Hide & eXec WRITEUP

Opening the downloadable zip of the challenge, we saw that there were two files:

- A png file containing a BarCode;
- A .zip file that required a password to be extracted.

By reading the BarCode, we found out that some code was present. By executing this code we obtained a string, which we utilized as the password to extract the zip. So, we just had to keep extracting the several .zip files. During the iteration process, we found out that the codes were written in different languages:

- Python
- PHP
- Java
- Javascript
- Bash

We coded a Python3 script to do the work of extraction/execution of the codes, which you can read in the following (note that 7zip is required to extract the zip files):

```
# General imports
import os
import sys
import re
# Read barcode
import zxing
# Execute codes
import brainfuck
import subprocess
from py_mini_racer import py_mini_racer

def read_barcode(img_path):
    """Return the text inside an image containing a barcode."""
    reader = zxing.BarCodeReader()
    barcode = reader.decode(img_path)
    return barcode.parsed
```

```
def execute_code(code):
    """Execute the given code and return its output"""

    # Javascript
    if "console.log" in code:
        ctx = py_mini_racer.MiniRacer()
        new_code = "function bella() { " + re.sub("\); *$", "", code).replace("c
        ctx.eval(new_code)
        out = ctx.call("bella")
        return out

    # Brainfuck
    if "+++" in code:
        bro = brainfuck.to_function(code)
        return bro()

    # Python
    if "print(" in code:
        if r"\n" in code:
            new_code = code.replace(r"\n", "\n")
        else:
            new_code = code

        with open("tmp", "w") as f:
            f.write(new_code)

        out = subprocess.check_output("python3 tmp", shell=True).decode().strip(
        return out

    # Java
    if "class" in code:
        with open("Main.java", "w") as f:
            f.write("public " + code)
        subprocess.run("javac Main.java", shell=True)
        out = subprocess.check_output("java Main", shell=True).decode().strip()
        return out

    # Others
    with open("tmp", "w") as f:
        f.write(code)
```

```
    # PHP
    if "php" in code:
        out = subprocess.check_output("php tmp", shell=True).decode().strip()
        return out

    # Bash
    if "done" in code:
        out = subprocess.check_output("bash tmp", shell=True).decode().strip()
        return out

    return "UNIDENTIFIED CODE"

def extract_zip(zip_path, pwd, out):
    """
    Extract a zip archive, provided the path to the zip, password and path for tl
    and returns a list containing the filenames of the extracted files.
    """
    a = os.listdir(out)
    subprocess.run(["7z", "x", zip_path, f"-p{pwd}", f"-o{out}"], capture_output=
    b = os.listdir(out)

    return list(set(b) - set(a))


# Take the filename to extract (without extension)
curr_file = sys.argv[1]
path = "./"
# Iterate the process until some errors occurs (something unexpected come out)
while(True):
    print("CURRENT FILE:", curr_file)

    code = read_barcode(path + curr_file + ".png")
    print("CODE:", code)

    pwd = execute_code(code)
    print("PWD:", pwd)

    extracted = extract_zip(path + curr_file + ".zip", pwd, path)

    # Get the next file to work with
    curr_file = extracted[0].split(".")[0]
    print("------")
```

Nearly to the end of the process, another language was added:

- Brainfuck

By implementing the execution of Brainfuck's code in our script, we concluded the iteration process by arriving at a last zip file containing the file flag.txt, which contains a useless YouTube address and the suggestion to look back into the logs we produced during the process.

So, we found out that the password we used to extract the last zip is our flag: `{FLG:P33k-4-b0o!UF0undM3,Y0urT0olb0xIsGr8!!1}`