January 1992 ISSN 0956-9979

VIRUS BULLETIN

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

CONTENTS

EDITORIAL

Popp Goes The Weasel

Dr. Joseph Lewis Popp, the alleged author of the notorious AIDS Information Diskette (*VB*, January 90, pp.2-10; *VB*, March 90, p.2) has been set free by a UK court. Judge Geoffrey Rivlin QC at Southwark Crown Court dismissed the case against Popp, who faced charges on eleven counts of blackmail, after testimony from London psychiatrist Paul Bowden to the effect that Popp, 41, was psychologically unfit to plead. *New Scotland Yard's Computer Crime Unit* initially had high hopes that Popp would stand trial; the *CCU*, in conjunction with the *FBI* with assistance from computer analyst Jim Bates had, over the course of a two year investigation, assembled a veritable barrage of forensic evidence to link Popp with the development and distribution in December 1989 of some 20,000 Trojanised diskettes.

According to the psychiatrist's report Popp's mental condition since his extradition to the UK from the United States in May of this year declined considerably. A spell of several days on remand at Brixton Prison is understood to have had a particularly adverse effect on his mental state. His recent antics have included wearing a cardboard box, putting hair rollers in his beard to protect himself from 'radiation' and 'micro-organisms' and wearing condoms on his nose. In recent months Popp has been resident at the Maudsley hospital, a psychiatric care unit in south London.

Following Judge Rivlin's decision, Popp, currently *persona non grata* due to the confiscation of his US passport by British immigration officials, will soon be free to fly back to his home town of Willowick, near Cleveland, Ohio. Investigators and plaintiffs have been dismayed by rumours that Popp has been approached to appear on television and intends to write a book. Prosecuting council Mr Richard Curtis mused 'we just hope that he doesn't do it again, and that we don't hear about him until his book comes out.'

The details of this incident were related in the January 1990 edition of *VB*. In December 1989 some 20,000 diskettes were posted in London to a mailing list of subscribers of *PC Business World* (which has since ceased publication), other business lists and delegates to a *World Health Organisation* conference on AIDS. The 5.25 inch envelopes which contained the disks bore first-class postage stamps. The disks contained an interactive questionnaire and risk assessment of exposure to the biological AIDS/HIV virus, the copyright for which was claimed by the '*PC Cyborg Corporation*'. However, the actual effects of installing and running the diskette amounted to what is construed by many as an elaborate but rather cack-handed attempt to extort money - after approximately ninety reboots the root directory of the hard disk was encrypted and a 'ransom' note demanding payment in US dollars was issued to any connected printer. The money was to be sent to a post office box number in Panama.

The Aids Information Diskette caused enormous disruption, not least to AIDS research. One AIDS organisation in Italy lost ten years of irreplaceable research as a result of panic after installing and running the program. A number of PC administrators were dismissed from European companies as a result of slack procedures exposed by the AIDS disk. Encrypted root directories, which occurred after the Trojan triggered, were still being reported up to one year after the initial distribution of the diskette. Copies of the *AIDSOUT* restorative program, written by Jim Bates, were requested in some ninety countries worldwide; Detective Inspector John Austen, who led the police investigation, estimates that the disk itself was installed by about 5 percent of those who received it, i.e. on approximately 1,000 computers. The police investigation itself amounted to the most intense and costly ever conducted in the history of computer crime.

Police officers leading the investigation were interested as to why no diskettes were mailed to the United States despite the appearance of US subscribers on the rented list; this fact suggested early on that the perpetrator might be familiar with American law. The lists themselves had been bought by a 'Kenyan businessman' by the name of E Ketema - neither he, nor Kitain Mekonen, Asrat Wakjira and Fantu Mekesse (the 'directors' of *PC Cyborg*, a company registered in Panama on 12 April 1989) have ever been traced.

The actual breakthrough in the investigation occurred on Christmas Eve 1989 when Detective Inspector John Austen of the *CCU* was telephoned by a Dutch colleague and informed that a Dr. Popp had been apprehended in an emotional state at Schipol airport. Popp had returned from a *World Health Organisation* seminar in Nairobi after reading of the disruption caused by the AIDS disk which was widely reported in PC magazines and newspapers worldwide. At this point he appears to have suffered a nervous breakdown. On arrival at Schipol he alerted the authorities by scribbling 'DR POPP HAS BEEN POISONED' on a passenger's luggage!

He was subsequently found to be in possession of materials relating to the *PC Cyborg Corporation*. However, Popp was released by the Dutch authorities and allowed to proceed from Amsterdam to Ohio where an FBI team kept the house in which he lived with his parents under surveillance. Extradition proceedings followed the issue of an arrest warrant by New Scotland Yard in early January 1990.

The motivation of Popp, who according to his lawyers admits having sent out the disks, remains the subject of much speculation. His UK solicitor Gareth Peirce is adamant that Popp is innocent of any attempted blackmail charge on the grounds of diminished responsibility. His defence council in the United States argued in earnest that Popp intended to donate any 'revenues' from his project to AIDS research - the HIV virus itself is reported to fascinate Popp. The whole

incident is explained by Popp's lawyers as the manifest actions of a mind at its most *irrational*. Certainly, his behaviour during the past 24 months would appear to be deranged.

However, a number of people conversant with this case harbour doubts. The cost of disk duplication and distribution alone exceeds £10,000. A massive logistic effort was undertaken in executing this crime - there was the bulk duplication of diskettes, mail-list purchasing and de-duplication, packaging, applying stamps and address labels, the hire of an accommodation address in London, the registration of *PC Cyborg* in Panama - if Popp was insane and did commit this crime, then there was method in his madness. If all the recipients of the disk had paid the full 'licence fee' for the programs (US$378), then the perpetrator(s) stood to net somewhere in the region of $7.5 million dollars! If just one percent of the intended victims had paid the *minimum* 'licence fee' (US$189), the '*PC Cyborg Corporation*' would have received a figure approaching US$38,000 - sufficient finances to recover its costs.

The technical evidence accrued from examination of Popp's computer and media in Ohio suggests a more calculated mind at work than Popp's lawyers have suggested. His US attorney confirmed earlier this year that Popp had been prepared to duplicate and distribute a further *two million* diskettes. One diskette obtained by the police contained an encrypted diary detailing the conception and development of the AIDS Information Diskette. Jim Bates, who decrypted this disk, discovered that the encryption key was 'Dr. Joseph Lewis Andrew Popp Jr.' - a discovery akin to a forensic jackpot. The police also obtained the entire source code to the AIDS disk, although the exact circumstances of this find have not been disclosed. As the diary information dated from April 1988, it appears this plan had been in development for some twenty months prior to its execution. Rarely, for a computer crime case, the technical evidence linking Popp to the development (if not the distribution) of the disk is close to being incontrovertible.

The legal eagles are still debating the fine print of the case; the fact that the 'documentation' supplied with the diskettes warned the user of unpredictable results, albeit in the smallest of small print, is seen by some lawyers as a loophole in *any* prosecution case. Moreover, filenames were encrypted rather than destroyed which reduces the impact of a charge of criminal damage. The only lasting outcome of this incident was the expedited passage of the *Computer Misuse Act* which became English law in August 1990.

The legal debate is now academic. By the time that this edition of *VB* is published, Dr. Popp will have returned to the United States and is unlikely to face further charges. Popp was no evil genius and his alleged crime was far from perfect. However, one could be forgiven for getting the impression that having learned the secrets of effective direct mail, he is now laughing all the way to his publisher. We look forward to reading his version of events.

INTERNATIONAL

Novell Ships Virus

A report by John Markoff in the *New York Times* (December 20th 1991) states that *Novell* has circulated a letter to approximately 3,800 customers warning of a virus infection of a disk shipped by the company on December 11th.

Novell Inc. of Provo, Utah has traced the infection to a particular part of its manufacturing process, although the company has not stated whether the virus had infected the master disk or whether infection occurred during duplication.

John McAfee's *SCAN* product identifies the virus which infected copies of the *Network Support Encyclopedia* as Stoned III (*VB* first reported this virus under the name of NoInt in September 1991). The virus has also been called the Bloomington virus for reasons which are at present obscure. The *Network Support Encyclopedia* is a diagnostic and reference program distributed to certified *NetWare* engineers and network administrators.

NoInt is a memory resident virus which infects the Master Boot Sector (Track 0, Head 0, Sector 1) as well as diskettes in drives A and B. A primitive stealth feature (similar to that first used in the Brain virus) returns the original boot sector when any attempt to read Sector 1 is attempted with the virus active in memory. The original boot sector is stored in Head 1, Track 0, Sector 3 on infected diskettes and on Head 0, Track 0, Sector 7 on hard disks. A reliable search pattern for the virus follows:

```
00B9 0002 161F 33F6 8BFE FCF3 A436 FF2E
```

Due to the fact that the virus infects boot sectors it will not spread over a network via the file server. Infection is spread purely by disk interchange. The *NetWare* boot sector is proprietary and not compatible with a DOS boot sector which means that *NetWare* file servers are effectively immune to DOS boot sector virus infection.

Mr John Edwards, director of *NetWare* marketing, says in the report that *Novell* plans to incorporate digital signatures and other unspecified protective features to its next major release of *NetWare* due in the autumn.

The *New York Times* report contains a degree of journalistic licence - the report warns of 'massive potential liabilities' and quotes John McAfee as saying - 'If this was to get into an organisation and spread to 1,500 to 2,000 machines, you are looking at millions of dollars of cleanup costs.' However, *Novell's* prompt warning, the inherent limitations of the virus itself, and the fact that the infected software was sent out to system administrators and engineers renders such a doomsday scenario fanciful in the extreme.

HEADLINERS

'Write' Said Fred

Dr. Frederick B. Cohen, regarded by many as the great-grandfather of self-replicating code from his pioneering Unix virus experiments in 1984, has recently astonished the so-called virus research community by devising and publicising a virus writing competition. The competition entitled *The Computer Virus Contest* offers a first prize of $1,000 donated by Cohen's company *ASP* and has been widely publicised in magazines such as *NetWare Solutions*.

Striking the Right Balance

According to Dr. Cohen the contest (which will take place annually) is designed to try and strike a reasonable balance between the uncontrolled experimentation that is now the norm, and an over-controlled environment in which no viruses may ever be written for any purpose.

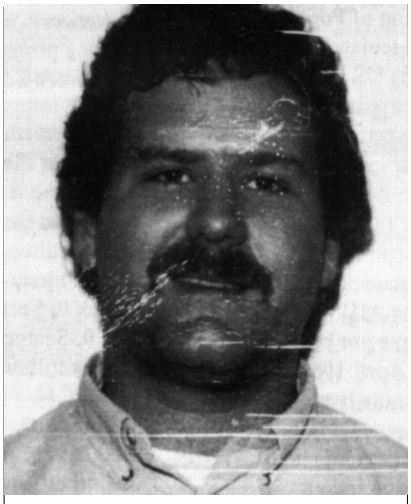
The competition rules state that the contest exists to promote the *beneficial* and *safe application* of computer viruses. 'It begins' he says 'to address the issue of how we can perform experiments in relative safety, which should be the subject of standards developed by the research community.'

Dr. Cohen evidently regards the contest as a platform by which the motivation of the virus writers may be channelled towards beneficial activities. In a recent communication to *VB*, he explained: 'We cannot stop all the virus writers, but maybe we can get some of them involved in legitimate activities.'

He continued: 'If you take the position that there is not and never can be a useful virus, you may close the door on a technology that could have a dramatic impact on the future of computing. There are some useful viruses already in use and more may come up over time. One vital issue is whether we throw out the baby with the bath water.'

Before discussing the relative merits (or otherwise) of such a scheme, it is worthwhile to outline some of the competition's more important clauses.

To be eligible for Dr. Cohen's prize, contestants must write and test their viruses only on systems where such experimentation is permitted by the systems administrator. Submissions will be accepted on all types of computer, although (not surprisingly) the guidelines state a preference for IBM-PC, Macintosh and Unix entries. The judging panel is not specified but is claimed to comprise 'members of the international computer virus research community' who are selected at the 'sole discretion of *ASP*.'



Cohen - 'don't throw the baby out with the bath water.'

Sponsorship for the competition is welcomed from any individual or organisation willing to submit a minimum $1,000 donation. Any sponsor is provided access to all entries submitted and can tender for 'licensing agreements' with the virus developers. This final clause is presumably included on the assumption that the virus code submitted will have genuine commercial value. Ironically, competition entrants are warned against plagiarism: 'DO NOT USE CODE FROM EXISTING VIRUSES!!!' state the competition rules.

Ominously, the rules also state: 'Contest entries and related information may be published by *ASP* in a contest publication. All contestants, by virtue of their submission, grant *ASP* the eternal, royalty-free, non-exclusive, non-transferable, non-assignable right to publish and sell the submitted materials, in whole or in part, in any form, for this purpose.' Finally, a legal disclaimer divests *ASP* of any responsibility for the submissions to the contest which is held and governed under the laws of Allegheny County, Pennsylvania, USA.

Courting Controversy

Dr. Cohen is no stranger to controversy, he was after all, the first man widely to publicise the theory of computer viruses and has enjoyed the unenviable position of computing's bête noire ever since. Never a man to shy from making unpopular statements (his assertion in *VB* that the AIDS Trojan and DOS *DISKCOPY* are computer viruses certainly raised a few eyebrows), his eccentricities have in the past caused relatively minor ripples - the same cannot be said of his latest brainchild.

Apparent Dangers

Dr. Cohen's contest is unashamedly Utopian and quite attractive for its seeming innocence. Unfortunately, there are obvious dangers associated with such a scheme.

Principally, this contest actively *encourages* and *legitimises* computer virus development but provides no safeguards against the distribution (intentional or unwitting) of the resulting object or source code. The term 'beneficial' in the context of the competition rules is so vague as to be meaningless - one man's meat is, after all, another man's poison. Dr. Cohen knows better than anyone that the effects of virus code, once developed, are unpredictable and its spread uncontrollable. Source code listings developed as entries for this contest could easily be modified to include malicious instructions, subsequently compiled and the resulting binary

uploaded to Bulletin Board Systems. There is already a mountain of documentary evidence to prove that so-called benign viruses have been quickly and easily converted into malicious samples and then re-released into the wild.

It is also probable that, by necessity, search data to detect viruses born of this competition will eventually eat into the limited and increasingly stretched resources of the world's virus scanners. Dr. Cohen's contempt for virus-specific detection methods is well known and documented but his posturing about integrity shells and other exotica will be of little comfort to hard-pushed scanner developers and anxious PC users. In a bout of arithmetic speculation, Dr. Alan Solomon of *S&S International* has already estimated that the real world financial penalties imposed on the *end user* as a direct result of this competition will amount to at *least* $20,000 - twenty times the total prize money (*Virus News International*, December 1991). This calculation is based on the dollar price of disk storage in Kilobytes and projected disk resources consumed by the inclusion of just *one* additional virus detection signature to scanning software. It does not begin to account for such imponderables as run-time cost, increased memory-resident footprints, update QA and despatch etc. Regardless of whether we accept Dr. Solomon's chosen criteria, few dispute his basic assertion that a price of some sort will be exacted as a result of this competition.

Pandora's Box

What has shocked many members of the loosely formed anti-virus community is Dr. Cohen's apparent belief that experimentation is necessary at all. The fundamentalist faction maintains that writing an experimental computer virus is an admission of intellectual defeat and is never excusable. 'Cretinous' was one such researcher's reaction when informed of Dr. Cohen's latest antics.

In fact, in international law, there are currently no clauses which forbid *experimentation* with self-replicating computer programs - that legislation which does exist criminalises unauthorised virus *distribution*. There is nothing to forbid a programmer from infecting his own computer or indeed that of any other consenting adult! Most researchers believe that if virus code can genuinely be used for beneficial purposes then by all means experiment, but do so in a *safe, methodical* and *controllable* way. This competition in the views of many falls short of these requirements.

Dr. Cohen's intellectual and moral integrity is not in question; there is no reason to doubt his stated aims. However, the road to hell is paved with good intentions. With Pandora's box now well and truly opened and virus code available even by direct mail, Dr. Cohen's activities might appear relatively innocuous: unless, of course, you believe he opened the box in the first place!

Love him or hate him, Fred Cohen has certainly started one hell of a debate.

RIGHT OF REPLY

'Wrong' Said Fred

Dear Ed,

Just some minor comments on the article 'Write Said Fred' (opposite).

1 - If Solomon is right, then the current disk cost of computer viruses scanning defenses on PCs is roughly 1,000 x $20,000, or $20,000,000! If this is true, the virus defenders should enter the disk marketing business - it's more lucrative!

Why does Dr. Solomon think we should scan for these benevolent viruses? The contest requires that they be well controlled and not spread without permission. The contest also requires that means and methods for removal be provided with the viruses. That sounds to me like any other software product on the market, We don't hear these complaints about *DR-DOS 6* which destroys many files on systems during installation and is distributed to hundreds of thousands of users by normal commercial means. Should our scanners identify *DR-DOS 6* and warn users about the current bugs in *Windows*?

And how about the savings? What if it turns out that benevolent viruses revolutionize computer programming in networked environments and result in massive reductions in cost and improvements in system operation? Some benevolent viruses have been in safe use for 4 years with no ill effects, no uncontrolled spreading, and substantial impacts on costs! Should we turn them off and explain to users that because of some people in the research community we have to change over to a less cost effective system?

2 - You are right about the path to hell, but the question is whether your path or mine leads that way. I think that the path to hell is the path where we criminalize technologies out of ignorance rather than seek ways to make good use of them. Your position, apparently, is that the path to hell is the attempt to apply potentially hazardous technologies in a safe way.

My path brings virus writers out of the closet and exposes them to the bright light of day. It provides positive incentives for good work and creativity and creates a responsible social environment for this work. It starts the debate on safe viral computing environments and encourages the creation of standards.

Your way forces them into an underground where they must hide their identity and seek refuge with criminals. It will prevent only the most honest people from writing computer viruses, while making those who write viruses susceptible to blackmail and a life of fear. And what of the social outcasts who enjoy writing viruses? Do we label them 'cracker' and

drive them into an underground group opposed to society, or do we embrace them for what they are and create a society in which they have the same rights as we do?

Which brings me to another interesting issue. How many of the 'legitimate' researchers who have proclaimed the writing of viruses to be utterly wrong and disdainful have NEVER written a virus themselves to find out how hard it is? I would guess that there are over 1,000 people who consider themselves legitimate virus researchers who have written viruses, and of those, I would guess that only a very small number have 'escaped' into the real world. What makes one self-declared researcher more legitimate than any other?

The path to hell is indeed full of good intentions, but who among us are so intimate with God that we presume to believe we know which path is right? Who is so confident in their judgement that they are willing to enforce their view on the rest of society? I am not that confident, and I don't think you are either.

Sincerely yours,

Dr. Frederick B. Cohen
President - *ASP*

Virus Bulletin Conference 1992

Call For Papers

Abstracts of 300 - 1,000 words are invited for papers to be presented at the *Second International VB Conference* in Edinburgh, September 2nd-3rd 1992.

The conference will be in two streams: Stream one will address the **management of the virus threat in the corporate environment**, the second stream will concentrate on **technical developments** including disassembly, detection and classification.

Abstracts are welcomed from individuals or groups active in research, software or hardware development, quality assurance, the law, corporate security management, or any field related to countering computer viruses.

Abstracts, which should be completed by February 15th 1992 and should be sent to The Editor, *Virus Bulletin*, 21 The Quadrant, Abingdon Science Park, Abingdon, Oxon OX14 3YS, UK.

CONFERENCE REPORT

Dr. Jan Hruska

NCSA Anti-Virus Product Developers Conference, November 25-26th 1991

Given the current state of relations between several anti-virus software vendors, it was a minor miracle that most of them should agree to assemble under one roof. The *National Computer Security Association* in the USA headed by Robert Bales, David Stang and their team, enabled such a miracle to take place. *The Marriott Hotel* in Washington DC provided the arena and various combatants came heavily armed with a variety of exotic weaponry and prepared ripostes. Despite eager expectations of mortal combat, the only battles were fought verbally over facts, statements, and views.

The proceedings opened with a presentation by the *National Institute of Standards and Technology* (Gaithersburg, Maryland, USA) on its proposed virus naming convention. Initially this proposal appeared promising but in the ensuing discussion the delegates agreed to disagree. Alan Solomon (*S&S Ltd*) enquired wittily whether he would receive a US Government grant to reprint all his product literature. It was only at the end of the second day, that *NCSA* agreed to adopt Patricia Hoffman's *VSUM* virus list as the official *NCSA* virus naming convention. It is yet to be seen whether this will have any effect on the commercial world.

The code of ethics proposed by Peter Tippett (*Certus International*) was an attempt to legitimise what a visiting journo called 'the slime industry'. The intention was to have all anti-virus researchers, publishers and professionals sign a Hippocratic oath before being allowed to practice. This is, in theory, a Good Thing, but in the absence of an enforcing organisation, any such document would be meaningless.

As one of the ways of raising funds, *NCSA* is running an anti-virus product certification scheme for different product categories. The testing will involve exposing each product to different viruses in the *NCSA* collection, as well as testing the product's compliance with advertised functionality. *NCSA* is also making its virus collection available on-line for downloading by authorised developers who have satisfied a number of conditions. The collection is supplied encrypted with a frequently changed password. Anti-virus software producers were invited to join the *NCSA* and were offered various categories of membership.

John McAfee (*McAfee Associates*), suffering from a throat infection that not even his software could cure, gave a presentation on some current virus problems. His Canadian distributor outlined some of the problems that the Canadians had in apprehending virus writers.

I tackled the increasing problems surrounding anti-virus software QA procedures. False positives and false negatives were the main problem, but in the ensuing discussion, the audience could not agree whether it were better to cry wolf a few times too many or once too few.

Presentations by those responsible for anti-virus policy in the *Boeing Corporation* and *Equitable Life* provided an insight as to the difference between corporate users' expectations and software manufacturers' abilities to fulfil them. David Figge from *Boeing* presented the spec-sheet for the anti-virus tool that he needs, and although a few enthusiasts tried to attract his attention claiming to have just the product, the majority of vendors were silent. The man was asking the impossible and we all knew it. Don Senges from *Equitable Life* asked the audience to suggest what items should be included in an 'anti-virus bag' when a layman is sent out to deal with a large-scale virus infection. Again, several vendors raised their hands in hope, but Mr. Senges was after more than just anti-virus software. The bag was eventually filled with DOS (all versions), comprehensive DOS documentation, unspecified but all-embracing anti-virus software for the PC and the Mac (presumably with bells and whistles), and a host of other all-singing, all-dancing gizmos.

A useful statistics to be revealed was the result of a recent poll of *Network World* readers, of which 32% stated that viruses were their major computer security preoccupation. Unauthorised use of computers worried 19% of the readers, hackers worried 18% and telephone time stealing on PABXs 8%.

Fridrik Skulason (*Frisk Software* and *VB's* Technical Editor) then spoke on the genealogy of viruses, despite continual interruption from a fire alarm (yet more false positives!) and he was followed by Carl Bretteville (*Arcen Data*) in Norway who discussed a risk assessment matrix to ascertain the level of danger presented by different virus samples.

Eric Babcock of *Novell* listed *NetWare* security features as they applied to virus control. He also gave a sneak preview into future security features which will be built into this operating system. *NetWare* currently accounts for 60% of the LAN software market worldwide. (See story on *Novell*, p. 3.)

During lunch Ken Wasch of the *Software Publishers Association* (the US equivalent of *FAST* - the UK's *Federation Against Software Theft*) told us about the work that *SPA* is doing to prevent unauthorised software use. He gave an assurance that *SPA* would not prosecute a virus researcher who takes a copy of infected software for analysis purposes.

The inimitable Alan Solomon spoke during the siesta hour with gusto and panache. He chastised users for ignoring the virus problem, he chastised software manufacturers for not comprehending the potential scale of it, he chastised lawmakers for not prosecuting virus writers. Very few escaped his mighty thunder. He even assaulted himself, nearly fatally, by inadvertently winding his microphone cable round his neck.

Peter Tippett then proposed legislation to combat computer viruses, which included the requirement to advertise any virus experiments in at least three national publications one week before performing the experiments. In the discussion which followed it became evident that the biggest virus problems are caused by people who never announce their intentions!

One of the most revealing sessions of the conference was the publication of a *Dataquest* survey to identify the nature and the extent of computer virus problem on PCs in the USA. 602 corporate end-users were interviewed by telephone during October 1991. All interviews were conducted on sites with 300 or more PCs and the majority of respondents were responsible for controlling the virus problem on all the PCs. 63% of respondents reported at least one encounter with a virus over the past year, and 9% had more than 25 PCs infected in the process. New Zealand was responsible for 48% of the attacks, Jerusalem for 37% and Joshi for 8%. Interestingly, the major entry points of viruses into the organisations surveyed were disks brought from home PCs (43%), followed by the virus-infected programs pulled down from bulletin boards (7%) and demonstration disks and service engineers (6%). Only 1% of infections came with the PC from the dealer or the factory. When asked to estimate the cost of each virus infection, 31% of organisations spent less than $2,000, 29% between $2,000 and $10,000, while 7% spent more than a staggering $100,000.

Equally fascinating were the reports from various parts of the world. It was with some incredulity that the delegates heard that the Whale virus is rampant in Western and Southern Australia and New South Wales. After all, in the best Stalinist tradition, the research community has declared Whale a non-virus. One well known researcher wrote: 'The virus is so large and clumsy that on most computers it doesn't actually work and when it does, it doesn't work for very long. As a result the main replication method of Whale is anti-virus researchers sending specimens to one another.' In fact, Whale replicates comfortably on the 8088 processor (as opposed to the 8086 or '286 and above) and is currently prevalent in Australian schools, where the 8088 is standard equipment.

Vesselin Bontchev (*University of Hamburg*) warned the anti-virus vendors of a new infection technique which is being discussed on bulletin boards in Bulgaria. When he explained the principle, beads of sweat broke out on a number of foreheads. Shimon Grouper (*Eliashim Micro*) from Israel stated that such a virus (Freddy) has already been seen in the wild. Bontchev's revelation necessitated a subsequent minor addition to the *Sophos Vaccine* user manual and I suspect it will in other anti-virus products.

The conference was a resounding success. Several people who had communicated only by e-mail met face to face, virus collections were exchanged and we could see for ourselves that our competitors have neither horns nor cloven hooves. The next conference will be held in Washington DC on 19th and 20th November 1992. I, for one, look forward to it.

RESEARCH MATTERS

Virus Collections - Sorting Sheep From Goats

Virus collections vary in size, some contain only a handful of viruses while others such as the *NCSA* collection consist of several thousand samples with numerous duplicates. Several such collections are used to conduct comparative reviews of anti-virus software, particularly of scanners.

Such collections contain files which no anti-virus program need ever detect; the very existence of these files leads to unfair comparisons between those programs which detect only real viruses and those which detect these extra harmless files as 'viruses'. A number of collections even contain perfectly legitimate copies of *CHKDSK*, *FDISK* and *FORMAT* - any scanner which detected such programs on the supposition that they were viruses would cause chaos! Other common 'viruses' include proprietary low level formatting programs, a variety of sacrificial goat programs in all shapes and sizes and even anti-virus programs - all purporting to be virus code.

Non-viruses in existing virus collections can be divided into the following groups: 1) *Trojans*; 2) *Joke programs*; 3) *Modified, non-working viruses*; 4) *Droppers*; 5) *Unknown.*

Trojans

Trojans are a legitimate threat and many of them are destructive. It make sense for an anti-virus program to include a check for common Trojans, but there are two distinct differences between viruses and Trojans, which make it unlikely that most users will encounter the latter. Firstly, Trojans don't spread unassisted - they do not replicate (unless the user copies them). Secondly, most Trojans are very obvious - when they activate they usually attempt a primitive act of destruction.

Only a handful of Trojans have been reported more than once in the wild. The AIDS information Diskette and the 12-Tricks Trojans are two well known and widely distributed examples. However, most Trojans are so limited in their distribution as to render concerted efforts to detect them pointless - they can usually be countered with a simple generic monitoring program capable of intercepting disk formatting requests and other potentially destructive routines. The majority of Trojans in virus collections around the world are extremely primitive programs which may never be found in the wild. At the very least they should be separated from the actual virus collection.

Joke Programs

Joke programs are often included in virus collections because they appear on Bulletin Boards in the company of virus code. Somebody may download the lot, send the files to a virus researcher, who simply merges the files with his collection, without checking them first. Until recently, two widely distributed joke programs (MUSHROOM.COM and DRAIN.COM) were frequently found in 'virus' collections. Producers of virus scanners sometimes detect joke programs (*Findvirus* version 4.26, for example, detected two such programs called BUGS and BUGSRES from the former Soviet Union), but as they are not a real threat they should be removed from collections used for comparative reviews.

Non-Working 'Viruses'

Many 'virus' collections contain files which do not replicate under *any* circumstances. One example is a file containing a modified Sylvia virus. Sylvia is one of the very few viruses which performs an internal integrity check by computing a simple checksum of its own code. This modified variant fails this check, displays a message and hangs the machine when run. It will never infect anything, but nevertheless this variant is included in many large collections and quite a few virus scanners identify it. Pentagon is another example of a crippled 'virus' - no working sample has ever been made available but despite this fact, most current scanners incorporate search data to detect it.

Some functioning viruses are processor-specific (i.e. they will replicate only on an 8088 processor or contain 'illegal' instructions on the 80286 etc.). Viruses which only replicate under certain highly specific conditions impose an enormous added research burden. The basic rule is that if a program replicates then it is a legitimate file to include in a comparative review test set, otherwise it should not be included.

Other 'viruses' that appear in collections have sections of their code 'nulled out', possibly in an attempt to disable replicating mechanisms or destructive routines. Using such deformed samples for comparative tests is unwise - the very sections of code richest in suspect code and therefore most likely to be selected for search data are likely to have been removed.

Droppers

A dropper program is a small program designed to launch ('drop') virus code but which does not replicate in itself. A dropper may decrypt a virus, or write boot sector code to a diskette or install a resident virus in memory from where the virus proceeds to infect files normally. There are numerous dropper routines in circulation but since these programs are non-replicating they should not be included in comparative test suites.

Unknown

Anything which has not been proven to replicate falls into the 'unknown' category. There are several samples which are included in many virus collections despite the fact that nobody has been able to make them infect anything. Examples include files such as SCORPIO.COM and the 'Pink Elephant' - programs which can be found in many virus collections. These 'unknown' files may contain viruses - careful analysis, such as a comprehensive disassembly may reveal necessary conditions for replication. Just as the non-working viruses, the 'unknown' files should not be used for comparative purposes - and if distributed to other researchers they should be marked as 'unknown'.

IBM PC VIRUSES (UPDATE)

Updates and amendments to the *Virus Bulletin Table of Known PC Viruses* as of 20th December 1991. Hexadecimal patterns may be used to detect the presence of a virus with a disk utility or preferably a dedicated virus scanner.

Type Codes

C = COM **E** = EXE files **D** = Infects DOS Boot Sector (logical sector 0 on disk)
M = Infects Master Boot Sector (Track 0, Head 0, Sector 1) **N** = Not memory-resident after infection.
R = Memory-resident after infection **P** = Companion virus **L** = Link virus

Seen Viruses

757 - CR: This virus displays a 'Bouncing-Ball' effect on the screen.

```
757             B907 00FC F3A4 585B 9DB8 0001 5350 CB9C 3D00 C774 DB3D 01C7
```

765 - ER: This virus is related to the '905' virus, perhaps an older version. Awaiting analysis.

```
765             53B4 368E 4602 8B76 0A26 8A14 80EA 40CD 213D FFFF 740E F7E3
```

907 - CR: An encrypted 907 byte virus, awaiting analysis.

```
907             83C7 0353 2EFF B55D 04BB DE03 B97F 0058 2E30 0143 E2FA 5BE8
```

1963 - CER: A Bulgarian virus, which does not increase the size of the files it infects. Awaiting analysis.

```
1963            B820 12BB 0500 CD2F 534B 4B26 881D B816 12CD 2F4B 4B26 891D
```

4870 Overwriting - EN: A strange overwriting virus which spreads in *LZEXE*-packed format. It is not possible to select a search pattern from the code portion of the virus.

Boojum - ER: A simple 334 byte virus which does nothing other than replicate.

```
Boojum          9C3D 004B 7510 5689 D646 803C 0075 FA80 7CFF 4574 075E 9DEA
```

Burger-Pirate - CN: This 609 byte overwriting virus is a simple modification of the original Burger virus, with a text message added at the end, which indicates the virus was written in Portugal. It is detected by the pattern published for the Burger virus.

Burghofer - CR: A simple 525 byte virus from Switzerland, which appears to do nothing of interest.

```
Burghofer       B448 CD21 5B48 8EC0 FA26 C706 0100 0000 2680 3E00 005A 7550
```

Cascade-1661 - CR: A rewritten version of the Cascade virus. It has been modified in several ways, changing the activation date to December of any year other than 1980 and 1990.

```
Cascade-1661    012E F684 9301 0174 0F8D BCB6 01BC 5A06 313D 3125 474C 75F8
```

Dutch Tiny-126 - CR: This virus from the Netherlands is an attempt to create the smallest resident virus, but it has no effect other than replicating.

```
Dutch  Tiny-126  930E 1FB4 3FCD 218B F280 3C4D 741C B002 E8CF FF97 B97E 00B4
```

Haifa - CER: This virus from Israel uses self-modifying encryption to hide itself. The length is around 2350 bytes, but variable. No search pattern is possible.

Hitchcock - CR: A 1247 byte virus. It activates a few minutes after an infected program is run, and plays the tune from the Alfred Hitchcock TV-series.

```
Hitchcock       2BD0 4A45 03E8 8EC5 4526 8916 0300 2689 2E01 0026 C606 0000
```

Illness - CR: This encrypted 1016 byte virus is probably of Polish origin. It contains the text 'WARNING : USE ONLY ORGINAL PROGRAMS DON^T COPY IT and now .. I AM ILL !!' The original sample was infected with Cascade-1701A, which has caused some confusion.

```
Illness         BAF8 0383 EA20 33FF 3E8A 86F3 043E 2883 1A01 473B FA75 F6
```

Jerusalem-1767 - CER: This 1767 byte version contains the text '** INFECTED BY FRIDAY 13th **'. Awaiting analysis.

```
Jerusalem-1767  7F33 C0F2 AF8B D783 C202 B800 4B06 1F0E 07BB 3500 1E06 5053
```

Jerusalem-Einstein - ER: An 878 byte variant of the Jerusalem virus, which is not able to infect .COM files. Awaiting analysis.

```
Einstein        7FF2 AE26 3805 E0F9 8BD7 83C2 0306 1F0E 07B8 004B 9C2E FF1E
```

Jerusalem-Miky - CER: A 2350 byte variant of the Jerusalem virus, which is reported to have originated in Bolivia.

```
Miky            7F32 C0F2 AE26 3805 E0F9 8BD7 83C2 038C C08E D88C C88E C0BB
```

Jerusalem-T13 - CER: An 1807/1812 byte version of the Jerusalem virus detected by the pattern for the Suriv 3.00 variant.

Jihuu - CN: A Finnish 621 byte virus, which may display various messages, depending on the current date and time.

```
Jihuu           8BCA 83EF 0489 0D89 4502 B800 4233 C933 D2CD 21B0 E988 4501
```

Liberty-SSSSS - CR: This 1170 virus resembles the Liberty virus, but may not be directly related. It is 1170 bytes long.

```
Liberty-SSSSS   FACD 21FA 0E1F B425 A02E 01BA FFFF 1F1E CD21 0706 0E1F BF00
```

Mosquito-Topo - ER: A 1536 byte variant of the Mosquito virus. Awaiting analysis.

```
Mosquito-Topo   5650 BE68 002E 8A24 2E32 263D 002E 8824 4681 FE49 0375 EE58
```

MPS-OPC 4.01 - ER: This virus is probably written by the same author(s) as the other MPS-OPC viruses - a Mr. Marek Pande, according to reports from Poland. Structurally it is very different however, and belongs to a different virus family. Awaiting analysis.

```
MPS-OPC 4.01    CD27 A12C 008E D833 FF8B 0547 0BC0 75F9 83C7 038B D7C3 3D00
```

Murphy-Bad Taste - CER?: This virus should be able to infect COM files, but during testing it only infected .EXE files, unlike other Murphy variants. It contains the text 'Bad Taste Ltd. (C) 1991 by Odrowad Trow.....who am I???'. This 1188 byte virus is detected in EXE files by the pattern published for Murphy-2.

NV71 - ER?: This virus has been reported elsewhere as '1840', but this name should be avoided, as the virus is only 1827 bytes long. It has also been reported to infect .COM files, but this has not been confirmed.

```
NV71            9CFA FC8C DA83 C210 2E01 1603 0033 C08E D881 3E86 0300 B875
```

Possessed-B - CER: A 2446 byte variant of the Possessed virus, and detected by the pattern previously published for that virus.

Pregnant - CR: A 1199 byte encrypted virus, related to the 1024 PrScr virus. It activates on Fridays, between 10 PM and 11 PM, making all infected files appear to be named PREGNANT.!!! if the user issues a DIR command. As the decryption routine is very short, only a 16 byte search pattern containing a wildcard is possible.

```
Pregnant        B99F 04BE 1001 B4?? 3024 46E2 FBEB 7990
```

Shadowbyte-2 - CR: A 635 byte variant of the Shadowbyte virus. When it activates it formats the first track of the first hard disk. .

```
Shadowbyte-2    B405 B280 B600 B500 B002 CD13 B405 B200 CD13 B400 B003 CD10
```

Tokyo - EN: A 1258 byte virus, reported to have originated in Japan. It appears to contain no side-effects.

```
Tokyo           B42F CD21 8C06 0600 891E 0400 0E07 8D16 0800 1E06 1F07 B41A
```

TPWorm - EN: This Bulgarian virus was first made available in source form only, but now an executable has appeared. It is 12969 bytes long. Due to the unreliability of search patterns for high level language viruses (they can be invalidated if the code is compiled with a different compiler) no pattern is given here.

VCS-Manta - CN: A virus generated by the German VCS program (virus construction set). Detected by the VCS 1.0 pattern.

VCS-VDV-853 - CN: This virus is detected by the same pattern as the VCS 1.0 virus, but is somewhat different and only 853 bytes long. Awaiting analysis.

Reported Only

1024 SBC - CER: Reported to be a 'stealth' virus.

1452 - CR: Unknown effects

ADA - CR: A 2600 byte virus from Argentina.

Argentina - CR: A 1249 byte virus from Argentina. It may display messages on various dates of patriotic significance in Argentina.

CRF - CN: 270 bytes. Contains the text 'OZR3'

Error - ER: A 628 byte virus from Argentina.

Guillion - MR: A boot sector virus from Argentina.

Reset - CN: A 440 byte virus, possibly identical to Omega, reported last November

V82 - CER: A 2000 byte virus, reported to be from Bulgaria.

Windmill - DR: A boot sector virus, reported to be from the Philippines

VIRUS ANALYSIS

James Beckett

PC-Flu II

Although the virus world sees rather few radically new ideas at the moment (a happy state of affairs!) certain trends and changes of approach have been noticed. Viruses are increasingly armoured with a variety of anti-debugging devices, the most concerted attempts to confound disassembly being Whale and Mark Washburn's V2Pn series.

The Maltese Amoeba virus (*VB*, December 1991) made a meagre attempt to protect itself from prying eyes while a new virus called PC-Flu II makes several, but even here, none are likely to confuse a competent analyst. The countermeasures used come into their own only when one tries to employ an 'intelligent' debugging tool (such as *Sourcer*) which analyses certain aspects of the virus code automatically. However, the relative complexities of the human mind can easily sidestep such attempted subversion - a task beyond the grasp of any program short of a virtual system emulator. Often the simple tools turn out to be the best for the job. Who needs more than *DEBUG* and a sober biological processor fired up on caffeine?

Categorisation

PC-Flu II is a lightly armoured, resident COM and EXE file infector with no payload, destructive or otherwise. Files are infected through both the DOS Load-and-Execute and File Open requests, and under certain conditions the virus will fake the contents of an infected file, making it appear untouched. The main body of the virus is encrypted, using a variable key, and the initial decryption routine mutates at each generation.

Armour

The PC-Flu II virus employs two tactics for disrupting analysis, which would potentially cause problems if more liberally used. The first method occurs just once, while the other one is invoked only four times. Neither technique necessitated a return to the virus sample.

In medieval times, it was often not only acceptable but considered good practice to write self-modifying code, where a program changed its behaviour not by modifying data values, but by changing the bytes comprising its own instructions. In fact with the sizes of memory available at the time it was often impossible to do a task any other way. In today's enlightened age, of course, no self-respecting programmer would dream of it. Enter the virus writer...

Designers realised some time ago that the efficiency of microprocessors can be increased by using spare bus time to pre-fetch the next few bytes of instructions. This has the curious corollary that if the memory is modified a very short distance in front of the current instruction, the processor never sees the change, as it has already read the relevant bytes. However, when being traced by a debugging tool, each instruction is interrupted to examine the state of the processor and this clears the queue of pre-fetched instructions. The program will then execute the modified instruction and with judicious choice of changes can tell that it is being traced. One gets the impression that the virus is watching *you* watching *it*!

However, this fascinating subterfuge can very simply be disabled. There is an art, or rather a knack to virus disassembly. The knack lies in never letting the virus code execute unless you have complete control of it, a target very difficult to attain in automated debugging software. The skill lies in spotting and defusing the virus writer's boobytraps before they explode in your face.

Stack of Lies

The other underhand trick that PC-Flu II employs could force the analyst to disassemble parts of the code by hand, or repeatedly return to the sample, wasting further time and effort. Subroutine calls, one of the definitive constructs of modern programming, allow execution of a block of commands from several different points in the program. The program stack is used to keep a record of the return address each time, which is invariably that of the instruction following that which initiated the subroutine. In this virus several subroutines modify the contents of the stack to return control one byte further along in the code, such that one byte is never used. A disassembly tool, blithely spewing forth its interpretation of the bytes it sees in sequence, has no way of knowing that one byte is never actually *used*, so the listing no longer shows a true picture of the code being run. If the unused byte happens to represent the start of a multi-byte instruction, the listing will become drunk and disorderly. Fortunately, it seems that many virus writers are incompetent - after setting up these obstacles, the author of this virus has simply inserted single-byte instructions into the gaps: the listing consequently remains quite undisturbed. One wonders why he bothered in the first place.

Monitor Subversion

Predictably, virus writers are responding increasingly to a barrage of unreasoned claims and provocation emanating from the producers of memory-resident anti-virus software (see *The Playground Approach to Virus Detection*, *VB*, March 1991). The virus writer does this by delving ever deeper into the gizzards of DOS to undo these manufacturers' flawed attempts to spot 'suspicious activity'. The use of undocumented calls for finding original system vectors, interrupt stripping and so on, require considerable research on the part of the virus writer, which leaves us wondering why these authors should expend so much effort for no personal gain.

PC-Flu II searches through the DOS system area for a specific segment of code (the superseded DOS 1.0 INT 27H Terminate-and-Stay-Resident handler) that leads it to find the original DOS Services interrupt 21H. After patching in some code to find the correct segment split, it links its own handlers in front of the internal ones, short-circuiting any routines that may have been there before.

As well as cutting out any virus monitors, this will neutralise any other resident programs providing additional DOS services, such as networking software.

Curiously, the writer has employed the same trick with the stack for this interrupt patch. Rather woolly thinking is this: if there is something between the virus and the INT 21H code, the modified return address refers to some code outside the writer's ken, which could produce some bizarre errors if not system failures.

Resident Infection Routines

The actual infection routines follow a standard pattern; many viruses these days become resident in memory, infect both COM and EXE files, override the DOS read-only attribute, preserve datestamps, and disable DOS critical-error messages.

An odd method of going memory-resident is used: the virus assumes control of part of the area of memory into which its host loaded, and runs the host itself as a child process, leaving DOS to allocate the relevant memory. When the child exits, the virus stays in memory through standard DOS calls.

The intercepting virus routines put up a constant fight to prevent anything else gaining control of the DOS interrupt, continually re-setting the vectors to point to itself. Little else will be able to wrestle back control (including many viruses) unless it too uses a similar tactic to find the original vector.

PC-Flu II infects on open as well as execute but treats the two file types differently. The initial 2112 bytes of a COM host are moved to the end of the file, and the virus goes in front; an EXE file is rounded to a 16-byte boundary and the code appended, in each case after the virus has formed a new encryption of itself. Repeated infection is avoided by checking for a message in the virus, requiring the infecting copy to decrypt fully what it sees.

Encryption

The encryption routine is trivial, comprising an eXclusive-OR and subtraction operation on each byte in the remaining code but the routine which performs this is different in each copy. The mutation engine employs a 'pick and mix' approach - each function in the decryptor can be coded in numerous different ways and a table in the virus contains five possibilities for each of six different operations, giving over 15,000 possible permutations in a 30-byte section of code. No simple search pattern is possible.

The final part of the interception routine reaffirms the stealth-like nature of this beast. While DOS functions are trapped, a request to read 64K of data from the start of an infected COM file will return an image of the uninfected file. A scanner or other tool which tries to optimise its operation by reading in as much as possible (viz. 64K) might just be caught in this way, but this limited capability is of rather questionable effectiveness.

Summary

This is an interesting virus with a few novel quirks which reveal the apparent lack of circumspection of the author. It is possible that the ideas came from another source and were not fully understood. They certainly aren't exploited to their full potential. Other clues give the impression that he is not even particularly proficient in the use of 8086 assembly language.

Detection Methodology

As the decryption code is variable no simple search string is possible for this virus, but some partial patterns can be extracted. The following discussion (which will be of no use to people other than programmers involved with virus scanner development) may provide the more general reader with an amusing insight into the convoluted and tedious processes necessary to detect viruses which employ self-modifying encryption. No wonder virus analysts groan when they read that now infamous *VB* description 'no search pattern is possible'!

The virus will commence at offset zero in COM files, and at the calculated start of execution of EXE files. The following hex wildcard sequence is constant:

```
50B? 2001 ???? B??? ???? ??B? 2008 ????
```

Certain other bits are constant, and many are interrelated e.g. byte 1 (counting from zero) can be BB or BA. If BB then bytes 4 and 5 are both 90, else any of the set {(8B,DA), (52,5B), (87,DA), (87,D3)}.

Byte 6 is B8 or BA; bytes 7 and 8 are the decryption key and can be anything; as above, bytes 9 and 10 can be {(90,90), (8B,C2), (52,58), (92,90), (90,92)}.

Byte 11 is B9 or BA; bytes 12 and 13 are a constant 20,08; bytes 14 and 15 are chosen from {(90,90), (8B,CA), (52,59), (87,CA), 87,D1)}.

Believe it or not, such patterns can be incorporated into an intelligent scanner as a heuristic for identifying the virus in a file. Reliable detection routines are currently being incorporated in a number of commercial and shareware scanners.

Automated disinfection of infected files is possible, but the easiest and most sensible approach is simply to replace them with clean master copies.

VIRUS ANALYSIS 2

Fridrik Skulason

Michelangelo - Graffiti Not Art

A number of reports of this virus spreading in the UK have been received in recent weeks, which have prompted the following analysis.

The Michelangelo virus resembles the New Zealand (Stoned) virus in several ways. It is more than a simple modification of New Zealand - large parts of the virus have been rewritten - but the overall structure and various bits of code are identical, so the virus might best be classified as belonging to the New Zealand family. It is obvious that the author has examined New Zealand and has attempted to correct the most serious problem associated with the original virus, i.e. its inability to infect diskettes larger than 360 Kbytes 'correctly'.

The origin of the virus is not certain, but it appeared first in Australia, and has now spread to Europe, being particularly prevalent in the UK and Scandinavia.

Operation

When the computer is booted from infected media, the virus gains control. It creates a 2K 'hole' in memory, by decreasing the number at 40H:13H, and copies itself to that area. After hooking into INT 13H, the virus checks whether it entered the system as the result of a boot from an infected floppy.

If so, the virus reads the Master Boot Sector, and checks whether it is infected. Just like the New Zealand virus it does this by comparing the first 4 bytes of the Master Boot Sector to the first bytes of itself, and attempts to infect the Master Boot Sector if it finds a mismatch.

Master Boot Sector Infection

The virus stores the original Master Boot Sector at Track 0, Head 0, Sector 7. The Partition Table itself (the last 66 bytes of the Master Boot Sector) is copied to the end of the virus, which is then written to Track 0, Head 0, Sector 1. After infecting the hard disk, the virus simply transfers control to the original Master Boot Sector.

Activation

If the computer is booted from the hard disk, or if the Master Boot Sector is already infected, the virus checks the current date, assuming the machine is equipped with a real-time clock. If the current date is the 6th of March, the virus will systematically proceed to destroy all data on the infected disk.

[*It was a researcher (not the virus writer) who named the virus after Michelangelo Buonarroti, the Italian Renaissance artist, on the grounds that Michelangelo was born on the 6th March 1475. The connection between the virus' trigger date and the anniversary of the birth of the artist is tenuous in the extreme - it is almost certain that the virus writer had a different reason for selecting 6th March as a trigger date.*]

Destruction

The virus first destroys any information on Track 0, then Track 1 and so on. On a 360K diskette, it will destroy sectors 1-9, heads 0 and 1, but on other types of diskettes it will destroy the first 14 sectors on each track.

On machines with an infected hard disk the destruction will be more severe as the virus may trash the entire disk, forcing the user to reformat it and restore everything from backups. On a hard disk the virus will destroy the first 17 sectors on every track, heads 0, 1, 2 and 3.

Destruction is accomplished not by formatting, but by overwriting with whatever is stored at memory location 5000H:0000H. This will probably be a block of zero bytes.

INT 13H Servicing Routine

The virus will only interfere with INT 13H operations if the user is accessing drive A and the drive motor is not already running. The original boot sector is then read into memory, and checked for infection, in the same way as the Master Boot Sector. If it is not infected, the virus attempts to infect it. The media descriptor byte (offset 15H) is checked to see whether it contains 0FDH, which indicates a 360K diskette. If so, the boot sector is stored at Track 0, Head 1, Sector 3 - the last sector of the root directory.

The major difference between Michelangelo and New Zealand has to do with high density diskettes. If the media byte does not contain 0FDH, the virus will write the original boot sector to Track 0, Head 1, Sector 14 - tactfully avoiding the problems associated with the New Zealand virus.

Detection

The following pattern will be found in the Master Boot Sector of an infected hard disk and the boot sectors of all densities of infected diskette.

```
BE00 7C33 FFFC F3A4 2EFF 2E03 7C33 C08E
```

Disinfection

Disinfection of the Michelangelo virus is relatively straightforward. The virus can be removed from hard disks even when it is active, but disinfection of diskettes requires a 'clean' machine. The hard disk may (under DOS 5) be cleaned

with the FDISK /MBR command. Alternatively, or under previous DOS releases, it can be restored manually with a disk editor by moving the Master Boot Sector from Track 0, Head 0, Sector 7 to its original position (Track 0, Head 0, Sector 1).

To disinfect a floppy disk it is necessary to determine first the location of the original boot sector. This can be done by examining byte 8 within the virus body, which will contain either 3 or 14, giving the sector number in which the boot sector is located (on Track 0, Head 1).

An alternative and more practical method of disinfecting diskettes is to transfer data and programs using the DOS COPY command. This must be done in a clean DOS environment. Once all items have been copied, the diskette should be formatted using DOS FORMAT. Do not use DISKCOPY as this is an image copier and and will transfer the exact contents of the disk including the virus code in logical sector 0.

VIRUS ANALYSES 3 & 4

Jim Bates

The Haifa Virus - How Low Can You Go?

I am occasionally asked whether I get bored with disassembling and analysing virus after virus. I must admit that there are occasions when I find the process tiresome but usually I find that as the inner workings of each virus are revealed, my disgust at the irresponsibility, nastiness and incompetence of the programmer begins to pump the adrenalin and I become totally absorbed in the work in an effort to negate the chancre that the code represents.

Most emotion is generated when examining the machinations of the virus code itself but there are occasions when the air in my office turns blue with exasperation as the nature of the trigger routines are revealed. This has never been more true than with recent work on the Haifa virus! This virus has been reported at large and the code is the first that I have come across which targets computer programmers within its trigger routines. This is done by *specifically* corrupting ASM and PAS files with infantile pieces of text.

Installation

The code infects by appending itself to COM and EXE files in the familiar way adopted by most viruses. When first executed, the virus immediately searches the machine environment data area for details of the COMSPEC setting. This is used on all MS-DOS machines to indicate the name and location of the main command interpreter program and will normally point to COMMAND.COM in the root directory of the boot drive. Once the virus has located this file, it is checked for existing infection and infected if found to be 'clean'. The virus code is then moved up into the highest available block of memory and finally hooks an intercept routine into the DOS Services interrupt vector at INT 21H. The pointer to the top of available memory is not modified so there will be occasions when DOS overwrites the virus code and causes unpredictable system failure. Once installed in memory, the virus passes execution control to the host program and execution continues normally.

During the installation routine, the code completes various checks, the results of which will change its subsequent operation. Firstly, the ubiquitous 'Are you there?' call is issued to ensure that this virus is not already resident. In this case the call consists of placing a value of 0D2H into the AH register and issuing an INT 21H request. If AH is returned unchanged then the virus is assumed not to be resident and installation continues.

A subsequent check is more unusual as the virus looks for a volume label of 'AT286' on the C: drive. If this label is found, the command interpreter is not infected although the code is still installed into memory and made active. Also during installation, a check is made on the system date and time. If the hundredths setting of the system seconds indicator happens to be 0, the machine will simply hang (a 100 to 1 chance). If the system date is 24th August or 8th April (any year), the PC displays a message and then hangs:

```
HAIFA VIRUS V1.12
WRITTEN BY Y.S
GUEST STARS T.S. & I.F.
MADE IN ISRAEL
I AM TIRED. PLEASE WAKE ME UP ON TUE 12.4.3456
PRESS RESET TO CONTINUE...
```

(If only we could be sure that he really would sleep that long!)

Interception

The intercept handler routine contains two sections: a small routine handles the 'Are you there?' call by checking for the 0D2H value and, if it is found, incrementing it before returning to the caller. The 0D2H value will cause malfunction of certain networks which use a similar call, notably *Novell NetWare* and *Banyan Vines*. The main interception routine looks only for a 4EH function request (which is the DOS FINDFIRST request) and all other functions are allowed to continue unchanged.

The intercept routine first completes some complex calculations involving checking the target filename but then negates them by issuing its own request to find the first available file (of any type) in the current directory. Once a suitable file is found, its attributes are checked and the file is rejected as unsuitable if the SYSTEM attribute is set.

Targeting

On suitable files, processing then continues by examining the file extension and here my blood pressure began to rise! The code searches specifically for files with ASM, PAS, TXT and DOC extensions (as well as COM and EXE). If found, each of these extensions results in highly specific corruption being introduced.

ASM files have the first 76 bytes overwritten with a puerile assembler routine which is designed (when assembled) to overwrite the first 16 sectors of Track 0, Head 0 of the first hard drive with garbage. This left me speechless at the sheer mindless stupidity of the contemptible individual(s) that could conceive of such a thing. Fortunately, such corruption would rapidly be detected by even trainee programmers during the course of program development.

"Intellectual freedom and individual rights are all very well, but who has to clear up the resulting mess?"

PAS files have the first 23 bytes overwritten with the text:

```
CONST VIRUS= "HAIFA";
```

Quite what this is supposed to achieve is not known since no further reference to it is made within the virus code.

TXT and DOC files are corrupted in a slightly different way by having text inserted at the approximate half-way point. The text inserted gives one tiny insight into the intelligence of the virus author(s):

```
OOPS! Hope I didn't ruin anything!!!
Well, nobody reads those stupied DOCS anyway!
```

Quite apart from the sentiment expressed (indicating at least one reason for his gratuitously malicious behaviour), the mis-spelling of 'stupid' somehow typifies the virus writing mentality - ignorant of everything except their own overriding obsession.

Apart from these deliberate corruption routines, the virus code infects both COM and EXE files with active copies of itself. Selection for infection is done using a primitive 'sparse' method by checking certain combinations of bits within the file time field. Infection recognition within a file is then accomplished by checking for similar combinations. My calculations indicate that around 70% of files will be infected.

At each interception the virus will attempt to infect up to four files within the current directory.

Encryption

Although this virus has no stealth capability, it does contain a self-modifying encryption routine such that each infection appears differently on disk.

The method used is so similar to that first introduced by Mark Washburn in his V2P6 virus that I cannot believe that it was developed independently. I have said before that in my opinion Mr. Washburn's efforts have added nothing to the anti-virus armoury but it is now becoming increasingly obvious that his code has provided valuable assistance to virus writers around the world. In the light of these developments, perhaps he can be persuaded to move out of computing altogether!

The code encryption does mean that no simple search pattern can be given to recognise this virus on disk. Randomisation techniques also make it difficult to give exact details of the length of the encrypted code. However, practical tests indicate an infective length of between 2370 and 2385 bytes.

Because the code lacks any stealth routines, any increase in file size is immediately obvious in a directory listing. During tests, no internal corruption of program files was noticed so specific cure routines may be capable of file recovery although replacement by clean copies is still the best way to remove this virus from infected files.

Conclusions

A continuing (if spurious) defence of some people who write virus code is that they do it for 'research'. This is a pretty weak argument but even if it is accepted, there can be absolutely no justification for the destructive trigger routines which are appearing ever more regularly in viruses.

The apparent adoption, here, of Mr. Washburn's variable decryption techniques confirms the self-evident dangers of developing and releasing 'research' viruses. One can only stand aghast at the likely repercussions of a virus-writing contest which has been reported as taking place recently in the United States! Intellectual freedom and individual rights are all very well, but who has to clear up the resulting mess? Rarely, if ever, it seems, do these 'researchers' engage themselves in the practicalities of actually *helping* stricken computer users.

The various routines contained within this code reach a new low in the intent of virus authors. If the trigger message is to be believed, an irresponsible group of computer 'enthusiasts' with the initials of Y.S., T.S. and I.F. at large somewhere in Israel are responsible for this garbage. Perhaps a reader of the *Virus Bulletin* in Israel knows of such a group and could identify the individuals to the relevant authorities.

The Einstein Virus - A Total Misnomer!

Most viruses display poor coding techniques. The Einstein virus, recently reported at large within a UK university, is no exception - but one of the mistakes within the code does help to destroy the abiding myth of 'benign' viruses and highlights a lesser known technical problem which has far-reaching implications for certain types of virus scanner.

The major part of the virus code is unremarkable and primitive. The virus becomes resident in memory when it is first executed and remains there until the machine is rebooted. However only the DOS service routines are intercepted, so a warm reboot (via Ctrl-Alt-Del) is sufficient to remove it.

'Are You There?'

During initial execution the code hooks into the INT 21H routines and specifically checks for requests to LOAD and EXECUTE. A small subsidiary routine provides the virus self-recognition capability which is a familiar feature of many resident virus routines. In this instance the 'Are you there?' call consists of placing a value of 0F0H into the AH register and then issuing an INT 21H function request. If this virus is resident and active, a value of 4EH is returned in the AH register and the flags remain unchanged.

This call may be used as a method to detect whether the virus is resident in memory but it should be noted that other programs use it too - notably *Novell Advanced Netware* version 1.00 (Connection ID Request) and the Menu Utility section of *Double-DOS*.

Once the virus has been installed and made active within the allocated memory block, the original host program is executed as a child process until it terminates, whereupon the termination code is collected and passed to DOS leaving the virus resident in memory.

Dummy Error Handler

The main interception 'hook' within the virus code examines all LOAD and EXECUTE requests (function 4B00H) but only interferes with files having an EXE extension. The intercept code first installs a dummy error-handling routine in place of the existing critical error handler at INT 24H (using the normal DOS GET Vector and SET Vector requests) and then collects and stores the attribute details of the target file.

File Infection

The current attributes are cleared and the filename is checked for an EXE extension. If the target file is an EXE file then the contents of its Date and Time fields are collected and stored and the file is opened for Read/Write access. The file is then checked for infection by examining 8 bytes located at a position 104 (68H) bytes before the actual end of the file. If the file is infected, these 8 bytes will contain the word 'Einstein' and the virus code will simply close the file, repair the attributes and allow the original request to continue, thus avoiding re-infection.

Multiple File Infection

This particular virus writer made a serious mistake in the design of the infection routine, which can result in the virus being unable to recognise its own existence within certain types of files and therefore cause multiple re-infections and irreparable damage to the target code.

On suitable target files which the virus 'thinks' are uninfected, the infection routine begins by reading the first 27 (1BH) bytes of the file into a buffer. Certain values of the header in the target program file are then modified to ensure the execution and survival of the virus code when the program is run. Finally, the virus code (which also contains the buffer with the original header information) is written to the end of the file and this is where the mistake occurs.

When attempting to detect the existence of the virus in a file, the search point is based upon the actual *end* of the file as it resides on the disk. However, when determining where to place the virus code during infection, the calculation is based on the end of the program load *image size*. Since the file size and the program load image are not necessarily the same (particularly with Windows 3 program files), the virus may well place its code *within* the target program file instead of appending it to the end.

A Simple Example

Perhaps a simple example will serve to illustrate this - consider an EXE file which is actually 200,000 bytes long but contains a program load image of 100,000 bytes.

The parameters contained within the primary file header include two size fields and these refer to the size (in 512 byte pages) of the initial load image and the number of bytes within the last page. Thus our example file would have 0C3H pages (= 99,840 bytes) with a further 0A0H (= 160 bytes) in the remainder field. The virus uses these values to calculate where it will place its code and then updates them to reflect the (notional) increase in file length.

The actual length of the virus code is 36EH (878 bytes) so our example file, in its infected condition will now have 0C5H pages (= 100,864 bytes) and a remainder of 0EH (= 14 bytes) marked in the header fields. The virus code will have over-written the original contents of the file between 100,001 and 100,878 bytes which in this case is within the body of the file.

The next time that this program file is run, the initial infection check routine within the virus code will not find the 'Einstein' marker and will therefore re-infect the file. However, this time the virus code will be inserted at offset 100,889, overwriting a further 878 bytes of the host file. This re-infection process

will continue each time the program is run until the multiple copies of the virus code spread beyond the actual end of the file (having destroyed all of the second 100,000 bytes in our example). On an actual sample (from an affected user) file, six copies of the virus code were found 'chained' in this way within the original host file.

It should be noted that the above description is slightly simplified and due to rounding corrections, the actual infective length of this virus will vary between 878 and 893 bytes. It is also obvious that the actual observed length added to a file could be anything from 1 byte to 893 bytes when multiple infections have taken place.

'Benign' Virus Myths

There are two distinct observations that should be made concerning the performance of this virus.

First, the virus does not contain any *deliberate* attempts to corrupt code or data and might therefore be classified by some researchers as 'benign'. This is a term which I find particularly annoying since its implication that there are 'harmless' viruses provides a spurious argument to those within our industry who will seize any opportunity to try to justify the development of virus code.

Let there be no doubt, computer viruses at the very least are stealing valuable processing resources and trespassing within systems without the authority of the owners and users. The Einstein virus is a classic illustration of how a supposedly 'benign' virus can totally destroy the user's code.

'Top and Tail' Scanners

The second point is somewhat more technical and concerns the methods used by some anti-virus scanning programs.

In these days of rapidly increasing virus numbers, two major areas of concern with scanners are available memory and the overall speed of operation. One simple way of avoiding the need to scan a whole file is to 'top and tail' it and thereby scan only those areas in which viruses are known to reside. This is done by selecting a maximum buffer size (say 15,000 bytes) and scanning files greater than this size by selecting only the first and last 15,000 bytes for known virus code.

Hitherto, this has been an effective tactic to increase efficiency but the possiblity that this virus could be *anywhere* within a target file effectively nullifies the method. In this instance, the effect is accidental rather than intentional. However, the appearance of the Brainy virus (*VB*, December 1991, Technical Notes, p.2) which introduces itself within a file as part of its infection strategy confirmed the fact that 'top and tail' scanners are easily circumvented. The risks posed by multiple file infection, the infection of overlay files and viruses which insert themselves into target files lend weight to secure scanners which conduct byte-by-byte file analysis.

Several scanners currently on the open market will need substantial changes in order to be 100% sure of detecting Einstein wherever it may reside.

Similarly, many 'disinfection' programs will be unable to confirm an effective removal of the virus code since the true size of the original file may be indeterminate.

Detection

Since this virus neither encrypts its own code nor attempts to 'hide' from scanners, detection is fairly simple.

It would be perfectly feasible to search program files for the word 'Einstein' but this could obviously produce false positive indications from legitimate files containing this word. A variant might also be produced which simply altered this text and thereby neutralised such a detection method.

A reliable search string from the body of the infection code is:

```
0042 CD21 7231 B96E 0333 D2B4 40CD 2172 193B
C175 15B8 0042
```

this will be found at offset 2C5H into the virus code. Using this string in conjunction with the word 'Einstein' at offset 306H in the virus code should greatly reduce the risk of false positive identifications. It is also worth mentioning that the 'MZ' header of an infected file will contain certain constant values in specific fields as follows:

Stack Pointer = 036EH
CheckSum = 1984H
Instruction Pointer = 0049H

These values remain the same on all infected files (even those with multiple infections) and may also serve to reduce the risk of false identification.

Removal

Because of the uncertainty concerning the internal positioning of the virus code, I would not recommend any attempt at specific disinfection of files infected by this virus.

The problem is that on files where the load image size does not match the physical size, the balance of the file may contain resource code/data, overlay code or just plain data and there will be no easy way of determining the extent of any damage.

An alternative method of disinfection might involve a pre-emptive check where accurate details of the file's appearance *before* infection have been collected and stored for generic disinfection purposes. Even then such methods may fail but at least the disinfection program will report the failure and the user can fall back to the trusted method of simply deleting the infected files and replacing them with known clean master software copies.

COMPARATIVE REVIEW

Scanners - The Acid Test

In *VB* comparative reviews, the scanners are usually tested against a large battery of infections which make up the *Virus Bulletin Test Set* (for details refer to *VB*, September 1991, p.18). This procedure has been criticised by some as an unrealistic test due to the inclusion of a range of so-called 'lab' viruses thought to be of academic interest only.

A different stance has therefore been adopted to start the new year: this month's test aims to determine the efficacy of a range of commercial and shareware scanners at detecting viruses known to be at large. From a risk assessment point of view, we believe this test is the most realistic so far conducted. It provides an insight into the varying degrees of protection available against *real* threats facing *real* PC users.

The latest releases of the most prominent scanners were tested against 34 parasitic viruses and 13 boot sector viruses positively identified as being in the wild. Where a parasitic virus infects both COM and EXE type files, an infection of *each* was generated and the scanners had to find **both** infections to pass. Similarly, multiple infections were generated of the encrypting viruses (Flip, Spanish Telecoms 1 and 2, Tequila and Whale) and the scanners had to find **all** progeny to pass.

Reports from a number of sources were used including *McAfee Associates*, *IBM*, *Leprechaun Software*, *Fridrik Skulason*, *Bates Associates* and *Sophos Ltd* to determine a realistic 'at large' test set.

It came as quite a surprise to learn that the Whale virus is now at large in Australia (it replicates on 8088-based PCs) and for this reason the virus is included in the test set. The DIR II virus is also included in the test-set; it has been detected at two sites in the UK and reliable observers inform *VB* that it is rife in many eastern European countries.

The Maltese Amoeba virus, which triggered on November 1st 1991 (see *VB*, December 1991, pp. 13-16) is the most recent sample in the test-set. Obviously, releases of any product prior to this date will not detect the virus. It is included in the test set because it is highly destructive, due to trigger again on March 15th 1992 (i.e. in approximately two months' time) and is already remarkably widespread.

Vendor	Product Name	Version	Parasitics	Boot Sectors	Total	Detection Percentage
Bates Associates	Viscan	3.29	34	13	47	100.00%
Central Point Software	Central Point Anti-Virus	1.00	29	13	42	89.36%
ESaSS	TBScan	2.80	29	13	42	89.36%
Frisk	F-Prot	2.01	33	13	46	97.87%
Harry Thijsen	HTScan	1.16	31	13	44	93.61%
IBM	Virscan	2.1.5	29	13	42	89.36%
Leprechaun Software	Virus Buster	3.75	29	13	42	89.36%
McAfee Associates	Scan	85	34	13	47	100.00%
Microcom	Virex-PC	1.8	32	13	45	95.74%
PC Enhancements	PC-Eye	2.1k	30	13	43	91.49%
RG Software	Vi-Spy	8.0.B194	34	13	43	100.00%
S&S International	Dr Solomon's A/V Toolkit	5.52a	34	13	47	100.00%
Sophos	Sweep	2.32	34	13	47	100.00%
Symantec	Norton Anti-Virus	1.5	Failed to complete test	13	N/A	N/A
Xtree Company (EliaShim)	AllSafe / ViruSafe	4.54	28	13	41	87.23%

All the viruses were stored on floppy disks and were scanned by each product in turn. The scanners were installed on the computer's hard drive to simulate the so called 'sheep dip' technique of scanning incoming diskettes.

The results are extremely revealing. All the scanners found all 13 boot sector viruses, but only five found all 34 parasitic viruses. Accolades go to *Bates Associates*, *McAfee Associates*, *RG Software Inc*., *S&S International* and *Sophos Ltd*, all of whom detected all the viruses in the test.

Symantec's Norton Anti-Virus did not complete the parasitic test - it failed to read any of the test diskettes. *Xtree's AllSafe* - which feature as a standalone product review this month (see pages 19-22), is the poorest performer among the scanners tested. It failed to find Flip, Maltese Amoeba, Spanish Telecom 2, Spanz, Tequila, and, Whale. For a 'new' entrant onto the market, this is an inauspicious start.

Fortunately for the Australians, *Virus Buster* had no problems with Whale. However, *Virus Buster* did trip up on Dir-II, Flip, Maltese Amoeba, Spanish Telecom 2, and Old Yankee 1.

Central Point Anti-Virus failed to detect Dir-II, Maltese Amoeba, Spanish Telecom 2 and Spanz. *Virex-PC* failed on Maltese Amoeba and on one of the ten Whale infections. Considering this product hasn't been updated since September 1991 (and therefore stood no chance of detecting Maltese Amoeba) this is a creditable performance.

PC Enhancements has always had a problem with the encrypting viruses and its *PC-Eye* still fails to detect Flip, Maltese Amoeba, Spanish Telecom 2 and Tequila. *IBM's Virscan* did not fare at all well, missing infections of Dir-II, Maltese Amoeba, PcVrsDs, Spanish Telecom 2 and Spanz. The shareware packages seemed to fare a little better than some of the commercial ones. Of particular note is *F-PROT* which only missed Maltese Amoeba. The *HTScan* product from Holland detected all but Dir-II, Maltese Amoeba and Spanish Telecom 2 while its compatriot *TBScan* in addition to these 'false negatives' also missed Spanz and Liberty 1.

'IN THE WILD' TEST SET

Parasitic Viruses:

1575, 4K, 777, Cascade (1701), Cascade (1704), Dark Avenger, Dark Avenger 2100, Dir-II, Eddie, Eddie-2, Flip, Hallochen, Jerusalem - Friday 13th, Keypress, Liberty 1, Maltese Amoeba, Nomenklatura, Nothing, PcVrsDs, Plastique, Plastique 5.21, Slow, Spanish Telecom (1), Spanish Telecom (2), Spanz, Syslock, Tequila, Vacsina, Vienna (2A), Vienna (2B), Virdem-Generic, Whale, Old Yankee 1, Old Yankee 2.

Boot Sector Viruses:

Aircop, Disk Killer, Form, Italian, Joshi, Joshi 1, Michelangelo, Music Bug, New Zealand 2, Print Screen, Spanish Trojan, Tequila, Yale.

PRODUCT REVIEW

Mark Hamilton

Xtree's AllSafe (ViruSafe)

Two years ago, the anti-virus software market was satisfied by small, dedicated, research-based companies. There were no rich pickings, so the majors stayed away. However, during 1990 the large utility software companies discovered that there was money, after all, to be gained from the worsening virus threat. First to seize the initiative was the *Symantec Corporation* with its *Norton Anti-Virus* (see *VB*, October 1990, p.2). Now they're all at it, fighting for market share like vultures picking at carrion.

Last month, two more such companies announced their 'unique' solutions and neither, in common with *Central Point* but unlike *Symantec*, has developed this product itself and neither has any particular expertise of its own in the virus field to offer. The two companies concerned are *Xtree* and *Fifth Generation Systems*. Actually, *Xtree* launched *two* products but whether you get the choice between them depends on where you happen to reside.

Background

Xtree has contracted with *EliaShim Microcomputers & Software* from Israel and distributes that company's *ViruSafe* in the US and Canada, but not elsewhere. According to *Xtree*, *EliaShim* wishes to market *ViruSafe* itself outside North America. But *EliaShim* has produced an enhanced version called *AllSafe* which *Xtree* is free to distribute worldwide, including the US and Canada.

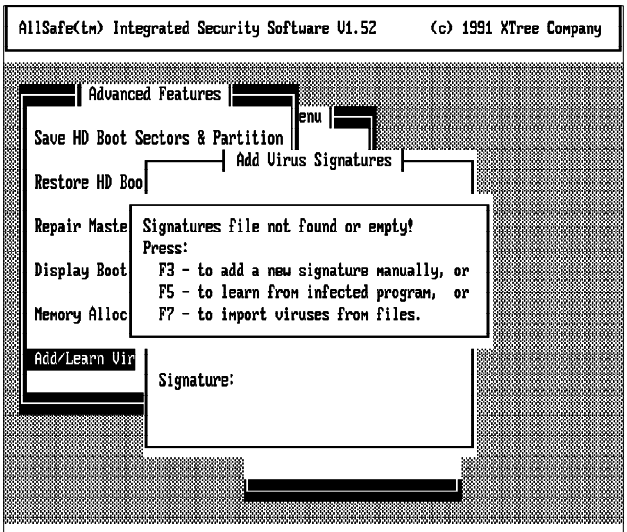
The packaging for *ViruSafe* proudly boasts the following:

> *'I think that ViruSafe is well thought out, has an excellent front-end menu system that makes initial operation quite painless, and is capable of detecting and removing a large range of viruses...I would recomend its use.'*
>
> *Virus Bulletin, April 1990*

Unfortunately, the reviewer, Dr Keith Jackson, didn't quite say that in his review (*Virus Bulletin*, April 1990). The exact wording he used was:

> *'I think that ViruSafe is well thought out, has an excellent front-end menu system that makes initial operation quite painless, and is capable of detecting and removing a large range of viruses. I would recommend its use but for the fact that it is copy-protected.'*

According to the editor, *VB* does not object to companies using extracts from its reviews to assist the promotion of



Xtree's Allsafe claims to 'learn' detection routines following infection by even unknown viruses - a false claim in the face of encrypting viruses.

products provided that the quotation is exact and not taken out of context. That said, the version submitted for this review was not copy-protected and the company has confirmed that neither of its products will be copy-protected.

Components

ViruSafe consists of the following principal elements:

- PCC.EXE This program is very similar to *Norton's SI* (*System Information*) program. It is a general purpose utility and has no specific anti-virus features. It says it can, however, fix hard disk boot sectors (both Master and DOS Boot Sectors) should these become corrupted.
- PIC.EXE This is a generic file checker which calculates checksums of a range of files. Like all integrity checkers, it compares checksums against those it has previously stored.
- VSMENU.EXE The menu system from which most actions can be performed. Interestingly, you can't run PCC from VSMENU which is slightly inconvenient.
- VC.EXE is a program that checks memory for the existence of a resident virus. This program claims to detect unknown memory-resident viruses.
- VS.EXE is a memory-resident all-purpose monitor which requires nearly 12K of memory (if the 'help' and virus-name reporting option is enabled, 9K otherwise). All but 1K of the footprint is eliminated if you have expanded memory.
- UNVIRUS is the main scanner and removal engine. Note that a wholly dissimilar program by the same name is part of *Dr Solomon's Anti-Virus Toolkit.*

AllSafe includes all the above-named elements, except that VSMENU is called ASMENU; it also includes:

- SAFER.PGM a device driver which checks passwords at boot time and consumes 9K of memory.
- XCRYPT.EXE is a file encryption/decryption program. If you buy *AllSafe* in the UK you should be aware that the US version uses the DES data encryption standard which is not the case in non-US versions. This is a nonsense because DES is widely documented and freely available in source code from a variety of sources including *CompuServe* and the UK's *CompuLink* (*CIX*).

Which One To Review?

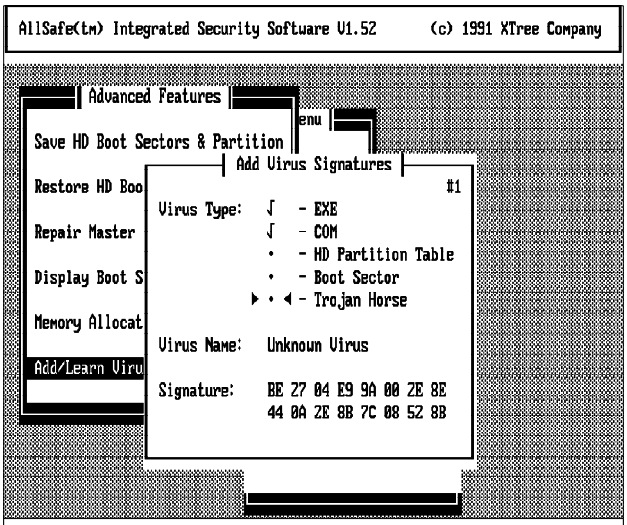
For review, I received the release version of *ViruSafe* and the final Beta version of *AllSafe*. I noticed that the file dates for *ViruSafe* are 17th September 1991 while *AllSafe's* are dated 6th November 1991. As far as anti-virus features are concerned, both products are identical; it is these aspects on which I will concentrate referring to both products simply as *ViruSafe*. Since there is a difference in file dates between the two products, I decided to use the later-dated version.

Automatic Identification?

One of *Xtree*/*EliaShim's* main claims for *ViruSafe* is that it can automatically identify unknown viruses. When scanning memory, it says it sets off trigger mechanisms to see whether a virus is lurking. If VC detects an unknown virus, it is supposed to save a copy of the viral code to a disk file called VIRUS.PGM - *UNVIRUS* does exactly the same while scanning the disk by invoking VC at the start of its run. You then add a recognition signature for the virus from within VSMENU/ASMENU. At least that is the theory.

To gain maximum protection, the authors recommend running VC after running each program - particularly if you execute a new program you've never run before. Unfortunately, in the light of recent developments adopted by the virus writers, *EliaShim's* philosophy is basically flawed. If the virus does not use self-modifying encryption techniques, VC does indeed detect its presence. VC did exactly what it claimed when memory was infected with Jerusalem.

However, not all viruses work in this way. VC cannot detect the so-called 'direct action' viruses - those which do not remain in memory - since there is no virus in memory when its host terminates. Of greater threat to users are viruses like Casper, SVC 6, Flip, V2P6, Maltese Amoeba and Haifa which use self-modifying encryption and randomise the bytes in the decryptor routine. These cannot be detected by a simple search pattern. If VC does detect a virus of this type, it generates its VIRUS.PGM file (it did this for all the above-named viruses, except, inexplicably, SVC 6) but that file will only ever represent one of the millions of possible variations of the virus. So the virus is always at least one infection ahead of the anti-virus software.



The first 16 bytes of unknown virus code to be executed by the CPU are selected as a pattern and displayed to the user for approval.

If you know that a particular file is infected (or if VIRUS.PGM has been created), then you can instruct VSMENU to 'learn' about that infection and the resulting search pattern is displayed in a dialog box. The pattern chosen by VSMENU is invariably the first 16 bytes that the CPU would process when the program is executed. That's fine and dandy for most older and less-sophisticated viruses but as already explained it's useless against the newer viruses that use variable encryption techniques where the chance of the same 16 bytes of static code appearing at all is minimal. Moreover, the likelihood of the first 16 bytes of the code executed being 'virus-specific' enough to eliminate the danger of false-positives is much reduced in comparison to good old-fashioned selection by disassembly.

Virus signatures such as those published in *VB* can be added. Many anti-virus programs identify encrypting viruses using identities which are hard-coded into the software. It's a pity that *ViruSafe's* authors haven't taken the same approach.

Scanner Accuracy

Of the 364 files in the standard Test Set (see *VB*, September 1991, page 18), *UNVIRUS* found viruses within 324 files, the remaining 40 infected files were passed as clean.

Xtree states that it gives priority to detecting those viruses known to be at large. This approach is fraught with problems particularly when marketing products of this type worldwide. What is at large in Europe may never have been heard of in the US or Asia etc. If a virus is known to exist it should surely be incumbent upon vendors of anti-virus products to enable users at least to identify it. *Xtree's* performance in tests restricted to viruses at large is documented on pages 18-19.

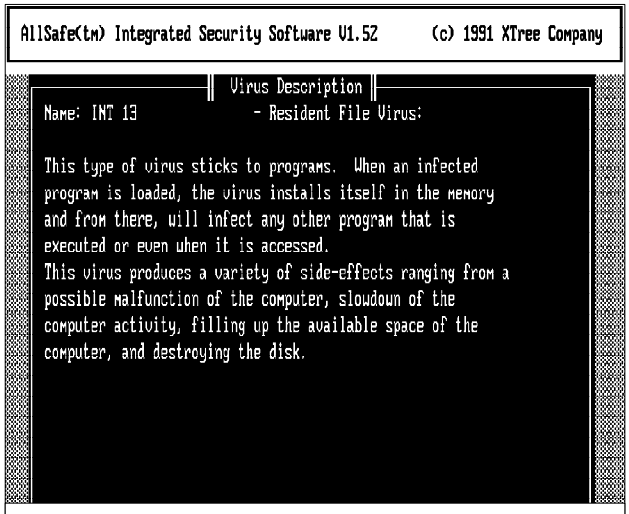
Interestingly, *UNVIRUS* reported that it had detected 431 viruses in those 324 files. Since the test set contains files infected with a single virus, I was alarmed by this because this means that *UNVIRUS* cannot distinguish between two or more similar viruses: the efficacy of its file repair capabilities must therefore be called into question. For example, it said that one of the Jerusalem variant-infected files contained Cascade as well as two Jerusalem variants. I don't subscribe to the 'repair' philosophy myself: it's much safer to delete any offending programs and restore from known clean backups.

[In unofficial testing, I used a larger, more up-to-date collection of 746 COM and EXE infections, *UNVIRUS* detected viruses in 572 files. As a comparison, *S&S's Findvirus* and *Sophos' Sweep* both detected 740 infections. Note that these results are **not** shown in the accompanying results tables.]

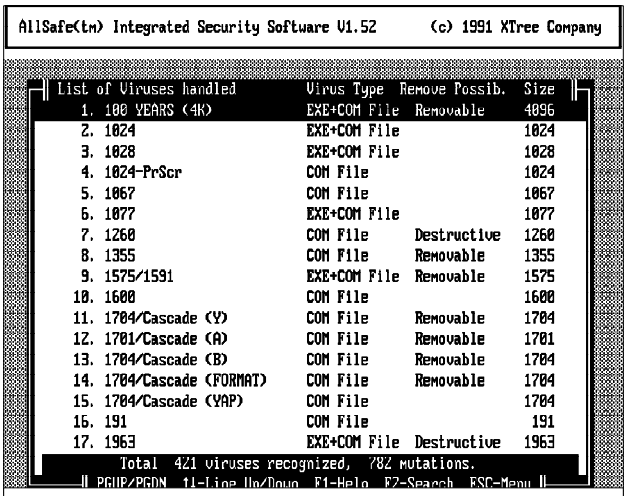
Execution speed is not too bad when checking just program files (COM, EXE, OVL and XTP - *Xtree* overlay files) but when checking all files, it was decidedly sluggish. I had to run this particular test three times because the first time I ran it, *UNVIRUS* crashed and failed to complete the check.

Virus Information

VSMENU can also display information about the viruses known to the package. Except in one or two cases, where it did display specific information on a particular virus, *UNVIRUS* displays information of such a general nature as to be misleading. For example, exactly the same text is displayed for Micro 128 (which merely replicates) and LoveChild which overwrites part of the hard disk when it triggers.



A database is provided containing information for each virus known to *AllSafe*. The information is extremely vague, the above description for the sophisticated INT13 virus applies to any number of other specimens.



Viruses known to *AllSafe* are listed along with target files and infective lengths. Disinfection routines, where available, are highlighted.

The generic checking program, *PIC*, on the other hand was found to be very quick in operation and performed exactly as it should, it detected one bit changes in files without any problem. I can not comment on the strength of the checksum algorithm which is proprietary (it should be irreversible).

Strategy

Xtree does not include updates in its selling price. If you buy a copy, you are entitled to one free update, but if you require regular updates these must be applied for and will cost between £10 and £20. Updates are only issued quarterly.

According to *Xtree*, 'It is important to note that because *AllSafe* prevents new viruses from entering the users machines in the first place, *Xtree's AllSafe* users will be most likely be less dependent on updates than users of other anti-virus products.' *Xtree's* sole presence in Europe is a product manager based in Paris, it has no sales office nor telephone support to offer in the UK. It provides support electronically via its BBS on which it will reply to enquiries within 24 hours of posting. The company says it will respond to faxed problems within the same time. *Xtree* has applied for a *CompuServe* forum in in the 'PC Vendor D' area (PCVEND). It was conspicuous by its absence when I checked on December 20th 1991. *Xtree* blames *CompuServe* for the delay.

With this product, *Xtree* has made extravagent advertising claims which, in the light of testing, are not substantiated. *AllSafe* can not be described as 'the ultimate in PC protection' as its advertisement in last month's *Personal Computer World* states. To the rival vendors who feared that they should pack-up and close down their companies in the belief that, thanks to *Xtree*, 'It's all over for the evil virus!', I say, 'unpack your debuggers, dust-off your DOS manuals and get back to work'.

ALLSAFE

Version Reviewed	4.54
Scanning Speeds	
Test 1 Hard Disk - Turbo	4 mins 36 secs
Test 1 Hard Disk Secure	15 mins 48 secs
Test 2 Floppy Disk Turbo	10 secs
Test 2 Floppy Disk Secure	22 secs
Scanner Accuracy	
Parasitic Viruses - Turbo	324 out of 364
- Secure	324 out of 364
Boot Sector Viruses	8 out of 8
Accuracy Percentage	89.25%

Stamina Test - Encrypting Viruses	
Multiple Test: Flip	Fail
Multiple Test: Suomi	Fail
Multiple Test: Tequila	Fail
Multiple Test: Spanish Telecom 1	Fail
Multiple Test: Spanish Telecom 2	Fail
Multiple Test: Group II	Fail
Multiple Test: Group III	Fail

[1] The speed test is outlined in the test protocol described in *VB*, April 1991, pp. 6-7.

[2] The test-set is outlined in *VB*, September 1991, p. 18.

[3] This test to determine a scanner's ability to detect encrypted viruses was first conducted in *VB*, October 1991, pp. 7-11.

Technical Details

Test Conditions: The testing for this review was conducted on two PCs. The first, a Compaq Deskpro 386/16, running under DR-DOS 6 was used for the speed tests. There are 37 megabytes in 974 files of which 440 files are binary executables occupying 19 megabytes.

For the floppy read tests, the 360 Kbyte Setup Disk for Microsoft C version 5.01 was used. This contains a total of 12 files requiring 354,804 bytes, of which 4 (238,913 bytes) are executable. The virus identification testing was conducted on an Apricot 486/25 which houses the test libraries.

Products: *AllSafe* and *ViruSafe*

Developer: *EliaShim MicroComputers Inc*

Marketed by: *Xtree Company*

Telephone: (USA) ++1 805 541 0604

Fax: (USA) ++1 805 541 8053

BBS: (USA) ++1 805 546 9150

RE-EVALUATION

Yisrael Radai

File Protector - A Grave Injustice?

Virus Bulletin has not received permission to reproduce this article on CD from the author. Readers can obtain a paper copy of the original issue directly from *VB*.

END-NOTES & NEWS

A **Computer Virus Market Survey** undertaken by *Dataquest* and sponsored by *NCSA*, *Xtree*, *McAfee Associates*, *Certus International*, *Fifth Generation Systems*, *Central Point Software*, and *Symantec* has been published. The survey aims to outline the extent of the computer virus problem on the IBM PC compatible in the United States, provide empirical data as to virus prevalence and define recovery costs in dollars to afflicted parties. 602 end-user sites with more than 300 machines (standalone or networked) were used to collate statistics. Information from *Dataquest*, USA. Tel 408 437 8000, Fax 408 437 0292.

VB '92, ***The Second International Virus Bulletin Conference***, *Edinburgh Sheraton Hotel*, 2nd-3rd September 1992. Information from Petra Duffield, Tel 0235 531889.

Certus International has introduced ***NOVI* anti-virus** software: 'With *NOVI* on your PC, you'll never need to worry about a virus again!!', '*NOVI* will clean your current software...it cleans automatically while you work, without interrupting you. With *NOVI* from *Certus* you need no updates!! Because *NOVI* will detect or prevent and repair both known and unknown viruses, updates are normally unnecessary.' Etc., etc. Information from *Certus International*, USA, Tel 216 752 8181.

Fifth Generation Systems has launched its anti-virus contender '**Untouchable**'. Network and standalone versions are available. The package comprises a scanner, a TSR monitor, file and system integrity checking and specific disinfection routines. It retails for £110, with quarterly updates supplied for an unspecified extra charge. Contact Mike Tait, *Fifth Generation Systems*, UK, Tel 0494 442224.

S&S International has ported ***Dr. Solomon's Anti-Virus Toolkit* to run under OS/2** enabling OS/2 users to search for DOS viruses anywhere on the system - not just within the DOS box. Upgrades will be made available monthly or quarterly. The recommended retail price for a single user will be £149 with quarterly updates and £249 with monthly updates. Server versions start at £399. Tel 0442 877877.

On Disk Software of New York has announced **version 1.3 of Quarantine** - its network-specific anti-virus software. The company claims to have tested the release on more than 400 servers with a maximum 250,000 files in registry. Contact *On Disk Software*, USA, Tel 212 274 8854.

5th International Computer Virus Conference, New York, March 12-13 1992. *Data Processing Management Association, Financial Industries Chapter*, Box 894, New York, NY 10269, USA.

The second edition of **The Computer Virus Crisis** (ISBN 0-442-00649-7), written by Fites, Johnston and Kratz has been released. Recommended price is £22.00. *Van Nostrand Reinhold*, 115 Fifth Avenue, NY, NY 10003, USA. In the UK, Tel 071 865 0066.

Springer-Verlag has published **PC Viruses, Detection, Analysis and Cure** (ISBN 3-540-19691-9) written by Alan Solomon (1948-). Price is £24.95.

VIRUS BULLETIN

Subscription price for 1 year (12 issues) including first-class/airmail delivery:

UK £195, Europe £225, International £245 (US$395)

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