompt for the CTF and all of the download links

To Start I am going to download all of these files

```
File Actions Edit View Help
(crop@DESKTOP-20N3VR3) - [~/Downloads]
$ ls
Addadshashanammu dictionary.txt level5.flag.txt.enc level5.hash.bin level5.py
```

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 x
1  import hashlib
2
3  ### THIS FUNCTION WILL NOT HELP YOU FIND THE FLAG --LT #####
4  def str_xor(secret, key):
5      #extend key to secret length
6      new_key = key
7      i = 0
8      while len(new_key) < len(secret):
9          new_key = new_key + key[i]
10         i = (i + 1) % len(key)
11     return "".join([chr(ord(secret_c) ^ ord(new_key_c)) for (secret_c,new_key_c) in zip(secret,new_key)])
12     #####
13
14     flag_enc = open('level5.flag.txt.enc', 'rb').read()
15     correct_pw_hash = open('level5.hash.bin', 'rb').read()
16
17
18     def hash_pw(pw_str):
19         pw_bytes = bytearray()
20         pw_bytes.extend(pw_str.encode())
21         m = hashlib.md5()
22         m.update(pw_bytes)
23         return m.digest()
24
25
26     def level_5_pw_check():
27         user_pw = input("Please enter correct password for flag: ")
28         user_pw_hash = hash_pw(user_pw)
29
30         if( user_pw_hash == correct_pw_hash ):
31             print("Welcome back... your flag, user:")
32             decryption = str_xor(flag_enc.decode(), user_pw)
33             print(decryption)
34             return
35         print("That password is incorrect")
36
37
38
39     level_5_pw_check()
40
41
```

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.

```

16
17 # Read the encrypted flag content from 'level5.flag.txt.enc'
18 flag_enc = open('level5.flag.txt.enc', 'rb').read()
19
20 # Read the correct password hash from 'level5.hash.bin'
21 correct_pw_hash = open('level5.hash.bin', 'rb').read()
22
23 # Define a function to hash a password string using MD5
24 def hash_pw(pw_str):
25     pw_bytes = bytearray()
26     pw_bytes.extend(pw_str.encode())
27     m = hashlib.md5()
28     m.update(pw_bytes)
29     return m.digest()
30
31 # Define a function to check the password for level 5
32 def level_5_pw_check():
33     # Prompt the user for a password input
34     user_pw = input("Please enter the correct password for the flag: ")
35     user_pw_hash = hash_pw(user_pw) # Hash the user-provided password
36
37     # Compare the user's password hash with the correct password hash
38     if user_pw_hash == correct_pw_hash:
39         print("Welcome back... your flag, user:")
40         # Decrypt the flag using XOR with the user's password
41         decryption = str_xor(flag_enc.decode(), user_pw)
42         print(decryption)
43         return
44     print("That password is incorrect")
45
46 # Call the function to check the password for level 5
47 level_5_pw_check()
48

```

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 x

dictionary.txt x

```
1 0000
2 0001
3 0002
4 0003
5 0004
6 0005
7 0006
8 0007
9 0008
10 0009
11 000a
12 000b
13 000c
14 000d
15 000e
16 000f
17 0010
18 0011
19 0012
20 0013
21 0014
22 0015
23 0016
24 0017
25 0018
26 0019
27 001a
28 001b
29 001c
30 001d
31 001e
32 001f
33 0020
34 0021
35 0022
36 0023
37 0024
38 0025
39 0026
40 0027
41 0028
42 0029
43 002a
```

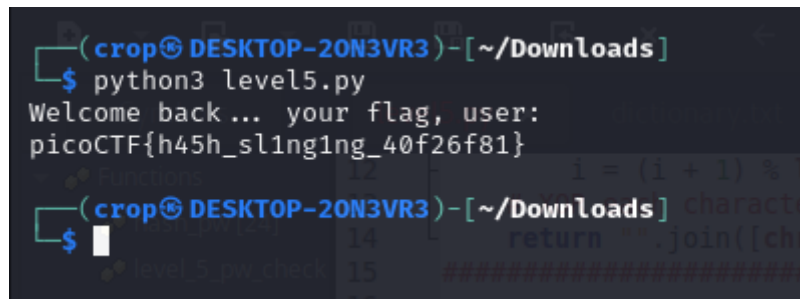
this dictionary has 65,536 different passwords in it.

```
# 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 incorrect 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 !

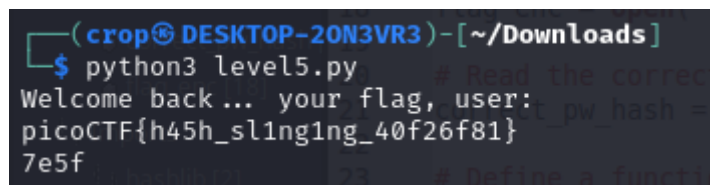


```
(crop@DESKTOP-20N3VR3) - [~/Downloads]
$ python3 level5.py
Welcome back... your flag, user:
picoCTF{h45h_sl1ng1ng_40f26f81}
```

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, the
```

This will print our password after it prints the decrypted flag



```
(crop@DESKTOP-20N3VR3) - [~/Downloads]
$ python3 level5.py
Welcome back... your flag, user:
picoCTF{h45h_sl1ng1ng_40f26f81}
7e5f
```

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