1989

A Computer Virus Primer

Eugene H. Spafford
Purdue University, spaf@cs.purdue.edu

Kathleen A. Heaphy

David J. Ferbrache

Report Number:
89-935

A COMPUTER VIRUS PRIMER

Eugene H. Spafford
Kathleen A. Heaphy
David J. Ferbrache

CSD TR-935
November 1989

A Computer Virus Primer[1]

Purdue University Technical Report CSD-TR-935

Eugene H. Spafford Kathleen A. Heaphy David J. Ferbrache

28 November 1989

Abstract

There has been considerable interest of late in computer viruses. Much of the information available is either of a highly theoretical nature, or describes a specific set of viruses. Neither is useful for providing an overview of how computer viruses work or how to protect against them.

This report is a condensed explanation of viruses—their history, structure, and some information on how to deal with their threat. It should provide a general introduction to the topic without requiring the understanding of excessive detail.

The interested reader is directed to the book from which this report is derived for further information, including references to related works and sources, more technical detail, and information on some of the legal aspects of computer viruses: *Computer Viruses: Dealing with Electronic Vandalism and Programmed Threats*, E. H. Spafford, K. A. Heaphy, and D. J. Ferbrache, ADAPSO, 1989.

Contents

1 What is a Computer Virus?

The term *computer virus* is derived from and analogous to a biological virus. The word *virus* itself is Latin for *poison*. Viral infections are spread by the virus (a small shell containing genetic material) injecting its contents into a far larger body cell. The cell then is infected and converted into a biological factory producing replicants of the virus.

Similarly, a computer virus is a segment of machine code (typically 200-4000 bytes) that will copy its code into one or more larger "host" programs when it is activated. When these infected programs are run, the viral code is executed and the virus spreads further. Viruses cannot spread by infecting pure data; pure data is not executed. However, some data, such as files with spreadsheet input or text files for editing, may be interpreted by application programs. For instance, text files may contain special sequences of characters that are executed as editor commands when the file is first read into the editor. Under these circumstances, the data is "executed" and may spread a virus. Data files may also contain "hidden" code that is executed when the data is used by an application, and this too may be infected. Technically speaking, however, pure data cannot itself be infected.

1.1 Worms

Unlike viruses, worms are programs that can run independently and travel from machine to machine across network connections; worms may have portions of themselves running on many different machines. Worms do not change other programs, although they may carry other code that does, such as a true virus.

In 1982, John Shoch and Jon Hupp of Xerox PARC (Palo Alto Research Center) described the first computer worms.[1] They were working with an experimental, networked environment using one of the first local area networks. While searching for something that would use their networked environment, one of them remembered reading *The Shockwave Rider* by John Brunner, written in 1975. This science fiction novel described programs that traversed networks, carrying information with them. Those programs were called *tapeworms* in the novel. Shoch and Hupp named their own programs *worms*, because in a similar fashion they would travel from workstation to workstation, reclaiming file space, shutting off idle workstations, delivering mail, and doing other useful tasks.

Few computer worms have been written in the time since then, especially worms that have caused damage, because they are not easy to write. Worms require a network environment and an author who is familiar not only with the network services and facilities, but also with the operating facilities required to support them once they have reached the machine. The Internet worm incident of November, 1988 clogged machines and networks as it spread, and is an example of a worm.

Worms have also appeared in other science fiction literature. Recent "cyberpunk" novels such as *Neuromancer* by William Gibson (1984, Ace/The Berkeley Publishing Group) refer to worms by the term "virus." The media has also often referred incorrectly to worms as viruses. This report focuses only on viruses as we have defined them.

[1] "The Worm Programs—Early Experience with a Distributed Computation," *Communications of the ACM*, 25(3), pp.172-180, March 1982.

1.2 Names

Before proceeding further, let us explain about the naming of viruses. Since the authors of viruses generally do not name their work formally and do not come forward to claim credit for their efforts, it is usually up to the community that discovers a virus to name it. A virus name may be based on where it is first discovered or where a major infection occurred, e.g., the *Lehigh* and *Alameda* viruses. Other times, the virus is named after some definitive string or value used by the program, e.g., the *Brain* and *Den Zuk* viruses. Sometimes, viruses are named after the number of bytes by which they extend infected programs, such as the *1704* and *1280* viruses. Still others may be named after software for which the virus shows an affinity, e.g., the *dBase* virus.

In the remainder of this report, we refer to viruses by commonly-accepted names. Appendix A gives further detail on many of these viruses, including aliases and particulars of behavior; Tables 2–4 list known virus names and aliases.

1.3 A history lesson

The first use of the term *virus* to refer to unwanted computer code occurred in 1972 in a science fiction novel, *When Harley Was One*, by David Gerrold. (The recent reissue of Gerrold's book has this subplot omitted.) The description of *virus* in that book does not fit the currently-accepted definition of computer virus—a program that alters other programs to include a copy of itself. Fred Cohen formally defined the term *computer virus* in 1983. At that time, Cohen was a graduate student at the University of Southern California attending a security seminar. The idea of writing a computer virus occurred to him, and in a week's time he put together a simple virus that he demonstrated to the class. His advisor, Professor Len Adelman, suggested that he call his creation a computer virus. Dr. Cohen's thesis and later research were devoted to computer viruses.

It appears, however, that computer viruses were being written by other individuals, although not named such, as early as 1981 on early Apple II computers. Some early Apple II viruses included the notorious "Festering Hate," "Cyberaids," and "Elk Cloner" strains. Sometimes virus infections were mistaken as trojan horses, as in the "Zlink virus," [sic] which was a case of the Zlink communication program infected by "Festering Hate." The "Elk Cloner" virus was first reported in mid-1981.

It is only within the last three years that the problem of viruses has grown to significant proportions. Since the first infection by the *Brain* virus in January 1986, up to August 1, 1989, the number of known viruses has grown to 21 distinctly different IBM PC viruses (with a further 57 minor variants; see Table 1). The problem is not restricted to the IBM PC, and now affects all popular personal computers (12 Apple Mac viruses and variants, three Apple II, 22 Atari ST and 18 Commodore Amiga viruses). Mainframe viruses do exist for a variety of operating systems and machines, but all reported to date have been experimental in nature, written by serious academic researchers in controlled environments.

Where viruses have flourished is in the weak security environment of the personal computer. Personal computers were originally designed for a single dedicated user—little, if any, thought was given to the difficulties that might arise should others have even indirect access to the machine. The systems contained no security facilities beyond an optional key switch, and there was a minimal amount of security-related software available to safeguard data. Today, however, personal computers are being used for tasks far different from those originally envisioned, including managing company

Table 1: The growth of the IBM PC virus problem (to August 1989)

Year	New	Viruses
1986	1	Brain
1987	5	Alameda, South African, Lehigh, Vienna, Israeli
1988	5	Italian, Dos 62, New Zealand, Cascade, Agiplan
1989	10	Oropax, Search, dBase, Screen, Datacrime, 405, Pentagon, Traceback, Icelandic, Mistake

databases and participating in networks of computer systems. Unfortunately, their hardware and operating systems are still based on the assumption of single trusted user access

The problem of viruses should be diminished considerably with the introduction of memory management (and protection), multiple users with compartmentalized environments, process privileges, and well-defined operating system interfaces. Operating systems that have recently become available for personal computers, such as IBM's OS/2 [2] and various versions of UNIX, [3] and using chips like the Intel 80386 and 80486, offer many of these facilities, but the problem of downward compatibility often exists. Furthermore, implementing enhanced operating systems and hardware is expensive and may mean the obsolescence of otherwise working equipment.

1.4 Formal structure

True viruses have two major components: one that handles the spread of the virus, and a manipulation task. The manipulation task may not be present (has null effect), or it may act like a logic bomb, awaiting a set of predetermined circumstances before triggering. We will describe these two virus components in general terms, and then present more specific examples as they relate to the most common personal computer: the IBM PC. Viruses on other machines behave in a similar fashion.

1.4.1 A Note About Mainframe Viruses

As we have already noted, viruses can infect minicomputers and mainframes as well as personal computers. Laboratory experiments conducted by various researchers have shown that any machine with almost any operating system can fall prey to a viral attack. However, there have been no documented cases of true viruses on large multi-user computers other than as experiments. This is due, in part, both to the greater restrictions built into the software and hardware of those machines, and to the way they are usually used. Anyone who can create a virus for a mainframe also can pursue other, more direct, forms of sabotage or disclosure that require less effort and present less risk than developing a virus. Our further comments will therefore be directed towards PC viruses, with the understanding that analogous statements could be made about mainframe viruses.

[2] OS/2 is a trademark of IBM Corporation.

[3] UNIX is a registered trademark of AT&T Technologies.

1.4.2 Structure

For a computer virus to work, it somehow must add itself to other executable code. The viral code must be executed before the code of its infected host (if the host code is ever executed again). One form of classification of computer viruses is based on the three ways a virus may add itself to host code: as a shell, as an add-on, and as intrusive code.

Shell viruses A shell virus is one that forms a "shell" (as in "eggshell" rather than "Unix shell") around the original code. In effect, the virus becomes the program, and the original host program becomes an internal subroutine of the viral code. An extreme example of this would be a case where the virus moves the original code to a new location and takes on its identity. When the virus is finished executing, it retrieves the host program code and begins its execution.

Add-on viruses Most viruses are add-on viruses. They function by appending their code to the end of the host code, or by relocating the host code and adding their own code to the beginning. The add-on virus then alters the startup information of the program, executing the viral code before the code for the main program. The host code is left almost completely untouched; the only visible indication that a virus is present is that the file grows larger.

Intrusive viruses Intrusive viruses operate by replacing some or all of the original host code with viral code. The replacement might be selective, as in replacing a subroutine with the virus, or inserting a new interrupt vector and routine. The replacement may also be extensive, as when large portions of the host program are completely replaced by the viral code. In the latter case, the original program can no longer function.

1.4.3 Triggers

Once a virus has infected a program, it seeks to spread itself to other programs, and eventually to other systems. Simple viruses do no more than this, but most viruses are not simple viruses. Common viruses wait for a specific triggering condition, and then perform some activity. The activity can be as simple as printing a message to the user, or as complex as seeking particular data items in a specific file and changing their values. Often, viruses are destructive, removing files or reformatting entire disks.

The conditions that trigger viruses can be arbitrarily complex. If it is possible to write a program to determine a set of conditions, then those same conditions can be used to trigger a virus. This includes waiting for a specific date or time, determining the presence or absence of a specific set of files (or their contents), examining user keystrokes for a sequence of input, examining display memory for a specific pattern, or checking file attributes for modification and permission information. Viruses also may be triggered based on some random event. One common trigger component is a counter used to determine how many additional programs the virus has succeeded in infecting—the virus does not trigger until it has propagated itself a certain minimum number of times. Of course, the trigger can be any combination of these conditions, too.

1.5 How do viruses spread?

Computer viruses can infect any form of writable storage, including hard disk, floppy disk, tape, optical media, or memory. Infections can spread when a computer is booted from an infected disk, or when an infected program is run. It is important to realize that often the chain of infection can be complex and convoluted. A possible infection might spread in the following way:

- A client brings in a diskette with a program that is malfunctioning (because of a viral infection).
- The consultant runs the program to discover the cause of the bug—the virus spreads into the memory of the consultant's computer.
- The consultant copies the program to another disk for later investigation—the virus infects the copy utility on the hard disk.
- The consultant moves on to other work preparing a letter—the virus infects the screen editor on the hard disk.
- The system is switched off and rebooted the next day—the virus is cleared from memory, only to be reinstalled when either the screen editor or copy utility is used next.
- Someone invokes the infected screen editor across a network link, thus infecting their own system.

1.6 The three stages of a virus's life

For a virus to spread, its code must be executed. This can occur either as the direct result of a user invoking an infected program, or indirectly through the system executing the code as part of the system boot sequence or a background administration task.

The virus then replicates, infecting other programs. It may replicate into just one program at a time, it may infect some randomly-chosen set of programs, or it may infect every program on the system. Sometimes a virus will replicate based on some random event or on the current value of the clock. We will not discuss the different methods in detail since the result is the same: there are additional copies of the virus on your system.

Finally, most viruses incorporate a manipulation task that can consist of a variety of effects (some odd, some malevolent) indicating the presence of the virus. Typical manipulations might include amusing screen displays, unusual sound effects, system reboots, or the reformatting of the user's hard disk.

1.6.1 Activating a virus

We will now describe how viruses are activated, using the IBM PC as our example. Viruses in other systems behave in similar manners.

The IBM PC boot sequence This section gives a detailed description of the various points in the IBM PC boot sequence that can be infected by a virus. We will not go into extensive detail about the operations at each of these stages; the interested reader may consult the operations manuals of these systems, or any of the many "how-to" books available.

The IBM PC boot sequence has six components:

- ROM BIOS routines
- Partition record code execution
- Boot sector code execution
- IO.SYS and MSDOS.SYS code execution
- COMMAND.COM command shell execution
- AUTOEXEC.BAT batch file execution

ROM BIOS When an IBM PC, or compatible PC, is booted, the machine executes a set of routines in ROM (read-only memory). These routines initialize the hardware and provide a basic set of input/output routines that can be used to access the disks, screen, and keyboard of the system. These routines constitute the basic input/output system (BIOS).

ROM routines cannot be infected by viral code (except at the manufacturing stage), since they are present in read-only memory that cannot be modified by software. Some manufacturers now provide extended ROMs containing further components of the boot sequence (e.g., partition record and boot sector code). This trend reduces the opportunities for viral infection, but also may reduce the flexibility and configurability of the final system.

Partition Record The ROM code executes a block of code stored at a well-known location on the hard disk (head 0, track 0, sector 1). The IBM PC disk operating system (DOS) allows a hard disk unit to be divided into up to four logical partitions. Thus, a 100Mb hard disk could be divided into one 60Mb and two 20Mb partitions. These partitions are seen by DOS as separate drives: "C," "D," and so on. The size of each partition is stored in the partition record, as is a block of code responsible for locating a boot block on one of the logical partitions.

The partition record code can be infected by a virus, but the code block is only 446 bytes in length. Thus, a common approach is to hide the original partition record at a known location on the disk, and then to chain to this sector from the viral code in the partition record. This is the technique used by the New Zealand virus, discovered in 1988 (see figures 1 and 2).

Boot sectors The partition record code locates the first sector on the logical partition, known as the boot sector. (If a floppy disk is inserted, the ROM will execute the code in its boot sector, head 0, track 0, sector 1.) The boot sector contains the BIOS parameter block (BPB). The BPB contains detailed information on the layout of the filing system on disk, as well as code to locate the file IO.SYS. That file contains the next stage in the boot sequence. (See Figure 3.)

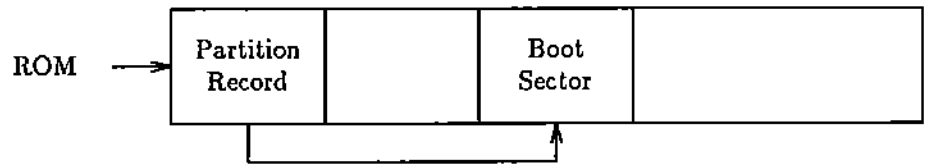

Figure 1: Hard disk before infection

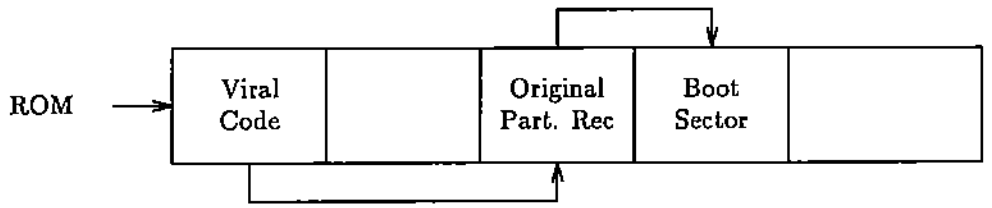

Figure 2: Hard disk after infection by New Zealand Virus

A common use of the boot sector is to execute an application program, such as a game, automatically; unfortunately, this can include automatic initiation of a virus. Thus, the boot sector is a common target for infection.

Available space in the boot sector is limited, too (a little over 460 bytes is available). Hence, the technique of relocating the original boot sector while filling the first sector with viral code is also used here.

A typical example of such a "boot sector" virus is the *Alameda* virus. This virus relocates the original boot sector to track 39, sector 8, and replaces it with its own viral code. (See Figure 4.)

Other well-known boot sector viruses include the *New Zealand* (on floppy only), *Brain*, *Search*, and *Italian* viruses. Boot sector viruses are particularly dangerous because they capture control of the computer system early in the boot sequence, before any anti-viral utility becomes active.

MSDOS.SYS, IO.SYS The boot sector next loads the IO.SYS file, which carries out further system initialization, then loads the DOS system contained in the MSDOS.SYS file. Both these files could be subject to viral infection, although no known viruses target them.

Command shell The MSDOS.SYS code next executes the command shell program (COMMAND.COM). This program provides the interface with the user, allowing execution of commands

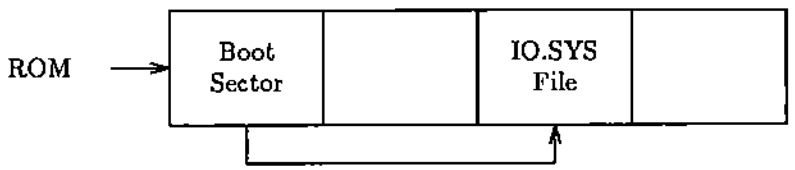

Figure 3: Floppy disk before infection

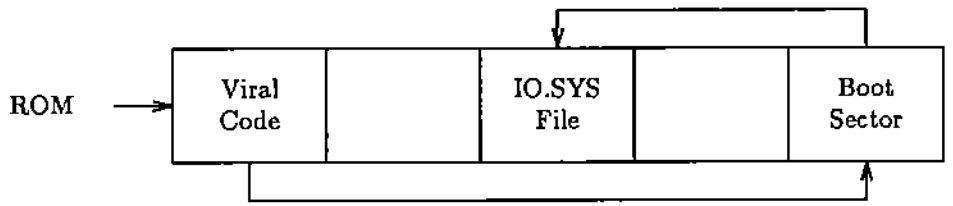

Figure 4: After Alameda Virus Infection

from the keyboard. The COMMAND.COM program can be infected, as can any other .COM or .EXE executable binary file.

The COMMAND.COM file is the specific target of the *Lehigh* virus that struck Lehigh University in November 1987. This virus caused corruption of hard disks after it had spread to four additional COMMAND.COM files.

AUTOEXEC batch files The COMMAND.COM program is next in the boot sequence. It executes a list of commands stored in the AUTOEXEC.BAT file. This is simply a text file full of commands to be executed by the command interpreter. A virus could modify this file to include execution of itself. Ralf Burger has described how to do just that in his book *Computer Viruses—A High Tech Disease*. His virus uses line editor commands to edit its code into batch files. Although a curiosity, such a virus would be slow to replicate and easy to spot. This technique is not used by any known viruses "in the wild."

Infection of a user program A second major group of viruses spreads by infecting program code files. To infect a code file, the virus must insert its code in such a way that it is executed before its infected host program. These viruses come in two forms:

Overwriting The virus writes its code directly over the host program, destroying part or all of its code. The host program will no longer execute correctly after infection.

Non-overwriting The virus relocates the host code, so that the code is intact and the host program can execute normally.

A common approach used for .COM files is to exploit the fact that many of them contain a jump to the start of the executable code. The virus may infect the programs by storing this jump, and then replacing it with a jump to its own code. When the infected program is run, the virus code is executed. When the virus finishes, it jumps to the start of the program's original code using the stored jump address. (See Figure 5.)

Notice that in the case of the overwriting virus, the more complex infection strategy often means that all but a small block of the original program is intact. This means that the original program can be started, although often it will exhibit sporadic errors or abnormal behavior.

Memory resident viruses The most "successful" viruses to date exploit a variety of techniques to remain resident in memory once their code has been executed and their host program has terminated.

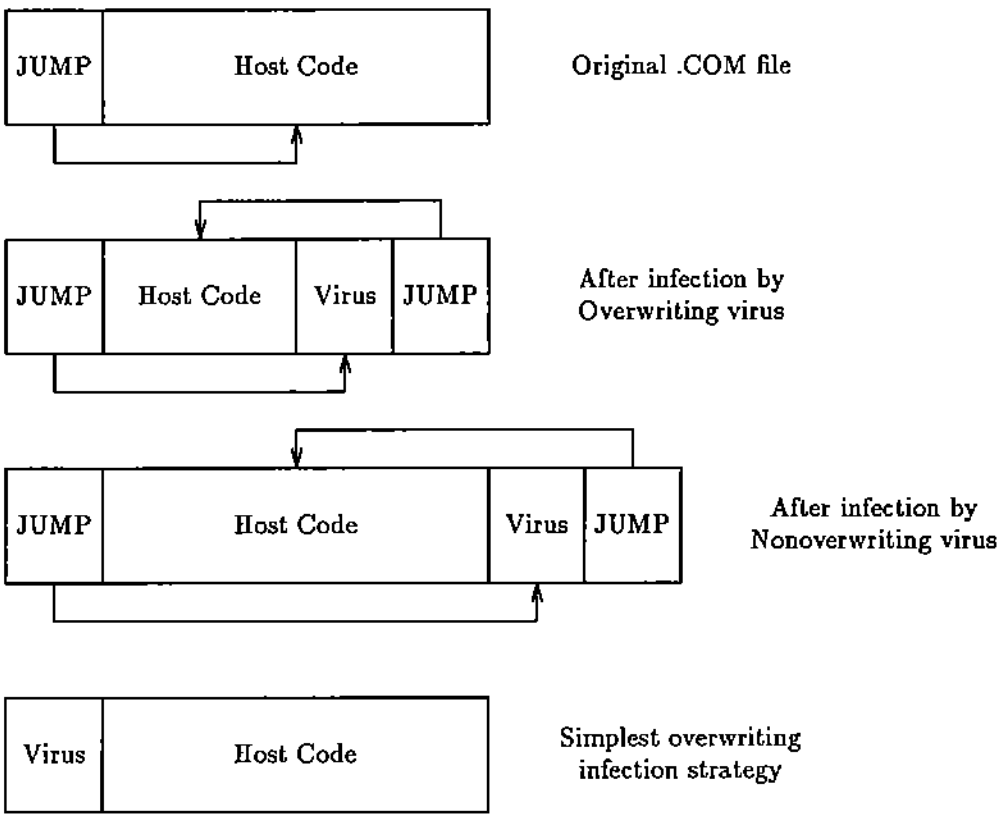

Figure 5: Infection of user applications

This implies that, once a single infected program has been run, the virus potentially can spread to any or all programs in the system. This spreading occurs during the entire work session (until the system is rebooted to clear the virus from memory), rather than during a small period of time when the infected program is executing viral code.

Thus, the two categories of memory-resident virus are:

Transient The viral code is active only when the infected portion of the host program is being executed.

Resident The virus copies itself into a block of memory and arranges to remain active after the host program has terminated. The viruses are also known as TSR (Terminate and Stay Resident) viruses.

Examples of memory resident viruses are all known boot sector viruses, the *Israeli*, *Cascade*, and *Traceback* viruses.

If a virus is present in memory after an application exits, how does it remain active? That is, how does the virus continue to infect other programs? The answer is that it also infects the standard

interrupts used by DOS and the BIOS so that it is invoked by other applications when they make service requests.

The IBM PC uses many interrupts (both hardware and software) to deal with asynchronous events and to invoke system functions. All services provided by the BIOS and DOS are invoked by the user storing parameters in machine registers, then causing a software interrupt.

When an interrupt is raised, the operating system calls the routine whose address it finds in a special table known as the *vector* or *interrupt* table. Normally this table contains pointers to handler routines in the ROM or in memory resident portions of the DOS (see figure 6). A virus can modify this table so that the interrupt causes viral code (resident in memory) to be executed.

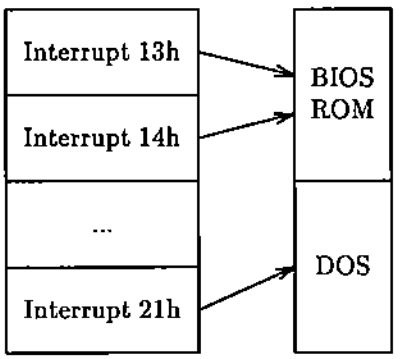

Figure 6: Normal interrupt usage

By trapping the keyboard interrupt, a virus can arrange to intercept the CTRL-ALT-DEL soft reboot command, modify user keystrokes, or be invoked on each keystroke. By trapping the BIOS disk interrupt, a virus can intercept all BIOS disk activity, including reads of boot sectors, or disguise disk accesses to infect as part of a user's disk request. By trapping the DOS service interrupt, a virus can intercept all DOS service requests including program execution, DOS disk access, and memory allocation requests.

A typical virus might trap the DOS service interrupt, causing its code to be executed before calling the real DOS handler to process the request. (See figure 7.)

1.6.2 Replication strategies

Types Viruses can be grouped into four categories, based on the type of files they infect:

- Boot sector viruses that only infect boot sectors (or rarely, partition records)
- System viruses that are targeted against particular system files, such as the DOS command shell
- Direct viruses that scan through the DOS directory structure on disk looking for suitable files to infect
- Indirect viruses that wait until the user carries out an activity on a file (e.g., execution of a program) before infecting it

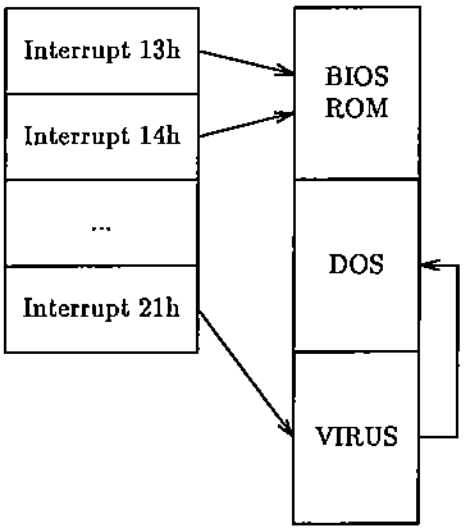

Figure 7: Interrupt vectors with TSR virus

Transient viruses are always direct in that they attempt to infect one or more files (usually in the same directory or home directory) before terminating. Resident viruses can be either direct or indirect (or worse, both). The recently reported *Traceback* virus infects any file executed (indirect), while also incrementally scanning the directory structure (direct).

In general, indirect viruses are slower to spread, but often pass unnoticed as their infection activities are disguised among other disk access requests.

Signatures to prevent reinfection One problem encountered by viruses is that of repeated infection of the host, leading to depleted memory and early detection. In the case of boot sector viruses, this could (depending on strategy) cause a long chain of linked sectors. In the case of a program-infecting virus (or link virus), repeated infection may result in continual extension of the host program each time it is reinfected. There are indeed some viruses that exhibit this behavior (e.g., the Israeli virus extends .EXE files 1808 bytes each time they are infected).

To prevent this unnecessary growth of infected files, many viruses implant a unique *signature* that signals that the file or sector is infected. The virus will check for this signature before attempting infection, and will place it when infection has taken place; if the signature is present, the virus will not reinfect the host.

A virus signature can be a characteristic sequence of bytes at a known offset on disk or in memory, a specific feature of the directory entry (e.g., alteration time or file length), or a special system call available only when the virus is active in memory.

The signature is a mixed blessing. The virus would be easier to spot if reinfections caused disk space to be exhausted or showed obvious disk activity, but the signature does provide a method of detection and protection. Virus sweep programs are available that scan files on disk for the signatures of known viruses, as are "inoculation" routines that fake the viral signature in clean

systems to prevent the virus from attempting infection.

1.6.3 Recognizing a viral infection (manipulation)

By reflecting on how viruses work, we can understand the causes of symptoms of a computer virus infection. A common symptom is a sudden change in the size of programs or files, or a sudden decrease in the amount of space available on your disks. This is caused as the viral code is copied into program files and to disk. A sudden increase in the number of sectors marked unusable or bad may indicate a virus that hides itself on disk. A reduction in available physical memory may signal the presence of a TSR virus.

A second common symptom of infection is odd behavior of system services. Resident viruses may not pass along system service requests correctly, or may alter those requests for their own purposes, thus leading to faulty behavior. Lost or garbled output to the screen or printers, corrupted images on the screen, or access to the disks that fail may signal a TSR virus—they also may signal a hardware problem or software bug. A system that suddenly seems slower may also signal the presence of a virus that is trapping service interrupts.

Since a virus needs to access disk to copy itself and to find new hosts to infect, excess or oddly-timed disk accesses can signal a viral infection. Newer viruses are more sophisticated in this regard as they piggyback their accesses on other, legitimate accesses.

A fourth and obvious symptom of a viral infection is the failure of some or all of your program to work normally. This occurs when the viral code overwrites your application, or when it botches the jumps or code changes necessary to infect the code. In particular, if your code behaves differently from machine to machine, or from hard disk to diskette, you should suspect a virus.

2 Dealing with Viruses

There are three components to a comprehensive technical policy against computer viruses: preventing them from infecting your software, detecting and containing them once they have entered your system, and recovering from an infection once viruses have been detected. A truly comprehensive policy will also address legal issues and user attitudes. These aspects are discussed in the book from which this article is derived.

2.1 Prevention

Preventing a viral infection is the best way to protect yourself against damage. If a virus cannot establish itself within your system, then it cannot damage your data or cause you to expend resources in recovery. Applying a few simple concepts can help keep your systems free of viruses.

2.1.1 Personnel

As with most security practices, personnel issues are often important—knowledge of the virus threat, together with a carefully planned and well-implemented anti-viral policy may be your best defense. It is crucial that users realize how much damage a viral infection can inflict, and that they are encouraged to take an active part in ensuring that this does not occur. If users are aware of the

losses they can incur, they are more likely to exercise caution in situations that might admit a virus to their computers.

It is especially critical that everyone with access to systems you wish to protect are educated about viruses. The executive vice president who runs a game program from a diskette brought from home is as much a potential source of a virus as are the members of the programming staff (perhaps even more so).

2.1.2 Policies

There should be a prepared policy that addresses at least the following issues:

- Use of foreign software obtained from bulletin boards and external organizations.
- Preventing viral infection with anti-viral software. This should include designation of individuals responsible for maintaining and running these utilities, especially in active development environments and those containing company-critical databases.
- Reporting of viral infection: to whom, the chain of command, format of reports, and follow-ups.
- Control of viral infection once detected—who is involved, what procedures are to be taken, when should outside help be obtained, under what circumstances are legal authorities notified.
- Recovery from infection, including backup and dump policies, and damage assessment.
- Education and information—what training will be provided to users, what sources of information will be used to keep current on new viral threats, and who will serve as a point of "rumor control" if a viral infection occurs.

This policy should be formalized and circulated in full to all users who might be exposed to computer viruses. The important point is that the policy should be flexible enough to deal with a variety of scales of infection, but concrete enough to allow rapid response. Furthermore, it should not be so restrictive or unreasonable that it encourages users to ignore it or seek shortcuts. The purpose of the policy is to establish sound guidelines for *everyone* in the organization without hampering their ability to do their work.

2.1.3 Sharing software

For viruses to spread they must have a transport medium—either an electronic network or a manual data exchange such as a disk or tape. One obvious action that can be taken is to restrict the use of foreign software to that obtained from reputable sources. In particular, the running of unnecessary games software (often copied and from an unknown source) should be discouraged. Important software should be obtained only from reputable sources.

Defining what constitutes a "reputable source" is difficult to do. Most software houses and major bulletin boards can be expected to have adequate in-house virus screening procedures, but this is not universal. However, viruses have been distributed in "shrink-wrapped" software provided by major companies. The size and reputation of a software source is not, by itself, a guarantee that the software is virus-free (or bug-free). The best policy is therefore to restrict incoming software

to what is necessary for the current project. The source of the software should always be known definitively.

2.1.4 Quarantine

It may be helpful to establish a quarantine station to screen incoming software. The quarantine machine should be segregated both physically and electronically from any other machines in the organization. This machine is used to test incoming software for a period of time, normally between a week and a month. The following suggestions can help make a quarantine station successful:

- Do not allow network connections except to other quarantine stations. (Some viruses may only manifest themselves over network links.)
- The entire hard disk should be reformatted at the end of each quarantine session.
- The station should have the same hardware and software configuration as the production systems for which the quarantined software is destined.
- The quarantine station should have a full suite of anti-viral software available.
- You may wish to isolate the station physically to prevent accidental use or contamination.
- *Clearly* identify any magnetic media ever used in the quarantine station. Brightly colored stickers and warning messages are highly recommended.
- Designate a small group of individuals who are responsible for the operation of the quarantine operation. They should be familiar with the proper functioning of the system so they can recognize a virus, if present.
- Keep a log of events related to operation of the quarantine station. This includes reformats, introduction of new software, the execution of anti-viral software, and the occurrence of unusual activity on the system.
- Run the system with an advanced clock, if possible, to trigger viruses that will remain dormant until some predetermined date.

It may be sufficient to run anti-viral software on the user's own machine, without the overhead that quarantine implies. However, you gain an extra measure of safety by running a well-designed program of quarantine and examination.

2.1.5 Diskless nodes

Another approach to preventing viral infection is to control the access to infectable media. One way to do this is to equip users with diskless machines connected via a network to a central file server. More specifically, these machines should not have any removable media such as diskettes or tape. They can be equipped with a hard disk so long as the disk cannot be removed. The file server is closely administered and users are not allowed to introduce new software to the central site, except through a regulated mechanism, such as a quarantine station.

2.1.6 Guard diskettes

Never routinely use original diskettes to load software (these are too valuable to risk corruption). Instead, make copies using a clean machine and clean diskettes. Write-protect and clearly label your copies, making sure that both they and the originals are kept in a safe place.

Do not share diskettes. If necessary, copy them using a clean machine and exchange those copies. Be sure to reformat returned diskettes before loading them in your own machine. Do not use your master diskettes in foreign machines, either—they can be infected and spread a virus to your machine the next time you use them.

Some PCs use optical sensors to determine whether the write-protect tab is in place on a diskette. Do not use cellophane tape or any other clear material to cover the slot—it will not work reliably and your disk may become infected.

2.1.7 Unusual symptoms

When strange behavior occurs, do not dismiss it as simply a bug. Instead, suspect a virus and respond accordingly—acting quickly may save your data. The following are possible symptoms of a viral infection:

- Strange screen graphics or displays
- Unexpected musical tones or sound effects
- Alteration of text or commands
- Unusual behavior on reboot
- Reduction in system performance
- Unexpected disk access patterns
- Changes in file length or alteration times
- Bugs in previously reliable software
- Bad sectors on floppy disks, or unusually large numbers of bad sectors on hard disks
- Reduction in available memory
- Unexplained changes in the system clock
- Unknown, new files or directories/folders appearing on disk
- Problems in time-dependent tasks such as communications or printing
- The system will not reset or reboot

Keep a log of such symptoms. This will be useful later for tracing or identifying any infection.

2.1.8 Segregation

Segregate your source code from binaries, preferably on a separate disk partition. If you do your development work on a separate machine from final production, you not only help to prevent cross-infection, but also preserve files against damage from any form of calamity.

2.1.9 Anti-viral software

Use anti-viral software on your systems. Many forms of virus prevention and detection programs are available for personal computers, including some good public domain and shareware programs that can be used at little cost. These programs may not detect or stop every virus that could infect your systems, but the added safety they provide against common viruses can be significant.

2.1.10 Hiding files

Some minimal protection can be obtained by encrypting or hiding selected files. If files that would be the target of a virus infection are not where a virus expects, or are in a format that the virus code does not understand, then it (usually) cannot infect them. This may work for a few files, but the approach is unworkable for most. It does, however, suggest a method of storing backup versions of important files that can be used in the event of an infection.

2.2 Detection of viral infection

Despite your efforts to prevent viral infection, a virus can still enter your system. When this occurs, it is critical to locate and remove it as quickly as possible, before it damages your data or programs.

2.2.1 Symptoms

The simplest method of detecting a virus is to observe its effect on the system as it replicates or executes its "logic bomb" or manipulation task. From the descriptions in Appendix A and Table 5, it can be seen that many viruses have specific effects on certain dates or periods of the year. A common technique used to provide advanced warning of destructive effects is to set the system clock ahead of real time. Designating one production system to run with its clock set a week ahead to monitor such effects (e.g., a disk reformat) can provide enough opportunity to analyze the infection and act appropriately.

Table 5 lists dates known to be of special significance to some viruses.

2.2.2 Checksums

The principal technique used in detecting alteration of program files or boot sectors is to generate a numerical value that is a function of the contents of the file or sector. Such a value is termed a checksum, and ideally will be sensitive to any alteration (however minor) to the contents of the file.

The checksum process is carried out by executing a checksum generation utility over the files in the clean system, resulting in checksums for all important files. These values then are stored for later reference.

Viral infection (and file alteration) is detected by regenerating checksums at regular intervals and then comparing the values to the original checksums produced on the clean system. Any change in file contents will be reflected in a change in the checksum value. Verifying the checksum of a binary file just before its execution is a good way to ensure that the file is not infected. Comprehensive checks can take significant amounts of time, although checksum testing of partition record, boot sector and COMMAND.COM can be carried out rapidly and regularly (perhaps by a resident program).

A virus can be designed to circumvent a particular checksum program by arranging dummy code so that the infected file has the same checksum as the original. This can be prevented by using multiple checksum algorithms, or by modifying the algorithm in a manner difficult to predict. Certain cryptographic algorithms can be used to generate checksums that are sufficiently complex that they are almost impossible to forge. In practice, because of the large variety of checksum programs available, it is unlikely that a virus writer would target his virus to defeat a particular system.

2.2.3 Access monitors

A second major class of anti-viral utilities is the disk access monitor. These programs take the form of resident utilities that intercept all disk access requests, and inform the user of any potentially suspect requests. Such requests include reads or writes directly to sectors, and writes to .EXE or .COM code files.

Returning to the interrupt table described earlier, a disk access monitor functions by intercepting a number of BIOS and DOS interrupts. (See Figures 8 and 9.)

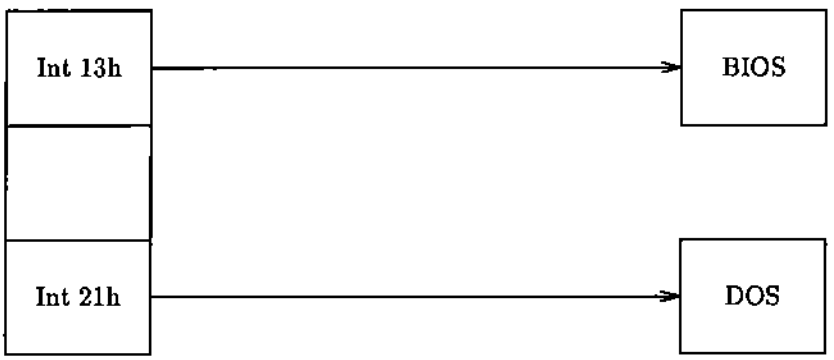

Figure 8: Standard interrupts

An example of such an access monitor is Ross Greenberg's FluShot+ utility that will display an "alert window" if such a suspect access is attempted by a program. (See Appendix B.)

The difficulty with access monitors is their inability to trap BIOS disk accesses by boot sector viruses. A boot sector virus becomes active before the monitor utility, and therefore can take a copy of the BIOS interrupt handler address itself. The monitor then installs itself later in the boot sequence, redirecting the interrupt handler. The virus's disk activity therefore bypasses the monitor utility entirely. However, other boot sector viruses can be trapped, as they do not store the BIOS

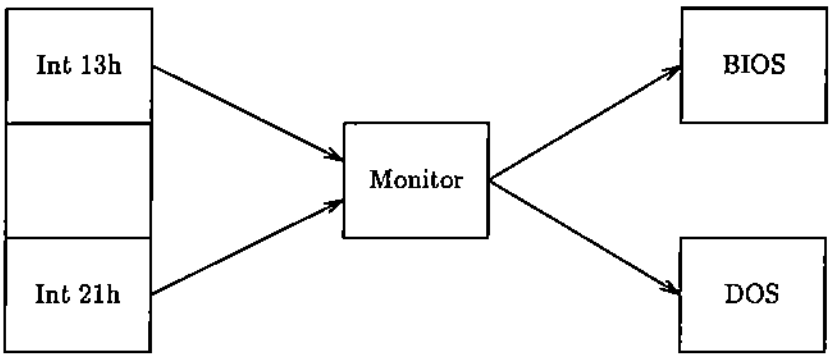

Figure 9: Interrupts with access monitor

handler address (making a direct call), but use the more conventional interrupt mechanism. (See Figures 10 and 11.)

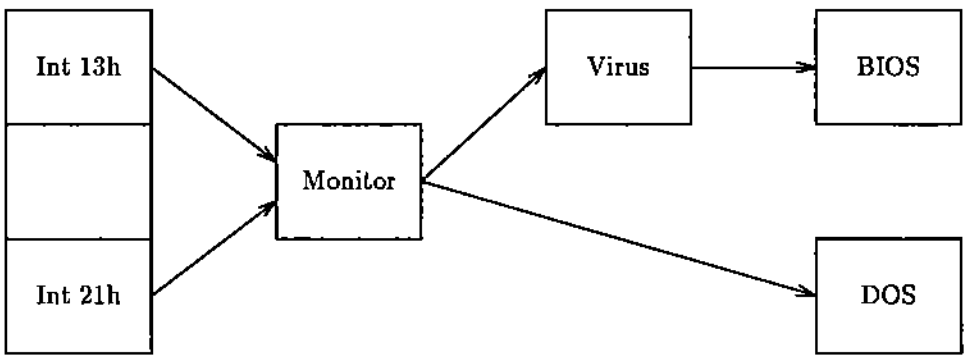

Figure 10: Interrupts and monitor with boot virus

Similar principles extend to Macintosh systems, where the monitor may intercept the resource manager traps that a virus would use to add resources to existing code or system files (e.g., VACCINE).

2.2.4 Vector table monitors

A second type of resident utility (often combined with access monitors) detects any changes that occur in the table of interrupt vectors, such as might be caused by a link virus installing itself in memory. Such changes are suspect, but may be caused by a legitimate memory-resident utility.

2.2.5 Detection by signature

The final class of detection software is designed to detect the characteristics of a specific or generic virus. This software is divided into three types:

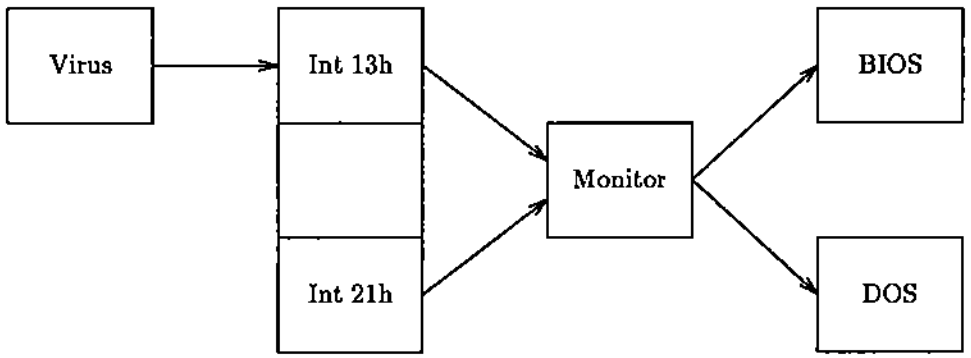

Figure 11: Boot virus detected by monitor

- Generic virus detectors. These detectors search code files for sequences that might indicate a virus, including reads or writes to sectors, disk format instructions, directory search instructions, terminate and stay resident commands, and suspect strings or file names. Recognition of such sequences is often complicated by unusual coding strategies or encryption of viral code.

- Specific virus detectors. These detectors search for code sequences that are known to exist in files infected by a specific strain of virus. Often these signatures may be those used by the virus itself to avoid reinfection.

- DOS signature call. These detectors monitor DOS calls that are added by known viruses, and they signal the user if they detect one of these calls. They also may return the correct recognition sequence to prevent the virus from infecting system memory, thus exercising an inoculation function.

Specific virus detectors are most common in the Macintosh environment, where they are often based on detection of the characteristic resource names and numbers added by viruses, such as nVIR, AIDS, MEV# and Hpat. Newer, non-resource adding viruses, such as ANTI, require searches for specific code sequences. These searches are often slow and cumbersome for large disk partitions.

2.3 Recovery

Once you have detected a virus, it is necessary to remove all traces of it from your system. You also need to remove it from any backup copies you may have made while the virus was present. Finally, as a responsible user, you should notify other users with whom you may have shared infected diskettes so that they may check for infection on their systems.

2.3.1 Link virus disinfection

Manual repair of infected programs is not recommended. While it is possible to restore the program code to its original condition, this process is not simple and may result in further damage to your

program. Disinfection utilities are available for most viruses; see Appendix B for a partial list. These programs can deactivate a virus present in a program and restore the original program code.

Disinfection utilities rely on their ability to identify a specific strain of virus, and then to apply detailed knowledge of the infection method to restore the original program. In the case of viruses that rely on the host commencing with a jump instruction, this can be as simple as restoring the original jump to the start of the host code.

2.3.2 Boot sector disinfection

It is possible for a non-specialist to disinfect a PC infected with a boot sector virus. This can be achieved with the *sys* utility that writes a new boot sector (together with new copies of the IO.SYS and MSDOS.SYS files) to the disk. Some virus-repair utilities can also remove specific viruses from the boot sectors.

2.3.3 Using backups

A regular backup policy is crucial to recovery from viral attack (as well as recovery from many other disasters). Following an infection by a link virus, it is possible to restore the system from a backup disk or tape. To do so, first make sure there are no viruses still active on the system. Next, place a clean, protected diskette containing copies of your restore and format utilities in your drive. Format and clean the disk onto which you are doing the restore.

At this point in the process, you also need to verify that no viruses exist on your backup. If the backup was a simple copy to a diskette, you can run anti-viral software on that diskette. If it was a selective backup, you need to restore the files and then run the anti-viral software. If you discover contaminated software, use an older backup.

To do this, the user should prepare (in advance) a disk containing a selection of anti-viral software and necessary utilities. This disk should be write-protected and stored with a clean system disk in a safe place.

3 Summary

Viruses are not necessarily complex, nor are they difficult to control. The most effective method of dealing with viruses is to use common sense. Once you understand how they operate and the damage they can do, taking some simple precautions can provide at least as much protection as many of the software and hardware products you can buy. At the same time, putting too many precautions in place can be a waste of resources and a nuisance—your exposure to viruses may already be limited, and too many precautions may be unnecessary.

It is important that you balance the risk of loss against the cost and effort involved in any planned anti-virus policy. A further concern is that you do not focus so much attention on computer viruses that you neglect precautions against other sources of loss—including trojan horses, crackers, buggy software, and disgruntled users. Computer viruses are only one form of threat to the security of your computers, and they may not be the most dangerous. Protecting your systems against the broader range of threats will result in some protection against viruses, too.

A Further Information on Viruses

A.1 Names of Known Viruses

Tables 2 and 4 list all known computer viruses affecting IBM-compatible PCs and Apple Macintosh computers (as of 15 August 1989). The page where each is described in detail is also given, as is the type of the virus.

A.2 Trigger Dates

Table 5 gives dates known to be special to selected viruses.

A.3 Known IBM PC viruses by characteristics

This section provides details on common IBM PC viruses, current as of 15 August 1989. The descriptions are divided into five groups, namely:

- Boot sector and partition record infectors
- System file infectors
- Overwriting .COM/.EXE viruses
- Non-overwriting transient .COM/.EXE viruses
- Non-overwriting resident .COM/.EXE viruses

A.3.1 Boot sector and partition record viruses

2730 Virus Boot sector virus reported in 1989, no details available to date.

Alameda Virus Aliases: Merritt, Yale, Peking, Seoul

The Alameda virus is a simple virus that first appeared at Merritt College in California in 1987. The virus infects by copying the original boot sector to a fixed location on the floppy disk, and then replacing it with viral code. As a result, the virus destroys any data stored on head 0, track 39, sector 8 when the original boot sector is relocated.

When the system is booted from an infected disk, the virus reserves 1K at the top of system memory and installs its code. The virus spreads only when a soft reboot is attempted via the CTRL-ALT-DEL keyboard sequence. This sequence is intercepted by the virus, which then simulates the soft reboot sequence using appropriate mode changes and delays. During this simulated reboot, the virus will infect the floppy being used to boot the system.

The virus will infect only drive A; it also contains an undocumented "POP CS" instruction that prevents its operation on 80286 or 80386 processors.

There are 9 common variants of the Alameda virus, including the "Golden Gate" strains.

Table 2: Catalog of IBM PC Viruses

Name	Identifier	Type	Page
2730	2730 virus	Boot sector	21
405	405 virus	Overwriting	26
Agiplan	Agiplan virus	Memory resident	27
Alameda	Alameda virus	Boot sector	21
April 1st	April-1-COM/EXE	Memory resident	27
Ashar	Brain variant		
Austrian	Vienna alias		
Autumn leaves	Cascade alias		
Basit	Brain alias		
Black hole	Israeli alias		
Blackjack	Cascade alias		
Bouncing ball	Italian alias		
Brain	Brain virus	Boot sector	25
Cascade	Cascade virus	Memory resident	28
Century	Israeli variant		
Clone	Brain variant		
Datacrime	Datacrime virus	Transient	27
Dbase	Dbase virus	Memory resident	28
Den Zuk	Search alias		
Disk eating	Icelandic alias		
DOS-62	Vienna alias		
Falling tears	Cascade alias		
Friday the 13th	South African alias		
	Israeli alias		
FuManchu	FuManchu virus	Memory resident	28
Golden Gate	Alameda variant		
Hebrew University	Israeli alias		
Icelandic	Icelandic virus	Memory resident	28
Israeli	Israeli virus	Memory resident	29
Italian	Italian virus	Boot sector	25
Jerusalem	Israeli alias		
Jork	Brain variant		
Lahore	Brain alias		
Lehigh	Lehigh virus	System File	26
Marijuana	New Zealand alias		
Mazatlan	Alameda alias		
Merritt	Alameda alias		
Miami	Brain alias		

Table 3: Catalog of IBM PC Viruses (continued)

Name	Identifier	Type	Page
Missouri	Alameda alias		
Mistake	Mistake virus	Boot sector	25
Music	Oropax alias		
New Jerusalem	Israeli variant		
New Zealand	New Zealand virus	Boot sector	25
Nichols	Nichols virus	Boot sector	26
One-in-eight	Vienna alias		
One-in-ten	Icelandic alias		
Oregon	Israeli alias		
Oropax	Oropax virus	Memory resident	29
PLO	Israeli alias		
Pakistani	Brain alias		
Pecking	Alameda alias		
Ping Pong	Italian alias		
Russian	Israeli alias		
SF	Alameda variant		
Sacramento	Alameda variant		
Saratoga	Icelandic alias		
Screen	Screen virus	Memory resident	29
Search	Search virus	Boot sector	26
Second Austrian	Cascade alias		
Seoul	Alameda alias		
Shoe	Brain variant		
South African	South African virus	Transient	27
Sys	Search alias		
Traceback	Traceback virus	Memory resident	30
Typo	Mistake alias		
UIUC	Brain alias		
UNESCO	Vienna alias		
Venezuelan	Search alias		
Vera Cruz	Italian alias		
Vienna	Vienna virus	Transient	27
Yale	Alameda alias		
sUMsDos	Israeli alias		
sURIV 1.01	April 1st COM		
sURIV 2.01	April 1st EXE		
sURIV 3.00	Israeli variant		

Table 4: Catalog of Apple Macintosh Viruses

Name	Identifier	Page
AIDS	nVIR variant	
Aladdin	-	
Aldus	Peace alias	
Anti	Anti virus	30
Brandow	Peace alias	
Drew	Peace alias	
Dukakis	Dukakis virus	30
Eric	Scores alias	
Frankie	-	
Hpat	nVIR variant	
Hypertext avenger	Dukakis alias	
Init 29	Init 29 virus	30
MacMag	Peace alias	
Mev#	nVIR variant	
NASA	Scores alias	
nFLU	nVIR variant	
nVIR	nVIR virus	30
Peace	Peace virus	31
Scores	Scores virus	31
Vult	Scores alias	

Table 5: Trigger dates for selected viruses

Date	Virus	Effect
13th October onward any year	Datacrime	Message and disk format
Friday the 13th any year	South African	File deletion
	Israeli	File deletion
April 1st	April-1-COM	Lock up system
	April-1-EXE	Lock up system
March 2nd 1988	Peace	Message and self-deletion
October-December 1988	Cascade	Cascade display
December 5th 1988 onwards	Traceback	Direct file infection
December 28th 1988 onwards	Traceback	Cascade display
August 1989 onwards	FuManchu	Character substitution
Friday the 13th 1990 or later	Jerusalem-D	Destroys FATs
Friday the 13th 1992 or later	Jerusalem-E	Destroys FATs
1st January 2000	Century	Destroys FATs and sectors

Brain Virus Aliases: Lahore, Pakistani, Basit

The Brain virus apparently originated in Lahore, Pakistan in January 1986. Some variants of this virus include the name, address, and telephone number of the alleged authors, namely Basit and Amjad, Brain Computer Services, Lahore, East Pakistan. The virus is a boot sector infector infecting only $5\frac{1}{4}$-inch floppy disks.

When a disk is infected, the Brain virus marks three consecutive clusters—six sectors in the file allocation table (FAT)—as bad. With this space now reserved from allocation, the virus copies the original boot sector to the first "bad" sector, replacing it with its own code. The remaining sectors contain further viral code.

When the system is booted from an infected floppy, the virus will reduce the available system memory by 7K, using the free area to install its code. The virus traps the BIOS disk interrupt and will attempt to infect any floppy disk in drive A or B when a read operation is attempted. The BIOS interrupt is also used to camouflage the presence of the virus by ensuring that any read of the disk boot sector returns the original stored version of the sector rather than the real boot sector.

The virus will modify the volume label of any infected diskette to be "(c) Brain". The virus has no known destructive effects other than reducing available disk space and memory.

There are 10 known variants of the virus, including one that will damage disk contents if activated after May 5, 1992.

Italian Virus Aliases: bouncing ball, ping pong, Vera Cruz

The Italian virus, reported in March 1988, is renowned for its unusual screen display. This display consists of a circular ball character that bounces around the screen, traveling through any text encountered. The text is restored after the ball has passed. The screen display is rare and is activated only when a disk read occurs in a small activation window (approximately one second per half hour).

The virus infects both floppy and hard disk boot sectors, marking a free cluster as bad in the FAT table. The original boot sector is then copied to the bad cluster, while viral code is installed in the boot sector.

When a system is booted from an infected boot sector, the virus will copy its code to 2K of reserved system memory. The virus then intercepts the BIOS disk interrupt and installs itself whenever a disk read is attempted.

The virus contains an invalid instruction that prevents operation on 80286 and 80386 processors.

Mistake Virus The recently-reported Mistake virus was discovered in June 1989 in Israel. This virus is a boot sector virus that installs itself in the top 2K of system memory. The virus intercepts printer interrupts, causing substitution of certain characters in printed text. Replacement occurs with a similar sounding letter (e.g., K by C), with the replacement extended to the extended ASCII codes for the Hebrew character set. Numeric strings are also replaced.

New Zealand Virus Aliases: Australian, Marijuana, Stoned

The New Zealand virus (first reported in Wellington) is unusual in that it is a partition record virus on hard disks, as well as a boot sector virus when on floppy disks. When the virus infects, it

relocates the original partition record/boot sector to a fixed location on the disk, replacing it with viral code.

When the system is booted from an infected disk, the virus will copy its code into 2K of reserved memory. The virus will occasionally display the message "Your PC is now stoned! Legalise Marijuana" when the system is booted from an infected floppy disk.

Booting from an infected floppy will also cause infection of the hard disk. The virus causes no damage other than destroying the contents of head 0, track 0, sector 7 on hard disks, and head 1, track 0, sector 3 on floppy disks. This is done during the copy of the original boot sector.

There are three variants of this virus known.

Nichols Virus Boot sector virus reported in 1989, no details available to date.

Search Virus Alias: Venezuelan, Den Zuk

This virus will infect floppy disk drives A or B every second time an attempt is made to read, write, verify, or format the information at track 0, head 0. The virus stores its code in the boot sector and in nine sectors commencing at sector 33 of track 40. This track is not normally used on floppy disks but is specially formatted by the virus. Many disk scanning utilities are unable to access information stored on track 40, and hence much of the virus may go undetected.

When active, the virus intercepts the CTRL-ALT-DEL soft reboot sequence, causing display of a striking "DenZuk" graphic to be displayed in large red letters on the screen of systems with C/E/VGA adapters. The CTRL-ALT-F5 sequence will cause a reboot if the virus is active.

The virus contains an infection count that is incremented on each new successful infection. When the counter exceeds two, the graphic is displayed and the disk label on any infected disk is changed to Y.C.1.E.R.P.

The virus will replace occurrences of the Brain virus on disk. There are six known variants of this virus.

A.4 System File Viruses

Lehigh Virus Aliases: none

The Lehigh virus was discovered in Autumn 1987; it infects only COMMAND.COM files. The date of the COMMAND.COM file is not restored after infection and can be used to detect the virus. The virus destroys the FAT table after infecting four further files.

A.5 Overwriting viruses

405 Virus The 405 virus infects one .COM file in the current directory in the next drive from current, using the sequence A, B, C, A. The virus overwrites the first 405 bytes of the infected file, or extends it to 405 bytes if shorter. The signature mechanism is faulty and multiple reinfections can occur.

A.6 Non-overwriting Viruses

A.6.1 Transient viruses

Datacrime virus Aliases: none

The Datacrime virus appends itself to a .COM file, causing an extension of 1168 bytes. When invoked, the virus will search the directory structure on drives C, D, A, and B (in that order) for an uninfected .COM file. This file is then infected. The COMMAND.COM file is explicitly excluded from infection.

If the virus is executed on or after October 13th, it will display the message "Datacrime virus, released 1 March 89" and then will format heads 0 through 8, track 0 of the hard disk. This format will destroy critical information including the partition table, boot sectors, and FATs.

There are two variants of this virus, one which extends the infected file by 1168 bytes, and one that extends the infected file by 1280 bytes. The behavior of the two is similar.

South African Virus Alias: Friday the 13th, COM virus, 512 virus

This virus first appeared in South Africa in 1987. When executed, it will attempt to infect two .COM files on the C hard drive and one on the A floppy drive. If invoked on Friday the 13th, the virus will delete its host program. Infected files are extended by 512 bytes.

Vienna virus Alias: Austrian, 648, one-in-eight

This virus was reported in London in Autumn 1988 and was published recently in disassembled form. It will infect a .COM file in the current search path when executed. The virus appends its code to the host, causing a 648-byte extension.

Files are marked as infected by setting the seconds field of the alteration time in the directory entry to 31 (the alteration times are stored in two second units, thus 31 would correspond to 62—an invalid value). There is a two in 15 chance that the virus will zero the start of the host file when infecting.

There are three known variants of this virus, differing only in the nature of the manipulation task.

A.6.2 Memory resident

Agiplan virus This virus was first reported in July 1988 in a German newspaper. The virus infects .COM files, extending their length by 1536 bytes. Reports indicate that after a four-month incubation period, the virus modifies write operations. After six months, the virus formats tracks 0-3 of any active disks, thus destroying the boot sectors, FAT tables, and directory entries.

April 1st viruses The predecessors of the Israeli virus include two viruses that have April 1st as their activation date. The first, the April-1st-COM virus, infects .COM files, causing an extension of 897 bytes. The virus installs itself in memory when run, thereafter infecting each .COM file executed. If the date is April 1st, the virus displays the message "April 1st ha ha ha you have a virus" after spreading to a .COM file. The virus then locks up the computer.

The April-1st-exe virus is similar to the above strain in that it will display the message when infecting memory, then cause a system lock-up. In addition, the virus will cause a lock-up one hour after infection of memory when the system is set to the default date of 1-1-80.

Infected .EXE files are extended by 1488 bytes.

Cascade virus Alias: blackjack, autumn leaves, falling tears, 1701, 1704

The Cascade virus was named for its distinctive screen display, which consists of letters on the screen detaching and "falling" slowly down the screen until they strike another object. This screen display was produced in the months of October, November, and December of 1988.

When an infected .COM file is executed, the virus installs itself in memory, and thereafter will infect any .COM file executed, resulting in an extension of 1701 or 1704 bytes.

The virus tests (incorrectly) for IBM proprietary BIOS code and attempts to avoid infecting true IBM machines. The virus code is encrypted in a manner dependent on the host file length, thus preventing text and code sequences from being recognized.

There are six variants of this virus.

Dbase virus The dBase virus is specifically targeted at the dBase application and its data files. When the virus is executed, it installs itself in memory and then infects all .COM and .EXE files in the current directory. The virus intercepts any *open* calls to files with .DBF extensions. When such a file is opened, the virus will copy the file handle and file length into its code space. When the user subsequently accesses the database file, the virus will transpose the first two bytes of any appends to the .DBF file. The location of the transposed bytes is stored in a BUG.DAT hidden file. The virus will transpose any subsequent reads of these bytes, thus returning correct data. When the virus is eliminated from the system, this second transposition will not be carried out, and the database appears corrupted.

FuManchu virus The FuManchu virus, discovered in the United Kingdom in May 1989, is a modified version of the Israeli virus. The virus infects both .COM files (prepending its code, causing a 2086-byte extension) and .EXE files (appending itself, causing a 2080-byte extension).

There is a random chance (one in 16) that, on infection, the virus will delay between 30 minutes and seven hours before displaying "The world will hear from me again," and then reboots the machine. The CTRL-ALT-DEL causes the same message to be displayed, but does delete the virus from memory.

The virus also triggers after August 1989, and begins to monitor the keyboard buffer for certain key words ("Thatcher," "Reagan," "Botha," and "Waldheim," two expletives, and the phrase "Fu Manchu"). The virus then expands the key words by adding additional text to the buffer. This additional text consists of derogatory statements about the political figures, deletion of the expletives, and addition of the text "virus 3/10/88 latest in the new fun line!" after the words FuManchu.

Icelandic virus Alias: saratoga, one-in-ten, disk-eating

The Icelandic virus is a .EXE infector, reported in June 1989. It installs itself in memory when an infected program is executed, first by allocating 2K of memory and then disguising that block as part of the memory allocated to the operating system. Every tenth program executed is checked—if

it is an uninfected .EXE file, then the virus code is appended to it, infecting it. At the same time, the virus will read the FAT and search for a free cluster, starting at the end. When it finds a free cluster, it is marked bad, thus reducing available disk space.

Israeli Virus Alias: Jerusalem, Hebrew University, PLO, Friday the 13th, Russian 1808, 1813, 1792, sUMsDos, WordPerfect

The Israeli virus infects by appending its code to a .COM file, causing an extension of 1813 bytes, or prepends its code to .EXE files, causing an extension of 1808 from the length stored in the .EXE header.

The virus installs itself in memory when executed, and thereafter will infect any program executed (except COMMAND.COM). 30 minutes after system infection, in any year except 1987, the virus will show two symptoms:

- A window at row 5 column 5, to row 16 column 16 is scrolled by two lines, causing a black rectangle to appear.
- A delay loop is inserted into the system time interrupt, causing a slowdown (up to a factor of 10) of the entire computer system.

The .EXE infection mechanism is faulty and will fail to recognize the virus signature, causing reinfection to occur.

The Israeli virus is specifically targeted to activate on Friday the 13th of any year except 1987. On this date it will not demonstrate the window or slowdown effect, but will delete any program executed.

There are twelve known variants of this virus, differing in effect and trigger dates.

Oropax virus Alias: Music virus

The Oropax virus was reported at the University of Hamburg in February 1989. This virus infects only .COM files, causing their length to be extended by 2756 to 2806 bytes, with the end length being divisible by 51 (providing the virus signature).

When an infected file is executed, the virus installs itself in memory and will spread whenever a delete, create, rename, open or get/set attribute operation is attempted on a file. The virus spreads by infecting one .COM file in the home directory for each such DOS call; the COMMAND.COM file is not infected.

When active, there is a random chance that the virus will delay five minutes after activation, then play three melodies repeatedly with a seven-minute delay between each rendition.

Screen virus The Screen virus infects all .COM programs in the current directory and then installs itself in memory. The virus cannot recognize previous infections and thus reinfects .COM files. When active, at intervals the virus transposes two digits out of any four-digit sequence it locates on the screen.

Traceback virus The Traceback virus, discovered in the United Kingdom in June 1989, is a .COM and .EXE infector that extends each by 3066 bytes. When an infected file is executed, the virus becomes resident in memory, and then infects all executed .COM and .EXE files (except COMMAND.COM).

After 5 December 1988, one .COM/.EXE file in the current directory is infected. If no such file is found, the virus infects the first eligible file in the directory hierarchy. If the virus encounters an infected file during the search, the search is terminated.

After 28 December 1988, the virus will cause a display similar to the cascade display an hour after infection. Depressing keys following the cascade will cause the virus to restore a character to its original position. Repeated keystrokes are ignored. After one minute the display returns to normal. The entire sequence repeats after one hour.

The virus is called the *Traceback* virus because an infected file contains the path name of the file that originally infected it. Thus, the path of infection can be traced.

A.7 Known Apple Macintosh Viruses

ANTI virus ANTI is the first virus for the Mac that does not add a resource to the infected file. Instead, the virus appends 1344 bytes of its code to the main CODE 1 resource of the application, and patches the code to ensure its execution before the application.

Again, it is not necessary to run an application to cause infection. ANTI is, however, less infectious than other viruses in that it does not infect the system file. Thus, the infection spreads only when an infected application is run.

Because of a bug, ANTI does not spread under Multi-finder.

Dukakis Virus This is an unusual virus—it propagates between Hypertext stacks. When an infected stack is executed, the *Open Stack* handler displays a message "Greetings from the Hyper-Avenger! ... Dukakis for President". It then installs the virus into the home stack, from which it will infect each stack as it is opened.

INIT 29 virus INIT 29 spreads when an infected application is run or selected. The system file is infected, and the virus patches the *Open* resource file trap.

Subsequently, any action that opens the resource fork of a file will cause that fork to be infected. This infection takes the form of the addition of a 712-byte code resource (numbered with the lowest free code resource number) to applications, or an INIT 29 to other files.

To cause infection, this virus does not require an application to be run. Only infected system files or applications can spread the virus, although many other files may be infected.

A characteristic of this virus is its attempt to infect the desktop on a newly-inserted disk, printing the message "The disk needs minor repairs" if the disk is locked.

nVIR Virus nVIR is probably the most ubiquitous of all Mac viruses—seven variants of this virus are known to exist. It spreads via infected system files and applications. When an infected application is executed, an INIT 32 resource is added to the system folder. Subsequently, when the Mac is rebooted, the virus becomes resident in memory. Thereafter, all application programs started

are infected (including the finder/multi-finder) through the addition of a CODE 256 resource. The virus is named for the nVIR resources added to the system file or application program (in addition to INIT/CODE).

The "nVIR A" variant incorporates a counter that is decremented by one from 1000 at each reboot, and by two each time an infected application is run. When the counter reaches zero, nVIR A will say "Don't panic" if *MacinTalk* is installed, or beep if not. This occurs once in every 16 reboots and once in every eight application runs.

The nVIR B variant beeps (does not use *MacinTalk*) once in every eight reboots, and once in every four application startups. Other variants of this virus allegedly exist that, when activated, will delete a random file from the system folder.

Further variants of the nVIR virus consist of slight modifications to the name or number used for the auxiliary nVIR resources. All four of these viruses exhibit symptoms similar to nVIR B:

- Hpat—renumbered code resource, nVIR changed to Hpat.
- AIDS—renamed code resources, nVIR changed to AIDS.
- MEV#—renamed code resources, nVIR changed to MEV#.
- nFLU—renamed code resources, nVIR changed to nFLU.

Peace Virus Aliases: Drew, MacMag, Brandow, Aldus

The Peace virus is designed to display a message of world peace on 2 March 1988 and then delete itself. The virus propagates by inserting an INIT 6 (Name "RR") into the system file. This virus does not infect application programs, but propagates only to system files present on hard or floppy disks.

Allegedly, this virus was developed by Drew Davidson for Richard Brandow of *MacMag* magazine.

Scores virus Alias: Eric, Vult, NASA

When an application infected with the Scores virus is run, it infects the system file, note pad, and scrapbook file. In addition, two invisible system files, named *Scores* and *Desktop*, are created.

Two days after system infection, the virus begins to spread to application programs. Any application executed is infected, and applications with "VULT" and "ERIC" resources are specifically targeted. If such an application is run, the virus will cause a system error after 25 minutes of use.

The virus enters its final phase of activity seven days after infection: 15 minutes after a "VULT" application is started, the virus will cause any writes to a disk file to return a system error.

Other than the extension of application files and its memory usage, Scores does not damage applications or destroy data.

B Information on Anti-Viral Software

There are literally dozens, perhaps even hundreds, of programs intended to prevent and detect computer viruses. They range from extensive security packages for network installations to small public-domain programs designed to fix a specific strain of virus.

It is beyond the scope of this article to provide a comprehensive listing or review of this software. Not only would such a list be too long, it would also be out of date in a matter of months as new viruses and virus-killers appear on the scene.

Instead, we will provide some suggestions for particularly effective shareware and freeware anti-virus programs, as well as published reviews of other available software. (*Freeware* is software that is available for public use at no fee. It is not necessarily in the public domain, because the authors may continue to claim copyright, but they allow the software to be used without fee. *Shareware* is software that the authors place in public locations (like bulletin board systems) with the request that users who like the program then pay a fee (usually small) for the code.)

Note that the information presented here is current as of Fall 1989 and is indicative of software available. The availability of software continues to change, and you should seek more current information if possible. The interested reader should obtain opinions from other users, user groups, and publications about the best software currently available. Selecting an appropriate mix of virus prevention, detection, and cure software is recommended.

B.1 Selected Reviews of Anti-viral Software

A description of viruses, along with capsule reviews of anti-viral software for Apple Macintosh computers, appeared in the November 1988 issue of *Macworld* magazine. The article was entitled "Mad Macs," and began on page 93.

One set of particularly good reviews of IBM PC programs that combat viruses appeared in the 25 April 1989 issue of *PC Magazine*. The reviews are in the article "Infection Protection," beginning at page 193.

Another set of reviews appeared in "A Virus Protection Sampler," starting at page 92 in the May 1989 issue of *Personal Computing*.

The June 1989 issue of *Byte* magazine contains a special section on security. The articles on pages 285 through 291, "Personal and Private" and "The Safety Zone," both provide information on security software. The latter provides product names, addresses of manufacturers, phone numbers, and prices; effective freeware and shareware products are not mentioned.

B.2 Easily Obtained Software

Although many viruses spread through bulletin boards and public software, those same bulletin boards also have proven to be a source of good software to prevent viruses. The user community, as a whole, deplores computer viruses, and many have devoted their time and talents to writing and distributing anti-viral software. Some of their freeware and shareware often are recommended by users:

- VIRUSCAN is designed to scan hard disks and floppies for infected programs. It can detect most IBM PC viruses. It is available through the Internet virus archives (see below), and from the HomeBase BBS @ (408) 988-4004.
- FluShot+ installs itself as an interrupt monitor at boot time. It watches disk activity on IBM PC machines, and warns the user of suspicious activity. It monitors for some known virus signature activities, and has options to encrypt and checksum files, as well as to monitor for

TSR activity. It can be obtained from the Internet archives, CompuServe, and from Ross Greenberg's BBS @ (212) 889-6438.

- Virus Rx is a free program distributed by Apple Computer and is available from many merchants and bulletin boards. The current version is 1.4a2 and was released in January 1989. It detects INIT 29, nVIR, SCORES, and variants of nVIR on Apple Macintosh systems. 1.4a2 does not detect the ANTI virus. Virus Rx has detected the presence of new viruses with its built-in self-checking feature. If it detects a change in itself while it's running, it alerts the user and changes itself into a document with a name asking the user to throw it away. This prevents Virus Rx from becoming a carrier if something infected it while running.

- Vaccine version 1.0.1 is a cdev (control panel device) that alerts users when most forms of virus attempt to infect their Macintosh systems. This is available on many bulletin board systems, and may be obtained from Don Brown of CE Software @ (515) 224-1995.

- Disinfectant version 1.3 is freeware available from Genie, CompuServe, Internet archives, and many user groups. It currently is able to detect all known Macintosh viruses (after infection) and safely remove them. It comes with a well-written manual and an easy-to-use interface.

- VirusDetective is shareware available from many bulletin board systems, and from the author:

  Jeffrey Shulman
  P.O. Box 50
  Ridgefield, CT 06877

  VirusDetective detects and removes the Scores virus from infected Macintosh programs.

There are undoubtedly many other programs available and worth considering for use. Your best bet is to talk with other users, through user groups or bulletin board systems, and discover what programs they recommend. Establishing contact with other programmers is also a good way to find expert assistance and experience should you discover that your system has been infected with something you do not recognize.

B.3 Internet Archives

Users with access to the Internet may obtain copies of public domain and shareware software to counter computer viruses. The master list of archives is regularly published in the *comp.virus* Usenet newsgroup, also available as the *Virus-L* mailing list. Ken van Wyk <krvw@sei.cmu.edu> is the moderator of that group.

The following sites are U.S. archive sites for these programs and for associated documentation. Each is listed with a contact address if additional information is needed on how to access the files via FTP.

- Macintosh archives are located on rascal.ics.utexas.edu in "ftp/mac/virus-tools". To use the archive, retrieve the file *00.INDEX* and review it offline. Contact is Werner Uhrig <werner@rascal.ics.utexas.edu>.

- Macintosh archives are also located on sumex.stanford.edu in "ftp/info-mac/virus". Contact Bill Lipa for additional information, if needed: <info-mac-request@sumex-aim.stanford.edu>

- A third Macintosh archive is located on wsmr-simtel20.army.mil and can be found in "PD3:<MACINTOSH.VIRUS>". Retrieve the file *00README.TXT* and review it offline. Contact is Robert Thum <rthum@wsmr-simtel20.army.mil>.

- An archive of Atari ST anti-viral software is available on the machine ssyx.ucsc.edu in the directory "ftp/pub/virus". The archive is also accessible as a mail-based server; instructions for the server can be obtained by sending mail to "archive-server@ssyx.ucsc.edu". Contact is Steve Grimm <koreth@ssyx.ucsc.edu>.

- An IBM PC anti-viral archive site is located on ms.uky.edu in the directory "ftp/pub/msdos/AntiVirus". The contact is David Chaney <chaney@ms.uky.edu>.

- An archive of anti-virus software for Amiga machines is also on ms.uky.edu in the directory "ftp/pub/amiga/Antivirus". The contact is Sean Casey <sean@ms.uky.edu>.

- A European archive site is available at Heriot-Watt University. Software for IBM-PCs, Apples, Ataris, and Amigas are present, as are back issues of the *Virus-L* mailing list. Information about operation can be obtained by sending mail with the text "help" to <info-server@cs.hw.ac.uk>. Sites in the U.S. should NOT use this server due to speed and cost constraints.

B.4 Other Places to Look

Most major commercial bulletin board systems such as CompuServe, Genie, and BIX all carry downloadable copies of public domain and shareware anti-viral software. Also, the PC support labs of many colleges and universities probably have copies of these programs, and many will give you copies of the software if you provide your own diskette.

Be sure to note, however, that it is possible to pick up a new virus if you load anti-viral software from an untrusted source! Be cautious about obtaining such software, even if it has been recommended to you (here, or elsewhere).

There are, of course, many fine commercial products that are available to combat computer viruses. Many of these products provide additional functionality that enhances your security and reduces your exposure to viruses—they are worth examining. Not all commercial programs do what they advertise, however. Asking the advice of other users and reading detailed reviews of the products involved may help you identify the products that are right for you.