#AUTHOR - Prajna Bhandary

WRITE-UP- Mysterious_mystery.py

CHALLENGE 2

The file this challenge gave had my resume sent back to me. But, at least the filename was different. In fact, it had a .py before .pdf. So the step number 1 for this challenge is to rename the file in .py format which means remove the .pdf form the filename.
After renaming it we can open the file in a python editor. The first 1000+ lines are filled with gibberish content but we can observe that parts of it has my resume's content in it. So scrolling further down I came across a few lines that looked like python code and had some base 64 decoding lines. That's interesting!! So, lets try running the code to see what happens. The hint given is and it gives the output as 'Hint: First Code Input: L' so I try to type L. Nothing really happens which is good because otherwise there is no challenge!

So I need to go through the code to see what happens. Let's get rid of my resume part because I know the program starts where the modules are imported. I can see modules like zlib, marshal, termios, new, sys, base64, etc. Termios is a linux specific module which means that they can only be run in an linux environment. Marshal is used to load byte or pseudo compile code and execute it as it is.

Let's try putting some print statements to make further sense of it. 'a0_func' has been base64 decoded (loaded using marshal.loads) and stored in the variable x. On printing that we can see some line like
<code object askdecrypt at 0x7f62d09329b0, file "mysterious_mystery.py",line 10272>

I used the uncompile6 module and imported the decompile method to understand the code a little better. On decompiling a0_func, I observed that it loads a function and reads the input as passphrase. The line below gives the idea that the value x is being XORed to the value of the passphrase and then it is mapping that value with the message (Message might be the result I am looking for in this challenge!):
res = ('').join(map(lambda x: chr(ord(x) ^ passphrase), message))

The location of the function is probably being sent to create a new function, new.function(). x_func points to a new function created with this x value and a dictionary.Okay.

Now there is a huge line of code which absolutely makes no sense but this line is being decompressed, decoded and executed at the end. So lets try printing the x value. That's just printing a huge value which looks to be encoded in base64. So let me just decode it and store this value in a file.

On observing the file I find (a new function c1(x,y) and a new string f12). The part at the end is interesting. The f12 value is being decoded and stored to the x, y variable. So let's try incorporating this in the program and see what x,y is going to print.

After running the program, It asks for the CODE and I type L and again nothing happens. So this is only possible if it is in an infinite loop. I do see a while loop so let's block that from executing continuously. After making multiple attempts using different logics I decided to decode the y value and store it in a file. On executing that I see it accepts L and actually stores values inside the new file. It is similar to the file I got previously. So looks like every time I run this code and replace the f values it is going to give me the next code. I am expecting the values to go till f0. But to get the next f11 - f0 values I need to guess the password character in every turn.

Okay so now we need to guess the password character every time I run the file. Now because the password is not stored, getting a brute force done seems like an option to me. So after brute-forcing we get the password as **'Lifelessneoss~'**
I get a scroll with some coordinates. Looks like they are the coordinates for the red balloon office. Cool!

The challenge is not over yet. I need to look at the memory to get the next step of the challenge. After some research I see that the dis module helps to disassemble the code. There I found the email address I need to send the challenges too.

Some important things I needed to note in this challenge:

- Running it in python 3 it did not work. It only works in python 2.7.
- It only runs in linux, that is known by the library termios
- The program accepts only one character input in every turn.