February 1991

ISSN 0956-9979

# VIRUS BULLETIN

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

Editor: **Edward Wilding**

Technical Editor: **Fridrik Skulason**, University of Iceland

## CONTENTS

# EDITORIAL

## A Troubled World

It would be profane and disproportionate at a time of such intense international crisis to talk of ‘war’ and ‘warfare’ in any context other than mortal combat - this journal has in the past made occasional reference to the ‘computer virus war’; the analogy may be apposite when peace prevails but is probably best shelved for the time being.

It would also be futile, as the level of conflict and violence intensifies, to rain emotional invective on those people who write computer viruses. Their activities, which many countries have designated crimes, are seemingly trivial (but not completely inconsequential) in the face of current world events.

It is now obvious that many lives will be lost in 1991 - a year that has already wrought a succession of sobering images in the electronic and printed media worldwide. Matters of life and death invariably sharpen peoples’ perspectives and help us to regain a sense of proportion. Regardless of individual loyalties and conscience, it is important to will *peace*, *justice* and *progress* for all people, everywhere.

This journal’s function is to report on a technical threat to computers - software, hardware and data. It attempts to address an increasing, but not yet overbearing problem which besets computer users in all the developed and developing nations of the world. Stopping this problem at its source will depend on *reason*, *clarity* and *logic*, both on the part of those people seeking to curtail the threat and from those who are actively promoting it. *In this respect, computer misuse, albeit unlikely to cause extreme trauma, is not dissimilar to the greater issues which trouble the world*.

In the very first *VB* editorial in July 1989, the indiscriminate nature of computer viruses, which victimise in a *random* and *unpredictable* manner, was presented as one of the clearest reasons for the virus writers to desist from their activities.

Parallels with terrorism are perhaps drawn too easily here; computer viruses are not *yet* designed to kill, although the repercussions of a multitude of safety critical systems being attacked by these means might well involve death and injury. *Reason*, *clarity* and *logic* combined may even (if certain academics and computer industry experts are to be believed) provide a rationale for developing such programs.

*It remains, however, stupefyingly difficult to find a rationale for the sort of vandalism which manifests itself in the random, indiscriminate destruction of peoples’ data and programs.*

Temporarily discounting matters of *peace* and *justice*, where is the *progress* in all of this? Inflicting such damage is more than just a hindrance - it is patently regressive.

The apologists for these activities invariably argue that it is the big organisations - the multinationals, the banks and all the other institutions that supposedly ‘oppress’ - which suffer most from computer misuse. In fact, these organisations are well aware of the dangers, are well defended and can respond quickly and appropriately to the threats they face.

The *real* victim of computer viruses is, and increasingly will be, the individual - be it the computer user; wholly dependent on his data, ignorant to the threat and woefully ill-prepared to recover from the effects of malicious software, or the ‘man in the street’; temporarily or permanently inconvenienced by essential medical, welfare, financial or other personal data becoming corrupted or inaccessible.

We are all responsible for our own actions and the people who develop and propagate viruses should, at the very least, realise that they are responsible for impeding other peoples’ freedom, creativity and progress indiscriminately. Millions of peoples’ livelihoods and welfare are now dependent on the humble personal computer - to attack such systems is irresponsible, if not palpably wicked.

The world is troubled and faces enormous dangers which makes many other problems appear quite inconsequential. *However, every responsible course of action by every individual, whatever his particular field of interest or knowledge, serves to lessen the world’s problems in some small but significant way*. This is as relevant to computer programming as it is to all other endeavours.

# TECHNICAL NOTES

## The Write-Protected System Floppy Disk

Long term readers of *VB* will be familiar with our continual warnings of the dangers of using scanning programs or disk utilities to search for viruses in the event that a ‘stealth’ virus, or a virus which infects program files as they are *opened*, becomes resident in memory. In the former case the virus will become ‘invisible’ providing no indication of infection, in the latter all files searched will become infected.

The need to repeat the warning has been re-emphasised by the release of the *Norton AntiVirus* (*VB*, Jan 90). The documentation with this product only mentions the **write-protected system disk** (or write-protected DOS disk as it is referred to) on page 20 of the manual (section QS 2) without sufficient emphasis.

This may lead the user to commence scanning directly in an infected DOS environment - the *Norton AntiVirus* does not claim to search for viruses in memory and tests have shown it unable to do so.

VIRUS BULLETIN **©1991 Virus Bulletin Ltd, 21 The Quadrant, Abingdon Science Park, Oxon, OX14 3YS, England. Tel (+44) 235 555139. /90/$0.00+2.50**

*SymantecUK*, which markets the product, has been informed of this oversight and *VB* understands that plans are underway to emphasise this essential requirement in the documentation and/or a banner. This problem is not confined to this particular product; unfortunately, anti-virus software developers often automatically assume that the user will be aware of the need to scan in a clean DOS environment and warnings are often omited or muted. **In fact, this requirement is fundamental to combating computer virus infections and cannot be over-emphasised**.

For the benefit of new readers and as a general reminder, a brief resume of the the contents and use of the write-protected system floppy disk follows:

> **A write-protected system floppy disk should be prepared and made available as an essential contingency measure for combating computer virus infections.**
>
> This disk contains all MS-DOS system files which are transferred from the hard disk to the floppy by issuing the command 'FORMAT A:/S'.
>
> The floppy disk should also contain the set-up and configuration files known as AUTOEXEC.BAT and CONFIG.SYS and any other system files or device drivers such as ANSI.SYS. These files are transferred to the floppy disk by using the DOS COPY command.
>
> Note that CONFIG.SYS usually refers to other files which load into memory before the system starts, using statements such as 'DEVICE = filename'. All files should be copied onto the floppy disk, and CONFIG.SYS should be modified to refer to files on the floppy disk, rather than files on the hard drive.
>
> The system disk must be **write-protected**, this is a hardware protection against the modification of the floppy disk by a virus or write command.
>
> **Should a computer become infected, or an infection be suspected, this floppy disk will be used to boot the computer ensuring that any items examined using anti-virus tools or disk utilities are 'viewed' through a clean DOS environment denying a virus the chance to employ hiding mechanisms such as interrupt interception.**
>
> In the event that developments go awry and that no clean write-protected system floppy disk is available, the infected or suspect machine will have to be inspected using backup copies of the **master MS-DOS disk**. (See also pp. 9-10). In this instance, configuration files, device drivers and other specific system information will be reconfigured manually.

*VIRUS BULLETIN*

## EDUCATION, TRAINING AND AWARENESS PRESENTATIONS

Education, training and awareness are essential as part of an integrated campaign to minimise the threat of computer viruses and malicious software.

From March 1991, a *VB* representative will be available to visit subscribing organisations and deliver a presentation about the threat and the measures necessary to minimise it. The presentation, aimed at computer users and non-technical staff, can be tailored to comply with individual company requirements and procedures in coordination with management.

These presentations are offered free of charge except for reimbursement of travel and any accommodation expenses incurred. Information is available from *Virus Bulletin*, UK. Tel 0235 555139.

### Direct Port Access

Examples of virus code continue to come in to researchers in ever-increasing numbers. Fortunately however, the number of new techniques used by the virus writers is diminishing and the task of detecting generic virus activity is thereby becoming somewhat easier. Most of the simpler parasitic virus types can be fairly easily classified and their capabilities are already well-known and adequately catered for within existing detection software.

Occasionally some new, unusual or unforeseen development necessitates some minor modification to existing detection/ prevention techniques and this is where detailed disassembly of the particular virus involved is so valuable.

An example of this was found recently in a virus known to be at large in Russia. It arrived at *VB* under the name of "ATTENTION" although disassembly of the sample revealed that the name was actually a part of the infected host program. However, it is referred to here by this name to avoid yet more confusion over nomenclature.

The virus is a small one (infective length is 377 bytes) and the major part of the code is unremarkable. There is no trigger routine (although there may be some additional strain placed on the floppy drive motor), the code simply replicates among files with an extension ending in “OM” (this obviously includes all COM files) where the length is between 786 and 64921 bytes inclusive.

Infection is invoked during the DOS LOAD/EXECUTE function (4BH), appending the virus code to the file and modifying the original host jump (having first saved the original values). During infection, file attributes are modified and then reset so that READ ONLY and HIDDEN files are equally vulnerable. The original file date is not maintained and infected files will show the date of infection when a DIR listing is done to the screen.

The interesting section of the code occurs within a Critical Error handling routine which the virus installs to the INT 24H vector. No attempt is made to check or link to the existing handler, and the new handler address is re-installed during each LOAD/EXECUTE request. Within this handler routine, after the flags and major registers have been saved on the stack, a retry count of three is set up and the code then goes into a timing delay loop before addressing the floppy disk controller *directly* through its port. The data mask is set to No Reset, Enable INT and DMA access and turn the drive motor off. Then there is another timing delay loop before the port is accessed again but this time with the Motor On bit set in the data mask.

This sequence is executed three times (via the retry count) and the routine finally restores the registers and returns with a value of three in the AL register. No immediate damage or corruption is caused by this routine, although it is possible that continued ON/OFF switching in this way might cause excessive stress to the drive motor.

One of the areas which is awkward to monitor within a PC environment is that associated with direct port access. This virus is the first known to *VB* which uses it (albeit for unclear reasons). **The virus accesses the floppy disk drive controller directly rather than through the ROM BIOS functions. Hence, this routine within the virus does not alter any interrupt vector**.

It has long been expected that virus code would appear which accessed disk drives by bypassing interrupt routines.**Most current anti-virus software scanners and all TSR monitors direct their attentions towards interrupts** Intentional and refined use of such unconventional tactics to subvert defensive software should be expected. There has also been ongoing speculation about the ability of computer viruses to cause hardware damage; in this instance it would appear that no damage is intended to the drive motor nor, indeed, is it likely to occur. However, the possibility of a virus inflicting hardware damage should not be discounted. Fortunately, the anti-virus community is forewarned and such techniques have been anticipated.

## Kamikaze - The Problem With HLL Viruses

The signature string for the Kamikaze virus in last month’s*VB* turned out to be unusable as it produced an unacceptably large number of false alarms. **An updated pattern for detecting the Kamikaze virus appears on page 8**

A false alarm (or ‘false positive’) may occur from one of two different causes. One *highly unlikely* possibility is that a block of random data just happens to contain the string. This could possibly happen if a large number of programs was packed, using *LZEXE* or a similar compression utility, as the resulting programs have a pseudo-random ‘look’ and can contain almost any sequence of bytes.

Long identification strings reduce this risk - and by using 16-byte strings as *VB* does, the chances of this happening are minimised. A much more common reason is a poorly selected identification string. In the case of the Kamikaze virus, false positives occurred because the virus is written in a high-level language (HLL), where the executable code is generated by a compiler. Any other program containing the same sequence of statements as the one which generated the original identification string will probably produce a false alarm. In simple terms, the compiler interprets a variety of different source code instructions in a *standard* fashion. This makes the selection of suitable identification strings for HLL viruses much more difficult than ordinary assembly language viruses.

## Processor Specific Viruses

Some of the first PC viruses functioned as intended on the 8088/8086 processor, but failed to replicate on 80286 or 80386 computers. The best examples of this are two of the early boot-sector viruses:

- **The Italian virus**(Ping-Pong, Bouncing Ball).

  The standard version uses the MOV CS,AX instruction (8EC8 Hex) which executes correctly on the 8088 and 8086 family of processors but which is trapped as an illegal instruction on 80286 or 80386 processors. An 80286 variant of the virus has been reported, but the only available sample does not seem to replicate.

- **Alameda** (Yale).

  The first version of this virus used the POP CS instruction (0F Hex), for the same purpose - which also generates an “invalid instruction” interrupt on later processors. Later versions, however, corrected this, and the original variant is now probably extinct.

Two of the recent viruses from Eastern Europe fail to execute on the 8088 and 8086 processors, but works perfectly on 80286 and 80386 computers.

The reason is the use of the PUSH IMMEDIATE instruction (opcode 68H), which did not exist on the 8088/86.

This instruction is used to transfer control back to the original host program, in the following way:

```
        PUSH     100H
        RET
```

This is one byte less than other common methods, which are:

```
        MOV      DI,100H
        JMP      DI
   and
        MOV      AX,100H
        PUSH     AX
        RET
```

It is improbable that the author used the PUSH 100H instruction to enhance optimisation - its use reduces the length of code by just one byte. This one byte effectively neutralises the virus on 8088 and 8086 machines. From the virus-writer's viewpoint this would be a high price to pay for meagre optimisation. It is far more likely that he was unaware that this instruction only exists on 80286 and later processors.

From a forensic viewpoint, this fragment of information about the development processor is one of a number of clues which can help researchers and analysts develop an overall picture of the programmer, his probable depth of experience and likely areas of programming knowledge.

### Overlay files

Overlays may be contained in a separate file (typically .OVL or .OVR) or they may be an integral part of an EXE file. The latter case presents particular difficulties for both virus-writers and anti-virus programmers.

A case in point is *Turbo C++*. This program is over 800 Kbytes bytes long - too large to be loaded into memory all at once. According to the header, the program is approximately 165 Kbytes. Upon execution only 165 Kbytes of the program are loaded into memory - the rest (approximately 635 Kbytes) will be loaded by the program itself as and when it is required.

A typical EXE virus, which normally appends itself to files, has two options when targeting this file. It may add its code *into* the file, just after the 165 Kbyte mark. The Jerusalem and 8-Tunes viruses do just this. The infected file will load, and the virus will work properly, but the infected program may fail to operate, causing the virus to be detected.

When this type of virus infects a program such as *Turbo C++*, the infection may not be detected by a scanning program, as the scanner may have been written in a highly specific way, i.e. it will only search for viruses such as Jerusalem near the *end* of infected files. As the virus is located *elsewhere* in the file, a specific pattern scan may fail to find it. When a file is infected in this way, it is not possible to develop a disinfection program to remove the virus because a section of the target file has been overwritten. The only possible solution is to delete the file, and replace it with a 'clean' copy.

A virus may also append itself to the *actual end* of the program, thus increasing its length in true parasitic virus convention. In this case anti-virus programs should be able to detect and remove the virus, but the infected program may not function correctly. This is the case with *Turbo C++*. The length of the program combined with the virus will exceed 800 Kbytes. In this instance, execution will fail, as a program of that size cannot be loaded into memory.

## The Computer Virus 'Underground'

A further insight into the virus-writing mentality can be gained by reading an edition of *Corrupted Programmers International* (sub-headed "C.P.I. We ain't the phucking Salvation Army."). This electronic publication appeared on underground BBSs in the United States in 1989.

Following a series of disclaimers ('FOR INFORMATIONAL PURPOSES ONLY, Remember we may talk alot, but we "just say no" to doing it.'), 'Ashton Darkside' describes a variety of virus-writing techniques to minimise detection and maximise destruction. Source code for a 'generic' virus (which is in fact crippled) is posted in its entirety as is that of a destructive virus called AIDS created by 'Doctor Dissector' (not the AIDS virus reported by *VB*). Predictably, Ralf Burger's *Computer Viruses: A High Tech Disease* (*VB*, October 1989) is plagiarised in a tutorial on BASIC viruses.

Fundamental 'ethics' are expounded:

```
And remember, don't target P/H/P boards (that's
Phreak/Hack/Pirate Boards) with ANY virus. Even
if the Sysop is a leech and you want to shove his
balls down his throat. Because if all the PHP
boards go down (especially CPI), who the hell can
give you all those nifty virus ideas? And
besides, it's betraying your own people, which is
uncool even if you are an anarchist. So target
uncool PD boards, or your boss's computer or
whatever, but don't attack your friends. Other
than that, have phun, and phuck it up!
```

A *CPI* 'enrollment form' appears at the end of the edition demanding detailed information about the prospective candidate including his contact addresses, telephone and modem numbers, police, government or telephone company connections ("YOU KNOW WHAT WE MEAN"), programming skills and other information. One of the questions reads: ''*Why would you ever want to release or aid in releasing a potential virus/trojan to the public? Answer in 4 Lines Or Less*''.

Why indeed?

# INVESTIGATION

***Fridrik Skulason***

## The Search for Den Zuk

The virus known as Den Zuk was discovered over two years ago, and infections have occasionally been reported since then. It is a boot sector virus with one major effect and one, probably unintentional, harmful side-effect.

One of the earliest reports of the virus came from Venezuela - leading to the incorrect conclusion that it was written there. The virus was instead written in Indonesia, where several related viruses are known to exist. It contains the following text message, which is not displayed.

```
    Welcome to the
       C l u b
   —The HackerS—
       Hackin'
    All The Time

     The HackerS
```

On a computer infected with this virus, pressing Ctrl-Alt-Del will not result in a simple reboot. If the computer has a colour display, it will display a picture on the screen for a fraction of a second. The picture shows the text “DENZUKO” and an unknown logo. Pressing Ctrl-Alt-F5 will reboot the computer, without displaying this picture. Ironically, the screen-effect eases detection and reduces  the virus’ chance of spreading.

### Seek and Destroy

It had been thought that “Den Zuk” meant “The search”, a reference to the ability of the virus to seek out and destroy copies of the Brain virus. If it finds a Brain-infected diskette, it removes the infection and replaces it with a copy of itself.

Normal 360 Kbyte disks only have tracks numbered from 0 to 39, but this virus was the first to use track 40 on diskettes - a practice which is now becoming more common. The author did not cater for 1.2 Mbyte or 3.5 inch diskettes on which track 40 is used. On these diskettes, the virus will overwrite that track, possibly damaging data or programs stored there.

The volume label “(c) Brain” on a Brain-infected diskette is changed to ‘Y.C.1.E.R.P’ - A mysterious text -*but it turned out to lead directly to the author*.

Den Zuk also removes another virus - which was (correctly) assumed to be an older version of itself. This variant was discovered much later, and is generally known as ‘Ohio’.

It is closely related to the Den Zuk virus, but it contains different text messages:

```
          V I R U S
             b y
         The Hackers
         Y C 1 E R P
        D E N Z U K O
        Bandung 40254
          Indonesia

(C) 1988, The Hackers Team....
```

### Who is Y.C.1.E.R.P ?

To a radio-amateur, ‘YC1ERP’ looks like a call-sign. Reference to the *International Callbook* revealed that this call-sign had been allocated to a person in Bandung, Indonesia.

There was no *proof* that this person was the author of the virus - it was also possible that the genuine virus writer bore a grudge against him, and included his call-sign in the virus to discredit him.

Obviously, the easiest way to discover whether this person was indeed the author was simply to ask him. A polite letter was sent to the Indonesian radio-amateur, asking  whether he was the author or not.

His reply is published here verbatim and in its entirety.

```
                          Bandung, September 20, 1990

Dear Mr. Skulason,

First, I want to introduce myself too.

Name:       Denny Yanuar Ramdhani
D.O.B.:     January 16. 1964
Address:    Jl. Ancol Timur XII/10
            Bandung 40254
            Indonesia

Occupation: -Student at PAT-Komputer Institut
Teknologi Bandung.

            -Freelance System-Programmer.

I want to explain about names which related with
viruses.

- Den Zuko is from Denny Zuko, my nickname.
  (from 'Danny Zuko', John Travolta's name at
  'Grease' the movie !)

- Hackers is from Hackers Technology, my hackers
  club.

- YC1ERP is my amateur radio callsign
```

Contd.

And about two viruses, DEN ZUKO and HACKERS:

The viruses were first Indonesian viruses. The designer and author is me. Viruses name is DEN ZUKO and HACKERS (not DEN ZUK and HACKER). Created on March 1988 in Bandung, Indonesia (not Venezuela, as reported in New York Times ?) The viruses were my experiment in PC operating system, low-level language, how fast its spread, and just to "say hello" to other hackers/computer users in my city (when they pressed Ctrl-Alt-Del !). I never thought or expected its spread nationwide and then worldwide. So, I was really surprised when I read 'Tempo' (Indonesian weekly news magazine) which reported about 'Den Zuk' virus (quoted from New York Times) attacked USA, cominf from Venezuela, but I'm sure it was Den Zuko. And what made me really sure it was my virus, when I checked diskette which invected by Denzuko virus with Turbo AntiVirus, IBM VirScan and Mc Afee Accociates Scanner, its reported the diskette contains 'Den Zuk' virus.The viruses have 2 versions:

Ver.1: - DenZuko, the color is white, the shadow is red - Hackers Technology (text explode),
Ver.2: - DenZuko, color is red, without shadow. - Hackers, color is white ('K' is red) shadow is cyan.

Version 2 will replace version 1 and Brain virus if find, and it has immunization for version 1 and Brain. Version 1 will replace & immunize Brain virus (Pakistani Brain) The viruses have stadium level counter at 2nd sector (sector 022H) track 028H, offset 03H (1 word). So, we can count the approximation of the virus population.

Version 1 identification is hexadimal (1 word) 0FAFAh (offset 02Bh)
"  2  "  "  "  "  0537Ch (offset 040h)

About other viruses from Indonesia:

The others are modification from Den Zuko or Hackers (except file-invected-virus, like Amoeba and Mystik). Some of virus researcher, Iknow them, but I don't have their address.

For more information, maybe you can get them, if you contacy Mr P.M.Winarno, he is the editor of MikroData and InfoKomputer Magazine. His complete address is:

P. M. Winarno
Editor Mikrodata
Jl Palerah Selatan 22 Lt. 3
Jakarta 10270
Indonesia
Phone 62-021-5483008 ext 3211,3212

If you contact him, say hi from me. Oo, I almost forget some information:

- Before stadium level 3 the viruses will not show DENZUKO or HACKERS logo, and will not change label to Y.C.1.E.R.P.
- There is a secret keys Crtl-Alt-F5, if you press them, the computer will reboot without show the logo.
- The others which modified from version 1 of mine, destroyed by version 2 and decreased the population.
- Location of the others various at track 028H, 029H, head 0 or 1.

If you want to put my statement and publish it, you can take it from this letter, all or portion of it is up to you. I have some questions too:

- Who send you information about Indonesian viruses ?
- How can you get my address, from amateur radio callbook, from one of my viruses at track 027H, sector 022H or else ?
- Why 'Hacker' = 'Ohio' ? I never give or put 'Ohio' in HACKERS.
- When did you get 'DENZUKO' or 'HACKER' and what is the stadium-level counter number (track 028h, head 0, sector 022h, offset 03h (1 word) ?

Sincerely

––denny––

(DENNY YANUAR RAMDHANI)

# KNOWN IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of January 28th 1991. Hexadecimal patterns can be used to detect the presence of the virus by the 'search' routine of a disk utility program or, preferably, a dedicated scanning program. The full table was published in the January 1991 edition.

### SEEN VIRUSES

**403** - CR: Destructive, overwriting 403 byte virus.

```
403            342E 8926 0301 2E8C 1605 012E A307 018D ; Offset 093
```

**Akuku** - CER: 889 byte virus, probably written by the same author as the Hybrid virus.

```
Akuku          E800 005E 8BD6 81C6 2A01 BF00 01A5 A481 ; Offset 24E
```

**AntiCAD-2576** - CER: Yet another variant of the AntiCAD/Plastique series from Taiwan. This 2576 byte variant is closely related to the 2900 byte variant reported in last month's *VB*.

```
AntiCAD-2576   595B 5807 1F9C 2EFF 1E3B 001E 07B4 49CD ; Offset 550
```

**Christmas Violator** - CN: A 5302 byte variant of the Violator virus.

```
Xmas Violator  11AC B900 80F2 AEB9 0400 ACAE 75ED E2FA ; Offset 1EB
```

**Doom2** - CER: This 1252 byte virus is not always able to infect files. The machine hangs immediately after a file is infected.

```
Doom2          803E 0A01 4574 052E 033E 0301 2E30 0547 ; Offset 017
```

**Hybrid** - CN: A 1306 byte encrypted variant of the Vienna virus which also marks infected files by setting the 'seconds' field of the timestamp to 62. On any Friday the 13th after 1991 the virus will format the hard disk. It may also overwrite files and cause reboots.

```
Hybrid         81EE 7502 8BFE B9DE 01AC 34DE AA49 75F9 ; Offset 007
```

**Leprosy** - CN: A 666 byte encrypted overwriting virus, similar to Leprosy B but using a different encryption method.

```
Leprosy        558B EC56 8B76 04EB 0480 2C0A 4680 3C00 ; Offset 25C
```

**MIX2** - CER: This is a 2280 byte Israeli virus based on MIX1 but improved with the addition of encryption and COM file infection.

```
MIX2           EE8C C803 C650 B826 0050 CB55 508C C0E8 ; Offset 01B
```

**Monxla-B** - CN: This 535 byte virus is probably an older version of the Time/Monxla virus reported in *VB*, January 1990.

```
Monxla-B       8994 1600 B42C CD21 80E6 0775 10B4 40B9 ; Offset 128
```

**Ontario** - CER: A 512 byte encrypted virus. It uses self-modifying encryption, and a full 16-byte search pattern cannot be extracted. The asterisks in the following string indicate a byte which may change from one infected file to another.

```
Ontario        8A84 E801 B9E8 01F6 **2E 3004 46E2 F8C3 ; Offset 1F0
```

**Paris**, TCC - CEN: The virus will infect all EXE files in the current directory, when an infected file is run. Length is 4904 bytes.

```
Paris          8CD8 03C3 8ED8 8EC0 8D3E 0301 B000 AAEB ; Offset 7EE
```

**Perfume-731** - CR: A slight variant of the Perfume virus, only 731 bytes long. This may well be an earlier variant.

```
Perfume-731    FCBF 0000 F3A4 81EC 0004 06BF BC00 57CB ; Offset 1AC
```

**Sentinel** - CR: This virus is written in *Turbo Pascal* and is 4625 bytes long.

```
Sentinel       FCAD 2EA3 0001 AC2E A202 0189 EC5D B800 ; Offset variable
```

**Spyer** - CER: This 1181 byte virus from Taiwan will always hang after an infected program is executed.

```
Spyer          8B36 0101 03F7 FCF3 A450 C38B 3601 01BF ; Offset 014
```

**SVC 3.1** - CER: This 1064 byte virus is probably an older version of the SVC virus.

```
SVC 3.1        C39D BA90 19CF 5A1F EBBD 33C0 8EC0 26C4 ; Offset 13D
```

**USSR-1594** - EN: A 1594 byte virus which uses a self-modifying algorithm indicated by the asterisks in the search pattern.

```
USSR-1594      1E07 BB15 002E 8037 **43 81FB 3A02 7CF5 ; Offset 005
```

**Wolfman** - CER: A 2064 byte virus from Taiwan with unknown effects.

```
Wolfman        8EC0 BE04 0026 837C FC00 7404 46EB F6EA ; Offset 07F
```

### AMENDED SEARCH PATTERN

**Kamikaze** - EN: Overwriting virus written in *Turbo Pascal*. Previous pattern caused false alarms. See *Technical Notes*, page 4.

```
Kamikaze       2C20 AAE2 F2B0 3DAA 1F1E 8E1E 3E00 8E1E
```

# COUNTERMEASURES

***John Sherwood***

## A Backup Strategy Based on Risk Analysis

In the December issue of *Virus Bulletin*, Dr. Keith Jackson explored some aspects of file backups in PC systems. In this article, the different types of file (i.e. system files, program files and data files) are examined.

An appropriate backup strategy must, by necessity, account for the types of risk to which electronic data is prone and exposed.

### Assets at Risk

In risk analysis the assets which are at risk should first be identified. In this instance the assets are logical in their nature, and they comprise the **system files** (i.e. the operating system), the **program files** (i.e. all executable files, particularly those with .COM or .EXE extensions and including **operating system utilities** such as DISKCOPY.COM, and all **application programs** such as word-processors, databases, spreadsheet managers, etc.), and the **data files** (i.e. document files for word-processing, spreadsheet data files, database files, etc.).

### Threats

The next step is to identify the threats which affect the identified assets. The main threats to the logical assets are:

- Viruses and other malicious software
- Physical disaster such as fire, flood, etc
- Operator error
- Mechanical failure of the disk

### Vulnerability

Next, the vulnerability of the assets to the threats is assessed.

**System files** (those which comprise the operating system) are vulnerable to infection by viruses, (which typically infect COMMAND.COM) and to damage caused by the payload carried in any type of virus.

The **operating system** is also vulnerable to physical disasters and to mechanical failure of the disk.

**Program files** (most notably those with .COM or .EXE file extensions) are vulnerable to infection by parasitic viruses and to damage caused by the payload carried in any type of virus. Programs are also vulnerable to operator error (typically inadvertent commands - 'DEL *.*' for instance), physical disaster and mechanical failure.

**Data files** are vulnerable to the damage caused by the payload of any virus. They are also vulnerable to operator error, physical disaster and mechanical failure.

### Developing a Model

Having analysed the risks, a good model on which to design countermeasures becomes available. In respect of viruses, there are a number of possible countermeasures, one of which is to take backup copies of files. This does not alter the threat, but it reduces vulnerability to the damage caused either by a virus infection or by the delivery of a virus payload. It also reduces vulnerability to physical disaster, mechanical failure and operator error. Hence we can see that backup copying reduces vulnerability to a wide range of threats and has wide-ranging benefits; it is also relatively cost-effective.

*How, then, should the backup process be implemented?*

To answer that question we need to look back at the specific analysis of the risks.

If a system file or a program file becomes infected by a virus, (i.e. it has its code modified by the virus), then copies of that infected version may replicate throughout the system, and infect recent copies of those files taken for backup purposes. Therefore the backup strategy for these types of file must take account of this added vulnerability, and must avoid the risk of restoring a backup file which is just as badly damaged as the one it is replacing!

### Software

Backup copies of the operating system and of all programs must be made from write-protected, original master diskettes. The master software should be write-protected as soon as it is removed from the box in which it is supplied.

**Every backup copy of a program should be a first generation copy, made directly from the write-protected master diskette supplied by the manufacturer. There should never be any second generation copies (or beyond)**

Additionally, the original masters should only ever be used for the production of first generation copies, those copies must be write-protected immediately, and all installation of software onto the PC must be done using those first generation copies.

**There should be at least two sets of backup masters for each program, and these must be stored in different physical locations, preferably on completely different sites** This strategy substantially reduces the vulnerability of your programs and operating system to irretrievable loss or damage.

### Data

Data files are updated regularly - usually every time that they are accessed. Hence it is the most recently available copy that you must restore. Backup copies of data must be taken frequently to minimise the potential loss if a file gets damaged. If the corruption is progressive (as with that caused by some virus payloads), multiple generations (as many as possible) further reduce the vulnerability, since you may be able to go back to a version prior to the onset of corruption.

The various techniques for backing up data are described in *VB*, Dec 90, pp. 12-14. A tape streamer system with automatic verification of stored files is recommended for large volumes of data**. However, you never, under any circumstances, backup programs to any backup media in use.**

### Selective Backup

To make this selective backup more manageable, I recommend the separation of data and programs in the directory structure. If you create a first-level sub-directory called C:\DATA, and store all data files with pathnames beginning with this, then all your data is in a self-contained sub-structure of the directory, which makes it easy to select when you want to make the backup copies. **If data and programs are combined in directories, selecting data files for the regular backup becomes a tedious and time-consuming operation** Programs can be arranged in their own sub-directories, which makes life easier if you ever need to restore from masters.

If you make use of an access control / user management package there is an added benefit from this approach, since you can also set up access rules for program directories which allow READ, OPEN and SEARCH on program files, but not WRITE, CREATE or DELETE. This should prevent current viruses from spreading as the management package will signal a system violation message if a virus attempts such unauthorised actions.

An example of the recommended directory structure to facilitate backup is shown in *Figure 1*.

### Configuration Data

There are other subtleties to consider, such as the configuration data for your programs - all the “settings” of the various parameters. This information is itself “data” and is used by programs to store your preferences for screen display colours, default settings for format, printer drivers for the printer, etc.

If programs are reinstalled from a backup master, you need to rebuild the same set of parameters that were in use before experiencing the problem. Since this restoration exercise is a rare occurrence, you can afford the time to re-configure your program manually when you install it, but to do this you need to have recorded the full specification of how the settings were configured. **This underlines the need to keep a written log of all system configuration details, including the actual directory structure and contents of each sub-directory, in order that the system can be restored to precisely its original configuration.**

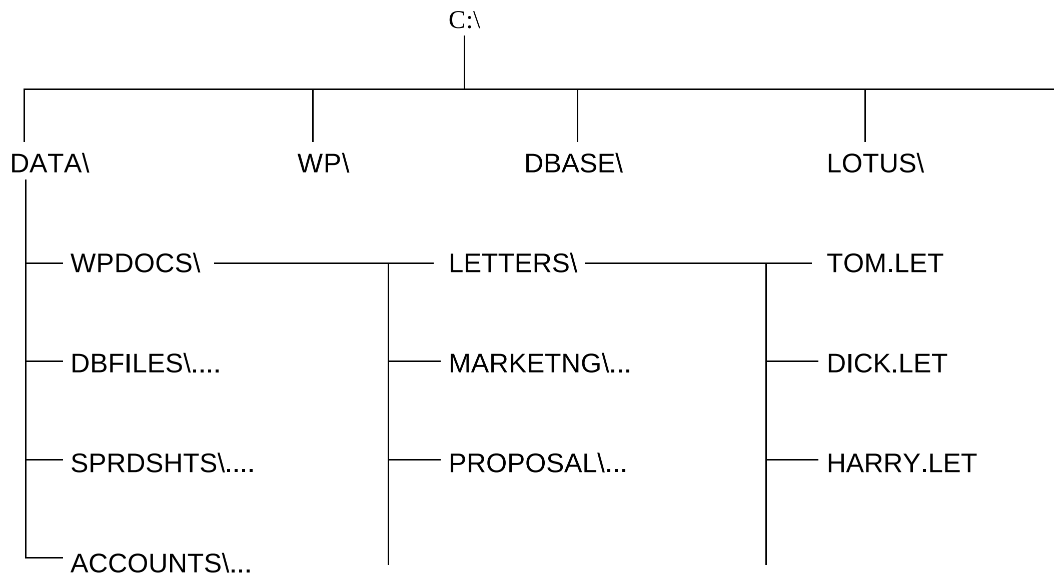

*Figure 1*. Selective backup directory structure

# VIRUS ANALYSIS 1

## The Beijing Virus - A Transatlantic Protest Against the Chinese Government

Recently (December 1990) specimens of a new Master Boot Sector virus were received from a UK University alongside two slightly different specimens of the same virus from the *Massachusetts Institute of Technology* in the United States.

The virus specimens appeared on both sides of the Atlantic within days of each other, which initially gave rise to the suspicion that this virus had been transmitted via modem. However, there are technical obstacles in transmitting boot sector viruses; subsequent analysis of the code revealed crude programming suggesting a restricted technical ability on the part of the virus writer. If this boot sector virus was transmitted via modem, it would almost certainly have required an accomplice to render the code active upon its receipt.

Direct comparison of specimens of the virus code from each site have shown that they are substantially the same. There are minor differences; one is accounted for by the storage of the INT 13H vector (which will be specific to the infected machine) and the others occur in the encrypted message information.

### Screen Messages

The UK version message decrypts as “Bloody! Jun. 4, 1989”, one of the *MIT* samples decrypts similarly but with a year of 1909 and the other sample has what appears to be a corruption (possibly introduced during modem transmission) where the “l” becomes graphic character number 250 (decimal point). Investigations are under way concerning how boot code could appear at two such widely separated sites within such a short space of time.

Because of the date (referring to the Tiananmen Square Massacre of two years ago in which hundreds of Chinese students were killed by forces loyal to the Chinese government), American sources have named this virus “Beijing”. UK researchers have favoured “Bloody” or “June 4” but there is no doubt that the code is the same. There is no explanation as to why the message contained in the virus is so vague - if this is an attempt at social or political protest, one might assume that the message would be rather more explicit.

### Virus Analysis

This is a single sector boot sector virus which infects the Master Boot Sector of fixed disks and the boot sector of floppy disks. As is usual with viruses of this type, a copy of the original boot sector is stored elsewhere on the disk.

### Installation

The code begins with an indirect far jump to an address stored in a double word location within the virus code.

The address is 0000:7C05 which is the normal place for boot code to be located. This jump precludes the code from being loaded and run at any other location. Installed at the 7C05 location, is a near jump into the code proper.

Processing then begins by zeroing the accumulator and the DS register, initialising the Stack Segment to zero and setting the Stack Pointer to 7C00. Then the existing INT 13H vector is collected from page zero of memory (offset 4CH and 4EH) and stored within the virus code for later use.

Next, the top of memory pointer maintained at 0000:0413H is collected and decremented twice before being replaced. The decremented value is then converted to a segment address and placed at two locations - the segment vector of INT 13H in low memory and at a location within the virus code which forms part of the relocation jump after the code has been relocated in high memory. The original INT 13H vector is then reset to point to the virus handler position in high memory. Since this is done before the code is copied to high memory, there may be unpredictable effects if problems occur after relocation. After the copying is copleted, processing transfers into high memory and immediately issues a Disk System Reset request to INT 13H.

The boot sector is then re-read from the default drive into the boot area at 0000:7C00H and a flag is checked to see whether the code came from a hard or a floppy disk.

### Virus Location and Boot Sector Relocation

If the flag indicates a floppy disk, processing reads Sector 3, Head 1, Track 0 of the floppy into the boot area and then an attempt is made to read the Master Boot Sector (Sector 1, Head 0, Track 0) from the first hard drive. If this is read successfully, the first six bytes are checked against the virus code to see whether the hard disk is infected. If the disk **is** infected, processing continues by passing control to the newly read code. This will then re-install the virus code and may give rise to unpredictable effects. If the hard disk is **not** infected, the original Master Boot Sector is written to Sector 6, Head 0, Track 0, and the contents of the partition table are copied from the original (uninfected) boot sector into the virus code, and the whole code (with the flag set to indicate a hard disk source) is then written to the boot sector of the disk.

If the flag indicates a hard disk source, processing reads Sector 6, Head 0, Track 0 of the first hard drive into the boot area. A counter byte contained within the virus code is then incremented and tested to see whether it has reached 80H (128). If the counter has reached 80H processing continues by resetting the counter to 7AH (122) and then decrypting and displaying the message which reads “Bloody! Jun. 4, 1989” (the date of the infamous Tiananmen Square protest). Once the message has been displayed, a copy of the virus code (complete with modified counter) is written back to the Master Boot Sector of the disk. Finally, the partition table information in the virus code is overwritten with uninitialised garbage. The reason for this is not clear. The normal boot process then continues by returning control to the boot area.

### INT 13H Handler

The INT 13H handler routine which is installed at boot time only intercepts Read and Write requests to floppy disk drives. First all registers are saved on the stack and then the original call request is completed using the “clean” INT 13H vector. Then the boot sector of the target floppy is read into a buffer within the virus code area. Once read, the initial six bytes of this buffer are compared to the start of the virus code to see whether the disk is infected. If the comparison succeeds, processing returns immediately to the calling program. If it fails, the buffer is re-written to Sector 3, Head 1, Track 0 of the floppy and the virus code is written to the boot sector of the disk (Sector 1, Head 0, Track 0) before continuing normally.

### Detection

Detection is a straightforward process and many commercial and shareware scanners have been updated to combat this virus, which employs no sophisticated encryption routines or stealth features. The following search pattern will be found in the Master Boot Sector (Sector 1, Head 0, Track 0):

```
80FC 0272 0D80 FC04 7308 80FA 8073 03E8 ;
Offset 01F
```

### Conclusions

It is obvious that this virus was intended by its author to be a political statement but no attempt has been made to ensure that damage does not occur to infected floppy disks. The choice of alternative storage sectors used (on floppy and hard drives) virtually guarantees that infected disks will become corrupted and in some cases unreadable. No distinction is drawn between floppies of differing densities and the effects of infection will vary with different disks. The virus also makes no attempt to re-route legitimate requests for the Master Boot Sector, so programs which collect such information as a matter of course will produce erroneous and completely unpredictable effects.

The virus, like so many before it, has appeared at large in a university environment. This tends to reinforce the suspicion that academic establishments, where computer resources and the people using them are often unregulated, are breeding grounds for the development of computer viruses and other forms of computer misuse.

# VIRUS ANALYSIS 2

***Richard Jacobs***

## Aircop - From the Taiwanese Virus Factory

The Aircop virus, which last month appeared as an entry in the ‘reported only’ category, has now been seen in the United Kingdom. It is believed that Aircop originated in Taiwan which has become the computer virus ‘capital’ of the Far East.

Its screen message (see following page) is possibly a veiled political statement decrying one of the Communist states in South East Asia (see also the analysis of the Beijing virus, opposite).

This particular virus, which can best be described as *singularly* inane, is proving to be of nuisance value only. However, it is possible that data may be destroyed on infected diskettes.

### Description

Aircop is a short memory-resident boot sector virus that consists of just one sector and only affects floppy disks.

It follows conventional boot sector virus strategy; it makes a copy of the boot sector and writes the virus code over the original boot sector. The virus executes when a PC attempts to boot from an infected disk. **This reinforces the need to remove diskettes from the floppy disk drive as soon as data transfer is completed. Non-system diskettes should never remain in the floppy disk drive when the PC is switched off**. Once the virus has executed, it becomes memory-resident, loads the copy of the original boot sector and jumps to it, before returning to the normal boot process.

Aircop is yet another example of a virus which does no intentional damage, but due to carelessness by the programmer it will cause corruption on some diskettes. The virus creates a copy of the boot sector in Sector 9, Side 1, Track 39 of the disk. On 360 Kbyte disks this is the last sector on the disk and conflict with data residing at this location is highly unlikely.

**On diskettes of other densities, the sector to which the virus will write is in the middle of the data area and may well contain data, which will be lost** The virus does not check the FAT for a clear cluster, or mark the FAT once it has copied the boot sector, so the copy of the boot sector may be overwritten by subsequent writing to the disk, rendering the disk unbootable.

The virus only infects diskettes in either of the first two floppy drives; fixed disks are **not** infected, presumably because the virus writer realised that the simple method of infection used would cause immediate data corruption leading to detection of the virus on fixed disks.

### Operation

When a PC is booted from an infected disk, the virus immediately gains control. It reserves 1 Kbyte of memory at the top of base memory by reducing the available memory by 1 Kbyte. The virus then captures the INT 19H vector (Reboot computer) before copying itself into the reserved 1 Kbyte block and transferring control to this copy. Once there, the original boot sector is loaded and INT 12H (Get base memory size) is captured and redirected to a routine that redirects INT 13H (BIOS disk services), resets INT 12H and then performs a normal INT 12H call. Finally the virus jumps to the original boot sector and the PC continues to boot normally.

If an error is reported when the virus tries to load the original boot sector, then the message "Non-system" is displayed and the virus waits for a key to be pressed. When a key has been pressed the virus calls INT 12H, which redirects INT 13H as described above. It then reads Sector 6 on Side 0, Track 0 of Drive 0. This will cause the disk to be checked for infection and subsequently infected should it not already be so. If an error is detected the virus returns to the "Non-system" message, this is repeated until no error is reported, when the PC will be rebooted by jumping to the captured INT 19H vector.

INT 13H is the only interrupt that is intercepted once the boot process has been completed. This routine first checks the drive requested and if the drive number is larger than 1, the normal INT 13H is carried out. Otherwise the logical sector number required is calculated, based on the assumption that the disk is 360 Kbyte. If the logical sector number is between 6 and 12, the disk is checked for infection, otherwise control is returned to the normal INT 13H vector.

The virus checks for infection by reading the disk boot sector and comparing 174 bytes with those of the virus in memory. If they match the disk is assumed to be infected and control is returned to the normal INT 13H routine. If the disk is not already infected, its BIOS Parameter Block (BPB) is copied on to the memory-resident copy of the virus, which is then written to the boot sector location andthe original boot sector is then written out to Sector 9, Side 1, Track 39 of the disk.

### Screen Message

Once a disk has been infected a counter is incremented in memory and logically ANDed with 7. If the result of this is zero the following message is displayed:

```
".Red State, Germ offensing –Aircop"
```

The message displays after every eighth disk is infected.

### Detection

The virus uses no sophisticated hiding mechanisms and its code resides only on floppy disks. The following hexadecimal pattern will identify Aircop:

```
32E4 CD16 CD12 33C0 CD13 0E07 BB00 02B9
```

### Disinfection

The PC should be booted from a clean, write-protected, system diskettes. Then all files can be copied safely from infected diskettes using the DOS COPY command. **DISKCOPY should not be used as this will transfer the virus code in the boot sector in addition to other images on the infected disk** The infected diskettes should be reformatted. As previously mentioned, only floppy disks can be infected.

## Boot Sector Viruses

When the computer is switched on, or a warm boot is performed (Ctrl-Alt-Del), a PC first executes the program held in ROM (Read Only Memory). The ROM program tests the drives for the first one containing a disk, loads into memory the contents of the first sector on the disk (known as the boot sector) which is a short program, and starts executing it. If the disk does not contain operating system files, the computer displays the message 'Non-system disk', or similar, and waits for the user to insert a system disk (i.e. a disk formatted with a boot sector and system files).

On hard disks, the Master Boot Sector (Track 0, Head 0, Sector 1) loads and executes the DOS Boot Sector (logical sector 0 in the active DOS partition selected by FDISK) which reads in DOS and transfers control to it.

Boot sector viruses modify either the **DOS Boot Sector** or the **Master Boot Sector**, depending on the virus and type of disk, usually replacing the legitimate contents with their own contents.

The original but modified boot sector is normally stored elsewhere on the disk, so that when the machine is switched on, the virus code is executed *first*. This normally loads the remainder of the virus code into memory, followed by the execution of the original version of the legitimate boot sector. From then on, the virus generally remains memory-resident until the computer is switched off. **A boot sector virus is thus able to monitor and intefere with the operating system from the moment it is loaded into memory**.

## Preventing Boot Sector Viruses

**Points to remember:**

- Boot sector viruses infect Sector 0 on floppy disks. At least one virus can now infect **all** densities of floppy disk. Some boot sector viruses also infect either the Master Boot Sector or the DOS Boot Sector on fixed disks. There are some hybrid ‘multi-partite’ viruses which can infect both programs and boot sectors.
- Any DOS formatted disk can spread a boot sector virus. **This is true regardless of whether the disk is used to transfer non-executable data or programs**
- Disks should **never** be left in floppy drives longer than is absolutely necessary. Instruct staff to remove disks from the drive immediately after data or program transfer is completed. Ensure that no disks are left in drives overnight or when machines are not in use.
- In the event of power loss, instruct staff to switch machines off and remove disks from drives. Otherwise when power is restored, the PC will attempt to reboot from whatever disk happens to be in the drive at the time.
- Make sure that a **clean write-protected system floppy disk** is readily available.

# VIRUS ANALYSIS 3

***Jim Bates***

## Faust

This virus was reported by a user as at large in the UK during January 1991. It apparently arrived attached to software imported from Hong Kong although there is a very slight possibility that infection occurred after importation.

### Description

Faust is a resident, parasitic virus which appends to executable files but does not infect COMMAND.COM. The infection process may possibly affect other file types if they are subject to the DOS LOAD & EXECUTE function request. There are **two** trigger routines, both of which activate on the 13th of any month, as well as a signature change during and after 25th December (any year).

**The primary trigger routine writes random garbage to a random position on the disk and detection of this virus must therefore be classified as a high priority requiring immediate and total disinfection in the event of its discovery**.

### Operation

There are two distinct entry points to this virus, depending upon whether the host file is a COM or EXE type. Both entry points begin by issuing an “are you there?” call to DOS by placing a value of 0E7H into the AH register and requesting an INT 21H. If the virus is resident, the interrupt request returns a value of 7BH in the AH register.

With COM files the original three bytes at the beginning of the program are repaired and program execution is returned to the start of the host program.

With EXE files an immediate jump is implemented to the CS:IP setting contained within the original program header. If the virus is not resident, the processing at both entry points relocates the virus code to offset zero of the code segment and jumps to the virus code. Processing then continues in a standard fashion for both types of infection.

A new stack is set up and a call is made to function 4AH of DOS INT 21H to allocate around 1700 bytes of memory. No check is made to see whether the memory was allocated successfully. A call is then made to obtain the system date and this is tested to see whether a) *it is earlier than 25th December*, or b) *it is the 13th of the month*.

If the date is 25th December or later, a signature used by the virus to recognise its own presence within a file is modified. This modification occurs each time the virus is installed and will result in multiple infections of target files. Once January is reached and files have been infected with the latest signature version, they will not be re-infected until the process is repeated on the next 25th December. It is not known why this particular process has been incorporated.

If the date is the 13th of the month, the virus installs three separate interrupt handlers - for INT 09H (Keyboard services), INT 13H (Disk I/O services) and INT 21H (DOS Functions). A temporary INT 24H (Critical Error) handler is also used within the INT 21H handler. If the date is *not* the 13th of the month, the INT 09H and INT 13H are *not* installed. Since these are concerned with the trigger routines this means that damage or interruption will not occur but file infection (via the INT 21H interception routines) will occur.

A description of these interrupt handlers follows.

### INT 09H (installed 13th only)

This is a simple interception routine which increments a counter within the virus code at every keystroke, and then tests its value. When the counter reaches 100 the video mode is set to 80 * 25 text (mode 2) and a short message is collected, decrypted and displayed before processing enters an infinite loop and the machine "hangs". The message is:

```
Chaos!!! Another Masterpiece of Faust...
```

### INT 13H (installed 13th only)

This handler invokes the primary trigger routine at every fifth disk access (*any* call to INT 13H) request. The counting process starts by incrementing a counter and testing for a value of 5. If the test fails processing continues unmolested, otherwise the trigger routine is executed. Counting does **not** start at zero but will vary according to the current month value recorded (and encoded) from the initial system date request. Thus for the months of January through to July (inclusive) and December, the initial count will start above 5 and will allow between 247 and 256 disk accesses before triggering. During August to November (inclusive) only 2 to 5 accesses are counted before triggering.

The trigger routine itself holds the original INT 13H request and issues a Write instruction having first generated a random track/sector address. The instruction is to write 9 sectors taken from the caller's buffer area and the write process is always to head zero. **No change is made to the drive specifier provided by the calling routine and this means that all local disks (fixed and floppy) are at risk.**

### INT 21H (installed every time)

This handler provides the infection routines and also the response to the "are you there?" call issued during initial execution. Apart from this function, the only other function intercepted is 4B00H (LOAD & EXECUTE). When this request is received, the virus first verifies that the amount of free space on the disk will allow the addition of virus code.

The extension portion of the target filename is checked in an unusual way: counting back from the end of the filename, if the second letter is the same as the tenth letter (as in COMMAND.COM where the Os match) then infection is aborted. Then the file attributes are collected, stored and reset to allow write permission. Next, the first and last letters of the three letter extension are checked against each other. If they are the same the virus sets a flag to indicate an EXE type file. This method obviously causes problems if a SYS file is processed with this function.

Target files are checked for previous infection by examining the word at offset 41 decimal from the end of the file. It is this word value which is incremented by four at every installation during and after 25th December. Thus the infection check will fail and files will gain multiple infections. In my sample, the value of this word was 1234H which may indicate that this version had **not** "mutated" in the way described.

The infection method is the (by now) fairly standard process of appending virus code to the file and modifying the initial program bytes (or header for EXE type files) to route processing through the virus code. The only major difference with this virus is that when an EXE type file is first loaded, the virus is installed and before becoming resident (using DOS Function 31H), the original file is loaded and executed using the DOS 4B00H function. For EXE files which require large amounts of memory, this will result in Out of Memory errors upon first execution.

### Conclusions

The general coding of this virus is extremely primitive and seems to have been written by a newcomer to assembler programming. Despite the Hong Kong connection already reported, the use of the word "Chaos" in the message may indicate a connection with that odious group known to propagate virus code from Germany and other places in Europe. Alternatively, it could simply be plagiarism.

Interrupt handlers are installed using DOS functions 35H and 25H and the whole code is made TSR with function 31H. No encryption (apart from the message) is used and the code is easy to detect and defend against. **However, the nature of the primary trigger routine is such as to make vigilance necessary since, like the Nomenklatura virus, the very presence of the virus code may indicate corrupted data which cannot be quantified or repaired.**

### Virus Information

Faust* is a resident virus which infects files via intercepted LOAD & EXECUTE function calls. The infective length is 1184 bytes. **A reliable search pattern is as follows**

```
B87A 0050 06B8 FD00 5026 C706 FD00 F3A4 ;
Offset 44H
```

I do not normally recommend disinfection of parasitic viruses, but if valuable code becomes infected, and no backups are available, COM files can be repaired by replacing the first three bytes of the file with the second three bytes within the virus code (offsets 3, 4 and 5). EXE type files are reparable since the virus does not overwrite any program code. **However, the disinfection process is somewhat involved and not recommended without accurate reference to a full disassembly of the virus code.**

**Editor's note. The life of George Faust (c. 1480-1540), a German necromancer and 'unscrupulous charlatan' has formed the basis for numerous artistic and literary works. Legend has it that Faust traded his soul with the Devil (Mephistopheles) in exchange for earthly pleasure and power. Marlowe's 'The Tragedy of Doctor Faustus', the first literary dramatisation of the legend, appeared in 1604.*

# VIRUS ANALYSIS 4

***Fridrik Skulason***

## The 'Illegitimate' LoveChild Virus

The report from Bulgaria in the December 1990 edition of *VB* mentioned the Russian LoveChild virus, saying "It is believed that the Russians may be using viruses for software copy-protection and their reportedly 'clever' LoveChild virus may have been produced for this purpose."

A careful analysis was performed to determine whether this was the case. The results show LoveChild to be a virus, written by a technically proficient but sloppy author, which has nothing to do with copy-protection. There are several 'bugs' in the code which indicate a lack of testing and the appearance of a 'production' version in the future should be anticipated.

**The most significant feature of the virus, is that it Trojanises certain program files and that the post-trigger effects are extremely pernicious.**

No infections by LoveChild have yet been reported outside Eastern Europe, but it is quite likely that sooner or later it will appear elsewhere. The mistakes in the virus reduce somewhat the risk it poses. However, when it strikes, the virus can have devastating effects - the complete destruction of the contents of the hard disk.

### Simple Structure

LoveChild only infects COM files, overwriting the first 4 bytes of the target file with two instructions, STI and JMP to the body of the virus, which is appended to the end of the file.

The length of the virus body is 488 bytes, not 467 as was incorrectly reported in the January table. Inside the virus one finds the following two text strings:

```
v2 (c) Flu Systems (R)

LoveChild in reward for software sealing.
```

The strings are never displayed, and are not used in any way, other than the "v2" - which probably indicates version 2 of the virus - being used to verify whether the virus is present in memory. It is generally assumed that "sealing" is a spelling error and should probably be "stealing".

### Installation

When an infected file is executed the virus first determines whether it is already present in memory. This is easily done, because the virus can always be found at the same address. The current version occupies the memory area from 0000:01E0 to 0000:03C8 - the upper half of the interrupt table. Interrupt vectors 78H - FBH will be overwritten, probably causing any program using them to crash the system. Examples of such programs include *Novell Netware* ˜ and *AutoCAD* ˜.

If the virus finds the characters "v2" at address 0000:01E0, it assumes that it is already active, and restores the first 4 bytes of the host program and transfers control back to it. If the self-identification signature is not found, the virus will transfer itself to this area and proceed with the installation.

### DOS 3.30 Installation

It can be assumed that the author of LoveChild was running MS-DOS version 3.30 because a special check is made for that version and a more sophisticated method used to hook into INT 21H than otherwise.

If the DOS version in use *is* 3.30, LoveChild will attempt to disable any program monitoring the INT 13H vector, by resetting the interrupt vector to the value it had directly following bootstrapping. All memory-resident programs, which might have hooked into the interrupt chain of INT 13H after that will be disconnected. **In particular, most programs which attempt to protect the hard disk from unauthorised 'write' or 'format' commands will be rendered ineffective.**

The address to which the INT 13H vector is directed is obtained at a fixed location in memory - a location which is only valid for MS-DOS 3.30.

Instead of changing the interrupt table to make the INT 21H vector point to the virus code, the virus overwrites the first 5 bytes of the original INT 21H entry point with a JMP FAR to itself. LoveChild assumes this entry point is at a fixed offset within the DOS memory segment. As before, this only holds true for MS-DOS 3.30. Finally, the virus restores the original first 4 bytes of the host program and transfers control to it.

### Non-3.30 Installation

If some other version of DOS other than 3.30 is in use, a much simpler method is used. INT 13H is not changed, and INT 21H is just set to point to the virus code.

Fortunately, there is a serious flaw in this part of the code - serious enough to prevent the virus from working under any version of DOS other than 3.30. In all other cases the computer will crash. Amazingly, this error could have been corrected by adding just *one* instruction!

### The INT 21H Handler

When any INT 21H function is called, a 16-bit counter is decremented, and as long as it is above 0 nothing will happen - the old INT 21H function will just be called normally. As this counter starts with a value of 5000, some time may pass before anything of interest happens.

If the command is ‘write’ (AH = 40H), the first two characters of the buffer are checked. If they are ‘MZ’, the virus may ignore the original data, and instead write a 64-byte Trojan to the file. Whether this happens or not is determined by a random value obtained from the system clock and in 3 out of every 4 cases the virus does nothing.

The check for ‘MZ’ at the beginning of the file is an attempt to identify EXE files (which commonly start with these two bytes), although ‘ZM’ is also permitted.

**Any EXE file which is copied or created after LoveChild activates has a 25 percent chance of containing the Trojan** (see below).

If the command is ‘create file’ (AH = 3CH), it may be changed into a ‘create directory’ command. This has a 1-in-8 chance of happening, but probably the likelihood was intended to be 1-in-32.

If the command is ‘execute’ (AH=4BH), ‘open file’ (AH=3DH) or ‘rename’ (AH=56H), LoveChild may - with a 1-in-8 chance - attempt to infect the file in question.

> *“The fact that this virus is so highly destructive actually reduces its chances of spreading.”*

### Infection

**Any file being** *opened*, *renamed* **or** *executed* **is a potential target for infection**.

To identify a COM file, the virus examines the last two characters of the file name for ‘OM’. The infection process is performed in a standard way - the virus first intercepts INT 24H, the fatal error or ‘Abort, Retry, Ignore ?’ interrupt, probably to prevent error messages appearing on the screen when a floppy disk in the drive is write-protected.

The virus then reads the first 4 bytes of the file. The method used to determine whether the file is already infected is simple - the virus just examines the first byte of the file - assuming that a value of 0FBH indicates an infection. All COM programs starting with an STI instruction are therefore immune to infection by the virus.

If the file is not already infected, the first 4 bytes are stored in the virus body and the virus appends itself to the file. Finally it writes an STI instruction and a JMP to the virus code to the beginning of the host program.

### Damage

If the file being executed, opened or renamed is not a COM file, it has a 1-in-8 chance of being deleted. **In particular this means that EXE files will slowly disappear from any infected system**. This damage is nevertheless minor compared to the potential damage which can be caused by the Trojan.

### The Trojan

**When an EXE file containing the Trojan is executed, sectors 1-16, heads 0-3 on every track of the first hard disk will be overwritten with garbage, starting with the Master Boot Sector, the FAT and the root directory.**

### Detection and Disinfection

As LoveChild uses no “stealth” methods, it is easily detected by searching for a pattern near the end of COM files. Any COM file not starting with an STI instruction can be eliminated quickly from further consideration. It is easy to write a disinfection program - disinfection is simply a matter of locating the contents of the original first 4 bytes, replacing them and removing the virus body from the end of the file.

Detection is straightforward. A reliable search pattern to detect the virus is:

```
33C0 8EC0 E800 005E 8BEE BFE0 01FC 2681
```

**However, removing the Trojan from infected EXE files is not possible and any such files must be replaced** A separate search string can be used to find the Trojan - usually located at the beginning of EXE files.

```
LoveChild Trojan
B901 00BA 8003 8BD9 B810 03CD 13FE CE79 ; Offset 0
```

### Conclusions

The fact that the virus is so highly destructive actually *reduces* its chances of spreading. The disappearance of EXE files will inevitably warrant investigation and early detection. However, the fact that the Trojan component of the virus inflicts such massive damage makes this virus a potent threat.

The appearance of this virus lends support to East Bloc reports (as yet unconfirmed) of viruses which Trojanise specific program files and trigger on attempts to remove the virus code from other files. It is also indicative of the ‘sabotage mentality’ which is understood to prevail in Bulgaria and the Soviet Union.

# PRODUCT REVIEW

**_Dr. Keith Jackson_**

## Turbo Anti-Virus

*Turbo Anti-Virus* has been developed by an Israeli company, is supported by a Research & Development headquarters in the United States and is marketed by distributors in some twenty five countries worldwide. This amply illustrates the international nature of the virus problem. Computer viruses are not contained by national boundaries and the same is true of anti-virus tools.

### The Manual

The manual provided with *Turbo Anti-Virus* claims that it "constitutes a revolutionary technique to cure computer systems of their viral ills ...". This is pure hyperbole. There is no such thing as a 'cure' for the computer virus threat, just a series of defences which are effective to a greater or lesser degree.

A decent index is provided in the manual, and an extensive question and answer section. The latter contains curious advice to someone posing the question *"I have no backups. Can Turbo Anti-Virus help?"* The best advice in such circumstances would be a stern admonishment explaining to the questioner that he is terminally stupid and should take a complete backup (or two) immediately. Not so - advice about backups can only be found tucked away in the glossary.

Although *Turbo Anti-Virus* has an installation program, this is not strictly necessary as installation copies all necessary files to the required destination subdirectory.

### The Software

Most of the features offered by *Turbo Anti-Virus* are contained within a single executable file. The opening screen of this program shows that the version of *Turbo Anti-Virus* provided for evaluation combats 312 viruses or variants.

The program provides facilities to search for viruses, disinfect programs and immunise files against specific viruses. These operations can be performed on a complete disk, in the current directory, on a user-specified set of directories, on a user-specified set of files, or on the boot sector. This profusion of choice illustrates one of *Turbo Anti-Virus's* strongest features; all conceivable options can be simply activated from drop-down menus.

Rather curiously there seems to be no means of saving the current settings to disk (or if there is I cannot find it), and after execution commences, the settings always revert to the original *Turbo Anti-Virus* default values.

Coupled with the main *Turbo Anti-Virus* program are three small utility programs. Two of these are memory-resident monitoring programs which detect and prevent virus activity. Two programs are provided as there is a trade-off between the amount of memory occupied and the facilities provided. The user can choose which is best suited to his way of working. The third utility program maintains the Master Boot Sector and DOS Boot Sector of a disk in a virus-free state.

### Scanning Speed

Programs that detect viruses by scanning for known patterns are judged by two criteria: how fast they scan and how well they detect viruses. I tested the scanning speed of *Turbo Anti-Virus* by searching the whole of the hard disk on my ancient PC compatible for viruses. This is an old slow workhorse, but it can nevertheless produce valid comparison times. With Turbo Mode switched on (which is the default setting), *Turbo Anti-Virus* took 4 minutes 7 seconds to report that it had searched the complete hard disk for viruses. For comparison purposes, version 4.5B66 of *SCAN* from *McAfee Associates* took 13 minutes 46 seconds to search the same disk, while version 2.19 of *SWEEP* from *Sophos* took 9 minutes and 57 seconds to search the whole of each file on the disk (equivalent to non-Turbo Mode, see below).

If the Sound Effects option is left on, then *Turbo Anti-Virus* makes a small chirping noise when it opens and/or closes an on-screen window for each subdirectory on the disk being searched. As a testament to how fast the search speed of *Turbo Anti-Virus* really is, activating the Sound Effects option makes the computer sounds like a demented budgerigar while *Turbo Anti-Virus's* search progresses.

When a virus is found, a menu appears which offers either to 'Clean' the offending file (remove the virus), to 'Clean & Immunise' (remove the virus and immunise the file against it), continue detection, or to stop.

The 'Clean & Immunise' option is particularly helpful as it secures files against reinfection during 'clean up operations'. (Note, that it is only possible to immunise a file against a single *specific* computer virus or closely related subset at any one time, Ed.). All the files contaminated by the specific viruses which the *Turbo Anti-Virus* documentation claims to to be capable of removing were successfully disinfected.

For three virus samples (December 24th, Kennedy and Virus-90), *Turbo Anti-Virus* correctly identified the virus, but displayed a message saying that the developers did not have a sample of the code along with a request that a copy of the infected file be sent to them on floppy disk. As an incentive to comply with this request, the user of *Turbo Anti-Virus* is offered a free upgrade version capable of correctly detecting this virus. (*Carmel Software* claims to offer a 14-day upgrade service from receipt of a new virus to providing detection and disinfection routines for it. Ed.)

When a sample of an unknown virus was requested, *Turbo Anti-Virus* could not disinfect the virus infected file. Given that it has no knowledge of this virus, and a wrong guess could be harmful, this is unsurprising. (The ability to detect these viruses but inability to remove them can be explained by the developer's incorporation of reliable search patterns or algorithms *prior* to analysing the structure and operation of the actual virus samples. *VB* effectively supports this approach by providing search patterns in advance of full disassembly and analysis. Tech Ed.)

The above quoted search times pale into insignificance when *Turbo Anti-Virus* is tested with Turbo Mode switched off. This is a text search mode whereby every file is scanned from its first byte to its final byte. The manual just comments that in this mode *Turbo Anti-Virus* searches the complete file for viruses, saying "...although this is a much longer process, it is nevertheless recommended in special cases". This hides the fact that with Turbo Mode switched off, *Turbo Anti-Virus* took 1 hours 49 minutes 48 seconds to search my hard disk for viruses. This is not a typing error, it really did take nigh on two hours to carry out this task: your intrepid reporter kept his stopwatch going the whole time. *Carmel* say that this feature is principally included to provide positive identification in the event of a suspected false alarm. However, even accounting for the need for a comprehensive and complete file scan, the program in this mode is astonishingly slow.

However, the *Turbo Anti-Virus* search speed when operated with Turbo Mode switched **on** (which is the default setting and recommended procedure) is undeniably impressive.

## Detection Rate

I tested the accuracy with which *Turbo Anti-Virus* could detect viruses by using the standard *VB* set of viruses (see Technical Details below). This test set is to be extended to include many more recent viruses, but this task was not completed in time for this month's evaluation. Even using the original test set, *Turbo Anti-Virus* failed to detect six of the 101 virus samples.

*Turbo Anti-Virus* failed to report an infection for files that were infected with the Anarkia, Devil's Dance, and Virdem viruses. The same result was found for one variant of each of the Datacrime, Vienna and Yankee viruses. *Turbo Anti-Virus* also wrongly identifying samples of the Prudents, PSQR, South African, Valert and Virus-B viruses. (The first three of these viruses are related to the Jerusalem virus which may explain this misidentification. Ed.). I find it rather incongruous that *Turbo Anti-Virus* proved very meticulous in reporting exactly the right variant of some viruses, but failed altogether to detect other viruses.

## Conclusions

If you avoid the hype, and the omissions, the manual is small, simple, but comprehensible to most readers. With its very well designed user-interface, *Turbo Anti-Virus* proved very easy to use. That said, I'm not too impressed by the virus detection capabilities, and the search speed with Turbo Mode switched off still produces wry smiles. However, when Turbo Mode is on, *Turbo Anti-Virus's* search really does fly. The package offers comprehensive facilities deserving a more thorough assessment than time or space permit. It will still be in use when many current anti-virus software packages have long since fallen by the wayside. However, initial assessment suggests that the developers do need to improve the detection rate.

### Technical Details

Product: ***Turbo Anti-Virus***

**Vendor**: There are 25 distributors worldwide. For customer support and marketing: *Carmel Software Engineering USA*, 177 Palisade Ave., Cliffside Park, New Jersey 07010, USA. Tel 201 945 5751, Fax 201 945 9029.

**Developer**: *Carmel Software Engineering*, POB 25055, Haifa, Israel, Tel 972-4-416976/9, Fax 972-4-416979.

**Availability**: IBM PC, AT, PS/2, or 100 compatible with either a 5.25 inch 360K floppy disk drive, or a 3.5 inch 720K (or larger) floppy disk drive. At least 256K of RAM is required, and MS-DOS v3.0 or above. LAN versions are supported.

**Version Evaluated**: 7.03A

**Serial Number**: 681556

**Price**: US $150.00

**Hardware Used:** An Amstrad PPC640 with a V30 processor, and two 3.5 inch (720K) floppy disk drives, running under MS-DOS v3.30. An ITT XTRA, a PC compatible with a 4.77MHz 8088 processor, one 3.5 inch (1.44M) floppy disk drive, two 5.25 inch (360K) floppy disk drives, and a 40Mbyte hard disk, running under MS-DOS v3.30.

**Virus Test Set**: This is a set of 49 unique viruses (according to the virus naming convention employed by *VB*), spread across 101 individual virus samples. It comprises two boot viruses (Brain and Italian), and 99 parasitic viruses. There is more than one example of many of the viruses, ranging up to 10 different variants in the case of the Cascade and Vienna viruses. The actual viruses used for testing are listed below. Where more than one variant of a virus is available, the number of examples of each virus is shown in brackets. For an explanation of each virus and the nomenclature used, refer to the list of PC viruses published regularly in *VB*:

405 (2), 4K (2), AIDS, Alabama, Amstrad (2), Anarkia, Brain, Cascade (10), Dark Avenger (2), Datacrime (3), dBASE, December 24th, Devils Dance, Eddie (2), FuManchu (3), GhostBalls, Hallochen, Icelandic (2), Italian, Jerusalem (6), Kennedy, Lehigh, Macho-Soft, MIX1 (2), Number of the Beast, Oropax, Perfume, Prudents, PSQR, South African (2), Stealth, Suriv (8), Sylvia, Syslock (2), Taiwan, Traceback (4), Typo, Vacsina, Valert, Vcomm, Vienna (10), Virdem, Virus-90, Virus-B (2), VP, W13 (2), XA-1, Yankee (5), Zero Bug.

# END-NOTES & NEWS

**The *Virus Bulletin Conference* on Combating Computer Viruses**, September 12-13th 1991, Hotel de France, St. Helier, Jersey. The programme is now complete and will be distributed presently. Speakers include Fridrik Skulason, Jim Bates, Vesselin Bontchev, David Ferbrache, Ross Greenberg, Jan Hruska, John Norstad, Yisrael Radai, Ken van Wyk, Gene Spafford and Martin Samociuk. Previously unannounced presentations include the *IBM High Integrity Computing Laboratory* (Steve White, *IBM T. J. Watson Research Center*, New York) and *Digital Equipment Corporation's* approach to preventing worm and virus propagation on distributed VAX systems. Specialist sessions on DOS, disassembly, forensics, anti-virus tools, recovery, Macs, DECNet/VMS, mainframes and networks, probable developments, malicious programming, corrupt work practices, blackmail and extortion. Information from Petra Duffield, *Virus Bulletin Conference*, UK. Tel 0235 531889.

*S & S Ltd* and *Virus News International* have moved. The new address for both companies is Berkley Court, Mill Street, Berkhamstead, Hertfordshire HP4 2HB, UK. *S & S* is also holding a **two-day seminar on the virus threat**, February 13-14th 1991. Information from Ann Creamer or Janet Rudkin, *S & S*, UK. Tel 0442 877877.

Successive **seminars on Computer Viruses and Computer Security** will be presented Dr. Frederick B. Cohen, London, UK, 11th and 12th March 1991. Details from *IBC Technical Services*, UK. Tel 071 236 4080.

Cohen has also authored ***A Short Course on Computer Viruses.*** The book costs U.S.$48.00 including postage and packing. Available from *ASP Press*, PO Box 81270, Pittsburgh, PA 15217, USA. Tel 412 422 4134.

**4th Annual Computer Virus & Security Conference**, 14-15th March 1991, New York, USA. Contact the *Computer Society of the IEEE*, USA. Tel 202 371 1013.

*Sophos Ltd* continue a series of **computer virus workshops.** Introductory (14th March 1991) and advanced courses (15th March 1991) are available. Further information from Karen Richardson, *Sophos*, UK. Tel 0235 559933.

**Computer viruses and network security** are the subjects of two seminars being held by *State of the Art Seminars*. The seminars will be presented by Dr. Douglas Tygar, Assistant Professor of computer science at *Carnegie Mellon University*, USA. The events will be held in Rome (10-13th April), Munich (17-19th April 1991) and London (22-24th April 1991). Information from *SAL*, UK. Tel 071 404 3341.

*Elsevier Seminars*, UK, is holding seminars on **Investigating Computer Abuse** (Oxford, UK, 4-6th March, 1991) **Commonsense Computer Security** (London, UK, 18-19th March 1991), and **Contingency Planning and Disaster Recovery** (London, 17-18th April 1991). Tel 0865 512242.

## VIRUS BULLETIN

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

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

### Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, 21 The Quadrant, Abingdon Science Park, Abingdon,
OX14 3YS, England

Tel (0235) 555139, International Tel (+44) 235 555139
Fax (0235) 559935, International Fax (+44) 235 559935

### US subscriptions only:

June Jordan, Virus Bulletin, 590 Danbury Road, Ridgefield, CT 06877, USA
Tel 203 431 8720, Fax 203 431 8165

No responsibility is assumed by the Publisher for any injury and/or damage to persons or property as a matter of products liability, negligence or otherwise, of from any use or operation of any methods, products, instructions or ideas contained in the material herein.