# Компьютерные вирусы и антивирусы:

## взгляд программиста

Климентьев К.Е.

ДМК ПРЕСС
ИЗДАТЕЛЬСТВО

**УДК 004.49**
**ББК 32.973-018.2**
**К49**

**Климентьев К. Е.**
К49 Компьютерные вирусы и антивирусы: взгляд программиста. – М.: ДМК Пресс, 2013. – 656 с.: ил.

**ISBN 978-5-94074-885-4**

Книга представляет собой курс компьютерной вирусологии, посвященный подробному рассмотрению феномена саморазмножающихся программ. Содержит неформальное и формальное введение в проблему компьютерных вирусов, описание принципов их работы, многочисленные примеры кода, методики обнаружения и удаления, а также лежащие в основе этих методик математические модели. Рассматривает все наиболее широко распространенные в прошлом и настоящем типы вирусов. Ориентирована на самую широкую аудиторию, но прежде всего на студентов и программистов – будущих и действующих специалистов в области защиты информации и разработки системного и прикладного программного обеспечения. Также может быть полезна и интересна «рядовым» пользователям, интересующимся проблемой компьютерных вирусов.

УДК 004.49
ББК 32.973-018.2



ISBN 978-5-94074-885-4

# Содержание

# ГЛАВА 2

# Загрузочные вирусы

В системных областях дисковых устройств – винчестеров и дискет – присутствуют специальные программные компоненты, обеспечивающие начальную загрузку операционных систем. Вредоносные программы, заражающие эти программные компоненты, известны с 1986 г. – они называются *загрузочными* или *boot-вирусами*. Этот тип компьютерной «заразы» доставлял немало хлопот пользователям 1980–1990-х годов, но в новом веке, казалось, был «окончательно побежден» и «вымер». Однако в последние годы появились очень сложные и опасные вредоносные программы, использующие в точности те же технологии, – «*буткиты*». И, судя по всему, их история только начинается.

## 2.1. Техническая информация

*...Всего-то в ней два медных диска с чайное блюдце... Нет, ребята, тяжело эту штуку описать, если кто не видел, очень уж она проста на вид...*

А. и Б. Стругацкие. «Пикник на обочине»

Для понимания принципов функционирования загрузочных вирусов необходимо разобраться в устройстве дисковых носителей и в том, каким образом после включения компьютера происходят загрузка и запуск операционных систем.

Все дисковые устройства имеют единую «геометрию»[1]. Очень грубо и упрощенно дисковое устройство можно представить в виде пакета круглых пластин, насаженных на общую ось (см. рис. 2.1). На поверхность пластин нанесен магнитный слой, который хранит ин-

---

[1] Здесь не рассматриваются CD/DVD или «флэшки». Заражение их возможно, но на практике не встречается.

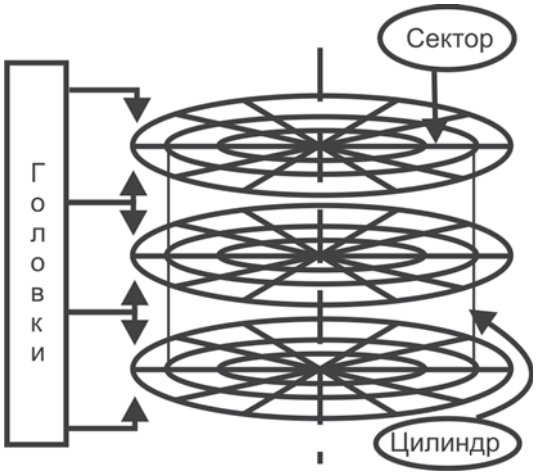


**Рис. 2.1** ❖ Логическая организация дискового устройства

формацию, записываемую или считываемую при помощи магнитных головок, количество которых равно количеству рабочих поверхностей. На винчестерах, как правило, присутствуют несколько рабочих поверхностей, а на дискетах только две: нулевая и первая.

Участки дисковой поверхности, предназначенные для хранения информации, образуют ряд концентрических окружностей – *дорожек* (*треков*). Самая ближняя к внешнему краю диска дорожка имеет номер 0, следующая – номер 1 и т. д. На дискетах количество дорожек бывает 40 или 80, на винчестерах типичное количество – несколько сотен или тысяч. Совокупность всех дорожек, равноудаленных от оси и расположенных на всех рабочих поверхностях, носит название *цилиндра*. Каждая магнитная дорожка разбита на ряд *секторов*, нумеруемых от 1 и до максимального значения, которое для дискет составляет 9, 15 или 18, а для винчестеров достигает иногда нескольких сотен. По умолчанию в сектор стандартного размера возможно записать 512 байт информации.

Любой сектор можно однозначно идентифицировать, задав тройку {цилиндр, головка, сектор}. Например, самый первый сектор на дисковом устройстве, в котором располагается крохотная программа начальной загрузки, имеет координаты {0,0,1}; первый сектор второй по счету головки {1,0,1}; первый сектор второго по счету цилиндра {0,1,1} и т. п. Такой способ нумерования секторов характерен для

*CHS-адресации* (от англ. *cylinder* – цилиндр, *head* – головка чтения-записи, *sector* – сектор).

Существует и другой способ адресации конкретного сектора. Можно присвоить стартовому сектору диска номер 0, следующему за ним 1, потом 2 и т. д. Такая «абсолютная» адресация секторов носит наименование LBA (от англ. *Logical Block Address* – Логический адрес блока). Формула пересчета из CHS в LBA выглядит следующим образом:

$$N_{LBA} = N_s N_h c + N_s h + s - 1,$$

где $N_{LBA}$ – абсолютный номер сектора (начиная с 0); $N_s$ – количество секторов на дорожке; $N_h$ – количество головок (рабочих поверхностей); $c$, $h$ и $s$ – номер дорожки, номер головки и номер сектора на дорожке соответственно. Именно LBA используется на современных винчестерах большого объема, но в эпоху расцвета загрузочных вирусов почти всегда применялась «троичная» система адресации.

Работа с дисковым устройством через контроллер дисковода или винчестера довольно сложна. Поэтому в ROM BIOS каждого PC-совместимого компьютера записаны стандартные процедуры, обеспечивающие доступ к дисковым устройствам. Большинство старых системных и прикладных программ, в том числе и некоторые операционные системы (например, MS-DOS), пользуются при обращении к дискам именно процедурами BIOS. Современные же операционные системы (Windows, клоны UNIX и т. п.) обращаются к BIOS исключительно редко, предпочитая работать с контроллером устройства напрямую. Тем не менее начальная загрузка любых операционных систем до сих пор возможна только средствами стандартных процедур BIOS, и никак иначе[1].

Доступ к дисковым процедурам BIOS возможен через программное прерывание 13h. Также эти процедуры (в той части, которая касается работы с дискетами) имеют еще одну точку входа – через прерывание 40h. Интересно, что обработчик прерывания 40h располагается в ROM BIOS практически всегда по адресу F000h:EC59h. Приведем примеры чтения с винчестера содержимого загрузочного сектора с адресом {0,0,1} с использованием этой процедуры.

Вариант на языке Ассемблера в MS-DOS:

```
mov ah, 2                    ; Команда "Читать сектор"
mov al, 1                    ; Количество читаемых секторов равно 1
```

[1] Планируется, что перспективные архитектуры на базе спецификации EFI/UEFI обойдутся без BIOS.

```
mov ch, 0                   ; Номер цилиндра равен 0; старшие биты числа
                            ; могут размещаться в cl
mov cl, 1                   ; Номер читаемого сектора равен 1
mov dh, 0                   ; Номер головки равен 0
mov dl, 80h                 ; 80h - код винчестера, 0 и 1 - дискет A: и B:
mov es, SEG Buffer          ; Сегмент буфера данных
mov bx, OFFSET Buffer       ; Смещение буфера данных
int 13h                     ; Собственно выполнение операции чтения
```

Вариант на языке Си в MS-DOS:

```
cmd = 2;           // Команда "Читать сектор"
nsecs=1;           // Количество читаемых секторов равно 1
track=0;           // Номер цилиндра равен 0
sector=1;          // Номер читаемого сектора равен 1
head = 0;          // Номер головки равен 0
drive = 0x80;      // 80h - код первого винчестера; коды 0 и 1
                   // соответствуют дисководам A: и B:
result=biosdisk(cmd, drive,head,track,sector,nsecs,buf);
```

Возможно это и в других операционных системах. Вот как можно прочитать сектор дискового устройства в Windows 9X:

```
#define WIN32_DIOC_DOS_INT13 4
typedef struct DIOCRegs {
    DWORD   reg_EBX;
    DWORD   reg_EDX;
    DWORD   reg_ECX;
    DWORD   reg_EAX;
    DWORD   reg_EDI;
    DWORD   reg_ESI;
    DWORD   reg_Flags;
} DIOC_REGISTERS, *PDIOC_REGISTERS;

DIOC_REGISTERS r;

...

 HANDLE h = CreateFile("\\\\.\\vwin32", 0, 0,
                       NULL, 0, FILE_FLAG_DELETE_ON_CLOSE, NULL);
 r.reg_EAX=0x201;
 r.reg_EBX=(DWORD) &Buf;
 r.reg_ECX=0x0001;
 r.reg_EDX=0;
 DeviceIoControl(h, WIN32_DIOC_DOS_INT13, &r,
                      sizeof(r), &r, sizeof(r), &n, 0);
. . .
```

Возможно это и в Windows-семействах NT:

```
BYTE mbr[512]; DWORD dwRead;
...
```

```
HANDLE hDisk = CreateFile("\\\\.\\PhysicalDrive0", GENERIC_READ,
                          FILE_SHARE_READ, NULL, OPEN_EXISTING, 0, NULL);
ReadFile(hDisk, &mbr, 512, &dwRead, NULL);
...
```

А вот пример для Unix-подобных операционных систем:

```
int f; unsigned char buf[512];
f=open("/dev/hda", O_RDONLY);
read(f, buf, 512);
. . .
```

Сразу после включения питания компьютера происходит аппаратный сброс всех устройств, а счетчик команд устанавливается на начало кода программы POST, которая размещается в запрещенном для записи регионе адресного пространства вместе с BIOS. Эта программа тестирует оборудование, производит, если нужно, его программную инициализацию, а вслед за этим начинает искать активное дисковое устройство, с которого возможна загрузка, – винчестер или дискету.

В весьма редком случае, когда ни одного подходящего устройства не найдено, «доисторические» IBM PC загружали кассетную (расположенную в ПЗУ) версию интерпретатора языка BASIC, игравшую роль встроенной операционной системы. Более поздние персоналки в этом случае просто извещали: «NO ROM BASIC, SYSTEM HALTED». Сообщения, выдаваемые загрузочным блоком современных компьютеров, могут быть различными.

В современных версиях программы SETUP есть опция, инструктирующая программу POST начинать поиск потенциального носителя операционной системы либо с дискеты, либо с винчестера, либо вообще игнорировать какие-либо устройства. Но в любом случае программа POST старается добраться до какого-нибудь дискового устройства, прочитать сектор {0,0,1}, загрузить его содержимое в ОЗУ по жестко фиксированному адресу 0:7C00h и передать туда управление. С этого момента BIOS компьютера снимает с себя всякую ответственность за ход процесса загрузки.

В зависимости от того, с дискеты или с винчестера производится загрузка, содержимое сектора с адресом {0,0,1} различно. Различны и сценарии процесса загрузки операционной системы.

### 2.1.1. Загрузка с дискеты

Предположим, что программа POST нашла в кармане дисковода A: какую-то дискету. В этом случае, прочитав сектор {0,0,1}, она загрузит в память так называемый boot-сектор. Содержимое его может быть

различным для дискет, отформатированных при помощи разных программ, таких как MS Format, FFormat A. Шамарокова, Central Point PC Toools и прочие. Но общая структура boot-сектора (см. рис. 2.2а) и функции содержащейся в нем программы-загрузчика стандартизованы:

```
                        org         0
Start:
                jmp     short Begin
                nop
; Таблица параметров дискеты
OEM_ID          db      8 dup (?)           ; ID форматирующей программы
SecSiz          dw      ?                   ; Размер сектора в байтах
CluSiz          db      ?                   ; Размер кластера в секторах
ResSec          dw      ?                   ; Число зарезервированных секторов
FATs            db      ?                   ; Число FAT на диске
ROOTSiz         dw      ?                   ; Размер корневого каталога
Nsecs           dw      ?                   ; Полное число секторов
MediaDsc db     db      ?                   ; Байтовый ID описания носителя:
                                            ;  0F0h - дискета 1.44 Мб; 0F8h -
                                            ;  жесткий диск и прочее
FATSiz          dw      ?                   ; Размер каждого FAT в секторах
; Поля, используемые MS-DOS версий старше 3.0
TrkSiz          dw      ?                   ; Число секторов на дорожке
Nheads          dw      ?                   ; Число головок
; Поля, используемые MS-DOS версий старше 4.0
HidSecs         dd      ?                   ; Число скрытых секторов
VolSiz          dd      ?                   ; Полное число секторов
DrivNum         db      ?                   ; Номер физического устр-ва
Reserved        db      ?                   ;
ExtSig          db      29h                 ; Признак расширенного загрузчика
VolSerNum       dd      ?                   ; Серийный номер тома
VolLabel        db      11 dup (?)          ; Метка тома
FSysID          db      8 dup (?)           ; Строка - тип файловой системы
; Начало программы загрузки операционной системы
Begin:
                cli
                xor     ax,ax
        ....
; Текст сообщения, выдающегося при невозможности загрузки
; операционной системы
Messag          db      'Non-System disk or disk error',13,10
                Db      'Replace and press any key when ready',0
; Имена файлов, в которых хранится ядро операционной системы
Name1           db      'IO      SYS'
Name2           db      'MS-DOS  SYS',0,0
; Байтовая сигнатура загрузочного сектора
                        org     1FEh
Sign            dw      0AA55h
```

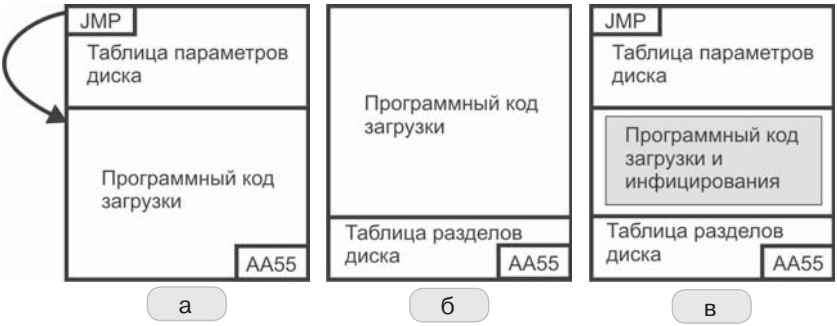


**Рис. 2.2** ❖ Различные варианты содержимого сектора с адресом {0,0,1}: а) Boot-сектор операционной системы (на дискете); б) MBR – главная загрузочная запись (на винчестере); в) типичное содержимое зараженного сектора

Первые три байта сектора зарезервированы под команду или группу команд, служащих для передачи управления на основной код загрузчика.

Далее располагается таблица параметров дискеты, которая необходима, чтобы стандартные процедуры BIOS сумели настроиться на конкретные характеристики носителя при обращении к нему. Она имеет разный размер в зависимости от того, какой программой и в какой версии операционной системы производилось форматирование. Но для того, чтобы дискета оставалась «читабельной», необходимо предусмотреть наличие в таблице по крайней мере правильно заполненных полей вплоть до «FATSiz».

Сразу после таблицы параметров дискеты располагается программный код загрузочной программы. Эта программа должна найти в корневом каталоге дискеты файлы операционной системы, прочитать их в оперативную память и передать им управление. Указанная операция выполняется успешно только в том случае, если дискета является «системной», то есть содержит файлы IO.SYS и MS-DOS.SYS[1]. Впрочем, имена этих файлов зависят от версии загружаемой операционной системы, например в «древних» версиях PC-DOS требовались файлы IBMBIO.SYS и IBMDOS.SYS. Если же программа-загрузчик не сумеет обнаружить «заветные» файлы (что, вообще говоря, практически всегда случается, если пользователь просто использует дис-

[1] Файл командного процессора COMMAND.COM в загрузке не участвует.

кету для хранения своих данных), то она должна отреагировать на попытку загрузиться с «несистемной» дискеты предупреждающим сообщением.

Последними двумя байтами в загрузочном секторе обязательно должны быть 55h и 0AAh.

## 2.1.2. Загрузка с винчестера

Теперь рассмотрим случай, когда карман дисковода пуст, а в качестве носителя операционной системы выступает жесткий диск. Загрузка с винчестера, в отличие от загрузки с дискеты, происходит в два этапа. В стартовом секторе винчестера располагается не загрузчик конкретной операционной системы, а так называемый *внесистемный загрузчик*. Кроме того, в том же секторе располагается таблица, описывающая логические разделы винчестера (Partition Table). Совокупность кода внесистемного загрузчика и таблицы разделов образует *главную загрузочную запись* – MBR (Master Boot Record).

Содержимое стартового сектора винчестера с адресом {0,0,1} обычно имеет следующую структуру (см. также рис. 2.2б):

```
                        org     0
; Начало программы внесистемного загрузчика
Start:
                cli
                ...
                jmp     0000:7C00h              ; Передача управления
                                                ; следующему загрузчику
;Предупреждающие сообщения
Mess1   db      'Invalid partition table', 0
Mess2   db      'Error loading operating system', 0
Mess3   db      'Missing operating system',0
                        ...
                        org     1BEh
; Таблица описания разделов (Partition Table)
; (приведена структура 1-ой "строки" таблицы из 4 возможных)
Active          db ?    ; +00 - Признак активного раздела (80h или 0)
BegHead         db ?    ; +01 - Головка 1-го сектора раздела
BegCylSec       dw ?    ; +02 - Цилиндр/сектор 1-го сектора раздела
Type            db ?    ; +04 - Тип (1/6/B - FAT12/16/32, 7 - NTFS,...)
FinHead         db ?    ; +05 - Головка последнего сектора раздела
FinCylSec       dw ?    ; +06 - Цилиндр/сектор последнего сектора
ReloSec         dd ?    ; +08 - Относительный номер 1-го сектора
PartLen         dd ?    ; +1C - Количество секторов в разделе
...
; Байтовая сигнатура загрузочного сектора
                        org     1FEh
Sign            dw      0AA55h
```

Конец загрузочного сектора также занимает уникальный признак, состоящий из байтов 55h и AAh.

Непосредственно перед этим признаком располагается таблица описания разделов, занимающая 64 байта и состоящая из четырех записей (строк), каждая из которых описывает один из разделов жесткого диска. Если на винчестере существует только один раздел, то строки со второй по четвертую просто заполняются нулями. Таблица содержит информацию, необходимую для распределения пространства жесткого диска под разделы, причем каждый из разделов может быть организован по своим правилам – в соответствии с файловой системой FAT, NTFS, EXTFS и др. Кроме того, один (и только один) раздел может быть объявлен «активным», то есть предназначенным для загрузки операционной системы (в поле Active соответствующей строки должен стоять признак 80h).

Верхнюю же половину MBR занимает специальная программа, которая ищет в Partition Table запись, соответствующую «активному» разделу, рассчитывает (при помощи значений полей BegHead и BegCylSec) местоположение стартового сектора этого раздела, считывает его содержимое при помощи команды 2 прерывания 13h на свое место – по адресу 0:7C00h – и «длинной» командой JMP передает туда управление. Разумеется, поскольку сама эта программа располагается по адресу 0:7C00h, то она предварительно «перетаскивает» свою рабочую копию в другой регион памяти (обычно в 0:600h). Запомните это обстоятельство!

Остается выяснить, что же именно загружается внесистемным загрузчиком из стартового сектора того или иного раздела? Ответ прост: boot-сектор конкретной операционной системы, описание которого приведено выше – при описании загрузки с дискеты. Таким образом, загрузка с винчестера выполняется в два этапа: загрузочные сектора различных типов поочередно считывают друг друга и передают друг другу управление.

Отметим также, что для винчестеров, разбитых на разделы при помощи стандартной утилиты FDISK, стартовый сектор первого раздела обычно размещается по адресу {1,0,1}. Поэтому практически вся так называемая «нулевая дорожка» винчестера, составленная из секторов с адресами вида {0,0,2}, {0,0,3} и т. д., остается пустой. Пустое пространство размером в несколько десятков килобайт активно используется вирусами и «буткитами», хотя там могут размещаться и данные какой-нибудь полезной программы. Например, именно на нулевой дорожке хранят свой код драйверы программы Ontrack Disk

Manager, предназначенной для поддержки работы старых компьютеров с «большими» дисками (то есть с дисками, у которых число цилиндров превосходит 1024) и выполняющей перекодировку из «троичной» системы адресации в LBA.

## 2.2. Как устроены загрузочные вирусы

*Он висел у меня над головой среди заплесневелых проводов... жалкий и нелепый, весь в лохмотьях от карбонной коррозии и в кляксах черной подземной грязи.*

А. и Б. Стругацкие. «Хищные вещи века»

В этом разделе мы кратко рассмотрим общие принципы функционирования загрузочных вирусов.

### 2.2.1. Как загрузочные вирусы получают управление

Обычно вирус просто замещает своим кодом стандартный загрузчик, располагающийся в начальном секторе винчестера или дискеты.

Если вирус заразил загрузчик дискеты, то он может получить управление в одном-единственном случае – если кто-то попытается с этой дискеты загрузиться. Вопреки распространенной среди малоквалифицированных пользователей легенде, «запрыгнуть» на компьютер во время обычных записи или чтения файлов с зараженной дискеты загрузочный вирус в принципе не может!

Если вирус заразил MBR или boot-сектор винчестера, то он получает управление в первые мгновения после включения питания (а также после перезагрузки компьютера «кнопкой» или «тремя пальцами»). Фактически это происходит еще до того, как первый компонент операционной системы оказывается в оперативной памяти. Таким образом, вирус всегда имеет «право выступки» – огромную фору по отношению к любому системному или прикладному программному обеспечению.

Чтобы процедура загрузки операционной системы не нарушалась, вирус может:

- сохранить оригинальное содержимое MBR или Boot-сектора в «укромном уголке» винчестера или дискеты, а после выполнения своих несанкционированных действий загрузить «оригинал» в память и передать ему управление, так что процедура загрузки продолжится и завершится естественным образом;

- выполнить (возможно, упрощенную) процедуру загрузки самостоятельно.

Элементарный анализ структуры размещения информации на дисковых носителях показывает, что «укромных уголков» на дискете или винчестере достаточно. Чаще всего вирус сохраняет оригинальный загрузчик на нулевой дорожке винчестера в обычно неиспользуемой области между секторами {0,0,1} и {1,0,1}.

До заражения (см. рис. 2.3):

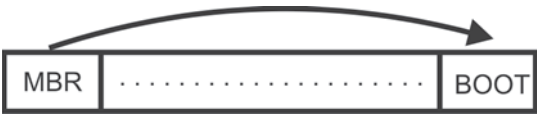


**Рис. 2.3** ❖ Правильная передача управления загрузчику

После заражения (см. рис. 2.4.):


**Рис. 2.4** ❖ Передача управления загрузчику на зараженной машине

Например, многие разновидности вируса **Stoned** используют для хранения старой MBR сектор {0,0,7}. Это, кстати, может привести (и неоднократно приводило!) к конфликтам между вирусами «одной породы», результатом чего являлась утеря оригинального содержимого MBR.

### 2.2.2. Как загрузочные вирусы заражают свои жертвы

Возможны два «сценария» активации вируса:

- вирус находился в одном из загрузочных секторов винчестера;
- вирус находился в загрузочном секторе дискеты.

В первом случае вирус располагает свой код (или часть его) в оперативной памяти компьютера, встраивается в цепочку обработчиков

прерывания 13h или 40h, отслеживает обращения к дискетам и заражает их.

Во втором случае вирус сначала записывается в загрузочный сектор винчестера. Далее он может выполнить действия по оставлению себя в памяти и перехвату дисковых прерываний и, таким образом, немедленно приготовиться к заражению других дискет. Но может и не делать этого, поскольку после следующей же перезагрузки компьютера ситуация автоматически начнет развиваться по сценарию «загрузка с винчестера».

### 2.2.3. Как вирусы остаются резидентно в памяти

Подавляющее большинство загрузочных вирусов пользуются тем фактом, что по адресу 0:413h программа POST помещает размер в килобайтах доступной основной оперативной памяти, а MS-DOS всецело «доверяет» этому значению в процессе своей загрузки и функционирования. Загрузочный вирус стартует после программы POST, но до операционной системы. Он корректирует содержимое ячейки 0:413h в сторону уменьшения (например, было 640, а стало 639) и копирует свой код в образовавшийся якобы «несуществующий» фрагмент оперативной памяти. Там его никто не тронет.

Этот фрагмент всегда располагается в конце 640-килобайтной «основной» памяти и имеет размер, кратный 1024 байтам. Таким образом, положение загрузочных вирусов в оперативной памяти, как правило, жестко фиксировано. Например, если вирус «откусил» 2 Кб памяти, то его код длиной 2048 байт всегда будет размещен, начиная с адреса 9F80h:0.

### 2.2.4. Как заподозрить и «изловить» загрузочный вирус

Загрузочные вирусы сами по себе очень редко конфликтуют с операционной системой, поэтому способны обитать на компьютере долгое время, оставаясь незамеченными и распространяя вокруг себя «заразу» через инфицированные дискеты. Обнаруживаются они, как правило, благодаря следующим обстоятельствам.

Во-первых, большинство загрузочных вирусов, согласно «древней традиции», рано или поздно проявляют себя какой-нибудь дурацкой шуткой или серьезной деструкцией. Например, классический вирус **Stoned** с вероятностью 1/8 блокировал процедуру нормальной загрузки компьютера, информируя пользователя: «Your PC is now

Stoned! LEGALISE MARIJUANA!» А вирус **Michelangelo (Stoned. March6)** активировался всего раз в год – 6 марта, но при этом перезаписывал «мусором» (случайными данными) обширные области на винчестере, с которого загрузился. Подобные несанкционированные действия происходят до загрузки операционной системы, и, следовательно, пострадать от них может даже самый современный компьютер с самой современной версией Windows. Поэтому довольно типичным – увы! – является обнаружение загрузочных вирусов лишь в результате «посмертного вскрытия» компьютера, переставшего загружаться.

Во-вторых, при запуске из-под MS-DOS системная утилита MEM сообщает о количестве занятой и свободной оперативной памяти, каковые на современных компьютерах в сумме должны составлять 640 Кб. Если вирус присутствует в памяти, цифры могут не сойтись:

```
C:\>mem

Тип памяти        Размер   Занято   Свободно
---------------   -------- -------- --------
Обычная           638K     100K     538K <- Надо 640K !
Верхняя           0K       0K       0K
Зарезервировано   0K       0K       0K
Память XMS        15360K   14068K   1292K
---------------   -------- -------- -------
Всего памяти:     15998K   14168K   1830K
```

Наконец, в Windows распределение памяти совсем другое, и MEM, работающая под управлением виртуальной машины, ничего «странного» не покажет. Тем не менее, даже не пользуясь антивирусом, заподозрить наличие нового вируса в системных областях винчестера или дискеты пользователь может и самостоятельно, визуально просматривая их содержимое при помощи утилиты типа Symantec DiskEdit (работает в MS-DOS) или Acronis Disk Editor (работает в Windows). Ведь ранее уже упоминалось, что загрузочные сектора имеют ряд характерных признаков (например, расположенные внутри строковые сообщения), по наличию или отсутствию которых можно отличить «здоровый» сектор от «больного».

Эти же утилиты позволят не только просмотреть, но и скопировать содержимое зараженных секторов винчестера или дискеты в указанный файл – для дальнейшего изучения. Есть одна тонкость: все это желательно делать, загрузившись с заведомо «чистой» системной дискеты или LiveCD/DVD, поскольку вирус, находящийся в памяти, может исказить картину и даже блокировать все ваши попытки.

# 2.3. Охотимся за загрузочным вирусом

*А Малышев в восторге. Прямо на седьмом небе. Режет мух и разглядывает в микроскоп. Говорит, что в жизни не представлял себе ничего подобного.*

А. и Б. Стругацкие.
«Чрезвычайное происшествие»

Проиллюстрируем все этапы обнаружения, анализа и удаления загрузочной «заразы» на примере вируса **Stoned.AntiEXE**. Наш выбор в значительной степени определяется следующими обстоятельствами.

Вирус этот давно известен, по крайней мере с начала 90-х годов XX века. Как свидетельствует бюллетень от Joe Wells, он изредка встречается (вероятно, на дискетах в старых архивах) до сих пор практически везде – и в России, и в США, и в Австралии. Вирусный код содержит ряд любопытных фрагментов, исследование которых следует признать весьма поучительным. Наконец, немаловажным является то, что этот вирус не форматирует винчестер, не обнуляет CMOS-память, не перезагружает каждые 5 минут машину и даже не выводит на экран нецензурных ругательств, а всего лишь «тихо и мирно» препятствует запуску какого-то очень древнего и давно всеми забытого антивируса – программы длиной 200 256 байт.

## 2.3.1. Анализ вирусного кода

Допустим, что, пронаблюдав работу компьютера и изучив при помощи DiskEdit загрузочные сектора винчестера и часто используемых на этом компьютере дискет, вы пришли к выводу о присутствии загрузочного вируса. В частности, шестнадцатеричный дамп MBR выглядит совсем не так, как ему полагается выглядеть. В нем «на просвет» не видно никаких предупреждающих сообщений, которые там обязаны присутствовать!

```
000 E9 14 01 4D-0D 00 00 20-33 2E 33 00-02 02 01 00  щ..M.. 3.3......
010 02 70 00 D0-02 FD 02 00-09 00 02 00-00 00 4D 5A  .p............MZ
020 40 00 88 01-37 0F E0 80-FC F9 74 52-2E A3 07 00  @.И.7.рА..tR.г..
030 CD D3 72 4A-9C 2E 80 3E-08 00 02 75-40 51 56 57  ..rJb.A....u@QVW
...
1C0 01 00 04 04-91 5A 11 00-00 00 26 C8-00 00 00 00  _.__CZ.......&+.
1D0 81 5B 05 04-D1 CF 37 C8-00 00 D9 7B-00 00 00 00  Б[__--7+ ..+{...
1E0 00 00 00 00-00 00 00 00-00 00 00 00-00 00 00 00  ................
1F0 00 00 00 00-00 00 00 00-00 00 00 00-00 00 55 AA  ..............Uк
```