July 1990 ISSN 0956-9979

VIRUS BULLETIN

THE AUTHORITATIVE INTERNATIONAL PUBLICATION
ON COMPUTER VIRUS PREVENTION,
RECOGNITION AND REMOVAL

CONTENTS

EDITORIAL 2

TUTORIAL

Detection and Brute Force Disinfection 3

VIRUS ANALYSES

Burger's Legacy I - 'Demonstration Disks' 6

Burger's Legacy Continued - The Vienna Virus 7

KNOWN IBM VIRUSES (UPDATES) 9

FOR PROGRAMMERS

The Structure of Virus Infection Part I .COM Files 10

SPECIAL FEATURE

Virus Writers and Distributors 12

CONTERMEASURES

Virus Monitoring Software - An Endless Battle 15

MAC THREATS 17

PRODUCT EVALUATION

Copy Protection: *VB* Policy 18

ENDNOTES & NEWS 20

EDITORIAL

The recent conviction and sentencing of UK hacker Nicholas Whiteley at Southwark Crown Court, London, is unlikely to deter computer misuse.

Whiteley launched a trail of destruction is 1988 causing £25,000 worth of software and data damage on networks at London's *Queen Mary College*, *Bath* and *Hull* Universities. On June 7th, he was sentenced to four months' imprisonment making him the first British hacker to go to jail.

However, Whiteley has little to lose by this verdict and probably much to gain. A book is to be published about his exploits, his photograph has appeared in national newspapers and, according to many observers, he will walk straight from the prison gate into a lucrative job as computer security consultant. Like Robert Morris, Edward Austin Singh and Robert Schifreen, Whiteley has jointed the 'hall of fame'.

There are reasoned arguments against criminalising pure hacking - the action of gaining unauthorised access to computer systems. The British legal system is fraught with inconsistency. Criminalising unauthorised access to electronically stored data contradicts the fact that a written communication marked 'private and confidential' is not recognised as such by law. Nor is physical trespass a criminal offence - which renders any concept of 'electronic trespass' highly questionable. Ultimately, information is not considered property and never has been.

In the United Kingdom the haste to introduce legislation led to the drafting of a Bill on computer misuse which is riddled with inconsistency and shortcomings. Nobody, for instance has so far provided any sensible prescriptions for its effective enforcement as statutory legislation. There are also quite profound issues of personal freedom, ownership and property which appear to have been overlooked.

What is urgently required is *tough, enforceable legislation* and *concerted international action against virus-writers*. There is a distinction between exploring a computer system and infecting it thus causing unwanted effects and destruction. It is not unreasonable to demand emergency legislation to address specifically the critical areas of data and software integrity in advance of drafting legislation about hacking and fraud. Viruses are the agents of widespread vandalism and are seriously undermining user confidence - they have for instance depleted a marketing industry reliant on the use of demonstration software and seriously impeded shareware and public domain software distribution.

What is significant about Whitely, is that he has been judged a *common criminal* - he has been found guilty of destroying £25,000 worth of magnetic media. A jury decided that Whitely's actions were akin to common yobbery, just like the habitual lout who smashes the neighbourhood telephone box, uproots newly planted saplings and scrapes the paintwork of your car with his key-ring. Those who write and spread viruses also fall into this category.

Viruses currently threaten integrity and availability. Lost or corrupt data is always a nuisance but can also be the cause of considerable distress - in the case of medical and safety-critical systems it can even cause pain and death. The stakes are high, our financial security, physical safety and welfare is dependent upon accurate computer data and operation.

Effective legislation to stamp out computer vandalism is desperately needed worldwide - to close the Bulgarian 'virus factory' to deter future software extortionists such as the notorious '*PC Cyborg Corporation'* and to restore user confidence.

With specific reference to computer viruses, the issues confronting the law makers can be defined clearly. If action against virus writers is to be taken the legislators must decide whether it should be criminal to:

Gain unauthorised access to a computer and subsequently delete, modify or tamper with its system, programs or data, whether intentionally or accidentally.

Distribute self-replicating machine code or source code.

Publish self-replicating source code.

Obviously, ethical questions surrounding the 'accidental' development of a virus (which has already occurred with commercial software), unintentional access and accidental deletion arise and must be taken into consideration. In most countries, software publishers already live under the threat of civil action in the event of their distributing contaminated or even bug-ridden software. Further legislation would probably enhance manufacturers' vigilance and quality control.

Ultimately, however, criminal legislation should not be designed to trap the hapless, the bewildered or the unlucky, *It should be aimed at the bloody-minded, wanton computer vandal.* The need for specific, watertight legislation will become patently obvious when the author of a pernicious virus is apprehended and the corresponding source code, development machine and other conclusive evidence impounded. Currently, only a handful of police forces in the world could press any charge whatsoever with any hope of conviction.

By addressing the virus issue directly and regarding it as a special category, wider legal wrangles over the ownership of data, its status as property, and the freedoms and rights of hackers are nearly circumvented. **A prolonged failure to legislate and extradite offenders will exacerbate the computer virus problem - virus writers will continue undeterred amidst an international user community stripped of any protection under criminal law.**

TUTORIAL

Detection and Brute Force Disinfection

This article outlines techniques for detection of three distinct categories of PC virus: **parasitic, partition boot sector and disk boot sector viruses**. It explains some of the pitfalls for the unwary as well as prescribing straightforward methods of disinfection. **In all cases the problem discussed is that of infection of the hard disk.**

Note: The disinfection techniques described here for removing boot sector viruses are neither subtle nor elegant, but do provide a reliable worst-case remedy. In fact it is usually possible to remove boot sector viruses without re-formatting the affected disk, but this should be attempted only with expert advice at hand.

The golden rule for all diagnostic and removal work concerning viruses is that the PC should undergo a clean bootstrap process before proceeding. This means that it should be switched off and then re-booted from a **write-protected clean system floppy disk**, such as the master disk supplied with the machine when purchased.

Parasitic Virus Detection and Disinfection

In the absence of information provided by a checksumming program, the technique for finding a parasitic virus such as Jerusalem or Cascade is to search all executable files, in all directories, for a pattern known to be contained within that specific virus. This 'search pattern' is generally a hexadecimal string of the sort listed in the VB Table of Known Viruses. The technique for removal is to delete the file containing the virus.

The basic approach is therefore quite simple:

1. Switch off the PC and re-boot it from a write-protected clean system floppy disk.

2. Do not run any software from the hard disk, or make it the current drive.

3. Find every executable (usually .COM or .EXE) file which contains the pattern identifying the virus, using a scanning program or general purpose utility running from floppy disk.

4. Delete each such infected file using the DOS command DEL.

5. Restore a clean copy of the file from the master disks on which the software was originally supplied.

There are two common methods of searching for patterns: **general-purpose utilities** (such a Norton Utilities of Mace) and **specialist virus-specific scanning programs.**

In general it is best to use a good scanning program since this will usually allow an entire disk to be swept for all viruses in a single operation, as well as ensuring that the search is carried out in a reliable way.

While Norton, PC Tools and Mace are powerful utility programs, there are several problems is using them for this purpose; consider, for example, use of the Norton Utilities:

1. As Norton can search for only one pattern at a time, a search for parasitic viruses will need to proceed pattern by pattern. Every time a new search is carried out, the patterns will need to be re-entered manually, which is not only time-consuming but also error-prone. It is not possible to maintain a library of patterns.

2. While Norton does allow a search of the entire data area of the disk to be made, this is done cluster by cluster, sequentially. As a result, Norton reports pattern matches in unused space as well as in files, which gives troublesome false positives after infected programs have been deleted and can give false reports when two adjacent clusters combine to form the pattern. Much more seriously, when used in this mode Norton fails to find any search pattern which, in a particular file, happens to fall across the boundary between two non-adjacent clusters. This in turn means that a reliable search using Norton can be carried out by selecting each file in turn and searching it for the pattern in question. For example, to check a hard disk containing 300 executable files for the 50 most common parasitic viruses would require the Norton user to go through the menu sequence 15,000 times - *a total of more than 130,000 keystrokes*. By contrast a dedicated scanning program will perform the same task, more reliably, in a fraction of the time and requiring fewer that 10 keystrokes.

3. Viruses such as 1260 cannot be found using a straightforward pattern search - while scanning programs can still find several such viruses, it is an infeasible task using general purpose utilities.

It should be noted that these points apply to most PC utility packages and not just to The Norton Utilities.

Partition Boot Sector Viruses - Detection and Disinfection

Partition Boot Sector viruses, such as the Italian or Disk Killer, infect the **Partition Boot Sector** (or just Boot Sector). **This is the first sector within a hard disk partition, or the first sector on a floppy disk. It is addressable as logical sector 0 within the partition or disk.**

Such viruses are found by searching the partition boot sector for a pattern known to be within the virus. They are most simply eliminated by using the FORMAT command to replace the affected sector.

A simple procedure is therefore:

1. Switch off the PC and re-boot it from a write protected clean system floppy disk.

2. With the aid of a scanning program or a utility program, running from a floppy disk, determine whether logical sector 0 contains a patter identifying the virus.

3. Back up all data files. The backup disks will not become infected, because the PC has been booted from a clean disk. Make sure master copies are available of all programs - if not, then back up these too.

4. Perform a high-level format of the hard disk by typing FORMAT C:/S. This will replace the infected boot sector with a clean copy as well as clearing up any bogus 'bad sectors' created by the virus.

5. Restore all programs and data from the appropriate backup copies.

It is also possible to kill a partition boot sector virus by typing SYS C: instead of FORMAT C:/S at stage 4 above. However in the case of most viruses this will still leave one or more bogus 'bad sectors' containing virus code. More elegant methods are also possible but should be attempted only with expert advice, given in the context of the specific system being disinfected.

Disk Boot Sector Viruses - Detection and Disinfection

The only known virus to date, albeit a very widespread one, which infects the Disk Boot Sector is the New Zealand virus. **The Disk Sector** (also sometimes referred to confusedly as Master Boot Sector, Master Boot Record, Partition Boot Record, Partition Record etc ad nauseam) **is the first physical sector on a partitionable hard disk, addressable as sector 1, track 0, head 0.**

As with a partition boot sector virus, the approach is to search the suspect sector using a virus scanning program or a standard utility, followed by replacing the sector with a clean copy.

A simple, if draconian, procedure is therefore:

1. Switch off the PC and re-boot it from a write-protected clean system floppy disk.

2. With the aid of a scanning program or a utility program, running from a floppy disk, determine whether sector 1, track 0, head 0 contains a pattern identifying the virus.

3. If possible (see note below), back up data files. The backup disks will not become infected, because the PC has been booted from a clean disk. Make sure master copies are available of all programs - if not, then back these up too.

If the disk has additional partitions, ensure these too are backed up.

4. Perform a low-level format of the hard disk, followed by rebuilding the partitions using FDISK and FORMAT. This procedure should be described in detail in the manufacturer's installation manual provided with the machine. It is vital that this documentation exits and is accessible.

5. Restore all programs and data from clean backup copies.

Again, there are neater ways of removing the New Zealand virus. However the success of this operation depends crucially upon the details of the infection and needs expert assessment. The above procedure, while brutal, will work. **Note however that on certain systems the New Zealand virus destroys part of the FAT, and therefore it may not be possible to back up data files in the normal way. If so, it is essential to seek expert advice or restore files from a pre-infection backup.**

Summary - Scope of Anti-Virus Tools

General-purpose utility programs can be quite useful in dealing with bootstrap sector viruses of either kind, but are fundamentally unsuited to finding parasitic viruses.

Virus-specific scanning programs provide the best ad hoc method of checking a disk from scratch, for known bootstrap or parasitic viruses. However they are less satisfactory for long term use than checksumming programs, which do not need updating and will detect future as well as current viruses.

Good scanning software should offer the following benefits:

- The ability to scan specified disks for known viruses of all types.
- Automatic identification of both the viruses and the infected programs or sectors.
- Avoidance of false positives such as viruses reported in unused clusters.
- A regular updating service to cope with new viruses and mutations.
- A virus pattern library which can be expanded by the user in emergencies.
- The ability to automate the virus detection process.
- Easy and fast operation.

Both scanning and checksumming anti-virus programs can be used for **software validation.** The establishment of a Software Quality Assurance Section to test incoming software on a quarantine PC is strongly recommended (see VB, May, 1990. pp 5 and VB, June, 1990 pp 3).

Software Validation

All incoming software should be screened prior to its installation on company PC's or LANs. This applies to all software including shrink-wrapped packages from reputable manufacturers. Testing should be undertaken on a quarantine PC and involves the following basic steps. **Make sure that the inspection disk is write-protected before examining it.**

* Compare the programs listed in the documentation with those which appear on the disk itself. Make the floppy disk drive current, install the suspect disk and inspect it using Norton or a similar utility. Note, the DOS DIR command run from the floppy drive will not spread a virus but could trigger a Trojan. Similarly, unexplained files should be read using a utility program. **Do not read unexplained files using DOS**. Any discrepancies should be queried with the software manufacturer and vendor. If there is no documented program listing contact the manufacturer and obtain one.

* Some programs use READ.ME files which contain instructions and/or updated information. View these using a special utility - do not use the DOS TYPE command. This could trigger a key-redefining Trojan horse.

*Search for unused clusters which might indicate an erased virus which could be intentionally recovered by technically proficient staff.

*Does the disk contain bad clusters? The presence of bad clusters is indicative of poor quality media but may also result from virus infection.

*Examine the boot sector and ensure that it is a legal boot sector. Modifications to the boot sector indicate the presence of a virus *(See Figure 1.)*

* Does the software modify executable file lengths or attributes? A cryptographic checksum of files stored on a clean PC can be compared to a checksum of the system after installation and execution of the suspect program.. The checksumming package should identify the type of modification and the name of the infected file(s). Any modification to system or program files should be regarded as suspicious.

* Scan the suspect disk for known virus patterns and identities using a scanning program. Floppy disks can be scanned in a matter of seconds.

Testing should be systematic and results of each test documented. One a disk has been cleared a backup should be taken using a known 'clean' machine. **The write-protected master disk should be safely archived while the backup disk (also write-protected) can be used to install the software.**

```
000000   eb 34 90 4d 53 44 4f 53   33 2e 32 00 02 02 01 00   .4.MSDOS  3.2.....
000010   02 70 00 d0 02 fd 02 00   09 00 02 00 00 00 00 00   .p......  ........
000020   00 00 00 00 00 00 00 00   00 00 00 00 00 00 00 0f   ........  ........
000030   00 00 00 00 01 00 fa 33   c0 8e d0 bc 00 7c 16 07   .......3  .....|..
000040   bb 78 00 36 c5 37 le 56   16 53 bf 2b 7c b9 0b 00   .x.6.7.V  .s.+|...
000050   fc ac 26 80 3d 00 74 03   26 8a 05 aa 8a c4 e2 f1   ...&.=.t  &.......
000060   06 1f 89 47 02 c7 07 2b   7c fb 8a 16 fd 7d cd 13   ...G...+  |....}..
000070   72 66 a0 10 7c 98 f7 26   16 7c 03 06 1c 7c 03 06   rf..|..&  .|...|..
000080   0e 7c a3 3f 7c a3 37 7c   b8 20 00 f7 26 11 7c 8b   .|.?|.7|   ...&.|.
000090   le 0b 7c 03 c3 48 f7 f3   01 06 37 7c bb 00 05 a1   ..|..H..  ..7|....
0000a0   3f 7c e8 94 00 b0 01 e8   a9 00 72 19 8b fb b9 0b   ?|......  ..r.....
0000b0   00 be c5 7d f3 a6 75 0d   8d 7f 20 be d0 7d b9 0b   ...}..u.  .  ..}..
0000c0   00 f3 a6 74 18 be 76 7d   e8 61 00 32 e4 cd 16 5e   ...t..v}  .a.2...^
0000d0   1f 8f 04 8f 44 02 cd 19   be af 7d eb eb a1 1c 05   ....D...  ..}.....
0000e0   33 d2 f7 36 0b 7c fe c0   a2 3c 7c a1 37 7c a3 3d   3..6.|..  .<|.7|.=
0000f0   7c bb 00 07 a1 37 7c e8   3f 00 a1 18 7c 2a 06 3b   |....7|.  ?...|*.;
000100   7c 40 50 e8 4d 00 58 72   cf 28 06 3c 7c 76 0c 01   |@P.M.Xr  .(.<|v..

000120   74 08 b4 0e b7 00 cd 10   eb f3 0e 07 b8 01 02 bb   t.......  ........
000130   00 02 b1 01 ba 80 00 cd   13 72 13 0e 1f be 00 02   ........  .r......
000140   bf 00 00 ad 3b 05 75 11   ad 3b 45 02 75 0b 2e c6   ....;.u.  .;E.u...
000150   06 08 00 00 2e ff 2e 11   00 2e c6 06 08 00 02 b8   ........  ........
000160   01 03 bb 00 02 b9 07 00   ba 80 00 cd 13 72 df 0e   ........  .....r..
000170   1f 0e 07 be be 03 bf be   01 41 4c 49 53 45 20 4d   ........  ..B.....
000180   03 33 db fe cl cd 13 eb   c5 07 59 6f 75 72 20 50   .3......  ..Your P
000190   43 20 69 73 20 6e 6f 77   20 53 74 6f 6e 65 64 21   C is now   Stoned!
0001a0   07 0d 0a 0a 00 4c 45 47   41 4c 49 53 45 20 4d 41   .....LEG  ALISE MA
0001b0   52 49 4a 55 41 4e 41 21   00 00 00 00 00 00 00 00   RIJUANA!  ........
```

Figure 1.

Normal boot sector (000H to 10FH) and a section of the New Zealand virus (commencing 120H to 1BFH).

Boot sector viruses **completely** replace the original boot sector (commencing 000H) and inspection of the boot sector will reveal their presence.

In this instance the distinctive New Zealand 'Stoned' message is shown.

VIRUS ANALYSES

Jim Bates

Burger's Legacy I - 'Demonstration Disks'

The book *Computer Viruses: A High Tech Disease* by Ralf Burger has already been reviewed (VB Oct, 1989) and the incredible irresponsibility and arrogance of the author in publishing virus source code has been noted both here and elsewhere. Burger's suggestion (in 1986) that viruses "*used properly may bring about a new generation of self modifying computer operating systems*" has still not been realised and subsequent events have proven his book to be *the* current source of reference for virus writers.

Another means whereby Burger has provided the viruses writers with working material to further their 'craft' is with so-called "demonstration" viruses. These not only mean that more recognition "signatures" need to be published, but they will also produce "families" of similar strains (like the Vienna group) which makes the whole process of fighting the threat much more complex than it needs to be. **International legislation formulated to make such activity a criminal offence is long overdue.**

The best known of Burger's demos is his VIRDEM virus, mentioned in the book. Fortunately, this is poorly written and contains bugs so that acolytes will need to disassemble and debug these programs before they can advance their own modifications. However, this will undoubtedly be done and new viruses can be expected to use some of the techniques used in VIRDEM and other "demo" virus code. For this reason, a brief description of VIRDEM may be useful to anti-virus researchers.

VIRDEM - Fact and Fiction

VIRDEM uses a "generation" number for each successive infection up to a maximum of 9. This is very simply done by incrementing a counter within the infection cycle and checking that once it reaches nine, the counter is disabled. This is used to collect indexed access to a table of entries which are, in turn, used as a basis for a guessing game when infected programs are executed. What happens is that the generation number is used as the upper limit of an integer to be guessed. A pseudo-random integer is generated by accessing and gating the system clock and if the operator's guess matches it, then program operation is allowed.

Otherwise the program does not run and an appropriate message is displayed (I disassembled the original Version 1.06 with German text).

Burger's own comments on this are worth repeating to indicate the weak reasoning and woolly thinking that this man indulges in:

"Unfortunately the source code cannot be published because with the help of the source code anyone would be able to change the manipulation task and have a non-overwriting viruses in 8088 machine language. In addition it would be almost unthinkable if there were suddenly numerous dangerously modified versions of VIRDEM.COM around."

This is either blindness or stupidity! He obviously assumes that anyone buying (?) a demonstration virus will be incapable of disassembling it. **VIRDEM is quite easy to take apart and equally easy to understand (and modify) and is almost as dangerous in it's code form as it would be in source code.**

It should also be noted that VIRDEM is not a "non-overwriting" virus, it reads the first 1280 (500H) bytes of the target program file, and appends them to the end of the file. It then overwrites these initial 1280 bytes with the virus code. Finally, a small stub recovery routine is also appended. The virus only infects floppy disks placed into drive A: and rather than reporting in detail on the virus operation, I shall list Burger's own comments regarding it (in italic) - together with my own findings after disassembly:

1) Al COM files up to the second sub-directory are infected.

(The virus is non-resident and infects only one file during each execution).

2) *The first COM file in the root directory (often COMMAND.COM) is not infected.*

(This is true although no reason is given. The possibility that COMMAND.COM will be the first file in the root directory is noted but no attempt is made in the code to avoid infecting it if it isn't).

3) COM files of more than about 1.5K in length are expanded by about 1.5K, shorter files are expanded by about 3K.

(Files longer than 1380 bytes are expanded by 1366 bytes - shorter files are first made 1280 bytes long and **then** expanded by 1366 bytes. No check is made on whether there is actually room for virus code and a work area within the 64K COM format limit, so **files greater than 62122 bytes are irreparably damaged).**

4) Infected programs remain completely functional.

(As noted above, **this is not true for large files** and there are also some unpredictable and possibly dangerous effects when the machine configuration includes certain device drivers).

5) An infected program is recognised and cannot be infected twice.

(True - an infected program is recognised by an initial word of 9090H).

6) VIRDEM.COM inserts an additional function into the infected program. This additional function is a guessing game whose difficulty level is dependent on the virus generation.

(True).

7) VIRDEM mutates up to the ninth generation. After that the propagation continues but no mutation takes place.

(Maybe the word "mutates" was an incorrect translation from the original German. Certainly the virus counter is altered but not as a result of a "mutation", rather because of an infection counter).

Detection

This virus uses only DOS function calls and since it is not resident, it has no defence mechanisms against anti-virus search programs. Therefore a version with only simple modifications will not be difficult to detect. More sophisticated changes will require separate disassembly and reporting. The hexadecimal recognition string for this version is:

```
Virdem 03C3 8BF0 268B 1C8B F3BF 0001 B900 0590;
offset 19BH
```

It should be noted that a Burger demonstration program also called VIRDEM which was disassembled by *VB's* Technical Editor contains English text as opposed to the original German text.

It is also possible that Burger released various completely different programs under the title VIRDEM, or even different version numbers of the same program.

This English text VIRDEM (also version 1.06) is substantially similar to the program described in this article. The search pattern as published will identify infection by either virus.

Burger's Legacy Continued - The Vienna Virus

The Vienna virus first came to the attention of the computer world-at-large as a result of the source code being published in Burger's book in 1987.

In my original review of the book (*VB, October 1989*) I *said "There is no doubt that some damage will result from attempted copies of the Vienna virus listing".* This has proven to be a huge understatement since I now have 17 separate examples of virus code which can be directly attributed to this listing. At least five of these display differences which occur as a direct result of the ways that different assemblers handle default addressing modes. Others, notably a group from Bulgaria, display concerted attempts at optimisation of the code to reduce the overall size. This multiplicity of versions complicates disassembly and analysis and makes recognition and disinfection more difficult.

The proliferation of the Vienna virus in this way raises questions concerning legislation against virus source publication. **Certainly the author of the original book has caused more harm than any single individual virus writer.**

The original virus is a relatively simple one which has well defined actions and makes fairly simple use of system resources. It affects only .COM files which are located in the current default directory and along the specified system PATH. The code does not become resident, relying solely upon program execution to be activated. This makes it easy to remove and it is not strictly necessary (although always wise) to reboot a machine to remove it. Infected files are marked by setting the seconds field of the file date/time stamp to 31 (equivalent to 62 seconds). Program files with this setting should be deleted and replaced.

As a result of the differences discussed above, the infective length varies considerably from around 623 up to around 670 bytes although there are some optimised versions (still awaiting full disassembly and analysis) as small as 353 bytes.

There is no recognised 'trigger' action but the virus does modify its actively according to a pseudo-random reading of the system clock which results in *corrupted* files.

Analysis

The Vienna virus uses the classic trick of appending its code to an infected file and routing processing into the

code by overwriting the first three bytes with an appropriate JMP instruction. The original three bytes are stores within the virus' own data area and processing begins by replacing these original three bytes at the start of the program file.

Next, a check is made to verify the version of DOS in use. This verification is about as simple as it's possible to get, consisting of a test to ensure that the major version number is not zero. If this test is passed, then the address of the DOS Disk Transfer (DTA) is collected and saved prior to the DTA being reset to a buffer maintained by the virus. The next routine collects the address of the Environment segment from the Program Segment Prefix (PSP) and this is then searched for the "PATH=" statement so that a pointer to the first directory noted therein can be maintained.

Processing then continues by commencing with a search for the first available .COM file in the current default directory. **Read Only and Hidden attributes are used for the search so using these attributes is not protection against infection**. As each file is found, its Date/Time stamp is examined for the 62 seconds marker. If the marker is found, the file is rejected and the search continues for the next file. If there are no un-infected files in the current directory, the path pointer is examined and the search continues in each directory in turn listed in the PATH= statement. If no un-infected files are found, the virus terminates and returns control to the original host program. **Note that whenever an infected file is run, this extended search is conducted and as more and more files become infected, there is a noticeable slowing in program execution time.**

Once a suitable (uninfected) file is located, its size is checked to ensure that it is between 10 and 64000 bytes - files outside these limits are rejected. Acceptable files then have their original attributes stored and are then set to allow read/write access and opened for the virus to examine. The original Date/Time settings are collected and stored and then the machine system clock is examined. This is he point at which a pseudo-random decision, based upon the contents of the system clock, is made on whether the file will become infected or corrupted. The first three bits of the system time field are checked to see if they are zero. If they are, the file will be corrupted, otherwise the file will be infected. **This means that on average, one is eight files will become corrupted rather than infected.**

Corruption in this instance consists of writing 5 bytes over the beginning of the target file. No attempt is made to save the original file contents and so the file is **irreparably** damaged. The published listing mentioned above is incomplete concerning just what these five bytes are but the earliest copies of the virus use a far jump instruction into the ROM reboot sequence. This means that trying to run a corrupted program will result in the machine rebooting and the original program function will not be executed. **This is the most dangerous aspect of this virus since the code is easily modified to increase the number and function of the overwriting bytes.** Thus it becomes impossible to say just what effects later versions might induce into the corruption.

The infection routine is quite standard - appending the virus code to the end of the file and modifying the initial 3 bytes to jump into it. Both corrupted and infected files are marked with the 62 seconds marker.

On the positive side, this virus uses only DOS functions and makes no attempt to hide its existence. It can therefore be traced quite easily using any of the numerous scanning programs now available. Its activity during infection also ensures that system monitoring programs such as FLUSHOT+ will detect the attempt to write to program files.

Removal

Removal is a relatively simple process once infected files have been identified. Although it is possible to repair infected files, it will generally be easier to replace them from master backup files. Corrupted files are a little more difficult since their identification is more awkward. However, once identified they must be replaced - they cannot be repaired.

Although the original virus used a 62 seconds marker in the time stamp of affected files, there have been reports of versions that use instead a 13 months marker in the date field. This is still being investigated, together with the minor changes introduced within some of the optimised versions.

Conclusion

The fact that Burger was allowed to publish a "cook book" and "demonstration" virus code for aspiring virus writers is an international scandal. His irresponsibility is matched only by his publishers (Date Becker, West Germany and Abacus, USA)

It cannot be emphasised enough that the publication of virus source code is every bit as damaging as production of the viruses themselves. The sooner that this too is made a criminal offence, the better we shall be able to combat the threat.

KNOWN IBM VIRUSES (UPDATES)

Updated and amendments to the *Virus Bulletin Known IBM PC Virus Table* as of June 27, 1990. The full table was last published in March 1990 and will be published again in the August 1990 edition.

Entries consist of the virus name, its aliases (if any) and the virus type (see Type Codes). This is followed by a brief description and a 16 bytes hexadecimal pattern which can be used to detect the virus using the 'search' routine of disk utility programs such as Norton PC Tools or virus specific scanning software. Offset (in hexadecimal) means the number of bytes from the virus entry point.

5120, CEN: This is one of the largest viruses known, 5120 bytes long. Complete analysis is not yet finished. Parts of the virus seem to be written in compiled BASIC.

```
5120 40B1 E88C DB03 C305 1000 8ED8 8C06 ; Offset 026
```

Anarkia-B, CER: Minor variant of Jerusalem detectable by Jerusalem strings already published.

Armagedon, CR: A 1079 byte virus from Greece, which interferes with the serial port. It will produce control strings for Hayes-compatible modems, dialling the number 081-141 which is a speaking clock in Crete. Virus name is mis-spelt with a single 'd'.

```
Armagedon 018C CBEA 0000 0000 8BC8 8EDB BE00 01BF ; Offset 3F0
```

Cancer, CN: Variant of the Amstrad virus, 740 bytes long - detected by signature strings already published.

Form, DR: This boot sector virus from Switzerland infects hard disk as well as diskettes. The virus contains no side effects.

```
Form B106 D3E0 8EC0 33FF B9FF 00FC F3A5 06B8 ; Offset 074
```

Jo-Jo, CR: This is a non-encrypted version of the Cascade virus (1701), produced by patching out the encryption code and making minor changes.

```
Jo-Jo B800 F08E COBF 08E0 813D 434F 751B 817D ; Offset 0D2
```

July 13th, ER: This virus activates on July 13th, but the extract effect have not yet been determined. The virus is 1201 bytes long and it is encrypted.

```
July 13th 2EA0 1200 3490 BE12 00B9 B104 2E30 0446 ; Offset variable
```

Mendoza, CER: Minor variant of Jerusalem. Detected by strings already published.

Shake, CR: A primitive 476 byte virus which repeatedly infects the same files. Infected programs sometimes produce a reboot when executed.

```
Shake B803 42CD 213D 3412 7503 EB48 90B4 4ABB
```

Solano, CR: Virus adds 1991 bytes in front of a file and 9 bytes at the end. It is still awaiting disassembly.

```
Solano B4C0 CD21 3D34 1275 OE2E 8BOE 0301 1E07
```

Svir, EN: this is a simple 512 bytes virus with no interesting effects.

```
Svir 33f6 4626 8BOC E302 EBF8 8BD6 83C2 04E8 ; Offset 049
```

Taiwan-2, CN: A new variant of the Taiwan virus, with a length of 743 bytes. The Taiwan virus activates on the 8th day of any month and overwrites the FAT and root directory on drives C: and D:

```
Taiwan-2 07E4 210C 02E6 21FB B980 00BE 0000 BB80 ; Offset 065
```

Tiny, CN: Variant of the Kennedy virus, but only 163 bytes long. No effects other than replication.

```
Tiny 408D 94AB 01B9 0200 CD21 B43E CD21 FFE5; Offset 088
```

Victor, CEN: A recent 2442 byte virus from the USSR. Awaiting disassembly.

Victor 8CC8 8BD8 B104 D3EE 03C6 50B8 D800 50CB : Offset 0C8

Virus 101, CER: An improved version of Virus-90 written by Patrick Toulme of the USA. This self-modifying encrypting virus uses and XOR encryption algorithm. **NOTE: No reliable search pattern can be extracted.**

Virus B, CN: 'Test virus' which was available as a 'restricted access file' from John McAfee's Interpath/NBBS bulletin board, USA. Virus -B is a modified version of the South African virus (*detection string, VB, March 1990).* Destructive code of the original has been disabled but could be reactivated.

REPORTEDONLY

Ambulanc

Suomi

Type Codes
C = Infects COM files
E = Infects EXE files
D = Infects partition boot sector (Logical sector 0 on disk)
M = Infects disk boot sector (Track 0, head 0, sector 1 on disk)
N = Not memory resident after infection R = Memory resident after infection.

FOR PROGRAMMERS

Fridrik Skulason

The Structure of Virus Infection Part 1: .COM Files

As the number of computer viruses grows, the need for various systems of classification increases. A classification system can be based on two criteria - functional and structural. The functional system classifies viruses depending on how they operated, how they select programs to infect, whether they are memory-resident, the methods they use to bypass virus monitors, and so on.

The structural system only accounts for the *structure* of infected programs. Structural analysis of infected files is of particular importance to anyone attempting to write a *disinfector program* or a program searching for a possible infection by a previously unknown virus. This primarily interested in preventive software might prefer to use functional classification, which will be the subject a later series of articles.

In part 1, infected .COM files are discussed, followed by .EXE and boot sectors in parts 2 and 3 respectively.

What is a .COM File?

There are subtle differences between .COM and .EXE files which are of more importance than the simple difference between the extension designation. These differences are easily overlooked due to the fact that renaming a .COM file as a .EXE file or vice versa does not affect program, functionality. However, DOS uses the first two characters of any file being executed to determine its true type. If these characters are 'MZ' or 'ZM', the file is assumed to be a true .EXE file, regardless of its extension name .EXE files, which in general terms have more complex structure than .COM files, are always relocated after loading.

A file starting with any series of characters other than 'MZ' or 'ZM' is treated as a true .COM file, which can be loaded directly without any relocation.

Some computer viruses identify files by their extension name rather than the first two bytes of their code. A virus that identifies a file by the extension .COM will cause unpredictable results should it infect a true .EXE file renamed with a .COM extension. However, this problem does not affect the disinfection of a file.

Viruses which infect .COM files can be divided into four groups - **simple overwriting**, **improved overwriting**, **prepending** and **appending**.

Group 1 - Simple Overwriting

The most primitive method used to infect .COM files is simply to overwrite the beginning of the program with the virus code. Programs infected by simple overwriting viruses seldom work upon execution - which diminishes the chances for these viruses to spread. The book by Burger (see pages 6-8) describes a few viruses which use this method of infection, but these types of viruses are rarely found in the wild. The 405 virus is the best known in this category and there are some Pascal viruses which use the method including the AIDS virus (distinct from the AIDS Trojan). The Pascal viruses are comparatively large at 12K or so. Restoring a program infected by one of these viruses us impossible without a backup, due to their destructive nature.

The Lehigh virus uses an interesting variation of this method, placing the virus code in an area which is normally unused. A JMP to the beginning of the virus code is then written at the beginning of the file.

It is not possible to restore a program infected with the Lehigh virus to its original state, but normal operation can be restored simply be locating the original first three bytes of the infected program and writing them to their original position.

Group 2 - Improved Overwriting

A more sophisticated variation of the overwriting method is to store the overwritten part of the program at the end of the file. By doing this the virus can restore the original program once it (the virus) has finished its infection routine. Program disfunction is less likely so the virus will avoid detection thus increasing its chance to spread, The following viruses use this method:

Name	Length
Taiwan	708,743
Pretoria	879
Virdem	1336
X-AL	1539

Files infected with any of the viruses in this group can be disinfected by reading the original beginning of the program, which had been stored at the end, and writing it back to its original location. The file is then shortened as necessary.

The 800 virus from Bulgaria uses a similar method. It overwrites 800 bytes of the program and appends the original overwritten part to the end of the file, but it does not overwrite the first 800 bytes of the file. Instead it overwrites a random part of the file and place a 3-bytes JMP at the beginning of the file which transfers control to the virus code.

The Number of the Beast viruses employs a variation of this method - it overwrites the first 512 bytes if the file but stores the original code in the free space after the end of the file without

altering the file length. This is possible because DOS allocates space for files in 'clusters' which are normally 1024 or 2048 bytes long.

Group 3 - Prepended

A number of .COM infecting viruses infect files by placing virus code in front of the original program. This method is used by the following viruses:

Name	Length
Amstrad/Pixel/Cancer	847,740,345,299,277
sURVIV 1 (April 1st)	897
Armagedon	1079
Amoeba	1089 (303)
Sylvia	1301 (31)
Zero Bug	1536
SunDay	1631 (5)
Jerusalem	1808 (5)
Solano	1991 (9)
Fu Manchu	2080 (6)

In addition, the Agiplan virus has been reported to use this method, but no researcher has yet obtained a sample of it, so this cannot be confirmed. Some of the viruses may also add several bytes to the end of the infected program. This is indicated by the figures in brackets in the table above.

Disinfecting a program infected with one of these viruses is done by reading the infected file, skipping over the virus code done by reading the infected file, skipping over the virus code at the beginning and writing the rest to a new file. The Armagedon (*sic*) virus require further work because it scrambles the first byte of the original program by adding 11 to it.

Group 4 - Appending

The most common method of .COM file infection is to place the virus code after the normal end of program and overwrite the first three bytes with a three-byte JMP or CALL pointing to the start of the virus code. The following viruses do this:

Name	Length
SouthAfrica/Virus-B	415, 416, 540, 544,
Shake	476
W13	507, 534
Stupid	583
Vienna (Lisbon)	684, 623, 627, 367, 353, 348, 435
Eddie-2	651
Perfume	765
Virus-90	857
Fumble	867
Devil's Dance	941
1260	1260
Murphy	1277, 1521
Datacrime II	1480, 1514
Cascade	1701, 1704
JoJo	1701
Dark Avenger	1800, 2000
dBase	1864
Ghostballs	2351
Victor	2442
Virus-101	2560
Yankee	2756, 2932, 2997 2885, 2901, 2981
Traceback	3066, 2930, 3031
5120	5120

A few viruses use a similar method - they overwrite more than just the first three bytes of the virus. The number of bytes which are overwritten are shown in brackets after the virus length.

Name	Length
Durban	699 (17)
VP	909-927 (16)
Datacrime	1168, 1280 (10)
Tenbyte	1554 (32)
8-Tunes	1971 (11)
Hallochen	2011 (6)
Oropax	2756-2806 (4)
Liberty	2857 (120)
Fish 6	3584 (6)
Syslock	3551 (35)
4K	4096 (6)

The following two viruses produce infections which appear similar to those listed above, but as the viruses only infect .COM files that already start with a JMP (E9H), they do not store the first byte, only the second and third.

Name	Length
Tiny	163
Kennedy	333

Disinfection is more complex for viruses in group 4 that in the other three groups. The original code which was overwritten must first be located, somewhere inside the virus body. In most cases it is stores at a fixed offset from the beginning or the end of the virus code, but the bytes need not be contiguous. To complicate the problem further, some of the viruses above (Cascade, Datacrime II, Syslock, 1260. Liberty) are encrypted, so the virus must first be decrypted, in most cases using a simply XOR or rotate algorithm.

The original bytes are then written to their original position, and the virus code removed by shortening the file as necessary.

SPECIAL FEATURE

Dr. Jan Hruska

Virus Writer and Distributors

'Attributable Viruses'

It is not easy to establish the origins of a computer virus and it is rare that positive indicators as to authorship can be found by examining virus code. There are, of course, notable exceptions to this.

The Brain virus, for instance, includes the authors' names, address and telephone numbers embedded in the boot sector. The virus code was written by two computer software retailers and was reportedly developed as a means of copy-protection - a measure to punish 'bootleggers'. (see *Figure 1*.)

Toulme

Patrick A Toulme of the United States is a high profile virus writer. Earlier this year he uploaded his 'Virus-90' to a number of Bulletin Boards in the United States ad requested a fee of $19.95 (see *VB, March 1990*). His latest release is 'Virus-101', an 'improved' version of his earlier creation. Virus-101 is a memory resident self-modifying and encrypting virus which makes the extraction of reliable hexadecimal pattern virtually impossible. Fridrik Skulason reports that patterns to search for the virus may be possible with the use of 'wild card' characters in the hexadecimal pattern although the virus is still under examination. Virus-101 will probably necessitate the use of a 'virus identity' to identify infected files (see '1260 Revisited', *VB, April 1990*).

Toulme claims that these viruses are *"designed to give both experienced programmers and novice computer enthusiasts experience in dealing with computer viruses".* There seems little legal redress against such activities, although an infection caused by Virus-90 or -1`01 on US federal interest computers might expose him to prosecution under the *US Computer Fraud and Abuse Act 1986.* It is probable that both viruses (and variants of them) will appear in the wild.

Virus-B

The development of 'Virus-B' by John McAfee's *InterPath corporation,* USA is another example of an 'attributable virus'.

Virus-B is not, in fact an entirely new specimen, but a modified version of the South African virus (which InterPath call X-12). Virus-B only infects .COM files and displays a clear infection message upon execution of infected programs. According to the documentation the virus will:

> *"...increase the size of the infected program by about 500 bytes. An infected program will cause no damage but it will be a nuisance if a large number of system's programs become infected."*

The documentation also acknowledges certain dangers including the possibility that *"Hackers could re-activate Virus-B to return to destructive mode".* It goes on to say that *"Such a person could just as easily write a virus from scratch if they were so inclined, but the potential for reactivation exists."*

Reference is made to 'built in protection mechanisms' which explains that the code segment for the destructive mechanism has been left intact ("*so that it may be analysed*") but that the branch instructions to these segments have been removed. A dire warning follows: *"DO NOT ATTEMPT TO DISASSEMBLE THIS VIRUS AND RECONNECT THE BRANCH INSTRUCTIONS".*

Virus-B was made available by InterPath as a 'restricted' access file' and was *"developed to be used in a research environment for studying virus replication activities and as a safe tool for testing anti virus measures".*

Problems abound with 'demonstration and 'test' viruses and one can only hope that *InterPath Corporation* exercised proper judgement in the people to whom it distributed Virus-B.

Burger and Morris

The case of Ralf Burger his 'VIRDEM' demonstration virus is discussed by Jim Bates on page 6 of this month's VB. It would appear that Burger distributed a number of different virus demonstration disks, some of which have now appeared in the wild. Burger's primary motivation appears to be financial gain from the virus phenomenon.

VB has, over the months, also covered the case of Robert Morris and the Internet worm program. Morris is the first man convicted under the aforementioned *US Computer Fraud & Abuse Act.* Intellectual challenge seemed to have been the motivating force behind the development and release of the program. This is not surprising from a computer science graduate at one of the principal US universities - Cornell.

The Bulgarians

Lubomir Mateev Mateev and Iani Lubomirov Brankov wrote and distributed the Murphy virus in Bulgaria - they included their telephone numbers and addresses in the source code which they circulated. No prosecution will result within Bulgaria which, like so many countries, has no applicable legislative power.

Sofia is home to a dedicated group of virus writers including 'T.P.' and Dark Avenger'. The writers of the Bulgarian viruses are almost certainly students from a research institute attached to the *Bulgarian Academy of Sciences* or from a faculty at *Sofia University* or the *'Lenin-V.I.' Higher Institute of Mechanical and Electrical Engineering.*

Apart from these rare examples, computer virus writers generally choose anonymity, although careful study of text strings and programming style can reveal details about the programmer's age, nationality and personality.

Other Possible Sources

Various motivations lie behind the development of the attributable computer viruses mentioned above. It is quite useful to speculate on other possible groups or individuals involved in virus writing and distribution.

A number of groups are readily identifiable as potential (high likelihood) originators of computer viruses.

Hackers and 'Technopaths'

In the book *Out of the Inner Circle,* the author Bill Landreth describes the various motivations behind computer hacking. He describes five hacker sub-classes - novice, *student, tourist, crasher* and *thief.* Of these sub-groups he identifies two categories as liable to inflict damage to computer systems. The 'novice' often causes damage unintentionally due to inexperience and carelessness but is also prone to vandalism. However, Landreth singles out the 'crasher' as

> *"a troublemaker motivated by the same elusive goals as a vandal. If it weren't for computers, her could just as easily be spray painting his name on the side of a building, or perhaps, even setting the building on fire".*

The author asserts that genuine hackers aspire to either 'student' or 'tourist' class and hate 'crashers' because the give hackers a bad name, they close accounts which hackers have spent time and effort to obtain and they crash bulletin board systems on which hackers communicate. The behaviour as described would be clinically defined as psychopathic. The computer world has adopted the term '*technopath*' to describe this type of personality disorder.

The willingness to inflict damage to computer systems makes the 'crasher' a potential computer virus writer.

```
000000  fa e9 4a 01 34 12 00 09  22 00 01 00 00 00 00 20  ..J.4... ``......
000010  20 20 20 20 02 20 57 65  6c 63 6f 6d 65 29 74 6f        We lcome to
000020  20 74 68 65 20 44 75 6e  67 65 6f 6e 20 20 20 20   the Dun geon
000030  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20
000040  20 20 20 20 20 20 20 20  20 20 20 20 20 20 20 20
000050  20 28 63 29 20 31 39 38  36 20 42 61 73 69 74 20  (c) 198 6 Basit
000060  26 20 41 6d 6a 61 64 20  28 70 76 74 29 20 4c 74  & Amjad (pvt) Lt
000070  64 2e 20 20 20 20 20 20  20 20 20 20 20 20 20 20  d.
000080  20 42 52 41 49 4e 20 43  4g 4d 50 55 54 45 52 20   BRAIN C OMPUTER
000090  53 45 52 56 49 43 45 53  2e 2e 36 33 30 20 4e 49  SERVICES ..730 NI
0000a0  5a 41 4d 20 42 4c 4f 43  4b 20 41 4c 4c 41 4d 41  ZAM BLOC K ALLAMA
0000b0  20 49 51 42 41 4c 20 54  4f 57 4e 20 20 20 20 20   IQBAL T OWN
0000c0  20 20 20 20 20 20 20 20  20 20 20 4c 41 48 4f 52              LAHOR
0000d0  45 2d 50 41 4b 49 53 54  41 4e 2e 2e 50 48 f4 4e  E-PAKIST  AN..PHON
0000e0  45 20 3a 34 33 30 37 39  31 2c 34 34 33 32 34 38  E  :43079 1,443248
0000f0  2c 32 38 30 35 33 30 2e  20 20 20 20 20 20 20 20  ,280530.
000100  20 20 42 65 77 61 72 65  20 6f 66 20 74 68 69 73    Beware  of this
000220  20 56 49 52 55 53 2e 2e  2e 2e 2e 43 6f 6e 74 61  VIRUS.. ...Conta
000120  63 74 20 75 73 20 66 6f  72 20 76 61 63 63 69 6e  ct us fo r vaccin
000130  61 74 69 6f 6e 2e 2e 2e  2e 2e 2e 2e 2e 2e 2e 2e  ation... ........
000140  2e 2e 2e 2e 20 24 34 40  25 24 40 21 21 20 8c c8  .... $#@  %$@!! ..
000150  8e d8 8e d0 bc 00 f0 fb  a0 67 7c a2 09 7c 8b 0e  ........  ..|..|..
000160  07 7c 89 0e 0a 7c e8 57  00 b9 05 00 bb 00 7e e8  .|...|.W ......~.
```

Figure 1.

The Brain virus was first reported after infecting floppy disks at the University of Delaware, USA, in October 1987. It has the distinction as being the first virus to strike world wide outside of a laboratory.

It is rare that it is in 'attributable virus' - its boot sector contains the Names of its originators along with their address in Pakistan. Today's virus writers generally choose anonymity.

Students

Most universities offer free, often uncontrolled, computer facilities to students (and often ex-students and even non-students). The conditions for both virus propagation and development are ideal. Illegal software copying is widespread, and virus attacks are continuously occurring at academic institutions.

The appearance of a virus at *Lehigh University* in 1987 (the Lehigh virus) which never infected any other sites indicates that the virus was developed on campus. There is also strong evidence to suggest that the Jerusalem virus was developed by a student or students at the *Hebrew University of Jerusalem*. The Italian virus is believed to have originated at the *Polytechnic of Turin.* The technical ability to write a virus is within the reach of first-year computer science student and, as in the case of Morris at *Cornell,* the primary motivation will be intellectual challenge.

This group are also a potential source of mini and mainframe viruses and worms. Whereas most members of the public can afford a cheap PC, they cannot easily gain access to an IBM 370 or a DEV VAX. The Internet worm is not the only example of this; the CHRISTMA EXEC 'Christmas tree' worm (VB, April 1990) originated at the *University of Clausthal Zellerfield*, West Germany.

Disgruntled Employees and Ex-Employees

Most organisations are acutely aware of the threat posed by this group. Although a computer literate employee might program a 'site-specific' virus, it is more likely that he/she would implant an existing destructive virus or add a destructive segment to a 'benign' (or demonstration) virus. Readiness to cause damage by programming has already been shown in cases of logic bombs being planted in computer systems by disgruntled employees.

Computer viruses, or the threat of unleashing such programs, could also be used during an industrial dispute as part of 'electronic picketing' or 'negotiation'.

Computer Clubs

In 1989 the *Chaos Computer Club* in Hamburg, Germany, devoted an entire private congress to the subject of computer viruses. Chaos have also released a 'Virus Construction Set' for the Atari ST and a diskette containing 'nightmare software'. A Chaos spokesman when asked what motivated the virus writer answered "You feel something wonderful has happened. *You have created something which lives. You don't know where it will go what it will do, and how it will live on".*

Other clubs have a history of creating viruses. The *Swiss Crackers Association* (SCA) released a virus for the Amiga which displays:

```
Something Wonderful has happened. Your Amiga is
alive...
```

Terrorist, Criminals and Politically Motivated Groups

There is no evidence, so far, that terrorist organisations have been involved in writing or disseminating computer viruses. However, the Italian *Red Brigade's* manifesto specifically includes destruction of computer systems as an objective, which should be done by means other than explosive or arson. In France, there is even an underground organisation dedicated to destroying information systems - CLODO - '*the committee to liquidate or neutralise computers'.*

The Jerusalem virus was reported in the *New York Times* as being written "as a weapon of political protest", but several researchers (including Yisrael Tadai of the *Hebrew University* who is the recognised authority on the Israeli Viruses) dispute this. The evidence to support this theory was that the original trigger date of the virus - May 13th 1988 - was the fortieth anniversary of the last day the Palestine had existed under British mandate. This virus is still referred to as the 'PLO' virus.

Computer viruses developed by terrorists and organised crime syndicates will probably make an appearance once their destructive capacity is realised and, significantly, once their potential as tools to commit fraud becomes more obvious. Computer viruses are an ideal way to cause disruption in order to conceal computer fraud. Rumours persist that the original Datacrime virus had been developed and circulated for criminal or terrorist purposes. It certainly caused a national panic among Dutch computer users in October of last year.

Future extortion bids, probably more targeted than that attempted by the '*PC Cyborg Corporation*' with their AIDS Information Diskette (*VB, January 1990*), will increasingly use destructive computer programs.

An underlying political motivation can be discerned in the on-screen messages of certain viruses - notably the Dukakis and Peace viruses on the Macintosh, and the Fu Manchu and New Zealand viruses on the PC.

More significantly, last year's threats involving the unleashing of computer viruses at poll tax offices in Scotland demonstrate an increasing awareness of the potential of these programs as political weapons.

COUNTERMEASURES

Virus Monitoring Software - An Endless Battle

The developers of interrupt monitoring software designed to intercept 'illegal' activity by computer viruses face increasing challenges - there are hundreds of undocumented features within DOS available to the aspiring virus writer.

Here, VB's Technical Editor looks at current virus programming techniques to evade detection by this type of software. The article also highlights the seemingly endless battle between the 'poachers' and the 'gamekeepers'.

Introduction

When the first computer viruses appeared, no anti-virus software had yet been written, so the virus writers had no need to utilise advanced methods for bypassing protection programs. This is no longer true. Protection programs are now installed on many computers and a virus which does not account for them is more vulnerable to early detection.

Designers of anti-virus software must, therefore prepare for viruses that make use of undocumented or obscure features of the DOS or BIOS to sneak past defensive software. Various methods of attack are possible and some of then are described below. Some of the more unusual methods are not used by known viruses but can be expected in the future.

Preventing Unauthorised Disk Writes

This is the single most important problem which developers of virus monitoring programs (memory resident TSR programs which monitor interrupt calls, the best known example of which is FluShot +) must address. There are other problems, for example the prevention of any tampering with data going to parallel and serial ports, but they are less important.

If unauthorised disk writes can be prevented, no virus will be able to spread.

Preventing unauthorised writes involves intercepting **all** write-requests, examining the parameters and taking any action required.

Possible actions include:

- Permit the request

- Ask the user for confirmation

- Return immediately with error status

- Produce an alert and halt the computer

The process becomes a duel between the virus writer and the monitoring program whereby the virus writer employs unexpected, obscure methods to bypass the software or to trick it into permitting invalid request.

Any totally effective virus monitoring program must be capable of preventing numerous requests for writing to the disk. Some methods are very straightforward to prevent but others are extremely difficult to intercept.

INT 21H

The most obvious way to write to a file is simply to open it with function 3D01H or 3D02H and write to it using function 40H. Virus monitoring programs therefore generally intercept this function. Unfortunately, they do not all intercept the other functions which can be used for writing to program files. Other INT 21H functions available to the virus writer are:

OFH, FCB open - possibly using an extended FCB

44H, IOCTL, subfunctions 5 and 0DH in particular

6CH, Extended open (DOS 4.0)

One trick is to open the file in Read-Only mode (function 3D00), but then modify the internal structures used by DOS, changing the access mode to Read/Write. The file can then be written to in the normal way.

The so-called 'Norwegian method' is similar. The file is opened in Read-Only mode and a part of it is read into memory. The disk buffers currently in use are then searched for the code just read and they are then modified in memory as necessary. Finally, the 'dirty bit' of the disk buffer is set and the file closed. DOS will then quietly write the modified data out to disk.

A further trick is used by several viruses. They hook into INT 21H, activating when an executable file is being written to. The virus stores the disk handle until the file is closed, when it will infect the file. This means that a simple COPY command from DOS can result in the infection of the program being copied. An anti-virus program might ignore

the write operation, as DOS is permitted to write to files. If it alerts the user to the fact that a program is writing to an executable file, the warning would probably be ignored - after all, the user is trying to write to the file.

Alternate DOS Entry Points

Anti-virus programs that monitor INT 21H are of little use if the virus is able to obtain the original INT 21H address and jump directly to it. A couple of viruses (Icelandic 2 and December 24th) do just this, but they are rare. The viruses contain a table describing the first few bytes of the original INT 21H routine in various versions of DOS. They obtain the DOS segment address by using an undocumented interrupt function (52H), and just scan the segment for the beginning of the INT 21H routine.

Instead of issuing and INT 21H call later, the virus will perform the following commands:

```
PUSHF
CALL  DWORD  PTR  CS:  [OLD_DOS_ADDR]
```

Another method for calling INT 21H without arousing suspicion is to use function 5D00H, which most anti-virus monitors ignore. This function can be used to call all other INT 21H functions.

Yet another method is to make a far call to location 5 in the PSP, with the function number in the CL register. This method is described in the DOS technical manual, but will only work for functions 00H-24H.

INT 26H and INT 13H

INT 26H and INT 13H are used for writing directly to the disk, bypassing the file system. Only a few programs have a legitimate need to do so including FORMAT, disk cache and disk repair programs, such as The Norton Utilities in additions to DOS itself. The problems facing anti-virus software is to determine whether a legitimate program is calling the interrupts or if the call is made by a virus. Using INT 26H and INT 13H is an obvious way to write to the disk, so anti-virus monitors usually intercept it.

INT 40H

If a hard disk is present, the original INT 13H vector will be redirected to INT 40H. As many anti-virus programs do not monitor this vector, it provides and easy method for writing to diskettes. This can be easily prevented by intercepting INT 40H, in the same way a INT 13H is intercepted. However, many current monitors ignore this vector.

ROM Jumps

On most IBM-PC compatible machines it is possible to write to diskettes simply by jumping to a fixed address in ROM. This address was hard-coded into DOS 2.0. This method is only used by the Ghostballs virus and is of limited effectiveness - it cannot be used to write to hard disks.

Some viruses are able to locate the original hard disk INT 13H entry point in ROM. This is true of a number of Bulgarian viruses, Dark Avenger and Murphy for example, both of which the authors made available as fully commented source code.

The Bulgarian viruses use two different methods for obtaining the INT 13H address. One involves scanning the ROM, but the other is a simple call to an undocumented INT 2FH function.

It is safe to predict that a number of future viruses will employ similar techniques for calling ROM directly. This is extremely difficult to stop. It is not, however, impossible and a couple of experimental anti-virus products are currently under development to intercept such calls.

Talking to the Controller

No known viruses use this method and it is unlikely that one ever will - it is just too difficult to program. This would require knowledge of the most common types of disk controllers available, but such a virus could then communicate directly with the hardware using IN and OUT instructions as necessary. Preventing this from working under current versions of MS-DOS seems practically impossible. This does not mean that viruses using this method cannot be stopped, only that virus specific programs must be used.

Extensive listings of undocumented MS-DOS features have been compiled and widely distributed and are being employed in 'second generation' or 'stealth' viruses.

Fortunately, these listings are also available to anti virus software developers. It is not, however, possible to accurately predict which features will be used in future viruses - this is one of the major problems faced by memory resident software which attempt to intercept 'illegal' calls or recognise 'viral behaviour'. The software also requires the user to exercise considerable judgement in responding correctly to its prompts as it detects 'suspicious' activity.

MAC THREATS

MDEF

MDEF, the nineteenth Macintosh virus strain, was discovered at *Cornell University,* NY, USA on May 16th.

The virus, also called 'Garfield', attacks application and system files by adding its viral resource and renumbering the system menu definition in the system file.

It does this by replacing the system file's native MDEF resource with a new resource called Garfield. The MDEF resource it part of the Macintosh menu generation system. The original MDEF resource is given an ID of 5378 and the substitute resource assumes a normal ID of zero.

The virus does not contain any trigger or 'payload' but propagates from system files to applications on the Mac Plus, SE, SE/30, II, IIx and IIcx. It dose not spread from system files to applications on the IIci or the IIfx.

Detection

The following search strings can be added to Virus Detective.

```
Resource MDEF & Name "MDEF"
Resource MDEF & ID = 5378
```

The Virus Detective 4.X search string reported by Jeff Schulman is:

```
Resource MDEF & ID=0 & WData 4546#58EA9AB#C3F#B6048
```

SAM 2.00 (Virus Clinic and Intercept) can be configured to find the virus during scans and applications launches with the new virus definition feature. Use the Add Virus Definition feature in Virus Clinic to add the following definition:

Virus Name:	MDEF
Resource Type:	MDEF
Resource ID:	0
Resource Size:	314
Search String:	2F3C434F44454267A9A0 (hex)
String Offset:	42

Add this definition to Virus Clinic and SAM Intercept. Update information is available from the Symantec Corp, USA, Tel 408 253 2167 or Symantec UK Ltd, Tel 0628 776343.

Virex version 2.7 has been updated to search and remove the MDEF virus. This is the thirteenth update to Virex since its release in November 1988. Microcom Software Division which markets Virex describe MDEF as 'unsophisticated' due its apparent inability to spread rapidly. Information is available from Microcom, USA Tel 617 551 1957 or Microcom, UK, Tel 0483 740763.

John Norstad's widely used Disinfectant 1.8 which identifies and removes the MDEF virus is currently available. Version 2.00 will be released within the next fortnight.

Apple Macintosh users are advised to use updated anti-virus utilities to search for and remove infection by Garfield.

The virus can be removed using ResEdit to remove the Garfiled MDEF and renumber the original MDEF ID=5378 back to ID=0. However, this operation should only be undertaken by the technically adept and is **not recommended to non-technicians.**

Trojan Horses

A Mac Trojan called 'Steroid' was reported by MacMash virus archive on June 5. Steroid is an INIT which claims to speed Quickdraw on Macintosh computers with 9 inch screens. The INIT scans for a date greater the June 6, 1990 and once ascertained, the Trojan proceeds to erase all mounted drives.

Disks which have been erased can reportedly be recovered using SUM II Disk Clinic (Symantec Utilities for Macintosh).

As is often the case with Trojan horses, the discovery of the 'Steroid' program came too late to issue a widescale and general warning prior to its triggering.

Trojan horses have been rate in the Mac world - the most notable recent incident s being the Mosaic and FontFinder programs which were uploaded to a BBS in February (VB, March 19990). The most notorious Macintosh Trojan was a HyperCard stack called 'Sexy Ladies', which displayed pornographic pictures while it erased data.

As *VB* reported in March 1990, the Trojan horse threat will continue until computer systems incorporate privileged operating system kernels with well defined entry points, hardware protection and memory management.

Acknowledgements to Information Systems Integrity & Security Ltd for their assistance in the preparation of this article.

PRODUCT EVALUATION

Dr Keith Jackson

Copy Protection: Virus Bulletin Policy

There have recently been a resurgence of copy protection techniques being used by companies selling anti-virus products. I have encountered many problems while testing such copy-protected programs, most of which have not been mentioned in print. As a consequence of this, and because we have no wish to inflict such problems on our readership, the editor of *Virus Bulletin* has decided that **form now on we will only review software which is not copy-protected**. I totally agree with this policy decision.

Secure operational methods involve taking backups to ensure that under no circumstances are you ever without a functioning copy of the original software. Copy-protection by its very nature circumvents this good practice, and forces the user into bad, insecure, habits where regular backups are difficult, if no impossible, to obtain.

The last straw which caused this decision to be taken was a copy-protected anti-virus product which did not explain how to deinstall the software. Upon telephoning the company involved, I was told that deinstallation was impossible, such a feature had 'not yet been implemented'. **Any company that expects a product to be installed forever on a computer is living in cloud-cuckoo land.** If the software had not been copy-protected, then removing it from the hard disk would be simply a matter of erasing the relevant files.

This article explains: *why we have decided to refuse to review copy protected software; how copy protection schemes work; and why using copy protected software is such a bad idea.* The companies who use copy protection have not been explicitly named in the following discussion; we have no wish to become part of a virulent debate on copy protection. There seems little to no point in going through the motions of a review, only to damn the product because of problems caused by copy protection.

Copy protection is probably justified with just one type of software: *computer games.* It is not the end of the world if the floppy disk containing the game becomes unreadable, and the copy protection scheme may even help matters by preventing the game from being installed on a hard disk. **For all serious computer usage copy protection should be avoided.**

What is Copy Protection?

Copy protection attempts to prevent a floppy disk from being copies under any circumstances. It is an attempt by manufacturers to ensure that only the legal owner of a software package actually has use of the software. This appears to be a noble aim, but in practice copy protection brings with it many undesirable side-effects, which can be so severe as to render copy protection schemes unacceptable.

For a copy protection scheme to work, it is imperative that some part of the original disk cannot be copied by the operational system under which the program executes. This condition is fulfilled in many different ways by the various methods of copy protection.

The simplest methods of copy protection use floppy disks with deliberate **bad sectors** written to them. Bad sectors cannot be copied by the operating system, as they look like errors in the disk media. Copy protection schemes know exactly how the bad sector(s) have been written to disk, and can retrieve any information they might contain. This type of code protection often extends to writing extra disk tracks in the area beyond the highest numbered track that can be addressed by the operating system. Other methods rely on the copy protection software recognising the existence of a physical defect (e.g. a small hole made by a laser) at a known location on the surface of the floppy disk.

Some types of copy protection write deliberate bad sectors to the hard disk during the installation process. Such sectors are again only readable by the copy protection software, and not by the operating system. It is usually imperative that these bad sectors remain at their original location on the disk for the copy for the copy protection to be able to find them, therefore utilities which rearrange the structure to the hard disk, can cause chaos if they are not used extremely carefully when such copy protection methods are in use.

"Any company that expects a product to be installed forever on a computer is living in cloud cuckoo land"

Most software packages do not advertise that copy protection is being used (are the manufacturers embarrassed by it perhaps?), and the presence of sectors that must remain at absolute disk locations may be completely unknown to the user. This can have dire consequences.

A different type of copy protection requires a small device known as a 'dongle' connected to one of the computer ports before a copy-protected program will execute.

The word dongle does not appear to mean anything. It seems to have simple evolved in response to the need for a name. Software that relies on a dongle is usually less problematic than other types of copy protection, as the copy protection method does not prevent multiple backup copies of the software being created. However it can run into trouble if many software packages on the same computer all require their own individual dongle. Simple dongles are little more than a device which provides a serial number on request while more complicated dongles execute a part of the program being protected on a processor contained within the dongle.

More bureaucratic schemes of copy protection rely on the user telephoning the manufacturer for a code number which is required before installation can proceed. Before the code number is released. The manufacturer usually asks questions to establish that the software is the result of a valid purchase, and is not a 'pirate' copy. This type of copy protection can prove a nightmare if installation has to be performed many times, if the manufacturer of the software is at a remote location (perhaps with an appreciable time zone difference), or if the manufacturer and the user can not communicate readily in a common language.

Finally there are copy-protections schemes which require the original master disk to be present every time that the software is executed. Pity the poor user who formats this master disk because it has accidentally been left behind in a floppy drive. If every software package required this type of copy-protection, imagine the consequences of needing up to a dozen master disks. *This is probably the worst type of copy protection: the floppy disk cannot be copied, but must be used every time that the program is executed.* The disk is likely to wear out quite quickly, and no matter what manufacturers may say as part of their sales pitch, they are always reluctant to replace misplaced or faulty master disks. If they replaced disks at a moments notice, the point of using copy protection would largely be negated.

What Types of Problem Can Be Encountered?

I have often been sent evaluation copies of anti-virus products which insisted on being installed using a specific hardware configuration. Insistence upon certain drives being used is often a consequence of using a copy protection scheme. It is inevitable that the software capable of operating the copy protection scheme reads data from the disk using its own resources, rather than the resources provided by the operating system, therefore the drive names are usually fixed when the software is written. If the operation system resources could be used, then by definition the disk would not be copy protected.

Very often the floppy disks provided with a software package are not the correct type for the computer used for testing. This can usually be circumvented by copying the disks to the required type of disk, and I have facilities to make copies to/from all types of floppy disks in common usage. However such tactics do not work when the software is copy protected, for instance while installing a package for a review, it quickly became apparent that the software was copy protected (even though the documentation did not mention this), and the installation process insisted on having the copy protected disk in drive A. The computer I usually use for test has a 3.5 inch disk as drive A, and 5.25 inch disks as drives B and C. Unfortunately the software came on 5.25 inch floppy disks.

Without reconfiguring the innards of my computer, something I'm reluctant to do on a system that has given me no trouble, I could not install the software as it insisted on loading from drive A. As the disks were copy protected it was inherently impossible to copy the 5.25 inch disks to 3.5 inch floppy disks (which would have solved the problem). Having reached this impasse I telephone the company involved, and they agreed that the only solution was for them the provide 3.5 inch copy protected disks. Delivery was promised in a few days, but actually took over four weeks, several impatient telephone calls, and a couple of letters, to arrive.

In another instance, I tried to install an anti-virus software package, again without knowing that it was copy protected, and encountered the same insistence on the floppy disk being inserted in drive A. Installation failed because the copy protection mechanism intervened, and I tried to obtain a 3.5 inch version to complete the review. However I found out later that the installation process had made a hidden subdirectory on my hard disk (with non-visible characters in the name ensuring that it was difficult to erase), containing two hidden files. The Norton Utilities was required to get rid of this debris. *Nothing in the documentation warned me that this would happen. Indeed many users would not even notice the presence of these hidden files in a spurious, curiously named, subdirectory.*

In summary, any software package that prevents one from taking as many backups as desired, requires some form of special hardware, or requires a floppy disk to be present at all times should be avoided. Copy protection prevents taking proper backups which are the first (and most vital) line of defence against viruses. How many book publishers do you know who print books on paper with special faint ink that cannot be photocopied? Any publisher who attempted this would immediately find sales plummeting, as the side-effects would outweigh the usefulness of the copy protection scheme.

Legal remedies are available to help prevent distribution of software in a manner that contravenes the manufacturer's licence agreement. Such remedies should be used in preference to the mirage offered by copy protection, which solves the problem if illegal copies at the user's expense.

Restating a conclusion from a previous review 'Life is complex enough without the ritual dance imposed by copy protection'.

END-NOTES & NEWS

The Japanese Ministry of International Trade has drafted a set of guidelines for combating computer viruses. According to *Computer Fraud & Security Bulletin* systems managers will be required to report all virus outbreaks to the *Japanese Information Promotion Agency.* Only two Japanese virus outbreaks have been reported to date. *The University of Tokyo's Earthquake Research Unit* was hit in September 1989 by two PC viruses and an NEC computer network also became infected earlier this year.

EVENTS

Information Security: Confidentially, Integrity & Availability, July 3-5, 1990, London, UK. *Unicom Seminars*, UK

An intensive two day seminar '**How to Recover Ddamaged Data, Files and Disks'** takes to the road in July. The course includes tuition on disk structure and format, DOS recovery utilities, hard disk drives, Norton, Mace, PC Tools deluxe, CHDSK,DEBUG, EDLIN and programming solutions for data recovery. Venues include London, Dublin and Leeds with the first course taking place in Manchester (23-24 July). Details from *IIR Technology,* UK. Tel 071 622 5444.

Implementing Data Security in the Financial Industry. A two day seminar from IBC Technical Services, UK, 24-25 September 1990. Encryption algorithms and standards, retail and corporate banking, key management, network security etc Tel 071 236 4080.

Local Area Network Security, July 23-24, Boston, Massachusetts, USA. *System Technology Institute*, USA, Tel 818 888 6100.

Computer Virus Workshop, July 24-25, Oxford. Technical and managerial streams available. *Sophos*, UK, Tel 0235 559933.

Disaster Recovery & Restoration of Networks, August 13-14, Minneapolis, USA. *Data*-Tech Institute, USA, Tel 201 478 5400.

Compsec '90, a three day conference on all aspects of computer security, October 10-12, 1990, QE II Centre London. *Elsevier Seminars*, UK, Tel 0865 512242.

VIRUS BULLETIN

Subscription price for 1 year (12 issues) including delivery:

US$ for USA (first class airmail) $350, Rest of the World (first class airmail) £195

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, Haddenham, Aylesbury, HP17 8JD, England
Tel (0844) 290396, International Tel (+44) 844 290396
Fax (0844) 291409, International Fax (+44) 844 291409

US subscriptions only:

June Jordan, Virus Bulletin, PO Box 875, 454 Main Street, Ridgefield, CT 06877
Tel 203 431 8720, Fax 203 431 8165