ISSN 0956-9979 APRIL 1996

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Anniversary scanners.** It has been one year since the last *VB* comparative review of *NetWare* anti-virus products. What has happened in those twelve months? Details from p.14.

• **Breaking old ground.** Klaus Brunnstein, activist and outspoken campaigner for his causes, has made his voice heard throughout the world. Find out more about the man on pp.6-7.

• **Much ado about nothing.** March saw the annual media hype over the Michelangelo virus: as usual, very few actual infections were reported. See the news report, p.3.

CONTENTS

EDITORIAL
Transformers: Files in Disguise 2

VIRUS PREVALENCE TABLE 3

NEWS
1. Michelangelo – A Damp Squib 3
2. *ON Technology* Acquires *Thompson* 3

IBM PC VIRUSES (UPDATE) 4

INSIGHT
Research and Other Hobbies 6

VIRUS ANALYSES
1. Zhengxi: Saucerful of Secrets 8
2. Nexiv_Der: Tracing the Vixen 11
3. What a Quandary! 13

COMPARATIVE REVIEW
Scanning *NetWare* 14

PRODUCT REVIEW
ViruSafe 21

END NOTES & NEWS 24

EDITORIAL

Transformers: Files in Disguise

As the months go by, *Virus Bulletin* receives more and more press releases from anti-virus companies about products which can check compressed files. The technique poses an interesting question: should this be the job of an anti-virus product?

First, however, some background. In today's electronic world, we see a baffling quantity of what can only be described as transformed files; those modified by programs which alter data in such a way that it can be later restored to its original state. This description covers compressors, archivers, scramblers, encryptors, encoders… the possibilities are endless.

"it is the job of an anti-virus product to find viruses, not to transform files"

An excellent example arrived in my mailbox recently: a file compressed with ZIP, then encrypted with PGP, then transformed into 7-bit ASCII with UUENCODE. The final twist came when I discovered that the sender's mailer had automatically translated the message into MIME, making a total of four transformations. Regrettably, the file within was not a dynamically-compressed executable – and dynamically compressed files must be scanned internally for different reasons [*see the discussion on the Cruncher virus at the end of the NetWare comparative on p.19 of this issue*].

Multiply-transformed files turn up ever more frequently. The custom of attaching files to email is one culprit: when Internet mail protocols were designed, it was assumed that only 7-bit data files would be sent – this is no longer enough, which is why UUENCODE is used. Another factor is the ever-increasing size of files – the *Excel 5.0* executable is over 4MB long, a quarter the size of the hard disk on my 1990 *Acorn Archimedes*. The increase in the amount of data computer users exchange has not been matched by a growth in floppy size (my *Archimedes* has a 3.5-inch 1.44MB drive virtually indistinguishable from that in my current desktop PC), meaning that we must compress more data than before. Internet transmission times can be reduced greatly by compressing files.

This brings me back to anti-virus software. In an organisation which receives floppy disks, electronic messages, and CD-ROMs full of compressed information, virus-checking them all can be very inconvenient. This situation led to developers creating the facility in anti-virus software enabling scanners to understand the most common compression format (ZIP): scanners could then check the files within a ZIP archive. It was a great idea – users love it because it saves time; scanner manufacturers love it because having such a feature in their product sets them apart from those without one.

Developers now also build scanners which can cope with other archives (e.g. ARJ, LHA). The current market for virus-checking systems attached to mail servers (such as *cc:Mail* and *Lotus Notes*) entails more effort still: the product needs to understand the 8- to 7-bit conversion systems, notably UUENCODE and MIME, so those are built in as well.

Network administrators, however, rely on this functionality, which by its very nature can never be complete. When presented with an RAR or an HA archive, virus-checking systems can do little but throw up their hands in disgust. If the archives are encrypted, things go up another level. It will be, one hopes, more than a few years before an anti-virus product can crack a PGP message without the help of the message's recipient and discover a virus inside.

However good an anti-virus product may be at looking inside transformed files, it can never do so completely. There are almost definitely more people out there developing file transforming programs than anti-virus products. It is the job of an anti-virus product to find viruses, not to transform files. We would be wise to remember the UNIX methodology: use small utility programs which combine to produce the desired effect. If your scanner cannot look inside transformed files, use shareware such as *SHEZ* (a DOS program for archive management) or *WinZip* to give the scanner access. If the files are encrypted, it should not be possible for third parties or third party applications to look inside.

The only cover-all solution is to scan files when they are restored to their original state – wait for the users to extract them; then scan them. This is the only way to be sure. The future will bring more forms of file transformation, and scanners should be scanning, not unpacking.

NEWS

Michelangelo – A Damp Squib

1996 has been no exception in the media circus which has taken place every February since 1992: *VB* was approached by several international radio and television stations, and by various publications, for information on the potential risk of 'catching' the Michelangelo virus. The media hype began four years ago when some self-styled experts forecast that hundreds of thousands of PCs would be affected by the virus: in the event, only a tiny percentage was damaged.

Every year, in the weeks preceding the virus' trigger date (6 March), speculation runs riot as to how many people will have their data wiped. Every year, only a small number of reports are received: this year, *VB* had no direct reports, but did receive some indirectly from vendors who contribute to the Prevalence Table.

Despite the press releases from certain vendors advertising free protection, and the international media coverage, users did not this year seem unduly concerned about the possibility of an infection – perhaps a sign of the times? As users become more aware of the risks inherent in any type of system, they also become less susceptible to scare-mongering from any quarter, and more cognizant of necessary precautions. It is to be hoped that this trend will continue ▮

ON Technology Acquires Thompson

Thompson Network Software, owned by Roger Thompson (formerly of *Leprechaun Software Pty*), has been merged with Cambridge, Massachusetts-based *ON Technology*, a provider of multi-platform communications and network management software. The merger, which was finalised on 23 February 1996, will see Thompson and his employees remaining in their Georgia (USA) offices, but working as part of *ON Technology*.

Thompson commented: 'The key factor is that this will allow us here to concentrate on product development. We won't have to do any of the marketing, sales, or documentation work – and that's a real bonus for a programmer!'

Before the takeover, *Thompson* had developed a new anti-virus software program, now called *Macro Virus Track 6.0*, to detect and remove the known *Word* macro viruses and Trojans before infection or corruption takes place. The product, since released by *ON Technology*, is said to provide full background protection by scanning every *Word* document in one of two ways; either on-access or automatically, without the user having to open each document individually.

For information on the product, contact *ON Technology* on Tel +1 919 319 7142 (or in the Continental US on 800 636 9520). For information on *ON Technology*, email info@on.com, or visit http://www.on.com/ ▮

Prevalence Table – February 1996

Virus	Type[1]	Incidents	Reports
Concept	Macro	58	14.8%
Form.A	Boot	40	10.2%
Parity_Boot.B	Boot	37	9.4%
AntiEXE.A	Boot	31	7.9%
Empire.Monkey.B	Boot	24	6.1%
Sampo	Boot	20	5.1%
AntiCMOS.A	Boot	19	4.8%
Ripper	Boot	17	4.3%
NYB	Boot	16	4.1%
Bupt.9146	Boot	10	2.6%
Junkie	Multi	10	2.6%
Manzon	File	9	2.3%
Telefonica	Multi	9	2.3%
Stoned.Angelina	Boot	7	1.8%
V-Sign	Boot	6	1.5%
Empire.Monkey.A	Boot	5	1.3%
EXEBug.A	Boot	5	1.3%
Parity_Boot.A	Boot	5	1.3%
Jumper.B	Boot	4	1.0%
She_Has	Boot	4	1.0%
Natas	Multi	3	0.8%
Peter II	Boot	3	0.8%
Quandary	Boot	3	0.8%
Stoned.Diablo	Boot	3	0.8%
BootEXE.451	Multi	2	0.5%
Joshi	Boot	2	0.5%
Stealth_Boot.C	Boot	2	0.5%
Stoned.Kiev	Boot	2	0.5%
Stoned.No_Int	Boot	2	0.5%
Stoned.Standard	Boot	2	0.5%
Unashamed	Boot	2	0.5%
Other[2]		30	7.7%
Total		392	100%

[1] Each month, the virus type will also be included for those viruses in the top section of the *Virus Bulletin* Prevalence Table. Information on virus type, for those infectors listed below, is available if required.

[2] The Prevalence Table includes one report of each of the following viruses: 2K-674, Byway, Cascade, Changsha, Da'Boys, DMV, Espejo, EXEBug.B, Fat_Avenger, FitW, Floss, Frodo.Frodo.A, Helloween.1376, J&M, Jerusalem.Barcelona, Keypress, Leandro, MtE:?, Music_Bug, Ontario, Quicky, Quox, Raajat.871, SF2, Stoned.Stonehenge, Swiss_Boot, Taipan.438, Trojector.1463, WBoot.A, Wonka.

IBM PC VIRUSES (UPDATE)

The following is a list of updates and amendments to the *Virus Bulletin Table of Known IBM PC Viruses* as of 21 March 1996. Each entry consists of the virus name, its aliases (if any) and the virus type. This is followed by a short description (if available) and a 24-byte hexadecimal search pattern to detect the presence of the virus with a disk utility or a dedicated scanner which contains a user-updatable pattern library.

Type Codes

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

Alfons.1344

CER: An appending (EXE) and prepending (COM) 1344-byte (EXE) and 1426-byte (COM) virus containing the encrypted text: 'Alfons ! Synchronizing drive C:(do not interrupt this operation! 0% Done'. During the first week of a month starting on Sunday, the virus may overwrite the first hard disk.

```
Alfons.1344          36B4 E283 CC1F C340 1EFC B406 CD52 3321 26ED 578B 8EFE 80DA
```

Asparagus.768

CR: An appending, 768-byte virus containing the plain-text strings: 'Her zaman iyiler K A Z A N I R ! Dogruluktan A Y R I L M A !' and 'ZEKVIR Virusu (c) 1 9 9 5 ASPARAGUS (tm) INTELLIGENT'. It tries to hide its presence in memory by restoring the original Int 21h vector when a file is loaded for execution. It does not check for a resident copy of itself: as a result, the system may run out of memory, being occupied by copies of the virus.

```
Asparagus.768        E834 00B4 40B9 0003 BA00 01CD 21B8 0157 5A59 CD21 B43E CD21
```

AT.156

CR: An appending, 156-byte virus residing in low memory. It tries to avoid infecting EXE files but only checks for programs starting with 'M', with the result that it corrupts files starting 'ZM'. Infected files carry the time and date of infection.

```
AT.156               8BE8 B19C 2BC1 3B44 0174 16B4 40CD 21B8 0042 33C9 CD21 B440
```

Awaits.500

CR: An appending, 500-byte virus showing the text: 'Hell awaits' when an infected file is run for the first time. When active in memory, the virus displays the text: 'Infecting...' when files are loaded for execution and repeats the message after infection is complete. Other plain-text strings read: 'Start works' and 'Press ENTER to continu' (this is located at the end of file). Infected files have their date set to the year 2151.

```
Awaits.500           B440 B903 00BA 3A00 CD21 5A59 B801 57B6 ABCD 21B4 3ECD 21E8
```

Baby.962

ER: An appending, 962-byte virus infecting on FindFirst call (e.g. when the 'DIR' command is executed). On 1 January it may display the text (usually encrypted): 'HELLO BABY! I LOVE YOU'.

```
Baby.962             B425 B01C BAC1 03CD 21B0 21BA E603 CD21 5A1F 078C D82E 0306
```

BadCom.600

CN: An appending , 600-byte direct infector infecting one file at a time. The virus occasionally displays the text: 'Bad COM format'.

```
BadCom.600           BF29 0303 F9B9 2900 303D 47E2 FBFC BF00 01BE 3903 5903 F151
```

BootCom.450

CRMD: A multi-partite, appending 450-byte virus infecting COM files, DOS floppy boot sectors and the Master Boot Record of hard disks (does not save original MBR).

```
BootCom.450          0733 C08E D8BB 2A01 BE00 7C56 FF0E 1304 A113 04C1 E006 2D10
```

Doperland.490

CR: An appending, 490-byte virus containing the plain-text strings: 'Happy Birthday Doperland!!!' and 'c:\command.com'. Infected files have the string: 'Trif' located at the end of their code.

```
Doperland.490        B8AC FDCD 213D DDBA 744B 9090 B9FF 0181 E918 008B F5B8 2000
```

DST.231

CR: An appending, 231-byte virus which infects files with a COM extension, and marks infected programs with byte 03h located at offset 0003h.

```
DST.231              3F8D 1603 00B9 0400 CD4B 2E80 3E06 0003 742C B802 4233 C933
```

DST.242

CR: An appending, 242-byte variant which infects files with a COM extension and marks infected programs with byte 03h located at offset 0003h.

```
DST.242              3FB9 0400 8D16 0300 CD4B 2E80 3E06 0003 742C B802 4233 D233
```

Fifo.333

CR: A prepending, 333-byte virus infecting on GetFreeDisk Space call (e.g. when the 'DIR' command is executed). It contains the plain-text strings: '*.COM' and 'FIFO'.

```
Fifo.333             CB3D A44B 7501 CF80 FC36 7403 E9DF 0050 5351 521E 0655 B419
```

Hi.378

CR: An appending, 378-byte virus detected with the template published in the August 1992 edition of *VB*. This variant infects COM files and adds 200 years to their time stamp.

Hi.512

ER: An appending, 512-byte variant containing the string 'Hi'.

```
Hi.512               8B16 1304 C706 6401 D32E 4AB1 0689 1613 04D3 E2B9 4000 8CC0
```

Hi.549 **ER:** An appending, 549-byte virus containing the text 'ACE OF BASE'. It is detected with the template published in the August 1992 edition of *VB*.

Hi.671 **ER:** An appending, 671-byte virus containing the text: 'ACE OF BASE 2'. It is detected with the template published in the August 1992 edition of *VB*.

Hi.806 **ER:** An appending, 806-byte virus detected with the template published in the August 1992 edition of *VB*.

Hi.833 **ER:** An appending, 833-byte virus detected with the template published in the August 1992 edition of *VB*. It contains a number of payloads, including overwriting the hard disk.

Kasia.495 **CR:** An appending, 495-byte virus residing in the Interrupt Vector Table. Infected files are marked by the string: 'aisaK' at the end of their code. The virus hooks Ints 21h, 1Ch and 17h. It disables the printer and occasionally displays the usually encrypted text: 'UNHANDLED SYSTEM ERROR #17 AT 0F00:0FFF'.

```
Kasia.495           BA81 03B0 17CD 21B8 2135 CD21 891E 7D03 8C06 7F03 BA80 02B4
```

Louse.919 **CER:** An encrypted, appending, 919-byte virus, which often crashes the system when an EXE file is infected or executed. It contains the text: 'Louse (pl. lice) - small insect living on plants, bodies of animals, human beings and disks under dirty conditions.' and 'Your disk is abso-fucking-lutely infested with lice!'. The following template detects it in memory.

```
Louse.919           B8BA A2CD 213D AAB8 743B FAB8 4E01 2687 0684 0089 8680 0126
```

Morgen.656 **CR:** An appending, 656-byte virus occasionally displaying (in Hungarian) a joke about someone called Jörgen Morgen: 'Jörgen Morgen dán váltóőrnek az Oslo-Koppenhága expreszvonat levágja a két kezét és a két lábát. Mit mond erre Jörgen Morgen? -Hát a faszom nem kéne? (C):BGS,1993'.

```
Morgen.656          B4FE CD21 3D47 4274 56E8 0000 5E83 EE0C 1E8C D848 8ED8 33FF
```

MrH.1000 **CR:** A polymorphic, stealth, appending, 1000-byte virus containing the text: 'MrHANDEL - Carmelite Vesperes.' and 'COMMAND.COM'. All infected files have the string 'Mr' located at offset 00003h.

```
MrH.1000            83ED 06BF A203 9003 FD2E 813D C3C3 7414 90B9 B903 BF2F 0003
```

MrR.1000 **CR:** A polymorphic, stealth, appending, 1000-byte virus containing the text: 'MrRAVEL - BOLERO.' and 'COMMAND.COM'. All infected files have the string 'Mr' located at offset 00003h.

```
MrR.1000            0300 95BF CF03 03FD 2E81 3DC3 C374 16B9 BE03 BF2A 0003 FDB2
```

MrR.1300 **CR:** A polymorphic, stealth, appending, 1300-byte variant containing the text: 'divide overflow:', 'COMMAND.COM', 'c:\dos\format.com' and 'CHKDSK.EXESCANDISK.EXENDD.EXE'. Infected files have two spaces (2020h) located at offset 00003h.

```
MrR.1300            7416 B9F3 04BF 2100 03FD B2?? 2E?? 152E 280D 2E28 1547 E2F4
```

Mururoa.2464 **CER:** A stealth, encrypted, appending, 2464-byte virus containing a message against French nuclear tests at Mururoa: 'I have one mesage to all people on earth: Stop all French nuclear testing in the PACIFIC Dont forgot :Comon people dont like nuc. tests! This is is a MURUROA 1.386 by Blesk PLUTONIUM IS BETTER IN POWER-PLANT!!!! My greet to VYVOJAR,SVL,METABOLIS and all IRC.'

```
Mururoa.2464        E205 EB1D 5EEB 1C2E 3014 EB12 B925 00EB 072E 8A94 4A09 EBF4
```

Nado.841 **CR:** A stealth, 841-byte virus containing the text: 'anti-vir.dat' and '[Yitzak-Rabin 1.00 (c) made by TorNado in Denmark'96]'. The stealth procedure has a serious bug: when the virus is active in memory, all clean files appear to be 841 bytes shorter.

```
Nado.841            3E8B 9621 038D B609 00B9 6501 3114 4646 E2FA C3E8 0000 5D81
```

Neumann.752 **CR:** A 752-byte virus containing a destructive payload. It triggers on 5 May, overwriting 1280 sectors of the C: partition. All infected files have their time-stamp set to 62 seconds. The virus contains the plain-text message: '[NEUMANN J. (1903-57) V2.7 HUNGARY]'

```
Neumann.752         B821 250E 1FBA 6001 CD21 2E80 3E56 0100 7437 33DB EB0E 2E8B
```

SillyC.316 **CN:** An appending, 316-byte fast, direct infector with a destructive payload. When a file carrying the twenty-first generation of the virus is executed, the hard disk is overwritten.

```
SillyC.316          DF81 C387 01E8 30FF C3B4 03B0 20B6 00B2 80B5 00B1 01CD 13C3
```

SillyC.373 **CN:** An appending, 373-byte direct infector which infects single files. The time stamp on infected programs is set to 62 seconds.

```
SillyC.373          8CC8 8ED8 8B16 0101 5B53 B972 01B4 40CD 2172 3EBA 0000 B900
```

SillyCR.214 **CR:** An appending, 214-byte virus, residing in low memory. It triggers on 1 January, overwriting the C: partition. Infected files have byte 9Ch (£) located at offset 0003h.

```
SillyCR.214         895D 058C 4507 C645 04EA BA63 00B4 25CD 215E BF00 0157 0E1F
```

SillyCR.354 **CR:** An appending, 354-byte virus marking all infected files with a time-stamp set to 62 seconds.

```
SillyCR.354         B900 9351 1F33 FF81 C758 012D 0300 8845 0188 6502 B800 4233
```

T555 **CR:** An encrypted, appending, 556-byte (sic!) virus containing the text: 'T555 the terminator - And when we will succed , a new era will begin for manthe century of the machine ,QzSaEcFD!'. Infected programs have the character 'A' (61h) located at offset 0003h. The following template detects the virus in memory.

```
T555                2EA3 0201 0706 1F58 BD00 0155 33ED C3F6 D480 FCB4 7407 F6D4
```

INSIGHT

Research and Other Hobbies

Many people have become 'names' in the anti-virus world, for research and for programming: the former category has Professor Klaus Brunnstein as a member. Head of Hamburg University's Virus Test Centre (VTC), his career has seen controversy and success. Brunnstein is one of the 'old guard' of computing: his first exposure to the technology was during his university studies (Hamburg 1957-1962), where he used IBM 7047s and Telefunken TR-4/TR-440s in his dissertation.

'Programs were written in Algol and, for the IBM 7047, translated into Fortran,' he said. 'We had to operate the computer ourselves, from booting (working in single process mode) to diagnosing and curing errors. Despite these limitations, they were great times, where users did not interrupt my work with hardware, software, and systems!'

A Career in Computing

Brunnstein's professional life began in 1964, when he took up a post at the German Electron Accelerator computer centre, working with minicomputers and mainframes. In 1967, he gave his first lectures on the operating systems of mainframes and minicomputers. That year, he also developed a Library Information System with one of the first SDI (Selected Dissemination of Information) devices.

1969/70 saw his secondment to a group founded for the purpose of preparing the ground for Hamburg University's Informatik-Fakultät (IT Faculty): Brunnstein's responsibility was to develop the library and computer centre. In 1973, he became the first European professor for the application of Information Technology, concentrating on education.

In the 1970s, Brunnstein changed the direction of his research to business applications for Data Protection and Security. Since 1983, he has been chairman of the Advisory Committee for Europe's first Backup Computer Centre.

A Virus Outbreak

It was not until 1987, when Jerusalem appeared in the wild, that Brunnstein became aware of the threats posed by viruses. The following year, his minicomputer laboratory was transformed into the Virus Test Centre (VTC).

At the VTC, reverse engineering at every level is studied: viruses, explained Brunnstein, are an ideal medium for this purpose, being complex but not impossible. Researcher Vesselin Bontchev spent several years studying at the VTC, as did Morton Swimmer. Brunnstein has also been teaching a two-year course on Computer and Communication Security, which was inaugurated in 1989: the course currently running has eighty students.

The Faculty for Information Technology includes the VTC and the Net Test Centre, where network safeguards are tested under various operating systems (*Novell*, *Windows NT*, *OS/2*, etc). Although the VTC has well-established relationships with many anti-virus software developers, it is not funded by any of them, as it is part of Hamburg University: 'We don't get *any* funding for our anti-virus work,' explained Brunnstein. 'Our funds come from the university,' and, in a rueful aside: 'And are consequently rather meagre.'

CARO as Colleagues

Klaus Brunnstein is one of the founder members of the *Computer Anti-Virus Research Organisation* (*CARO*). This group was initially 'a little community aware of the threats and aiming at public information more than at business'.

'Most *CAROts* had research and academic interests,' he said, 'even when Alan (Solomon) and Fridrik (Skulason) began to develop their products. When Vesselin joined the VTC in 1990, he and Fridrik became active via email (Virus-L was indeed sometimes Vesselin-L!), attracting many new friends.

'CARO's international membership means new threats can be immediately analysed,' he added. 'Moreover, virus detection methodology has improved through discussion, including awareness of user needs and demands of testers. Vesselin in particular has contributed significantly to testing methodology during his studies.'

The Virus Threat

Brunnstein sees viruses as a threat essentially only on PCs and on the Amiga: less than 50 Macintosh viruses are known, and only a few UNIX viruses (none of which are in the wild).

'Virus authors write (for the most part, they merely modify) viruses as an experiment in program and system behaviour on related platforms,' commented Brunnstein. 'Very few viruses are written with the intent to damage.'

Fledgling authors, he feels, do not realise how damaging their creations might be. Only when they mature do they understand the ethical (and in some countries legal) implications of viruses and their potential for havoc.

'As long as schools and universities do not educate students on the malicious impact of what they are doing,' asserted Brunnstein, 'the wave of viruses will continue. The race between virus authors and anti-virus producers will continue. New methods will be more difficult to understand and detect. Macro viruses, which work on a higher software platform, are a new threat. Such viruses have been predicted and demonstrated by Professor Harold Highland as early as 1989/1990, but even today their malicious potential has not been realised.'

There are also dangerous developments outside the virus field, in Brunnstein's view. He feels in particular that Trojan horses and worms will be a more serious problem, especially to enterprise LANS and WANS.

'This threat will grow,' he continued, 'when software to install and autonomously distribute intelligent agents is more broadly available. Even when such languages (e.g. Java) have inherent security features, misuse will not be prevented as new threats exploit safety rather than security.'

Brunnstein forecasts that viruses will become less important, despite their increasing numbers, and that network threats will grow significantly, eventually dominating in enterprise and public networks. The only anti-virus producers he envisages surviving this change are those which enhance their products to become anti-malware packages.

'The advent of complex self-hiding methods,' Brunnstein said, 'is part of the game virus authors play with anti-virus developers. Creating a polymorphic engine does not need a genius – reverse-engineering is more difficult. Few producers seem to analyse code: developing and testing algorithmic detection is more time-intense and costly than extracting scan strings. Heuristics can help detect new viruses and variants with similarities, but is inherently less reliable than traditional methods.'

Klaus Brunnstein, academic and researcher, is constantly seeking new challenges.

On the Legal Front

Brunnstein does not believe that present legislation can cope with the problems posed by viruses: 'Laws often require prerequisites which cannot be proven,' he explained. 'In German computer sabotage and espionage laws, for example, deliberate intent is required, which is not easily provable when virus authors argue that their intent was purely educational.

'Second,' he continued, 'Lawyers and judges rarely understand technical terms or have a basic understanding of computers. Third, PCs and most networks have no inherent auditing which stores traces of malware; prosecutors (even if capable) always have difficulties proving damage.

Man in the Mirror

A controversial article, Rächer im Datennetz (Network Avengers) appeared in the German magazine *Der Spiegel* (German for 'mirror') some time ago, accusing Brunnstein of being little more than a virus profiteer. The article quoted him, amongst other things, as having said that the Michelangelo virus would destroy hundreds of thousands of PCs.

Brunnstein's official response was that the article was so full of misconceptions and false allegations that he was not willing to comment. He recalled numerous confrontations with the journalist in question: in fact, not only was Brunnstein never asked if the quotes were correct, but there was no contact between the two when the article was written: 'I decided,' he said simply, 'that non-reaction was best.'

Outside the VTC

Although Brunnstein plans to remain at the Virus Test Centre and to carry on with his work in education, he does have other areas of interest, both personally and professionally. His main area of work at the moment is analysing incidents on computer and net-based systems, from hacking and malicious software to real-time systems such as electronic flight control and medical systems.

He is at present writing two books; one on the methodologies of incident analysis and response, another based on his university courses. Together with colleagues, his work on the ethical and legal aspects of information technology is ongoing, and he is also currently analysing potential net threats from intelligent agents.

Although currently not active politically, Brunnstein has also been a Member of the German Parliament, and leader of a German coalition party. His political activities include a campaign to stop the Census Law of 1983: the term 'informational self-determination' was taken by the courts from the action of 'Brunnstein et al against the Federal Republic of Germany'. Brunnstein in fact feels that some of the controversies may relate to different political positions on the social implications of IT.

Sailing is another of his passions: he and his wife Gunda ('A garden architect responsible for some English-style parks in Hamburg,' said Brunnstein. 'She is interested in computing, though garden architects don't use computers much presently!') are often to be found on their 48-foot Newfoundland schooner. 'Based on the Bluenose's lines, but rigged as a staysail schooner,' he explained.

Brunnstein professes to enjoy life with his family, of all of whom he is manifestly proud: 'My daughter Anke is just now working on her doctorate in ornithology, specialising in rare birds, and my son Jochen is studying Economics, including computer methods and applications, here at Hamburg University.'

Music and literature take up much of the rest of his leisure time, and what remains is spent with the family's animals; two dogs and a horse.

Klaus Brunnstein has strong views on many subjects: whether controversial or conservative, these views will doubtless continue to be aired, and to provide, as before, ever more topics for discussion.

VIRUS ANALYSIS 1

Zhengxi: Saucerful of Secrets

Eugene Kaspersky

Only rarely does an anti-virus researcher come across a virus which cannot be dissected thoroughly and for which disinfection routines are difficult. Occasionally, however, it does happen, and Zhengxi is the latest on the list.

At 7K long, it is one of the most involved I have ever seen. It is a sort of 'all-in-one' virus, which infects EXE and OBJ files and attaches infected COM droppers to ZIP, ARJ and RAR archives (both static and self-extracting). Its very complex polymorphism resembles that of SMEG, but has loops often exceeding 2K in length, concealed by vast quantities of junk subroutines, and Int 21h and CP/M calls.

Zhengxi infects EXE files either by appending its code to the end of the file in the standard manner, or by looking through the file for C or Pascal subroutines and modifying these to execute the virus (as Lucretia). The OBJ infection technique is similar to that of Shifter [*see VB March 1995, p.11*], and that of archive modification was first seen in Dementia [*see VB November 1995, p.12*].

Initial Disassembly

The first step in analysing a virus of this type is to strip away the polymorphism and expose the virus body. This then usually succumbs fairly quickly to the experienced eye. In the case of Zhengxi, after executing the hundreds of junk instructions and subroutines concealing the decryption loop, the following text-strings become visible:

```
Abnormal program termination The Virus/DOS
0.54 Copyright (c) 1995 Zhengxi Ltd Warning!
This program for internal use only!
```

However, apart from these strings, the virus' inner workings are far from obvious. Even after many hours of hard work with debuggers, disassemblers and other tools, the code refused to give up all its secrets.

One of the things that makes it so complicated is its use of two methods to reference data: direct (CS:[address]) and indexed (SS:[BP+offset]). In many cases, both are used to access the same data at different points within the virus code, meaning that not even good disassemblers could build the complete reference tables which are often so useful in analysis. In addition, the virus uses 'hidden branches', where the destination address is concealed and may vary.

As I was doing this analysis, I was also watching an old movie about Pink Floyd, and playing their album *Saucerful of Secrets*. It occurred to me that analysing a virus such as Zhengxi in the mid 1990s could be compared to listening to avant-garde music such as Pink Floyd's in the early 1970s.

Checksums: An Interesting Idea

At several points within its code, Zhengxi uses an intriguing technique to decide whether or not to infect a file. Instead of directly comparing information from the file with data stored within the virus (which eases analysis, as it is possible to see what the virus is seeking), it computes a checksum (CRC) of the data and compares that with one stored within the virus.

This makes it considerably harder to discover what the virus is seeking – it is necessary to 'reverse-engineer' the checksum in an attempt to find all the byte patterns which could have generated it. Zhengxi first uses this technique when checking the name of a file it is considering for infection.

It takes the first five bytes of the filename, and uses them to compute a two-byte checksum which it compares with sixteen such sums stored within its body. If it matches, the file is not infected. There are over 25,000 five-letter strings which match one of these sixteen. From these I discovered twelve EXE files likely to match eleven of the checksums: UUENCODE, PKLITE, LZEXE, NDD, DIET, AFD, SD, SPEEDISK, DEFRAG, LINK, TLINK and AVSCAN. The other five checksums remain a mystery.

Zhengxi uses the checksum technique again when examining the contents of the file. Here it compares various checksums computed over the header against eight sums stored within its body. The same technique revealed which ones matched:

- 80h, the one-byte OBJ file marker
- 4D5Ah and 5A4Dh, the two EXE file markers
- 69EAh, the ARJ-archive marker
- 504Bh ('PK') 0304h, the ZIP-archive marker
- 52617221h ('Rar!'), the RAR-archive marker

The purpose of the two remaining CRCs is unknown. It appears (from the code paths taken following a match) that one should match a type of OBJ file, and the other matches some form of archive.

Going Resident

The virus also uses checksumming as it attempts to go resident. When an infected object is run, once the virus has decrypted its code, it hooks Int 1h (Single Step interrupt), and calls Int 21h indirectly (via the obsolete CP/M call 19h, Get Current Drive). Courtesy of its Int 1h handler, the virus receives control as the Int 21h handler starts to execute.

It now computes a two-byte checksum of the first twelve bytes of the Int 21h handler, and tests this against two stored possibilities. There are over two billion possible twelve-byte sequences which produce one of the two checksums in question, but the intent is clearly to identify two common

configurations. When one is found, the virus performs the following additional checks before going resident. It will terminate if any of these conditions are met:

- *Windows* is active (Int 2Fh, AX=1600h)
- Boot drive is A or B (Int 21h, AX=3305h)
- Int 8h, 13h and 28h vectors point into the same segment
- The date on the host file is the same as the current date

Next, Zhengxi creates space for its TSR copy using Int 21h, AH=4Ah and 48h (Change and Allocate Memory). It copies into its body the bytes from the Int 21h handler which matched the checksum, and patches that with a FAR CALL.

Using this technique means that the virus does not require an 'Are You There?' call. Zhengxi goes resident only when the Int 21h handler matches one it is seeking: when it goes resident it changes this handler, so it will no longer match.

What is the destination of the FAR CALL? 'The virus' Int 21h handler,' any expert would say, as did I. That would be wrong: it is the address of the system Int 25h (Absolute Disk Read) handler. The virus patches this handler with a FAR JMP to the TSR copy of the virus, so both Int 21h and 25h calls end up at the same address. To separate them, the TSR handler checks the code which generated the interrupt, and passes control to the correct virus interrupt handler.

All that remains is for the virus to copy its code into the memory created for it, and return control to the host program. How this is done depends on the host, and the method by which it was infected. At this point the COM dropper displays the message 'Abnormal program termination' and exits.

Stealth Functionality

The stealth capabilities of the virus are realised using both of the virus' interrupt hooks (21h and 25h). The Int 25h handler allows the virus to intercept attempts to access the directory entries, at which point it returns the original file length for infected files. This handler also restores the headers of such files to their original condition.

DOS functions 3Fh (Read), 42h (Lseek), 3D00h (Open, Read Only), and 11h, 12h, 4Eh, 4Fh (searching for files) are used by the Int 21h handler to provide limited stealth functionality. The virus replaces the file's original header, does not allow seeking outside the original bounds of the file, and returns original (i.e. uninfected) file sizes. In addition, the virus disinfects a file when it is opened (Int 21h, AH=3Dh, 6Ch) for writing, or deleted (Int 21h, AH=41h).

Miscellaneous Int 21h Functions

The virus utilises several other Int 21h functions. Functions 0h (Exit), 31h (Go TSR), 49h (Free Memory) and 4Ch (Exit with return code) are used to allow the virus to relocate itself in memory under certain conditions – it simply allocates a new block of memory, copies itself there, and changes the FAR JMP to the virus' Int 25h handler.

Zhengxi uses functions 67h (Set Handle Count), 48h (Allocate Memory), 4Ah (Resize Memory Block) and 4Bh (Exec) to allow it to attempt to hide its code within memory: it manipulates the MCBs to render it invisible when using memory browsing utilities.

When it sees a call to function 3D00h (Open, Read Only), the virus scans the code of the calling process and patches it. It seems Zhengxi is attempting to modify an anti-virus utility; however, it is irrelevant as there is a bug at this point, and that branch is never executed.

General Points of Infection

The virus uses Int 21h, AH=4Eh, 4Fh (FindFirst/Next by name) to infect files. Every file accessed by FindFirst, and every fifth accessed by FindNext, has its name stored in a buffer within the virus: the name stored there previously is overwritten. When next a file is closed, the virus infects the file whose name is stored in the buffer. In addition, on one out of every four tries, the file being closed is infected. Before a file is infected, Zhengxi checks that:

- the file was not created that day (similar to the test made when the virus goes resident)
- the file is on a local drive (not A or B)
- the file name does not match the pattern *.?V? (i.e. it is not an overlay file)
- sufficient disk space is available (Int 21h, AH=36h)

When all these conditions are met, the virus checks the file name and header using the checksum method described above. Execution then branches according to file type.

EXE Infection: Appending and Archive Modification

If the header is that of an EXE file, the virus determines whether it is a self-extracting archive. If it recognises a ZIP, ARJ or RAR self-extracting archive, it uses the code described later. If it does not, Zhengxi tests the length – a file is not infected if it is less than 400h bytes long. Next, it examines some fields in the file header: if the EXE module length is greater than 32K, the file is not infected in this manner, but by insertion. If this is not the case, the virus reads the file header, encrypts it, and writes it to the end of the file.

Next, the polymorphic generator is run, and the virus saves its newly-encrypted body at the end of the file. The infection process is completed with the virus marking the file as infected. This is done by padding the file to a length that, when divided by 9Dh, gives a remainder of 25h, and then modifying some fields within the EXE header.

EXE File Infection – Insertion

As stated above, if the EXE module length is greater than 32K, Zhengxi attempts to infect it by inserting the call to the virus code somewhere within the host program's code. It accomplishes this by reading in the first 6K of program code (starting from just after the EXE header), and searching it

for C or Pascal subroutines. These usually start with one of the following two three-byte sequences, which disassemble to the same two operations:

```
55          PUSH BP       55
8B EC       MOV BP,SP     89 E5
```

The virus searches for these sequences using the checksumming described above. If a match is found, it checks the next 80 (54h) bytes of code for a RET or CALL FAR instruction, to prevent it overwriting a return or call to another subroutine.

Once it has determined this subroutine is safe to patch, the virus overwrites the 54h bytes with the first part of the virus loader code, and encrypts and writes the virus body to the end of the file. It then encrypts the overwritten code of the subroutine and the remainder of the loader code (also saved to the end of the file). Next, it marks the file as infected by modifying the length, in the same way as for infection by appending – all in all, a very insidious infection technique.

As with Nexiv_Der [*see p.11*], it is impossible to detect files Zhengxi has infected in this way simply by scanning the file's entry point. The virus may not be called every time the file is run; it may well depend both on the overall environment and the individual options given.

Archive Infection

Archives are infected by placing a COM dropper within them – these begin with a JMP instruction, which is followed by random data, before the virus code begins. The JMP passes control to the decryption loop. The name of the dropper is chosen at random, but always ends in .COM.

Zhengxi does not require an external utility to handle the archives – it knows how to deal with ZIP, ARJ and RAR archives without external help. It does not compress the dropper, but inserts it as is into the archive: these archive formats all allow for this.

Special treatment is given to ZIP archives – the virus does not attach a dropper to an archive containing files which are 'stored' (PKZIP's name for a file not compressed in any way), allowing it to avoid placing multiple droppers into the same ZIP archive. In the case of the other archive formats, the virus places certain known values into header fields of the archives to prevent multiple infection.

OBJectionable Stuff

When infecting OBJ files, Zhengxi searches the fields of the OBJ file before creating and inserting new records containing the polymorphically-encrypted virus code.

Multiple infection is prevented by modification of the OBJ file fields so they do not match its requirements next time around. This is done by scanning the code of the OBJ files for the C/Pascal subroutine entry code, as described above. If this can be found, the OBJ file is infected. Unlike inserting into EXE files, the bytes overwritten by the CALL instruction are not saved for later restoration.

Once the OBJ file is linked into an EXE, the virus code becomes 'runnable'. The CALL passes control to the virus' decryption routine, and virus execution proceeds as normal. Just before control returns to the host, the virus overwrites its CALL with 558BECh, the C-style subroutine header. As with inserting into EXE files, this technique can result in the virus code lying dormant for some time – it may only become active under certain very specific circumstances.

Trigger Routine

Zhengxi contains one trigger, called when a ZIP file is being processed, other than the message displayed by the COM droppers (which does not count as a trigger *per se*). If the archive contains a 'stored' file, and the last modification date of the archive is 1996 or later, the virus finds and deletes all files and directories on drives C through Z.

Conclusion

Zhengxi is an exceptionally complicated virus, which does its job efficiently. It is not believed to be in the wild, but it does exhibit several new techniques, and is very destructive.

Zhengxi

Aliases: None known.

Type: Memory-resident, stealthy, polymorphic parasitic infector. Infects EXE and OBJ files, and inserts COM droppers into both static and self-extracting ZIP, ARJ and RAR archives.

Self-recognition in COM Droppers and in EXE Files (inserting):

File length = (9Dh * n) + 25h, where n is any integer. [*NB Self-recognition in memory not applicable – see text*].

Self-recognition in EXE Files (appending):

Length condition as above. EXE header values: CS = SS + 1, 79h < SP < 90h.

Self-recognition in OBJ Files and Archives:

See analysis.

Hex Pattern in Memory and in EXE Files (inserting):

```
9C0E E800 0051 5650 5352 551E
83EC 708B F436 836C 7E05 B451
```

Not possible in other infected objects.

Intercepts: Ints 21 and 25h.

Payload: Wipes all files/directories on drives C-Z.

Removal: Under clean system conditions, identify and replace infected EXE and OBJ files. Check ZIP, ARJ and RAR archives for COM droppers.

VIRUS ANALYSIS 2

Nexiv_Der: Tracing the Vixen

Peter Szor

Every year there are new developments in virus-writing; virus authors are constantly working to keep the anti-virus vendors on their toes. Occasionally, a new infection method pops up which can require a fundamental change to the way a virus scanner works: Nexiv_Der belongs in this category.

This virus uses a peculiar method of infection. It traces the execution of a victim file and randomly inserts a call to its own code in the middle of the host: this makes it difficult to use emulation or entry point scanning methods to detect it. It is multi-partite, and polymorphic in files and in boot sectors, but most interesting is its tracing method, which provides it with generic anti-goat and anti-heuristic capabilities.

The encrypted string 'Nexiv_Der takes on your files' found within the code gives the virus its name – but the message is never displayed. Indeed, the virus has no trigger; it just spreads.

Theory

'When an infected program is first executed, the virus code is executed first. Then the virus takes control and goes resident…' This definition is true for most file-infecting viruses, but Nexiv_Der is different: it traces its victim's execution and inserts the patch to its code at a random point.

Consider the DOS utility MODE.COM: MODE is a multi-purpose tool which can be used to control such things as keyboard speed, COM and LPT port settings, code pages and the number of text lines on screen. Different operations are controlled by command-line parameters.

Let's look at two cases where Nexiv_Der is resident and ready to infect this file. First, the user executes the command MODE COM1 BAUD=19200. The virus traces the program to the COM-handling routines of MODE, located mostly in the last third of MODE.COM. It then patches a call to its code in the middle of these COM port handling routines. Now the file is infected, but the virus is unable to spread unless the user runs it with a similar command-line.

In the second example, the user executes MODE CON RATE=32 DELAY=1. Nexiv_Der traces the keyboard handling routines of MODE.COM, and inserts the patch in another part of the file, probably near the beginning.

So, a Nexiv_Der-infected file is usually only able to spread further if the infected file is executed in a similar environment and with similar settings. Since most of the more advanced anti-virus programs use emulation to detect viruses, they will have problems with this sort of infection technique. How does an anti-virus program locate where the execution flow transfers from host program to virus? If the file was originally infected when used with a special command-line switch or in a special environment, how can the anti-virus program emulate the circumstances?

Infection

When an infected file is executed and Nexiv_Der takes control, it first decrypts – the first byte of the polymorphic decryptor is always a PUSHF command. Next, the virus calculates a checksum across its own code, and if the checksum has changed, returns control to the host program.

If the checksum matches, Nexiv_Der issues its 'Are You There?' call, Int 21h AX=304Eh, and expects the answer AH=30h. This is the DOS version query command: normally, the AL value is ignored. This means that many programs leave AL uninitialised when calling this interrupt and can accidentally trigger Nexiv_Der's 'Are You There?' call, getting odd results for the DOS version number.

If the call is not answered, Nexiv_Der reasons that it has not infected the DOS boot sector of the hard drive, and proceeds to do so. Unlike most other multi-partite viruses, Nexiv_Der infects the DOS boot sector, not the MBR.

The original DBR is moved to track 0 sector 6, the DBR is modified to contain a polymorphic loader (see below), and the rest of the virus is written to the twelve sectors from sector 7. After the DBR has been infected, Nexiv_Der does not stay resident, but waits for the next reboot to continue.

Going Resident

When the machine is rebooted from the hard drive, the virus is run. It first decrypts a small section of loader code which brings the virus body and uninfected boot sector into memory. Like almost all boot sector viruses, Nexiv_Der obtains the amount of DOS memory from the BIOS data area and copies the resident part of its code to just below the top of memory. Then it reduces the amount of system memory by 5KB, enough for its resident copy.

After storing the address of the original handler within its code, the virus hooks Int 13h. Finally, it re-starts the boot procedure, with the new interrupt handler in place.

Int 13h Handler

The first purpose of the Int 13h handler is to detect when DOS has been loaded so the virus knows when to hook Int 21h. To do this, the Int 13h handler checks the start of every sector read for the 'MZ' file marker. When the third such marker is seen, the virus hooks Int 21h. As modern device drivers are stored in EXE files, this usually happens when the third device driver is loaded from CONFIG.SYS.

Many multi-partite viruses detect when DOS has been loaded by hooking Int 1Ch (Timer) and waiting for the Int 21h vector to change, but Nexiv_Der has simplified this and made it more reliable. The second purpose of this handler is to infect floppies. When a request for the boot sector of drive A or B occurs, the virus checks the floppy type (it infects only 1.44MB diskettes), and compares the second byte of the sector with 45h – this will show if it is already infected. If it is not, the virus inserts JMP NEAR 47; NOP at the start of the sector and writes nine unchanging bytes to offset 47h.

Next, the virus generates a polymorphic decryptor, writing it into the boot sector after the nine constant bytes. As the polymorphic generator sometimes produces more code than can fit there, the virus changes the last word of the buffer to contain the marker bytes 55AAh. Despite this, these sectors usually function correctly. The new boot sector is written to floppy; the rest of the non-encrypted virus body to track 79.

The result is a boot infection which, whilst not completely polymorphic, contains insufficient distinctive constant code to be useful in a scan.

Int 21h Handler

In addition to watching for 'Are You There?' calls, the new Int 21h handler watches for program execution by trapping Int 21h, AX=4B00h. When such a call is made, Nexiv_Der opens the victim file, takes a copy of its first 32 bytes, and checks the file type. If it is an EXE file, infection aborts: only COM files are hit.

Next the virus checks whether the file is infected – if so, the seconds field of the time stamp is already set to 7. In this case, infection aborts. Then the virus checks the file size: a file will be infected only if 1000-55000 bytes long inclusive.

Now, tracing starts. The virus hooks Int 3h (Break Point) and re-hooks Int 13h, temporarily replacing its primary Int 3h handle with a new one. Then the victim file is closed, and the virus resets the disk system and lets execution continue.

Stepping Out

This Int 13h handler waits for a sector to be read. It will not have long to wait, as DOS will be ready to start executing the victim file, which will clearly involve loading it from the disk. When this happens, the virus compares the first 32 bytes to those saved by the virus' Int 21h handler previously. If they are the same, the first byte of the buffer is overwritten with an Int 3h instruction – as DOS is about to execute the file, the first instruction it will now see is a break point, which will trigger the virus' Int 3h handler.

This handler first hooks Int 1h (Single Step), and resets the Int 3h vector to its original state. The temporary Int 13h hook is also removed. Next, the virus generates a random number N between 256 and 2048. Finally, the trace flag of the processor is turned on, causing the Int 1h handler to be called after execution of every instruction.

The Int 1h handler allows program execution to continue for N instructions, where N is the random number previously chosen. If execution stops before N instructions are executed, the file is not infected, meaning very simple programs (such as goat files) will not be hit. If execution moves to a segment different from that of the victim file's code segment, it is simply not counted. This way, interrupt handlers do not affect the number of instructions the virus will trace.

When N instructions have been traced, the virus checks if the last executed instruction was CALL xxxx; JMP xxxx; ADD <8 bit register>, xx; or OR <8 bit register>, xx (all of which take three bytes). If so, the virus saves it, overwriting it with a call to the end of the victim file. If not, it does not infect, although it might the next time the file is run: N will be different. Such instructions occur often in normal programs.

Next the virus calls its polymorphic engine and writes a decryption routine and an encrypted copy of its body to the end of the file. Finally the seconds field of the time stamp is set to 7, marking the file as infected.

Nexiv_Der has a critical-error handler in effect during infection to prevent write protect errors, and it clears the files attributes during infection to allow it to infect read-only, system and hidden files.

Conclusions

Nexiv_Der is a complicated virus. It has some bugs which may cause problems under multi-tasking environments, and sometimes the virus crashes of its own accord. In any case, it is a complex and interesting virus which cannot be detected with searchstrings alone. It is also not easy to detect it by emulation, and disinfection is difficult. It was received from Italy in December 1995 in a collection of new viruses: thankfully, it is not thought to be in the wild.

Nexiv_Der

Aliases:	Red Vixen.
Type:	Resident COM and DBR infector.
Infection:	COM files 1000-55000 bytes long, hard drive DBRs, 1.44MB floppy boot sectors.
Self-recognition:	Seconds field set to 7 in files, value of byte at offset 2 in boot sectors is 45h.
Hex Pattern:	No simple hex pattern is possible for files or boot sectors. In memory:

```
3D4E 3074 0A3D 004B 7406 EA??
???? ??CF 5053 5152 5657 551E
```

Intercepts:	Ints 1h, 3h, 13h, 21h, and 24h.
Payload:	None.
Removal:	Replace infected files with clean copies; replace DBRs with SYS.

VIRUS ANALYSIS 3

What a Quandary!

Kevin Powis

Quandary is an in-the-wild boot sector infector. It infects floppy and hard disks and employs a modest encryption system to help avoid detection. If a diskette infected with this virus is in the floppy drive when a PC is booted, the infected boot sector is loaded into memory at offset 7C00h in segment 0 and run.

Encryption/Decryption

The first eight bytes of virus code set up the entry registers before calling an encryption/decryption routine. This decrypts 34 bytes of virus code using a simple XOR algorithm, a method of encryption which is often used, as it is reversible. If it is called once it will encrypt the target bytes, and when called a second time, the bytes revert to their original values.

Control passes to the decrypted code; instructions to take control of Int 13h (ROM BIOS disk interrupt) and reduce the amount of available memory by 1KB. They must be encrypted, as they contain now-standard instructions seen in most boot sector viruses and are thus detectable by scanners with even a modest generic capability. Once executed, Quandary re-encrypts them, as they are no longer required.

Taking Control

The virus now copies 512 bytes of its own image to offset 7C00h in the new segment it has created in the top of memory. At this point, a virus will normally issue a jump to the virus image at the new location. Quandary, however, issues an Int 19h, forcing the PC to reboot. This is neither a warm nor a cold reboot: all interrupt vectors and memory contents are left intact and the firmware loads the boot sector into memory and passes control to it.

When the firmware tries to read the Master Boot Sector, the request passes through the virus disk handler, which performs the read via a far call to the BIOS which was set up when Quandary captured the interrupt vector. Should this fail, the virus returns to the caller, returning the correct error code.

In the case of a successful disk access, Quandary checks the sector for the signature 55AAh in the sector's last two bytes: this should appear in all MBRs. If for any reason the MBS is missing its signature, the PC's hard disk will not be infected.

If the boot sector is considered valid, another test is made to see whether the word at offset 125h contains value 1405h. This is a puzzle, as it will not do so if infected with Quandary or if clean. It seems the virus is identifying another infection, replacing it with itself – a free upgrade, perhaps! If the boot sector contains the value 1405h here, Quandary performs the 'upgrade', taking the contents of sector 14 (which appears to be where the previous virus placed the MBR copy) and copying it onto sector 15. It then replaces the MBR with its own image and returns control to the caller.

In the absence of this other virus, Quandary checks for itself by comparing the word at offset 1BBh with C928h. If found, this means the disk is infected, and Quandary then retrieves the original MBS from sector 15 and returns it to the caller, thus realising the virus' limited stealth capabilities.

Infection

In all other cases Quandary has now identified an uninfected disk and begins infection. It writes the untouched MBS to head 0, sector 15, copies 64 bytes from offset 1BEh in the MBS to its image in memory (to preserve the partition table) and writes itself to head 0, sector 0, completing infection.

Floppy infection is similar: the virus preserves the original BIOS Parameter Block by copying 60 bytes from the start of the sector over itself. It stores the original boot sector at head 1, sector 15. Quandary only infects 1.44MB diskettes.

Summary

As viruses go, Quandary has little to make it stand out from the crowd. Its programmer appears to have written at least one other virus, but the code has signs of inexperience and a lack of understanding of how some instructions work. It is in the wild, however, and works well. Its stealth routines do not protect it being overwritten, and it contains neither trigger nor payload: perhaps we should be grateful.

Quandary

Aliases:	NewBoot_1, IHC, Parity-enc, Boot-c.
Type:	Boot sector infector.
Infection:	Hard disks, 1.44MB diskettes.
Self-recognition:	Word at offset 1BBh equal to C928h.
Hex Pattern on Hard/Floppy Disks and in Memory:	`81BF 2501 0514 7478 2681 BFBB` `0128 C974 3CB0 01E8 9400 B801`
Intercepts:	Int 13h.
Trigger:	None.
Removal:	For diskettes, salvage required files (which will be completely unaffected), and format. Use FDISK /MBR to disinfect hard drive MBR.

COMPARATIVE REVIEW

Scanning NetWare

It has been a year now since *Virus Bulletin* last did a comparative review involving products designed to protect a *NetWare* network. In that year, many changes have taken place, not least the advent of macro viruses (included for the first time in the In the Wild test-set). In addition to that, the overall number of viruses has been rising consistently, increasing the challenge to anti-virus manufacturers.

The test-sets have been updated since the DOS comparative review in January of this year. As usual, of special note is the In the Wild test-set: this contains the viruses, listed in the *WildList*, which are a threat in the real world. One test-set is missing: NLMs are not tested against boot sector viruses. The Polymorphic test-set continues to grow in size and complexity: it now contains 7500 samples of fifteen viruses.

Testing NLMs

The task of testing NLM anti-virus solutions is very different to that of testing DOS scanners. There are more features on offer, and the packages are more complicated. It is also not possible to do as much automated testing for NLMs as for DOS products, which further increases the time required.

It should be assumed that unless otherwise stated, the product is a standard NLM, and offers standard server-based, on-demand, scheduled, and on-access scanning.

Special Considerations

Two of the products included here are not NLM-based: *ESaSS' ThunderBYTE* and *RG Software's Universal NIM*. These function in a manner different from the others, but are included as they are what these companies would sell to a user requesting anti-virus software for *NetWare*.

There are several advantages to this approach, most notably that it is not necessary to develop a server-based product for every networking system which comes along. In addition, the product will work on minority networking systems for which it would not be economic to provide a native solution (*LANtastic*, *NetWare Lite* etc). The disadvantage is that the customer does not get a solution so integrated into their type of network – on the other hand, they will be able to use the same product on any of their corporate networks.

In addition to these, two other products – *Sophos' Sweep* and *EliaShim's ViruSafe* – have a technique for providing on-access scanning different to many other products. These use a resident program on the workstation which sends accessed files to the server for checking. If a file is found to be clean, information about it is added to a local checksum file so that it is not checked next time (unless the checksum has changed). This means that every file and disk accessed by a workstation will be, if necessary, checked: it does not have to pass through the server. There would be a longer delay in waiting for the file to be authorised the first time, but after that, things are much faster. It should also be noted that the resident software on the client will use less base memory.

Speed

The speed of NLMs is clearly of interest, and offers many chances for the tester to make unfortunate errors of technique. The most obvious errors would result in comparing the speed of a server-based product with one which is network-independent (such as those described in the previous section). On a network where the server is a Pentium 90 and the workstations are 386s with various clock speeds, this would be grossly unfair to the workstation-based products.

There is a relevant point here, however: for sheer flat-out scan speed, a server product has a distinct advantage. It does not have to pull the files across the network to scan them, and the server is likely to be a more powerful machine than its clients in the first place.

Central Point AntiVirus v2.5

CPAV is now very much a legacy product – supported with signature file updates alone, it is not receiving the functionality upgrades it so desperately requires. With this in mind, its continued drop in detection rates is not surprising. The signature files supplied for the review resulted in less than 60% In the Wild detection, demonstrating that the product does not constitute an adequate defence for a network.

All this is disappointing, as the configuration management options and programs are very impressive, all the more so considering *CPAV's* age. It offers centralised updating of servers grouped together into a *CPAV* domain, and a *Windows* administration program which is both complete and easy to use. Also supplied is *CentralAlert*, a notification package supporting multitudinous methods of alerting administrators in the event of a problem.

Nevertheless, in view of the detection, steer clear: if you want a *Symantec* product, *NAV* is better maintained and offers many of the same features.

Cheyenne InocuLAN v4.0

Comparisons between this product and that of *Intel* are inevitable, as they both approach the market from the kitchen sink perspective – throw in everything they can

think of, and it will be bound to suit everyone. For, like *Intel*, this is an example of a product with everything – all the features mentioned there are also present here, and done just as well. There is a minor difference when it comes to the automatic downloading of updates: *Cheyenne* requires a modem attached to the server, in contrast to *Intel*, which needs one connected to a workstation. There are advantages and disadvantages to both approaches.

In one area, however, there is a clear distinction: *InocuLAN's* detection rates are much better than those of *Intel*. Alas, not sufficient to place it near the top of the heap in those terms, but with a little improvement it could get there. Distressingly (at least for *Cheyenne* customers), the last comparative said much the same thing, but we still hold out some hope.

Command Software F-Prot v2.21

Since the last comparative, *F-Prot Professional for NetWare* has sprouted a nice *Windows* configuration and monitoring program, which allows complete control of multiple servers running *F-Prot*. There is also a much thicker wodge of documentation. Servers may now be grouped into domains to allow centralised administration and updating (this is referred to as 'deploying').

Detection is also vastly improved from one year ago, at least in the important In the Wild (where it missed out on one of the Concept samples) and Standard test-sets. The Polymorphic score displays little improvement since then: detection in this area will need some attention before too much longer.

For alerting functionality, a copy of *Alert Track* from *e.g. Software Inc* is provided with *F-Prot Professional for NetWare*. This caters for seven different types of notification: network broadcasts, two types of pager, *FaxWare*, two types of email, and SNMP – with all that choice, there can surely be nowhere for the network administrator to hide.

The *Windows* configuration program is very easy to use and flexible, and it is clear that a lot of thought has gone into its design. The price paid is that it is not possible to control the NLM from the console (or via RCONSOLE), but this is only relevant to command-line and text-mode die-hards.

Cybec VET_NET v1.0

VET_NET provides single server protection in the form of a console-configurable NLM. It uses the standard *NetWare*-style menu interface which is used by most of the other products, and offers all the features one would expect: scheduled, immediate, and on-access scanning. These are backed up by exclusion lists and multiple configurations. The interface, which is simple and uncluttered, is available on the server console, but *Cybec* says that it can also be configured from the remote console.

Its detection rates are by no means startling, but are not shameful either – they could be easily improved to pull the product into Division One.

Dr Solomon's AVTK for NetWare v7.55

This product has undergone major surgery since the comparative review of one year ago, as documented in its last standalone review [*VB, December 1995, p.17*]. It now has an interface which is much easier to use – this has been added over and above the powerful algorithmic language which was previously the only way to control the product. Server-based on-access scanning has also been added.

The install process is a little curious: it involves running a batch file from the installation disk, the first argument of which is the directory into which the files are to be placed; the second, the version of *NetWare* being run (i.e. 3 or 4). It is puzzling that this cannot be detected, a factor which other products seem to manage somehow. However, the process works, so it seems churlish to complain.

The *AVTKN* provides its features via several NLMs, which are loaded in an order dependent on the specified configuration. The result is a powerful, if fairly confusing, system; however, it does have the advantage that the user does not need to load bits of the system which will not be used.

The only things which some products do, and the *AVTKN* does not, are cross-server features – specifically, domain administration and centralised updating. To be useful in very large organisations with multiple servers, these should be the next features to be added.

Detection is, as expected, very good: a perfect score in the In the Wild set, and only missing one (Cruncher) in the Standard set. Detection against polymorphic samples is also good, but with the same problem experienced by the DOS product (incomplete detection on a few viruses) lowering the result. This is definitely a product to consider seriously.

EliaShim ViruSafe v2.15

Eliashim offers a similar solution to the problem of on-access scanning as that provided by *Sophos'* InterCheck: their InterServer system provides resident software on the client which sends items to the server for checking. However, whereas *Sophos'* version worked without problems, *EliaShim's* caused several hiccups, which eventually made it impossible to provide on-access scan scores.

The default installation of the resident software resulted in a long pause after each command was executed – seemingly identical to that described in the DOS product's standalone review on p.21. The removal of the client machine's SCSI drivers solved the problem, but this should not be necessary.

	In the Wild (286)				Standard (304)				Polymorphic (7500)				Overall (%)	
	On demand		On access		On demand		On access		On demand		On access		On demand	On access
	No.	%	No.	%	No.	%	No.	%	No.	%	No.	%		
CPAV	176	58.0	176	58.0	216	75.4	216	75.4	1872	22.1	1872	22.1	51.8%	51.8%
Cheyenne InocuLAN	271	93.9	271	93.9	274	92.3	274	92.3	4799	59.7	4799	59.7	82.0%	82.0%
Command F-Prot	285	99.0	285	99.0	285	96.0	285	96.0	3836	41.7	3836	41.7	78.9%	78.9%
Cybec VET_NET	270	94.9	269	94.4	292	97.7	291	97.1	4633	53.0	4633	53.0	81.9%	81.5%
Dr Solomons AVTK	286	100.0	286	100.0	303	99.7	303	99.7	6973	83.1	6973	83.1	94.3%	94.3%
EliaShim ViruSafe	240	86.2	n/a	n/a	257	86.3	n/a	n/a	2529	27.0	n/a	n/a	66.5%	n/a
ESaSS ThunderBYTE	286	100.0	286	100.0	302	99.4	302	99.4	6475	74.8	6475	74.8	91.4%	91.4%
H+BEDV AntiVir	233	81.8	233	81.8	282	95.1	282	95.1	3778	47.8	3778	47.8	74.9%	74.9%
IBM AntiVirus	285	99.5	285	99.5	297	97.6	297	97.6	5340	63.4	5340	63.4	86.8%	86.8%
Intel LANDesk	240	84.6	236	82.6	269	90.8	269	90.8	4469	54.7	4470	54.7	76.7%	76.0%
KAMI AVP	270	94.2	270	94.2	292	97.1	292	97.1	6255	74.2	6255	74.2	88.5%	88.5%
McAfee NetShield	285	99.7	281	97.6	286	95.4	286	95.4	4931	64.3	4931	64.3	86.5%	85.8%
Norman Virus Control	280	97.8	280	97.8	298	98.7	298	98.7	6993	88.3	6993	88.3	94.9%	94.9%
Norton AntiVirus	281	98.8	281	98.8	276	93.1	276	93.1	3733	47.3	3233	40.7	79.8%	77.5%
Sophos Sweep	282	98.0	282	98.0	304	100.0	304	100.0	7496	96.6	7496	96.6	98.2%	98.2%
RG Software Vi-Spy NIM	272	95.6	272	95.6	297	98.0	297	98.0	5013	56.8	5013	56.8	83.4%	83.4%

The results as shown above, detailing detection results, reveal that a perfect score overall was achieved by nobody: in fact, many overall scores were somewhat lower than might have been expected. Each and every product had at least one weak area.

Once the product was working correctly, it became clear that an important piece of information was missing from the server log file: it contains the workstation and user IDs and virus name discovered, but not the name of the infected object, an omission which will make the administrator's job considerably harder in the event of a workstation infection.

Finally, and most seriously, it was discovered that issuing two instructions to the client TSR will cause the network server to abend immediately and, in our tests, without fail. In addition, at other points the InterServer client would become unstable and crash the workstation; all of which means that it is impossible to present on-access detection rates for this product.

The package includes a DOS control program which allows multiple servers running *ViruSafe* to be configured from a single workstation. This program is both easy to use and attractive to the eye.

The problems described above, combined with the distinctly unexciting detection rates, especially in the tricky polymorphic test-set, mean that more work needs to be done on the product before it becomes a mainstream contender.

ESaSS TBAV for Networks v6.52

As noted above, this product is not an NLM like most of the others, but a more general network solution – in this case, one machine on the network acts as a virus-checking server, handling requests from *TBAV* clients on other machines with which it communicates via shared data areas.

There is a sophisticated control system on the virus checking server (rather confusingly called the client by the documentation), which allows the administrator to change the configuration of his domain, and to modify files on any client PC running the workstation software. The administrator can even reboot workstations remotely!

TBAVN uses a powerful programming language to create 'procedures' which can then be retrieved and used at will. These can also be used for the scheduler.

The detection rates are what we have come to expect from *ThunderBYTE* – very good. A new heuristic engine has helped to raise the polymorphic score back up; with a little further attention it could be unbeatable once again.

H+BEDV's AntiVir for NetWare v0.97c

AntiVir for NetWare offers a complete set of basic configuration options and features, without the more elaborate domain management and centralised updating features of products such as *CPAV* and *InocuLAN*. Also missing are the whizz-bang GUI configuration editors offered by other products, but the simple and uncluttered console interface is a pleasure to use, and should suffice for most administrators.

Over the past year, the on-line help (previously only available in German) has been translated into English, which eases the review process no end for the linguistically challenged among us…

Regrettably, however, the detection rates do need some improvement, which is strange, since the last time it was reviewed it did very well in this area. However, there is no reason to believe those days are gone forever, and it should be possible to restore the figures to their former glory.

IBM Anti-Virus for NetWare v2.4

Famous in past *VB* reviews, comparative and standalone, for its 'unique' (or peculiar, depending upon your point of view) interface, this product's detection is nonetheless rather good. It shuns the conventional *NetWare* menuing systems, and opts instead for a split-screen approach. The top third of the screen describes the configuration, the middle third the latest event, and the bottom third, the status.

The product is extremely flexible, but it will probably take quite a long time to become an expert with it. However, this should not be regarded as too much of a barrier to purchasing the product, as time thus spent would not be wasted. As regards detection, the product fares very well, detecting very nearly all of the In the Wild set, and missing only a few of the Standard set. Polymorphic detection still leaves a little to be desired, however.

IBMAVN has a clever way of dealing with sudden bursts of file accesses: instead of scanning the requested file immediately, it places it at the end of a queue for later scanning. This will speed up access to the file whilst still ensuring that it is checked.

This technique does not scan the file before the user gets it; therefore, the first person who accesses an infected file has a good chance that his request will succeed (depending on how busy the server is). When the file is finally processed, it may be locked against subsequent accesses, should a virus be found.

Detection rates continue their upward trend of the last few months: nearly all the samples in the In the Wild and Standard test-sets are now detected, and the polymorphic score is creditable, especially on this very tricky set.

Intel LANDesk Virus Protect v3.0

The phrase 'bells and whistles' springs to mind when reading the manual for this one. In fact, 'bells, whistles, and 72-piece symphony orchestra' may be more suitable. It has *NetWare 4* namespace support, *Macintosh* support, GUI control programs, domains, centralised updating, programs to produce pretty graphics showing events of various types, an alert system which supports every notification method up to and probably including smoke signals, and automatic downloading of new virus signatures. It does not appear to make coffee, but that's about the only thing missing…

With all this, it is a shame that the detection rates are not better – if they were, perhaps this would be the product for everybody? However, at fractionally under 85% on the In the Wild test-set, it clocks in third from bottom: a definite 'could do better'. In terms of features, however, this is one of the best.

KAMI AVP for NetWare v1.00

This product is making its first appearance in a *VB NetWare* comparative: previously, *KAMI* has only produced a DOS scanner, which in the last two such *VB* comparatives has conquered all in terms of detection, whilst at the same time it was sadly lacking in terms of speed.

However, for some reason, the NLM product does not display the same impressive detection rate. The virus information files with which the product came are a couple of months old, but no instructions were received to add new ones, so the product was tested as received. Whether or not that is the sole cause of the very poor (in *AVP's* terms) detection rates is impossible to say, but when fixed the NLM should be brought up on a level with the DOS product.

As far as functionality goes, the product is distinctly basic. Its thirteen-page manual describes the features available: on-access, on-demand and scheduled scanning (one job only) is about it, although on-access scanning can use a feature called 'postponed checking', similar to *IBMAVN's* delayed scanning. There are no cross-server or remote administration facilities, and *AVPN* will not find a niche in the market until either its detection matches that of its DOS sibling, or it offers more advanced features.

McAfee NetShield v2.3

Along with *LANDesk Virus Protect*, this product arrived at the *VB* office on a gold CD-ROM, an increasingly frequent occurrence reflecting both the growth in size and complexity of anti-virus products and the decreasing cost of in-house CD-ROM duplication.

As far as detection goes, a near miss (by one) in the In the Wild set is, as expected, the high point. *McAfee's* polymorphic detection continues to improve: against this difficult set, 64% is far from disgraceful.

In terms of features, *McAfee* is improving fast – since last year we have a new *Windows* control program, allowing multiple servers to be examined and configured from one workstation. Cross-server updating is also included – a group of servers can be told to check regularly with each other to find out who has the latest version of the signature database and update each other accordingly. Notification features are also increasing – now broadcast, mail, and pagers are supported.

NetShield also offers several enhancements to basic *NetWare* security, which, whilst not immediately virus related, may well help the network administrator – detailed access monitoring and logging features are the most useful here. Also available is the option to grant temporary access to an area to specific users – if someone needs access to an install system for ten minutes to update software on their PC, *NetShield* can be told to grant them access immediately and remove it after ten minutes.

One minor personal quibble – the documentation, at least for the copy under review, was provided on CD-ROM. This is something which people either think is the best thing since sliced bread, or a terrible idea to be discouraged at all costs. The personal opinion of this reviewer is that documentation should also be included in a printed form.

Norman Virus Control for NetWare v3.5

NVC, which in its *NetWare* incarnation is referred to as *FireBreak*, puts in an impressive performance in this review, perhaps most notably for its second place in the Polymorphic scan test. The In the Wild and Standard scores are almost as impressive.

As far as features go, *NVC* falls neatly into the 'basic but complete' category. There is neither a remote configuration program, nor native *NetWare 4* support, and the only concession to a multiple-server environment is a useful feature whereby one *FireBreak* server can act as a so-called 'communications hub' – an alert clearing house, if you will. In spite of this, *NVC* has a marketplace due to its simple effective approach: the server screen is easy to drive, and the manual is concise and straightforward.

Norton Anti-Virus for NetWare v2.0.2

The main anti-virus product from *Symantec* offers considerably better detection that *CPAVN*: whilst not quite breaking 50% against the polymorphics, it fares well against the In the Wild and Standard test-sets.

Norton Anti-Virus supports *NetWare* Directory Services (the distributed resource database used by *NetWare 4*), and comes with a sophisticated *Windows* control program which allows network administrators to configure multiple servers and domains. Centralised updating is available, and alerts can be transmitted by network broadcasts, MHS and pagers.

Overall, *Norton Anti-Virus* is difficult to classify. Its detection is not yet sufficient to place it at the top of the heap, but its features make it one of the best products tested. The *NAV* developers should perhaps place slightly more emphasis on finding viruses, as this is ultimately their product's *raison d'être*.

Sophos Sweep v2.81

This version does not offer server based on-access scanning (although *Sophos* states that v2.83 does) – instead it uses InterCheck, a system utilising resident software on the client to send files and boot sectors to the server for checking.

On the NLM side, *Sophos* offers a simple-to-control, fully-featured, if somewhat basic, product. The menus are intuitive, and the screen well laid out. Since last year, log file management and viewing features have been added, but the product can still only be controlled from the server console; there are no programs supplied to allow remote administration. Also missing is support for *NetWare 4* Directory Services (present in v2.84) and multiple server administration and updating.

However, the detection figures are, in the Standard and Polymorphic test-sets, the highest in this comparative – an impressive feat. The all-important In the Wild test-set result is marred only by missing the Concept infections – a lack also fixed in v2.83.

This is a product whose impressive detection history looks set to continue, and which is simple to use and administer. The addition of multiple server functionality would bring the product back to the cutting edge of *NetWare* protection.

RG Software Vi-Spy NIM v12.0

This product is another which is not an NLM but a more general network product. Essentially, the system is one allowing the administrator to configure his machines so they will automatically have their versions of *Vi-Spy* run and, if necessary, updated when they connect to the network.

Of necessity, features offered by the product are somewhat sparse when compared to its more exotic competitors. In a heterogeneous network, however, it may well find a home.

The detection rates, whilst unexciting, are creditable enough and, as for so many other products, a little work here and there would do the trick for the trustworthy *Vi-Spy*.

General Conclusions

One very noticeable change over the last twelve months is the convergence of the on-demand and on-access detection rates of the products. This point is discussed in more detail in the next section.

In terms of detection, which is the main thrust of this review, *Sophos' Sweep* comes out a clear winner. On the important In the Wild test-set, only *Dr Solomon's AntiVirus Toolkit for NetWare* and *ThunderBYTE* from *ESaSS* scored top marks.

When it comes to features, the products divide fairly neatly into two categories: those which cater for multiple servers and those which do not. Those which focus on one server now offer broadly the same list of features: it is difficult to separate them in this regard.

Multiple server products are much more in their infancy, so the specifications are quite different across the various products. If you are interested in this marketplace, the section 'How to Choose a Network Solution' will be of special interest to you.

Another division between different types of product occurs when you look at remote administration tools – here too there are the haves and the have nots. Some administrators will consider such a program important, others will not object to using RCONSOLE to get to the server to make configuration changes. The best of both worlds is to allow configuration both remotely and at the console.

Specific Comparisons

As mentioned above, one of the most interesting things about this review is the convergence between on-access and on-demand scan rates. A year ago, these were different for around 50% of the products; now they are the same for most of them. This is a step forward – it is only right that users will receive the same level of protection in all the products' various modes of operation.

Of further interest is the polymorphic test-set. This was updated before testing started with some fairly new (i.e. received from various vendors within the last couple of months) polymorphics, specifically Code.3952:VICE.05, PeaceKeeper.B, Russel.3072, and Sepultura:MtE-Small. These viruses use varied techniques and are of varying degrees of complexity, and help to keep this test-set up to date by enabling it to separate the sheep from the goats in terms of detection.

Such updates are necessary because the percentages for the In the Wild test-set are now almost universally in the high nineties. This can only be a 'Good Thing', and is a result of the emphasis placed by the *NCSA* and other organisations on this aspect of product functionality. It is to be hoped that the upward trend in this area continues, as it is of significant benefit to users.

The Standard set continues to serve as a station-keeping test; a set in which products are expected to score well: it contains nothing tricky with the possible exception of Cruncher.

Cruncher continues to cause problems for anti-virus products: looking at an infected file from the outside, it is almost impossible to tell whether it has Cruncher, or is simply compressed with DIET. To make the determination, the product must work harder; to decompress the file so that it may look inside to be certain.

It is interesting to watch products slowly 'learning' to detect Cruncher – the virus has been around for at least three years now [*see VB June 1993 p.8*], but the trade-off (speed versus thoroughness) is the same for anti-virus developers now as it was then. Until the virus becomes prevalent in the wild (as opposed to the few isolated incidents seen up to now), slow changes in the detection rate of this particular virus are likely to continue.

How to Choose a Network Solution

When it comes to choosing a product with which to protect your corporate network, the basic rules apply more vitally than for choosing a DOS product; there is more at stake.

Whilst *Virus Bulletin* can tell you how good the products are at detecting the virus against which they are tested, and can describe their features and point out any bugs or omissions, it cannot tell you which one is right for you. A third party can only ever inform you which products to avoid, not which ones to choose.

So, how is it done? A wise decision would be to use reviews such as this to narrow the field down to those products which provide an adequate defence. Exactly what this means depends on the situation under consideration – the small corporate, using computers for administration purposes, is probably at less risk than a code shop, for example.

Next, obtain evaluation copies of the chosen products. Anti-virus companies, by and large, are keen to supply evaluation copies of their software: the risk of piracy is low, as without updates such software rapidly becomes close to worthless. If the company is unwilling to supply an evaluation copy, you should ask why.

If you have the resources, set up a small test system to practise administration; and then, in turn, let the products loose on as small a subset of your network as possible. This will allow you to form an opinion of how the product fits in to the way in which your systems work. The best detection around is no use if the product is not being, or cannot be, used correctly.

Nevertheless, a product should not be ruled out simply because it is not the easiest one to use: the investment of time in learning to use a network product will not be wasted. After all, it is to be hoped that the chosen product will continue to be used for several years at least. The most

likely conclusion you will reach at this point is that none of the products precisely fulfil your needs: this is to be expected. In fact, it would be surprising if your dream product existed. Keep in mind the distinction between what you need and what you want, and always remember that this is something which you (as administrator) are going to have to maintain, configure, and support; and which your users are going to have to live with, every working day.

Make a bad decision, and a lot of people are going to be unhappy, not least yourself. Make a good one, and you will be saving yourself a lot of trouble down the line.

Technical Details

Server:

Compaq Prolinea 590 with 16MB of RAM and 2.1GB disk. Version of *NetWare*: 3.12 (CLIB 3.12j); five-user licence; one 500 MB volume.

Workstations:

(1) *Compaq Deskpro 386/20e* with 4MB of RAM and a 540MB hard disk (*Norton* info: Average Seek: 12.16ms, Track to Track seek 2.63ms, Data Transfer Rate: 938.3KB/s), DOS 6.22, *Windows v3.1*.

(2) *Compaq Deskpro 386s* (16MHz) with 2MB of RAM and 82MB disk (*Norton* info: Average Seek: 21.91ms, Track to Track seek 4.34ms, Data Transfer Rate: 781.7KB/s), DOS 6.22, *Windows v3.1*.

All three computers use *3com* 3C509 (revision C) Ethernet cards, and communicate via thin Ethernet using 802.3.

Test-sets

Where more than one variant of a virus is used, the number of samples included is given in parentheses after the virus name.

In the Wild: There are 286 viruses in this set.

_814 (3), Accept.3773 (5), Anticad.4096 (4), Anticad.4096.Mozart (4), Arianna.3375 (4), Avispa.D (2), Bad_Sectors.3428 (5), Barrotes.1310.A (2), BootEXE.451, Bosnia:TPE.1_4 (5), Byway.A, Byway.B, Cascade.1701.A (3), Cascade.1704.A (3), Cascade.1704.D (3), Cawber (3), Changsa.A (6), Chaos.1241 (6), Chill, Coffeeshop (2), Concept (in .DOT), Concept (in .DOC) (3), CPW.1527 (4), Dark_Avenger.1800.A (3), Datalock.920.A (3), DelWin.1759 (3), Die_Hard (2), Dir-II.A, DR&ET.1710 (3), Fairz (6), Fichv.2.1, Finnish.357 (2), Flip.2153 (2), Flip.2343 (6), Freddy_Krueger (3), Frodo.Frodo.A (4), Ginger.2774 (2), GoldBug (4), Green_Caterpillar.1575.A (3), Helloween.1376 (6), Hi.460 (3), Hidenowt, Jerusalem.1244 (6), Jerusalem.1808.Standard (2), Jerusalem.Sunday.A (2), Jerusalem.Zero_Time.Australian.A (3), Jos.100 (3), Junkie, Kaos4 (6), Keypress.1232.A (2), Lemming (2), Liberty.2857.A (2), Little_Brother.307, Little_Red (2), Macgyver.2803.B, Maltese_Amoeba (3), Manzon (2), Markt.1533 (3), Mirea.1788 (2), Natas.4744 (5), Necros (2), Neuroquila, No_Frills.Dudley (2), No_Frills.No_Frills.843 (2), Nomenklatura (6), November_17th.768.A (2), November_17th.800.A (2), November_17th.855.A (2), Npox.963.A (2), One_Half.3544 (5), Ontario.1024 (3), Pathogen:SMEG (5), Phx.965 (3), Predator.2448 (2), Quicky, Sarampo (6), SatanBug.5000.A (2), Sayha (2), Screaming_Fist.II.696 (2), Sleep_Walker (3), SVC.3103.A (2), Tai-Pan.438.A (3), Tai-Pan.666 (2), Tequila.A, Three_Tunes.1784 (6), Trakia.653, Tremor.A (6), Trojector.1463 (6), Vacsina.TP-05.A (2), Vacsina.TP-16.A, Vampiro, Vienna.648.Reboot.A, Vinchuca (3), Virogen.Pinworm (6), VLamix, Xeram (3), Yankee_Doodle.TP.39 (5), Yankee_Doodle.TP.44.A, Yankee_Doodle.XPEH.4928 (2).

Standard: There are 304 viruses in this set.

405, 417, 492, 516, 600, 696, 707, 777, 800, 905, 948, 1049, 1260, 1600, 2100 (2), 2144 (2), 5120, 8888, 8_Tunes, AIDS, AIDS-II, Alabama, Ambulance, Amoeba (2), Amstrad (2), Anthrax, Anti-Pascal (5), Argyle, Armagedon, Athens (2), Attention, Bebe, Big_Bang, Black_Monday (2), Blood, Burger (3), Butterfly.Butterfly, Captain_Trips (4), Cantando.857, Casper, CeCe.1998 (6), Crazy_Lord (2), Cruncher (2), Dark_Avenger.2100.DI.A (2), Dark_Avenger.Father (2), Darth_Vader (3), Datacrime (2), Datacrime_II (2), December_24th, Destructor, Diamond.1024.B, Dir, DiskJeb, DOS_Hunter, Dot_Killer, Durban, EarJob.405 (3), Eddie, Eddie-2.A (3), Fax_Free.Topo, Fellowship, Fish_1100, Fish_6 (2), Flash, Fu_Manchu (2), Genesis.226, Greets.3000 (3), Halley, Hallochen.A (3), HLLC.Even_Beeper.A, Hymn (2), Icelandic (3), Internal, Invisible_Man (2), Itavir, Jerusalem.PcVrsDs (4), Jo-Jo, Jocker, July_13th, Kamikaze, Kemerovo, Kennedy, Lamer's_Surprise, Lehigh, Liberty (5), Liberty.2857.D (2), Loren (2), LoveChild, Lozinsky, Macho (2), Maresme.1062 (3), MIX1 (2), MLTI, Monxla, Murphy (2), Necropolis, Nina, Nothing, NukeHard, Number_of_the_Beast (5), Old_Yankee (2), Oropax, Oxana.710 (3), Parity, Peanut, Perfume, Phantom1 (2), Pitch, Piter (2), Plague.2647 (2), Poison, Polish-217, Power_Pump.1, Pretoria, Prudents, Rat, Revenge, Riihi, SBC, Screaming_Fist.927 (4), Semtex.1000, Senorita.885 (3), Shake, ShineAway.620 (3), Sibel_Sheep/Haifa.Mozkin (2), Sofia.432 (3), Spanz (2), Stardot.789.A (6), Stardot.789.D (2), Starship (2), Subliminal, Sunday (2), Suomi, Suriv_1.01, Suriv_2.01, SVC.1689.A (2), Sverdlov (2), Svir, Sylvia, Syslock, Syslock.Macho (2), Syslock.Syslock.A, Taiwan (2), Telecom (4), Terror, Tiny (12), Todor (2), Traceback (2), TUQ, Turbo_488, Typo, V-1, V2P6, Vacsina.634, Vacsina.Penza.700 (2), Vacsina.TP.? (6), Vcomm (2), VFSI, Victor, Vienna.Bua (3), Vienna.? (11), Virdem, Virdem.1336.English, Virus-101 (2), Virus-90, Voronezh.1600.A (2), VP, Warrier, Warrior, Whale, Willow, WinVir_14, Yankee_Doodle.TP.? (5), Zero_Bug.

Polymorphic: These are 500 samples of each of the following 15 viruses in this set.

Code.3952:VICE.05, DSCE.Demo, Girafe:TPE, Groove and Coffee_Shop, MTZ.4510, Neuroquila.A, Nightfall.4559.B, One_Half.3544, Pathogen:SMEG, PeaceKeeper.B, Russel.3072, Sepultura:MtE-Small, SatanBug.5000.A, SMEG_v0.3, Uruguay.4.

On Reaching the Results:

The test results for this review are calculated in the same way as for the last DOS Scanner comparative. The polymorphic test-set score includes a weighting that favours those products which manage complete detection of one group of samples. In the Standard and the In the Wild test-sets, the scores are first normalised so that each group of samples of a particular virus contributes the same amount to the final percentage. The overall score is made up of a simple non-weighted combination of the three subsidiary scores.

For information on exactly how the percentages have been calculated, readers are advised to refer to the articles on testing protocols for DOS scanner comparative reviews which were published in *Virus Bulletin*, issues February 1995 (p.12) and November 1995 (p.14).

PRODUCT REVIEW

ViruSafe

Dr Keith Jackson

ViruSafe is an anti-virus package incorporating component programs to scan for viruses, remove viruses from infected files, detect unknown viruses, and calculate and verify file checksums. A memory-resident program is included.

I last reviewed *ViruSafe* for *VB* in April 1990; too long ago to draw meaningful conclusions about how *ViruSafe* has changed. However, I was intrigued to note in the original review that 'Within a matter of months, *ViruSafe* has gone from having knowledge of only 6 viruses … to a total of 79 viruses'. Oooooh! Life was simpler in those days.

Installation

ViruSafe was provided for review on two 1.44MB (3.5-inch) floppies; one for DOS, one for *Windows*. Installation was straightforward, and placed 3.19 MB (39 files) on my test computer's hard disk, the DOS components being installed first. After requesting the drive and subdirectory location in which to place the *ViruSafe* files, the installation program initiates a scan. If all is in order, the *ViruSafe* files are copied to hard disk, each file described and named onscreen.

After this is complete, the *Windows* components may be installed. This install program requests a subdirectory location, and offers to change AUTOEXEC.BAT to check memory, test for file changes once a day, and load the memory-resident component of *ViruSafe*, with full tables or monitoring for the most common viruses. I chose 'full tables', which gobbles up more conventional memory but provides a higher virus capability.

ViruSafe now says it will 'mark' all programs on my hard disk. That sounds like inoculation (adding anti-viral executable test code to extant executables) – I found out later that it wanted to create a database file in each subdirectory. Such hidden files are irritating for the user: a tidy program would maintain its database in its own subdirectory; however, this could leave it open to malicious attack.

The install program offers to make an 'Emergency Rescue Disk' (this can be done later), and requests the user name and company, and entry of *ViruSafe's* serial number. The program stated that the serial number must be 'one letter & 6 digits' to be valid, but though that of the product provided contained only one letter and five digits, nothing complained.

Documentation and Help

ViruSafe's documentation is a 240-page long A5 book. Each component is described competently, and the appendices contain a useful list of 'Frequently Asked Questions' and an explanation of the error messages. However, I doubt that *ViruSafe* only has six unique error messages, and would recommend that this section is completed. Very helpful is a fifteen-page appendix explaining how to cope with various pieces of hardware or software which may adversely affect installation. The manual has several minor errors, the most amusing of which is on the outside back cover, where the developers state that *ViruSafe* works with PCs running under *Windows 6*!! Do they know something we don't?

The *Windows* help file had me foxed at first – it simply displays a picture of the main *ViruSafe* window. It turns out that you can select the option on which you want help. Whilst trivial with a mouse, this is tricky with only the keyboard – the selection order is non-obvious.

ViruSafe claims knowledge of 2194 viruses; 5046 mutations. The documentation does not define clearly what it means by a mutation, so I am not quite sure what the figures mean.

Using ViruSafe

A DOS program called VSMENU is provided which permits integrated execution of *ViruSafe's* components. It is easier to use *ViruSafe* via VSMENU than to determine from the documentation how to operate individual components. This is not meant to knock the documentation; it is a testament to the ease with which even beginners should take to VSMENU.

A *Windows* program is offered which can control all the functions provided by the suite of programs. Despite the fact that *ViruSafe's Windows* components had been properly installed, nothing was visible when *Windows* was executed. This was at least partly due to the fact that *ViruSafe* put the file VIRUSAFE.GRP in its own subdirectory, where *Windows* probably could not find it.

The manual explains that installation should have modified PROGRAM.INI and WIN.INI. However, a visual inspection showed that neither had been modified, and a short session with a text editor was needed before things were sorted out.

Although the *ViruSafe Windows* program is easy to use, it is difficult (perhaps impossible) to operate fully without a mouse. This is unfortunate for laptop owners, where keyboard-only operation is still fairly common.

ViruSafe installs two files in the root directory of my hard disk (VSCHK.COM and VSBOOT.VS) – this I did not like: I want to decide where files go on my computer. *EliaShim's* reasoning is that files thus placed are easier to find when *ViruSafe* is run from a floppy after a clean boot.

In addition, its introduction to my PC meant that after any program finished execution there was a several second delay before the command prompt returned. During the boot

process, this meant that the test computer went off for a fifteen-second think at several points, making booting intolerably slow. I'll explain why it happens later.

Scanning Speed

It is impossible to provide a single statement of how fast *ViruSafe* can scan a disk. The only fair answer for a product with such a multitude of options is 'It depends'.

In default mode, *ViruSafe* for DOS checked my test computer in 1 minute 18 seconds. The *Windows* version performed the same task in 2 minutes 26 seconds. The hard disk contained 12.9MB of executable programs across 253 files. Scan time increased to 1 minute 27 seconds when the 'check and remove' option was used, even though no viruses were on the hard disk. Checking all files on the hard disk (24.5MB across 675 files) increased scan time to 2 minutes 26 seconds.

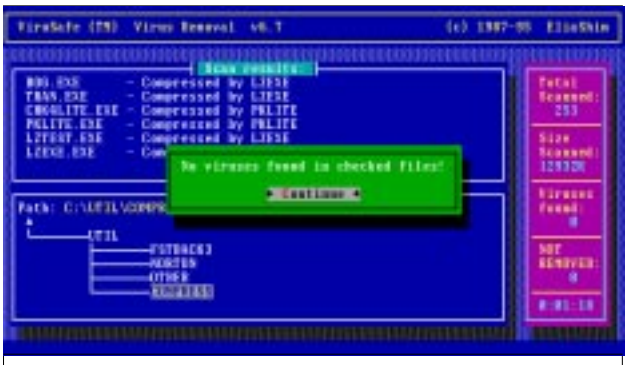
ViruSafe has three scanning options: Turbo, Quick and Normal. The default setting is Quick – using the Turbo option (scanning 'for common viruses only') reduced scan time to 1 minute 5 seconds. Both Quick and Normal scanned at the same speed (the default speed described above).

Comparative results of scanning speed are the only fair way to measure how fast a scanner can operate, thus I compared the above scanning times with two other well-known scanners. In comparison, *Dr. Solomon's AVTK* scanned the hard disk of my test computer in 4 minutes 16 seconds, and *Sophos' Sweep* required 3 minutes 23 seconds for the same scan. When all files were scanned, scan times increased to 6 minutes 28 seconds and 6 minutes 16 seconds respectively.

Memory-resident Software

The impact of memory-resident software on PC operation is difficult to quantify. I copied 40 files (1.25MB) from one subdirectory to another – this normally took 23.6 seconds, but rose to 25.9 seconds with the memory-resident software.

This test misses the real impact of *ViruSafe's* memory-resident software. As mentioned, it causes a delay before control returns to the operating system after any program is executed. Of the options available, I traced this delay to 'Checking known Viruses' – i.e. scanning. By comparison, the overhead introduced by any other option was almost trivial. This could be due to almost anything, but the fact remains that with it the product was practically unusable.



ViruSafe for DOS communicates with the user via a variety of windows and menus, making it reasonably easy to use.

In its default state, the *ViruSafe* memory-resident software occupies 10.3KB of memory. Many tailoring options are available: if the default choices are altered, the amount of memory required will change correspondingly.

Scanner Detection

As with scanning speed, because there are so many options available, the detection capabilities provided by *ViruSafe* are difficult to express in a few words. This is due to the myriad setup options, many of which affect virus detection.

ViruSafe's DOS version did not recognise the Magneto-Optical drive on which the virus test-sets were stored (the *Windows* version had no such problem). The polymorphic test-sets are too big to be copied to hard disk, so these viruses could only be tested using the *Windows* version: this would not make a log file of viruses detected, and only reported to the screen. It was therefore impossible to get at the fine details of polymorphic virus detection.

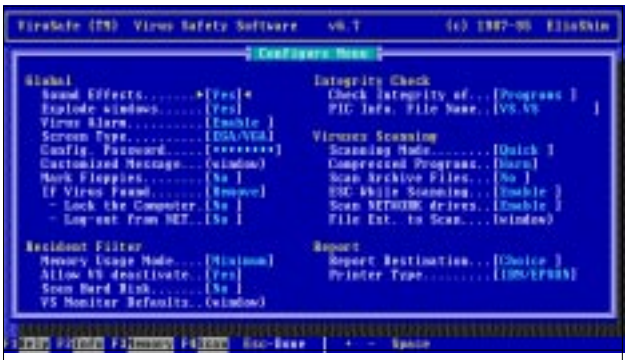
When the DOS version of the product was run against the In the Wild test-set, it detected 202 of the 286 test samples (71%) using default settings. Against the Standard test-set, again using default settings, *ViruSafe* detected 177 of the 265 samples (67%). After detecting these, *ViruSafe* recommended that all files should be checked, but when I used the 'Check and Remove' option on all files the error message 'Too many viruses found in a single source' was displayed. Checking terminated after 115 viruses were detected.

When the *Windows* version was tested against these test-sets, it found 280 and 286 viruses respectively. This is more than the DOS version, and in the case of the Standard test-set meant *ViruSafe* was detecting 286 viruses in 265 samples. Amazing. A closer inspection of onscreen logs explained the anomalies: *ViruSafe* had found more than one virus in several samples. In the worst case, *ViruSafe* thought that four samples were each infected with three different viruses.

The detection rate of over 100% occurred when the *Windows* version was in Normal mode. *EliaShim* agrees that it only happens in this mode: in Quick or Turbo mode, the In the Wild and Standard test-sets were thought to contain 242 and 236 viruses respectively. *ViruSafe* detected nineteen of the twenty boot sector samples – it missed only Urkel.

When tested against the polymorphic samples, *ViruSafe* reported that it could detect 2435 of the 5500 test samples (49%). Closer examination of the onscreen log showed that it detected most of just four types of polymorphic virus (Girafe:TPE, Groove and Coffee_Shop, One_Half.3544, and Pathogen:SMEG), plus just one sample of Nightfall.4559.B.

This does not make sense. There are only 500 test samples of each type of polymorphic virus. Even assuming 100% detection of the test samples listed, *ViruSafe* cannot possibly



ViruSafe has many different configuration options, allowing scanning to be configured to suit.

detect more than 2001 viruses. The only sensible explanation is that *ViruSafe* is detecting more than one infection of the same polymorphic virus in some of the test samples.

ViruSafe seems to detect a variable number of viruses depending upon the configuration settings used. Does the product's habit of over-detecting viruses mean that some of the previous *VB* comparative reviews for this product have over-estimated what is already a poor detection rate?

Correlation

ViruSafe includes a method of testing for viruses (be they known or unknown) called 'Correlation'. The manual explains this succinctly, and as I cannot think of a better way to explain what this does, I shall let it speak for itself:

'Correlate compares a sample of executable files on any path defined by the user … Correlate reads an equal number of bytes from each file and compares them using an intelligent algorithm. If similarities are found on the files, the probability is high that a virus is present.'

The manual explains that files created by the same compiler may exhibit correlation, and give a false alarm: this happened when I tried the feature on my test computer. 'Correlation' was reported between three *Stacker* programs, and three programs used by my Magneto-Optical drive. Such a high level of false alarms is problematic, as it can take longer to confirm an incident is a false alarm than to eradicate a virus.

Correlation has some problems. On executing the DOS version, all went well, until after twenty seconds an error message 'Runtime error #204 at 6125:1646. Please contact your dealer' appeared, and the program hung. I rebooted and tried again: the program functioned properly. The DOS version took 59 seconds to check my hard disk, whilst the *Windows* version took 1 minute 2 seconds for the same task.

Analysis and Integrity

Another feature 'analyses' executable files and 'searches for specific virus codes using heuristic analysis'. The DOS version took 1 minute 4 seconds to check through the files on the hard disk of my test computer, and the *Windows* version required 2 minutes 58 seconds. Both found nothing. I have no idea why the *Windows* version takes so much longer to carry out the same test as the DOS version.

ViruSafe's last major feature calculates and verifies checksums for all files stored on a disk. As mentioned earlier, *ViruSafe* first calculates the checksums when it offers to 'mark' all files on a disk; it then places a small database file into each subdirectory. Exactly how the checksums are calculated is not stated in the documentation: this says simply that *ViruSafe* uses 'encrypted digital signatures'.

The DOS version of *ViruSafe* took 58 seconds to 'mark' 252 files on my test computer, but the checksums could be verified in just 49 seconds. It took fifteen seconds to remove the integrity protection. The *Windows* version of this feature does not state how long it takes to carry out the same tests.

The Rest

ViruSafe is able to remove viruses from infected files, but in common with my usual stance, this feature has not been reviewed. A program called TIMERUN.EXE is included (for both DOS and Windows) which can execute programs automatically at preset intervals.

Beyond the features described in this review, *ViruSafe* lets the user manipulate/inspect/repair boot sectors, produce a memory allocation map, include new virus signatures and execute test programs – too many to describe individually.

Conclusions

ViruSafe is quick at scanning, and offers several methods to detect known or unknown viruses. I am not sure users will understand the differences between the methods incorporated in this product, some of which may produce false alarms.

The detection capability offered by *ViruSafe* could be much improved, especially as far as polymorphic viruses are concerned. Overall, *ViruSafe* suffers from many niggling problems – basic testing would have solved most of these. In summary, *ViruSafe* is a somewhat confusing product which is very full-featured but would benefit from further testing.

Technical Details

Product: *ViruSafe, v6.7*, serial number V22427.

Developer/Vendor: *EliaShim Microcomputers Inc*, 22 Ha'Ashlag Street, Haifa 31091, Israel. Tel +972 4 872 8899, fax +972 4 872 9966. WWW: http://www.eliashim.com/.

Price: Dependent on region; ranges from US$99 to US$250. Available with quarterly or monthly updates. New virus signatures are posted to company BBSs every two weeks.

Hardware used: A *Toshiba 3100SX*; a 16 MHz 386 laptop computer with one 3.5-inch (1.4MB) floppy disk drive, a 40MB hard disk and 5MB of RAM, running under *MS-DOS* v5.00 and *Windows v3.1*.

Viruses used for testing purposes: A list of all viruses used for this review can be found in *VB*, January 1996, p.20. For a complete explanation of each virus, and the nomenclature used, please refer to the list of PC viruses published regularly in *Virus Bulletin* (pp.4-5).

SUBSCRIPTION RATES

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, Oxfordshire, OX14 3YS, England

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email editorial@virusbtn.com
CompuServe address: 100070,1340

US subscriptions only:

June Jordan, *Virus Bulletin*, 590 Danbury Road, Ridgefield, CT 06877, USA

Tel +1 203 431 8720, fax +1 203 431 8165

END NOTES AND NEWS

Two-day courses in investigating **computer fraud and misuse** using forensic techniques will be held in Australia (Sydney 15/16 April 1996, Melbourne 17/18 April, and Brisbane 22/23 April), presented by *Network Security Management Pty* in association with *IBC Conferences Australia*. The speakers will include *Virus Bulletin* founding editor Edward Wilding. For information, contact *IBC Conferences* on Tel +61 2 319 3755, fax +61 2 699 3901, email IBCC@geko.com.au.

On 15/16 April, and 13/14 May 1996, *S&S International* is presenting **Live Virus Workshops**. The two-day courses will be held at the *Hilton National* in Milton Keynes, Bucks, UK. The company has also announced two new developments: its Toolkit now scans files compressed in *MS Expand*, and a clean bootable DOS disk, called the 'Magic Bullet', has been added to the Toolkit for use in emergencies. Details from the company: Tel +44 1296 318700, fax +44 1296 318777.

First.Base is hosting a series of IT security and Internet workshops in Sussex, UK, throughout the next three months. Sessions will include **Internet security (incorporating defence against viruses)** and disaster contingency planning. Information can be obtained from *First.Base* on Tel +44 1903 879879, fax +44 1903 879274.

The next rounds of **anti-virus workshops presented by *Sophos Plc*** will be held on 22/23 May 1996 at the training suite in Abingdon, UK. The two-day seminar costs £595 + VAT; one single day, £325 + VAT (day one: Introduction to Computer Viruses; day two: Advanced Computer Viruses). The company has also launched InterCheck client support for *Windows 95*. Contact Julia Line on Tel +44 1235 544028, fax +44 1235 559935 for details on either.

The release of ***The Enforcer v3***, the disk authorization package from *Precise Publishing Ltd*, has been announced. Features include a utility to identify and expand compressed files (allowing scanners access), real-time protection, and *Windows 95* support. [*Watch for a review in the near future. Ed.*] Also, the company is holding more Live Virus Workshops (15 May, 12 June, 17 July). Details on either are available from the company; Tel +44 1384 560527, fax +44 1384 413689.

Symantec Corporation has released a new version of ***Symantec Anti-Virus for Macintosh***, with patches giving 'comprehensive detection and repair' for the *Microsoft Word* Macro viruses. Users can download the patch from the company BBS (+1 541 484 6669), or via the Internet – ftp://ftp.symantec.com/public/mac/sam).

Reflex Magnetics has several courses coming up: **Live Virus Experiences** (12/13 June, 9/10 October), The Hacking Threat (10-12 April, 24-26 July), Internet Security and Firewalls (30 May, 22 July), and DTI Security Codes of Practice (31 May). For further information, contact Rae Sutton: Tel +44 171 372 6666, fax +44 171 372 2507.

From 3–5 June 1996, the *Computer Security Institute* (*CSI*) will be sponsoring ***NetSec 96***. The conference, to be held in San Francisco, will focus on security issues, problems, and solutions in networked environments. Further details, and a free catalogue, are available from the *CSI* via email at csi@mfi.com, or Tel +1 415 905 2626, fax +1 415 905 2218.

SecureNet 1996 will be held at the London Olympia hotel (UK) from 30 April–2 May 1996. Topics covered will include **network viruses, firewalls, risk assessment, and email security**. For more information, contact Alex Verhoeven on +44 1865 843654, fax +44 1865 854971, email a.verhoeven@elsevier.co.uk.

The first anti-virus scanner for *Solaris* has been launched by *McAfee*: ***ViruScan for Solaris*** is *McAfee's* 'latest move to upsize its product line' to support all network and security platforms. For further information, access the company web page at http://www.mcafee.com/.