Byte Crypt (Medium) - Rev

Sunday, June 11, 2023 2:55 PM

Description

"Unravel the bytecode labyrinth, decipher the encryption scheme, and reclaim the hidden flag."

Solution

In this challenge, we're given the python byte code of a python file.

We need to analyze the byte code and rewrite the original python code.

After reversing the code, the original code would be as follows.

```
import base64
import zlib
import os

flag_file = "flag.txt"
if not os.path.exists(flag_file):
    raise FileNotFoundError("Flag file flag.txt not found!")
with open(flag_file, "r") as file:
    original_flag = file.read().strip()

def xor_apply(data, key):
    decrypted = ""
    for i in range(len(data)):
        decrypted += chr(ord(data[i]) ^ ord(key[i % len(key)]))
    return decrypted

def super_encryption(encrypted_flag):

    # Step 1: XOR Decryption
    key = "CUwRn048*r$gUuE"
    xored_data = xor_apply(encrypted_flag, key)

    # Step 2: Reversing
    reversed_data = xored_data[::-1]

    # Step 3: Zlib Compression
    compressed_data = zlib.compress(bytes(reversed_data, 'utf-8'))

    # Step 4: Base64 Encoding
    encoded_flag = base64.b64encode(compressed_data)

    return encoded_flag

# print(original_flag)

flag = super_encryption(original_flag)
```

```
print(f'Encrypted Flag: {flag}')
```

To the flag.txt, it first applies xor_apply to xor the flag with the key.

Next it reverses the produced output.

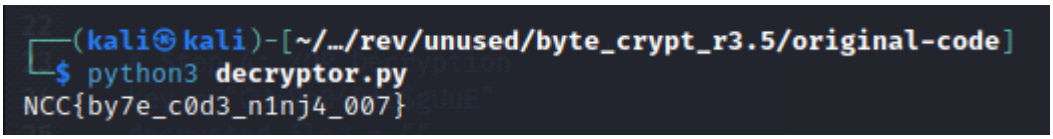
Then it applies zlib compression on the reversed output.

Finally it base64 encodes the data.

To reverse this, we'll need to reverse each step.

Final decryption script.

```
import zlib
import base64
import os

flag_file = "flag.enc"
if not os.path.exists(flag_file):
    raise FileNotFoundError("Encrypted flag file flag.enc not found!")
with open(flag_file, "r") as file:
    encrypted_flag = file.read().strip()


def super_decryption(encrypted_flag):

    # Step 1: Base64 Decoding
    decoded_data = base64.b64decode(encrypted_flag)

    # Step 2: Zlib Decompression
    decompressed_data = zlib.decompress(decoded_data)

    # Step 3: Reversing
    reversed_data = decompressed_data[::-1]

    # Step 4: XOR Decryption
    key = "CUwRn048*r$gUuE"
    decrypted_flag = ""
    for i in range(len(reversed_data)):
        decrypted_flag += chr((reversed_data[i]) ^ ord(key[i % len(key)]))

    return decrypted_flag

# Decrypt the flag
flag = super_decryption(encrypted_flag)
print(flag)
```



Flag: NCC{by7e_c0d3_n1nj4_007}