ISSN 0956-9979

OCTOBER 1997

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

Editor: **Nick FitzGerald**

Editorial Assistant: **Francesca Thorneloe**

Technical Editor: **Jakub Kaminski**

Consulting Editors:

Ian Whalley, Sophos Plc, UK
Richard Ford, IBM, USA
Edward Wilding, Network Security, UK

IN THIS ISSUE:

• **Virus goes ape:** With so much recent attention on macro viruses do not forget the lessons from the past. A simple boot sector virus caught people napping last month – read about Baboon on p.3.

• **Another direction:** The editor’s comments about macro virus up-conversion in August provoked a concerned response from well-known anti-virus researcher Vesselin Bontchev. Make up your own mind, starting on p.14.

• **Macro a-go-go(-go!):** Our three virus analyses focus closely on macro viruses this month. Two employ new polymorphic techniques, but our first analysis examines a virus with an innovative approach to infecting Microsoft Word documents; these articles start on p.6.

CONTENTS

EDITORIAL

Marketing Madness

Deliciously ambiguous title that. Do I mean ‘madness in marketing’ or ‘selling crazy ideas’?

We all know about the Good Times email virus hoax, and its bedpals Irina, Join The Crew, Penpal Greetings and the like. Apparently *McAfee* do not – at least, not those in their German offices. In September, in a fax titled ‘Viren im Internet’, a *McAfee GmbH* representative warned recipients of three viruses currently being distributed by email.

Opening messages entitled ‘Join The Crew’, advised the fax, wipes everything from your hard drive. Reading ‘Penpal Greetings’ messages infects your boot sector and the ‘virus’ forwards copies of itself to everyone in your email address list. ‘Returned or unable to deliver’ attaches itself to your computer components, rendering them useless.

“ Good stuff – except that it is all untrue ”

Good stuff – except that it is all untrue. As it was possibly a marketing move, the latter may not seem that surprising. However, not only is it untrue, it is impossible (as described) with existing email technology. The cynics amongst you will be saying ‘but as it was possibly a marketing move…’. So, was it a marketing move?

Clearly!

After warning of all this supposed email-borne mayhem, the fax went on to push McAfee web and email scanning products as the solution to the problems posed by such ‘threats’.

Given that the Penpal Greetings and Join The Crew email virus hoaxes are almost as widely known *for being hoaxes* as the venerable Good Times, it seems incomprehensible that *anyone* working for an anti-virus company could have been fooled into genuinely believing the latest widespread email virus hoax message. Currently, the three-tier hoax omnibus sandwiching the messages from the *McAfee* fax is the one about which I receive most enquiries.

But maybe this is not so surprising, after all. We are talking about *McAfee*, the company whose advertising claims about their network management products prompted *InfoWorld* to write earlier this year ‘…we’re wondering who exactly is working over there that could be guilty of both grossly misrepresenting our results and coming up with such weaselly fine print (which, it turns out, is wrong)’.

Of all the complaints I have received at *VB* about the ‘creative’ use of our test results in marketing or advertising, more are directed against *McAfee*-originated material than against any other vendor.

The *McAfee* advertisement that spurred *InfoWorld’s* test centre staff to write their opinion piece ignored other products that were included in the original test, added results for a product that was not tested and attempted to cover this sleight of hand with small print disclaimers. The *InfoWorld* staff were so niggled they concluded with the statement: ‘And were we a company rooted in the anti-virus market, we certainly wouldn’t do anything that would bring our ethics into question.’

In light of its apparent track record, and particularly this latest incident from *McAfee GmbH*, maybe it is time for *McAfee* to take a serious look at what its marketing department is up to!

Another ethical issue this month arises from the rather reactionary response from a major anti-virus vendor’s senior spokesperson to the news that Michael Ellison will be speaking at *VB’97*. The front page of a UK computer magazine featured the usual, preconceived judgement of a virus writer, glibly repeated by this spokesperson. This ‘you could never trust a virus writer’ attitude ensures that the anti-virus stalwarts will never actually talk to any reformed virus writers, so they will never be in a position to make an informed decision. It won’t, however, prevent them mouthing the platitudes they feel the rest of the industry expects to hear.

I’m looking forward to meeting those of you who make it to *VB’97*.

NEWS

Patently Obvious

Just when you thought it was safe to design an email virus scanner, news breaks of yet another suit claiming software patent infringement by extant email scanner products. In mid-September, *Hilgraeve Inc*, possibly best known for its widely-distributed *Windows* and *OS/2* communications software (e.g. *HyperTerminal* bundled with *Windows* 95 and *NT*), filed suits against *McAfee* and *Symantec*, alleging these companies' email scanners infringe its in-transit anti-virus patent, granted in 1994. *Hilgraeve's* patent covers searching for virus signatures in data that is being copied between media to inhibit virus infection automatically, such as in files being downloaded from the Internet.

Hilgraeve claims to have licensed *IBM* under its patent and that *McAfee* and *Symantec* failed to respond to licence offers in 1996. The patented process is at the heart of the so-called HyperGuard technology included in *Hilgraeve's* own communications packages since the early 1990s

Simian Says

With all the recent attention on macro viruses and Internet-borne threats, it can be easy to overlook the older, perhaps more mundane, forms of computer virus. You should not, of course. Just because *VB* has been full of talk about macro viruses does not mean you should forget the sometimes hard-learned lessons concerning boot and file infectors.

A recent outbreak of a very simple, and hitherto unknown boot sector infector, Baboon, has been a salutory reminder that basic anti-virus procedures should still be followed, even in the age of macro viruses. Baboon is a very run-of-the-mill boot infector which, nevertheless, successfully infiltrated several sites across Europe, with a few isolated outbreaks reported from the US and South America.

Despite its simplicity and ease of prevention, Baboon still triggered its disk-trashing payload on several hundred computers worldwide, on 11 September

Harold Highland

As this issue was going to print we received the sad news that Professor Harold Joseph Highland, FICS, FACM had passed away peacefully in his sleep. Well-known in the computer security field, he was a respected teacher, researcher and speaker. Attendees at the *VB'95 Conference* in Boston will remember his fascinating keynote address. He was a long-standing member of our Advisory Board and was interviewed in an Insight column in *VB*, January 1996. Professor Highland was the founding editor of *Computers & Security* and the author of several hundred technical articles and 27 books. The self-professed dinosaur of computing will be sadly missed

Prevalence Table – August 1997

Virus	Type	Incidents	Reports
CAP	Macro	145	28.5%
Concept	Macro	51	10.0%
NPad	Macro	39	7.7%
Dodgy	Boot	26	5.1%
Parity_Boot	Boot	24	4.7%
Form	Boot	21	4.1%
AntiEXE	Boot	19	3.7%
Temple	Macro	16	3.1%
Laroux	Macro	15	3.0%
Wazzu	Macro	14	2.8%
Junkie	Multi	11	2.2%
Ripper	Boot	10	2.0%
Showoff	Macro	9	1.8%
AntiCMOS	Boot	7	1.4%
Appder	Macro	7	1.4%
Empire Monkey	Boot	7	1.4%
Cebu	Macro	5	1.0%
WelcomB	Boot	5	1.0%
NYB	Boot	4	0.8%
Johnny	Macro	3	0.6%
LBB Stealth	File	3	0.6%
Lunch	Macro	3	0.6%
Sampo	Boot	3	0.6%
Switcher	Macro	3	0.6%
Bandung	Macro	2	0.4%
Cruel	Boot	2	0.4%
DMV	Macro	2	0.4%
Feint	Boot	2	0.4%
Galicia.800	File	2	0.4%
Helper	Macro	2	0.4%
HLL.5850	File	2	0.4%
Imposter	Macro	2	0.4%
MDMA	Macro	2	0.4%
One_Half.3544	Multi	2	0.4%
Paycheck	Macro	2	0.4%
Swlabs	Macro	2	0.4%
V-Sign	Boot	2	0.4%
Others[1]		32	6.3%
Total		508	100%

[1] Comprising one report each of: Brother.276, Cascade.1661, Colors, CountTen, DamnFog.1748, Date, Delwin.1759, Demon, Diablo, DZT, Edwin, Havoc, Int10, Int12, Int40, Irish, Jumper.B, Maniak, Manic.2143, Manzon.1484, MME.969, Muck, Nop, Obscene.2374, Rapi, Rehenes, Shell.10634, Stoned.A, Stoned.Angelina, Tentacle, Urkel, and Xxx.1060.

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 15 September 1997. 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	**M**	Infects Master Boot Sector (Track 0, Head 0, Sector 1)
D	Infects DOS Boot Sector (logical sector 0 on disk)	**N**	Not memory-resident
E	Infects EXE files	**P**	Companion virus
L	Link virus	**R**	Memory-resident after infection

Babe.1584 **CER:** An appending, 1584-byte virus which intercepts the keyboard interrupt (Int 16h) and swaps some characters (G with O, H with P, I with Q, K with M, S with R). The virus' 'Are you there?' call, AH=2Ch, BX=B0B0h, CX=BABEh Int 21h returns BX=BABEh and CX=B0B0h.

```
Babe.1584           B42C BBB0 B0B9 BEBA CD21 81FB BEBA 754B 81F9 B0B0 7545 2E83
```

Burglar.833 **ER:** An appending, 833-byte virus which recognizes already infected programs by checking two bytes at offsets 0012h and 0014h ([0012h] XOR [0014h]=74h).

```
Burglar.833         56B4 7080 C404 B0F0 86E0 CD21 0BC0 7451 BF74 7490 4F90 9075
```

BW.327 **CN:** An encrypted, appending, 327-byte direct infector containing the texts '[BW]' and '*.COM'.

```
BW.327              BB?? ??BE A800 2E81 07?? ??83 C302 4E75 F5??
```

Dec2nd.1167 **CER:** An appending, 1167-byte virus containing a payload that triggers on 2 December. Instead of infecting a new file, the virus reads the contents of the hard disk and copies track after track to the video memory (first VGA page at B8000).

```
Dec2nd.1167         E8D7 FDE9 F0FE B8DD E7CD 213D E7DD 7438 1E8C C048 8ED8 B88F
```

Eddy.1316 **CER:** A stealth, prepending, 1316-byte virus. Infected files have their time-stamps set to 62 seconds.

```
Eddy.1316           B0FF B40F 86E0 90CD 213D 0101 7434 B821 3590 CD21 2681 3E0A
```

Hideous.1024B **CEN:** An appending, 1024-byte virus with a payload triggering on the eleventh of every month, which overwrites the contents of the first hard disk.

```
Hideous.1024B       B440 B900 0483 06C8 0201 CD21 803E D802 0074 0C32 C0E8
```

Hideous.1024D **CEN:** An appending, 1024-byte virus containing the text 'Minosse'. This variant is sometimes called .1024C and the variant mentioned in *VB* (July 1997) '.1024D'. The payload triggers on 25 March.

```
Hideous.1024D       B440 B900 0483 0613 0001 CD21 803E 2300 0074 0C32 C0E8
```

HLLO.4592 **CEN:** An overwriting, 4592-byte virus containing the texts '*.COM', 'COMMAND.COM', '*.EXE' and 'I don t want to run today... ...because I m DeSTROYED forever!!!'.

```
HLLO.4592           BF68 461E 57BF 0C33 1E57 B8F0 1150 31C0 5050 9A74 0865 00BF
```

ILoveDOS.3622 **CER:** A stealth, encrypted, 3622-byte virus containing the texts 'IOSYS', 'COMSPEC=', 'EXECOMP' and 'AVG AVGW AVGSYS SCAN CLEAN ASTA VSAFE MSAV TURBO WIN BP TRAP TBAV TBDRIVER VCOP GUARD VS 286 386 CHKDSK'. The payload modifies the CMOS data.

```
ILoveDOS.3622       88A4 4E01 8D9C 4F01 8A84 4D01 2807 D0C0 43E2
```

IVP.426 **CN:** An encrypted, appending, 426-byte direct infector containing the texts '^*^ OZZY ^*^ –Yeew duh-meE iTs ReeEl!! [kR]' and '*.com'.

```
IVP.426             8D9E 0E01 B984 012E 8A27 2E32 A692 022E 8827 43E2 F2C3
```

IVP.924 **CEN:** An encrypted, 924-byte direct infector containing the texts 'StealthHacker2 Virus-=SH=-I,the almighty StEaLtH hACkeR, ..FUK VIVEK MUNDKUR!!', 'SH LIVES!', '*.com' and '*.exe'. All infected files have the word 5348h ('HS') at offset 0003h (COM) and 0010h (EXE).

```
IVP.924             2E8A A6A0 048D B61A 01B9 7303 2E30 2446 E2FA C3??
```

Konkoor.1844B **CR:** A prepending, 1844-byte virus; a minor modification of the Konkoor.1844A variant. It contains the encrypted texts 'C:\DOS', 'Bad command or file name', 'C:\', '*.COM' 'Warning !!!', 'KONKOOR Must be destroyed !', 'Sig: -= The Real Computer Scientists =-' and 'KONKOOR V1.50 — Date: 26 September 1993'. The payload, triggering on the sixteenth of every month, encrypts the contents of the first physical hard disk and destroys CMOS data. Both variants can be detected with the same template.

```
Konkoor.1844B       B9BE 0090 0205 8A25 FECC 8825 47E2 F53C 8675 03B9 9A02 5F58
```

LaLiberte.224 **CR:** An appending, 224-byte virus containing the text 'La Liberte'. The string can be found at the end of the virus code. Infected programs start with NOP (90h).

```
LaLiberte.224       3D00 F573 21B8 0040 B9E0 00BA 0002 CD21 7214 B800 4233 D233
```

Leprosy.792C

CEN: An encrypted, overwriting, 792-byte virus containing the texts ‘*.EXE’, ‘*.COM’, ‘Program too big to fit in memory’, ‘Acid Virus’ and ‘Legalize ACiD and Pot -By: Copyfright Corp-’. It also contains a text-art picture of a skull.

```
Leprosy.792C        E801 00C3 BB32 018A 2732 2606 0190 8827 4381 FB4A 047E F0C3
```

Leprosy.797

CEN: An overwriting, encrypted, 797-byte virus containing the text ‘Wouldn’t it suck if bytes were like brain cells? Hello... I would like to introduce myself. I am the DOSE-A virus, (C)1993 by DIMWIT! Your computer is now tripping its balls off, on my deadly psychedelic venom. Wouldn’t it suck if bytes were like brain cells? If they were, this computer would never think again.. -=< Have a nice day... DIMWIT! >=-’.

```
Leprosy.797         E801 00C3 BB32 018A 2790 3226 0601 8827 4381 FB4F 047E F0C3
```

Leprosy.808M

EN: An encrypted, overwriting, 808-byte virus containing the text ‘sKISM rYTHEM sTACK vIRUS-808. sMART kIDS iNTO sICK mETHODS dONT ALTER THIS CODE INTO YOUR OWN STRAIN, FAGGIT. hr/sss nycITY, THIS IS THE FIFTH OF MANY, MANY MORE.... yOU SIS’.

```
Leprosy.808M        BB40 018A 2F32 2E03 0190 882F 83C3 0181 FB68 047E EE59 C390
```

Moran.2720

CER: An encrypted, appending, 2720-byte virus containing the texts ‘——————hello! moran! ——————’, ‘————————welcome to the world——————’, ‘command.com’, ‘I LOVE YOU!’ and ‘love’. All infected files have the plain-text word ‘love’ at the end of the code.

```
Moran.2720          B93B 0931 0C31 1446 4942 0BC9 75F5 C390 8BDC 36C4 4706 FA33
```

Morganism.470

CN: An appending, 470-byte slow infector, containing the texts ‘*.com’ and ‘You have been infected by a living MORGANISM’. All infected files have the byte 60h at offset 0003h.

```
Morganism.470       BF00 008A B6D9 0230 B39F 0247 E2F9 C38B 86D5 0289 862C 038B
```

MPCA.442

CN: Three, encrypted, appending, 442-byte viruses with different texts: 442A ‘[SCHRUNCH]’, ‘[pAgE]’ and ‘*.com’; 442B ‘[ZEB(C)1992]’, ‘[ranger]’ and ‘*.com’; 442C ’[MPC]’, ‘[Michael]’, ‘Michael’ and ‘*.com’.

```
MPCA.442 (A, B)     B9D6 00BF ???? 8135 ???? 4747 E2F8
MPCA.442C           B9D6 00BB ???? 8137 ???? 4343 E2F8
```

MustDie.1207

CER: An appending, 1207-byte virus containing the text ‘MUSTDIE v.1 (a) <xxxxxxx x.x. must die> xxxx(C)1996-97’ (x denotes characters in Cyrillic). Infected files have their time-stamps set to 62 seconds.

```
MustDie.1207        E83D 00C3 BA00 01B9 B704 B440 E80B FEC3 B802 4233 C933 D2E8
```

NanJing.2976

ER: Two new, appending, 2976-byte minor variants containing the same text ‘!!!—<<WELCOME>>—!!! *.*.*.* Don’t panic I’m harmless! You have a new friend: TY_VIRUS V3.0 I hope I haven’t inconvenied you! (c)copyright 1994.3.25 Made by: NANJING NORMAL UNIVERSITY COMPUTER DEPARTMENT OK! Please wait a minute...’.

```
NanJing.2976B       2EFF 1E4C 00B4 FFCD 2180 FC00 751F 2EA1 0700 0510 002E 8B1E
NanJing.2976C       B130 B600 B406 CD1A B4FF CD21 80FC 0075 1F2E A107 0005 1000
```

No.1744

CER: A stealth, encrypted, appending, 1744-byte virus containing the text ‘Made in No.25 middle school which lived in Beijing China.’.

```
No.1744             80FC 3075 1083 FB9A 7403 E9E5 02B0 02E8 C104 E9F4 0280 FC11
```

Qres.402

CR: An appending, 402-byte virus, recognizing infected files by the byte 90h (NOP) at offset 0003h.

```
Qres.402            B440 512B CAE8 48FF 5AB9 9201 B440 E83F FF33 C933 D2B8 0042
```

Sza.2174

CN: An appending, 2174-byte, direct infector containing the text ‘JLLSafe’. Starting from September, on the first day of a month the virus overwrites the C: drive and displays a full screen message in Hungarian (this includes ‘SZÅMALK vírus V1.10’).

```
Sza.2174            BE1A 07BF 1101 8A04 3005 4680 3C01 7503 BE1A 0747 81FF 1004
```

Tedy.4350

ER: A polymorphic, appending, 4350-byte virus containing the text ‘¡¡ TEDY, el primer virus interactivo de la computación. !! Responda el siguiente custionario: ¿Los programas que usted usa son originales? [s/n] 5 archivos menos por mentiroso. 2 archivos menos por ladrón. ¿Los de Microsoft son unos ladrones? [s/n] Te doy otra oportunidad para responder bien.’. The payload, which triggers on 12 November, writes its code to the MBR (moving the original to cylinder 0, head 0, sector 7). The following template can be used to detect the virus in memory:

```
Tedy.4350           80FC 4E74 1380 FC4F 743D 3D00 F074 052E FF2E AB01 B8FF FFCF
```

Trivial.113B

CN: An overwriting, 113-byte virus containing the texts ‘*.COM’ and ‘Insufficient memory to run program’ which is displayed when an infected program is executed.

```
Trivial.113B        3D8B D483 C220 CD21 8BD8 B440 BA00 01B9 7100 CD21 B43E CD21
```

VCL.1725

CN: An encrypted, 1725-byte virus containing the texts ‘YOU DIDN’T SAID THE MAGIC WORD! (BLIEB!)’, ‘GREETINGS FROM: NIGHTMARE’, ‘C:\WINDOWS\*.INI’, ‘I wish you an UNHAPPY newyear!’, ‘ISN’T it time for you to:’, ‘NIGHTMARE RULES YOUR SYSTEM !’ and ‘-VCL The Next Generation- THE ALIEN WILL EAT YOUR C: IF YOU’RE RUDE TO HIM! Greetings from :NIGHTMARE!’.

```
VCL.1725            8DB6 1201 B94F 0381 34?? ??46 46E2 F8C3
```

VIRUS ANALYSIS 1

Anarchy in the USSR

Igor Daniloff
DialogueScience

The Anarchy family, first seen in 1994, was named after a string in its code and a reference to an anarchist musical ensemble from Omsk. In the West, aliases include GrOb, Unfo, and Vivat. In June 1997, two years after the release of Anarchy.6503, the last specimen of the series, I was surprised to receive a copy of the latest addition to the family, Anarchy.6093 – intelligent enough to infect not only DOS COM and EXE files, but also *Word* documents.

Installation from an Infected DOS File

On starting an infected COM or EXE, the polymorphic decryptor explodes the main virus body and passes control to it, which first computes a 16-bit CRC of its code. This is compared to the CRC saved at file infection. If there is a discrepancy, it hangs the system by looping back to itself.

If all is well, the virus checks whether it is already resident by opening the file 'JANKA DYAGILEVA', reading the first 81 bytes, and comparing them to some strings in its body. If Anarchy.6093 is already in memory, it simulates opening that same file with the file handle BX=FFFFh and reading the necessary bytes for verification. If it is already active, then the newly-run copy simply returns control to the host program. If it is not active, it searches the DOS environment for variable names ending 'IR=' and 'EC=' (normally WINDIR and COMSPEC). On finding them, it infects both the WIN.COM file in the WINDIR directory, and the command processor specified by COMSPEC.

Further to this, the virus writes its code in the segment immediately after the PSP of the loaded program, beginning from offset 0142h, and intercepts Int 2Fh, taking over the Get Job File Table Entry function (AX=1220h). When this function is called, the virus' handler sets the JFT ES:DI pointer to the virus data, whose first byte is FFh, as though the requested file had not been opened by the system. Thus, the virus tries to prevent analysis of the System File Table. Since Anarchy is invisible at the Int 21h level, denying any program access to the SFT is a reliable tactic for hiding itself in an infected file.

Anarchy also checks whether *Windows 95* is loaded and sets an internal flag for later reference. It then intercepts Int 21h, adjusting the memory block size necessary for the operation of the resident copy, and sets a DOS memory allocation flag. Then it starts the infected program (Int 21h AX=4B00h), receives the return code (Int 21h AX=4Dh), and terminates the host program (Int 21h AX=4Ch). In this way, Anarchy stays resident and controls interrupts Int 21h and Int 2Fh.

Int 21h Handler

Aside from providing its 'Are you there?' call, the Int 21h handler controls nineteen functions for infection and stealth. These are typical of DOS stealth viruses, such as Find First, Find Next, Create, Open, Close, Read, Write File, etc.

When the Load Program (AX=4B00h) or Close File (AH=3Eh) function is called, Anarchy again checks whether *Windows 95* is running and if its internal flag was set. If *Windows 95* was not loaded, and started subsequently in an infected DOS environment, the virus, using the function Int 15h AH=87h (Copy Extended Memory), reads the six bytes located at physical address 110000h, and checks whether the first machine word at this address is 60FCh (CLD, PUSHA). If this word is detected, the virus assumes that a procedure of driver VMM32.VXD (Virtual Memory Manager) is loaded at 110000h, and modifies the initial bytes of this procedure so that control is first passed to the part of the virus' 32-bit code that is designed to hide the virus in infected COM and EXE files.

This stealth mechanism intercepts file functions of virtual device drivers and, if necessary, 'corrects' the results of the requested operations to hide the virus' presence. I could only identify the 60FCh in computers running the Pan European edition of *Windows 95*. This part of Anarchy's functionality is probably specifically linked to that edition of *Windows 95*, which is widely used in Russia. The virus disables its DOS stealth mechanism if it detects that any of the archiving utilities ARJ.EXE, ZIP.EXE, RAR.EXE, or RAR20.EXE are running.

On intercepting file Create, Open and Close functions, the virus checks the extension of the file being accessed. If it is COM, EXE or DOC, the file is infected. For COM and EXE files, the virus makes an encrypted polymorphic copy of its code, appends it to the file, and modifies the header of EXE files, or the initial bytes of COM files, to transfer control to the decryptor in the virus' body. DOC files are treated in an unusual manner. The infection technique used had not been seen prior to the release of Anarchy.6093, which is the first virus to infect OLE2 documents by independently analysing their structure.

When programs named AI??????.EXE, WE?.EXE, or DR???.EXE (AIDSTEST.EXE, WEB.EXE, DRWEB.EXE are Russian scanners) open a document file, the virus disables its document infection engine.

Document Infection Engine

On detecting access to a DOC file, Anarchy reads the first 512 bytes into a buffer and checks the first word for the OLE2 signature. It also checks the machine word at offset 1Eh. This word must be equal to nine, which corresponds to the 512 bytes in the document cluster. If these conditions are satisfied, the virus proceeds to analyse the DOC file. It computes the offset of the stream directory in the document, sets the pointer in the file to the header of the second stream (256 bytes past the Root Entry), and reads 256 bytes into a buffer. Then it attempts to find the 'Word Document' header of the stream in the second or third directory entry. The header of this stream is verified only by the first character, 'W', in the stream name. Having detected this, the virus computes the offset and reads 288 bytes from the File Information Block (FIB).

At this point probably the only serious bug in the virus occurs – it does not verify the FIB signature, but assumes that the FIB offset computed in the previous step is correct. This is true only for documents that are longer than 4096 bytes. The virus' FIB computation algorithm is ill-designed for documents with the so-called mini-FAT. With such documents, instead of reading the FIB, Anarchy mistakenly reads the mini-FAT, and all subsequent operations designed to inject the virus code into the document are implemented incorrectly, so the document will obviously be corrupted. What will happen when such a document is opened is only a matter of conjecture. All these points apply equally to *Word 8* documents, as they utilize a format different from that of *Word 6/7*. *Word 8* documents will most likely be corrupted by an attempted Anarchy.6093 infection.

Word generally creates documents longer than 4096 bytes, and for such documents, Anarchy computes the FIB offset properly. Next, the virus checks whether the file is a template and whether it is password-protected. In either case, the virus does not infect the file. Otherwise it sets the template bit in the FIB, then it checks the Macro Table size, which as a rule is equal to two. Finally, it checks that the document size is a multiple of 512 bytes.

Documents deemed 'infectible' have their FATs modified so the virus can append a macro to the file. The virus then writes 512 bytes at the end of the infected document, and adds a reference to the Macro Table, which it creates after the 512-byte block. It also specifies the size of this Macro Table and creates the Macro Table. Anarchy writes a flag for only one macro, a randomly-chosen macro encryptor key and the two macro name tables. In the macro name table used by *Word's* internal macro handling procedures, the macro is named AUTOOPEN, so it is automatically executed on opening the document. However, in the other name table, which is where the macro names displayed in Organizer, Tools/Macro and the like are stored, the virus leaves the macro nameless. More correctly, there is a name, but it is a line containing no printable characters. 'Normal' macro viruses cannot play such tricks, as they depend on *Word* itself setting up these tables, and *Word* always creates a displayable name entry in the second table.

Thus, the virus attaches the Macro Table at the file end and then proceeds to create the macro in a buffer. It copies 104 bytes from its body to the beginning of the macro. These 104 bytes start with the standard macro 'SUB MAIN'. Then follows a simple routine to find a currently unused EXE filename and create a file with that name. This file is then opened for writing with the file handle '#1'. Using string manipulation and print commands, the virus builds a WordBasic routine in the macro to drop a copy of the virus' binary code. To achieve this, it scans its resident code, splitting it into sub-strings up to 128 characters long, and writes these into the macro. As a simplified example:

```
C$=”MZNE♣ _ ☺♥☺”
C$=C$+”äú↑êäô↑êäo↓ÖçVüãB☺ï”
PRINT #1, C$;
```

Incredible! Anarchy assigns characters to C$ that cannot be created at the keyboard. It transfers the binary virus code, including all the characters from 0 to 255, to the variable C$.

After scanning the resident code and creating commands in WordBasic, Anarchy writes the concluding instructions:

```
Close           ; close the newly created EXE
Shell N$        ; start the newly created EXE
Kill N$         ; delete the EXE file
END SUB         ; the end of the macro
```

The macro in memory is then encoded with the previously-generated encryption key and written to the end of the document. This completes the infection mechanism. It is clear that on opening an infected file in *Word*, a file in NewExe format will be created, run, then deleted. It only remains to discover what this program does.

Installation from the NewExe File

The *Windows* NewExe file dropped by the virus' macro is quite novel. Its NE signature is immediately after the MZ signature of the DOS EXE file and, as a result, there is no DOS stub for the NewExe file.

On execution, this program calls the KERNEL.192 (Global-PageLock) and KERNEL.172 (AllocAlias) functions. Then control is transferred to the 16-bit code that is also used in infected DOS files for finding and infecting the command processor defined in the COMSPEC variable, and the WIN.COM file in the WINDIR directory. After these operations, the functions KERNEL.176 (FreeSelector) and KERNEL.192 (GlobalPageUnlock) are called, and the program exits by calling Int 21h AX=4C00h (Terminate).

Payload

On 8 and 30 April, and 9 May, the virus writes a Russian quatrain to randomly-selected hard disk sectors. This routine runs only if the disk handler of Int 13h has 63h (ARPL) as the first byte – this is true under *Windows 95*.

Summary

Anarchy.6093 is the first multi-platform virus to infect COM and DOS EXE files, drop a *Windows* NewExe virus, and inject a dropper into *Word 6/7* documents. It is also the first virus known to hide its presence under *Windows 95* by modifying the system driver VMM32.VXD, which operates in protected mode as a supervisor.

This virus affects the four operating environments DOS, *Windows*, *Windows 95*, and *Word*. This is remarkable for its 6093 bytes – more so in light of its polymorphism and stealth under both DOS and *Windows 95*. It is an almost instant infector, spreading widely and quickly. Possibly the biggest surprise is the elegant route it takes to 'infecting' documents. This mechanism operates only if *Windows 3.1x* or *Windows 95* is started in an infected DOS environment. As soon as *Word* or *WordPad* begins to open a file, the virus code is planted in the 'clean' document. This works with 32-bit *Windows 95* applications because they use the 16-bit DOS interrupt handlers for their file operations.

The ability to migrate along with documents explains why Anarchy.6093 hit Russia at such a rate. I received the first specimen from Siberia. A day later I received an avalanche of infected files from Moscow and other central Russian cities miles away. The epidemic spread the length and breadth of Russia in literally two or three days and even visited the LAN at Duma (the Russian State legislative assembly). I cannot recall any other virus that hopped from one computer to another at such lightning speed.

Anarchy.6093

Aliases: None known.

Type: Memory-resident parasitic, stealth, polymorphic COM, EXE file and DOC OLE2 infector.

Infection: COM, EXE, and DOC files and drops a *Windows* NewExe file from Trojanized DOC files.

Self-recognition in COM and EXE Files:
Bit 15 in year field of file time-stamp. Russian text strings at file end.

Self-recognition in DOC Files:
Will not infect if template bit set.

Self-recognition in Memory:
Open file 'JANKA DYAGILEVA' and read 81 bytes; see description.

Hex Pattern in Files:
None possible.

Hex Pattern in Memory:

```
9C3D 0042 751B 83FB FF75 162E
381E 6C19 750F 23D2 750B 23C9
```

Intercepts: Int 2Fh for stealth routine and Int 21h for infection and stealth routine in DOS. Patch VMM32.VXD in *Windows 95*.

Trigger: On 8, 30 April and 9 May, overwrites random disk sectors.

Removal: Infected files are identified and restored from a clean system.

VIRUS ANALYSIS 2

New Kid in Town

Beáta Ladnai
Sophos Plc

Uglykid reared its head in the anti-virus industry about three months ago, carrying in its code a vague clue as to its origin. The virus contains text fragments '(c) Nasty Lamer & Ugly Luser, Slovakia', and 'Do not forget: SlovakDictator is mother of all macro viruses of the new generation !' Uglykid's author may have been inspired by SlovakDictator (*VB*, August 1997, p.15) or it is possible both viruses have the same author, who is overly proud of his creations. Whatever the relationship between these two, the Uglykid virus certainly makes its own contribution to the guidelines of macro virus writing.

Yet Another Polymorphic

Uglykid has only one macro in infected documents – an execute-only AutoOpen. A special tool is needed to 'decrypt' the macro, thus enabling *Word* to edit it. Once it is in an editing window, the macro reads:

```
Sub MAIN
JIGLOU = 23.792904114089
On Error Goto MHHT
ScreenUpdating 0
OGVH = Rnd()
LFGGPB$ = "A"
PRSF = 73.02454248902
SetDocumentVar LFGGPB$, "JJSRW"
JHGFC$ = WindowName$()
...
```

… or at least it did in one replicant. A quick glimpse at the macro reveals that the virus must be polymorphic. All variable names and labels seem to be chosen randomly, and the code is teeming with junk lines of seemingly random WordBasic statements. Statements of arguments, occurring in every replicant, may appear as

```
ToolsMacro .Name = "KUTQOQ", .Edit, .Show = 3
```

in one, but may have a different syntax in another:

```
ToolsMacro .Show = 3, .Name = "TWWMVM", .Edit
```

The virus alters the order of arguments to make the code more diverse. Beside these signs of polymorphism, an interesting feature is also seen at a glance – the AutoOpen macro is too short to hold the polymorphic engine. The bulk of the code must be hidden elsewhere!

Build a Macro on the Fly

On opening an infected document, AutoOpen first ensures that the screen does not update before the macro's run is complete. Next, it surreptitiously starts a macro editing window, specified by a random name like JLDEI – unlikely to be in use by another macro. An EditAutoText statement, which was then quite unexpected in a virus, inserts the content of an AutoText entry into the edited macro.

AutoText is a useful tool in *Word* for storing and restoring frequently-used text or graphics. Uglykid demonstrates that it is also a splendid hiding-place for virus code. The AutoText entry used by the virus contains the major part of its code and has to be associated with the document when the virus infects it. After closing the macro editor and reactivating the document window, Uglykid's AutoOpen makes a call to its newly-created macro. When control returns from this macro, AutoOpen deletes it.

Behind the Scenes

The transient macro is in charge of infecting the global environment. If there is no FileSave macro in the global template, the polymorphic engine of the virus will build one by editing it line by line. To support this there are some string manipulation functions. One of them generates a random string of capital letters, four to seven characters in length. This is used to replace all variable and label names in each replicant with new ones. Another function shuffles three arguments in a statement.

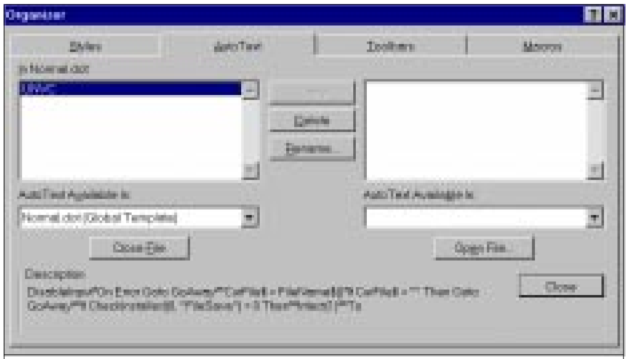
The FileSave macro has the same structure and mission as AutoOpen – to build and call a macro on the fly. Following the insertion of each source line relevant to the macro's functionality, the polymorphic engine calls a subroutine that generates one of five 'junk lines' and appends it to the new virus macro. The strings referring to SlovakDictator both have a 0.3% likelihood of being added as a remark, while one in seven times this line will be blank. In some cases no extra lines are inserted into the developing macro.

When the FileSave macro is ready, an identical copy of the AutoText stored in the document is passed onto the global template through the Organizer. Needless to say, the name of the entry changes as well. A macro called ToolsMacro is also created here, its content copied from FileSave. Thus, Uglykid takes control on every FileSave, and whenever the ToolsMacro option is invoked.

If the transient macro is created and called by the intercepted FileSave or ToolsMacro, the current document is infected in a similar fashion. First it is saved as a template file and checked for an AutoOpen macro. If the macro is not installed, the virus generates one and makes a copy of the constant AutoText in the document.

Uglykid's Fingerprints

Although Uglykid is armoured with some concealment techniques, its author was certainly not concerned about implementing perfect stealth. All its macros are execute-only, and the standard ToolsMacro is overridden by the viral ToolsMacro, which make access to the macros, and their source, less straightforward. The virus also removes the AutoText option from the Edit menu so as to minimize the chance of any user interference with the viral AutoText entry. However, when the virus removes the Templates option from the File menu, the customization is not carried out thorougly enough. After closing every document, the Templates option becomes available, providing a route to the macros and the AutoText entry as well.



Getting to Uglykid's AutoText through the Template Organizer.

Conclusion

There are two things worth noting about the Uglykid virus. First, it is polymorphic. The sudden increase in viruses of this nature shows that polymorphism may soon be a common feature of macro viruses. Uglykid is sufficiently polymorphic that, although it does not use encryption, the maximum number of consecutive bytes present in every replicant is only seven – clearly not enough for a reliable scan string.

Secondly, the clever technique with the AutoText makes this virus less prone to heuristic detection. All statements in Uglykid that could be described as very typical of viruses (like MacroCopy or Organizer) are hidden in the AutoText. The macros are built from statements less likely to be used in viruses.

Another aspect of the AutoText technique relates to disinfection. Removing the viral macros only leaves the major part of the viral code in the AutoText entries. Although it is only the 'passive' text of the virus, it remains available for editing and possibly for others to use. If this technique becomes widespread, AutoText entries in *Word* documents will surely need some anti-viral attention.

Uglykid

Aliases:	None known.
Type:	*MS Word 6/7* macro infector.
Hex Pattern in Files:	
	There is no reliable hex pattern for the virus – see text for explanation.
Trigger:	A dialog box displays if Edit|AutoText is selected in an infected environment.
Removal:	See text.

VIRUS ANALYSIS 3

Veni, Vidi, Vicis?

Vesselin Bontchev
Frisk Software International

I frequently receive virus collections that do not come from real infections but straight from virus writers or from some of the many virus exchange (VX) sites. In the first case, the virus authors may think that I, an anti-virus researcher, am the best person to 'appreciate' their work. In the second, it is usually 'helpful souls' who get the viruses from VX sites and send them to me (and very likely to other anti-virus researchers), probably with the feeling that they are helping us. In both cases, the collections are usually uploaded to our ftp site or sent to us by email and contain mostly junk.

The 'junk' consists of corrupted virus samples, programs intended to contain a virus but actually too buggy to work, virus creation tools, text files, programs falsely flagged as infected by one or other anti-virus product, totally unrelated files and so on. Even when they do contain viruses, they are usually known to me. If they are not familiar, they are the usual crop of boring, stupid, buggy viruses, often created by a slight modification of some well-known one. Processing such collections is most often a waste of time, yet it has to be done; it is (an unpleasant) part of the job of the anti-virus researcher.

Occasionally, however, these collections contain something interesting. This was the case with the latest batch of macro viruses from the virus-writing electronic magazine SLAM. SLAM was started by the amazingly inept German virus writer called 'The Nightmare Joker', who has authored probably more than a dozen rather trivial macro viruses and macro virus construction kits. This time, the magazine contained a macro virus obviously written by someone else – somebody with some talent for virus writing.

What's in a Name?

The author calls this virus 'Vicissitator' – whatever that is supposed to mean. The name looked too long to me, and besides, I enjoy spoiling part of the virus writer's fun by renaming their creations. I therefore decided to call it WordMacro/Vicis.A, or Vicis for short.

Vicis is a polymorphic macro virus… that is the very least that can be said about it, and it is a major understatement. Polymorphism in DOS viruses is usually achieved by encrypting most of the virus body and prepending a randomly-generated decryptor to it. The same idea has also been attempted in the macro virus world (see *VB*, August 1997, p.15). However, WordBasic is a slow language, not very suitable for character manipulation, so the encryption/decryption process is always slow, which makes such a virus very noticeable. WordBasic is much more suitable for string manipulation. Furthermore, WordBasic is a syntactically simple language. All these properties make it easy to implement a different kind of polymorphism, one not based on encryption. The idea was first described by Dr Fred Cohen several years ago, but this is the first virus I have seen implement it properly.

Replication and Polymorphism

Vicis replicates using the system macros attack. In infected documents, it consists of a single macro named FileSave. When an infected document is opened, edited and saved using the File/Save command, Vicis receives control and copies itself to the global template (NORMAL.DOT). It also uses the 'ToolsMacro .Edit' command to create a new macro, ToolsMacro, which resides only in the global template. The contents of this macro are simply:

```
Sub MAIN
'You have been Infected by the Vicissitator
    Macro Virus.
'(C)1997 CyberYoda A Member of the SLAM Virus
    Team
End Sub
```

The purpose of this macro is to disable the Tools/Macro command to prevent inspection of the macros in a document, and thus prevent discovery of the virus. Needless to say, such a primitive form of stealth is likely to be noticed quickly. Much more advanced forms of stealth have been developed by other macro virus writers. However, the strong point of Vicis is not its stealth but its polymorphism.

When Vicis is active in *Word*, clean documents are infected every time they are saved with the File/Save command. The virus invokes its polymorphic routine only when replicating from the global template to a document, not in the opposite direction. If the global template or the document contains a macro named FileSave, the virus considers it to be already infected and leaves it alone.

Vicis' main mutation routine works by editing itself in the macro editor. It marks a random number of consecutive lines (between one and ten inclusive), then checks that the contents of the marked lines satisfy a number of conditions. No line may begin with 'Sub', 'Dim', 'End Sub', 'On', nor end with a colon. Every 'For' in the block needs a matching 'Next', and every 'While' a matching 'Wend'. Each 'If' must have a corresponding 'End If' (with an optional 'Else' before it), and there should be no orphan 'Else' operators (from 'If' operators outside the marked area).

If all these conditions are met, the marked lines are cut and replaced with a 'Call' operator to a randomly-generated subroutine name. Then the virus pastes the cut lines at the end of its body between a Sub/End Sub pair it has created, using the same randomly-generated subroutine name.

After Vicis has invoked its main mutation routine, several different procedures can be executed. One in four times, it tries to 'unmutate'. It does this by looking for a line in its body beginning with 'Call', finding and copying the body of the called subroutine, and then deleting the subroutine. Next, Vicis goes to the place where the subroutine is called, and replaces the 'Call' operator with the lines copied from the subroutine.

The general idea behind this is obviously to reverse the main mutation, so that the virus not only fragments its body into subroutines but also occasionally re-combines the fragments. This allows the larger parts to be split again later into other fragments, therefore making its contents more dynamic. However, this process can 'unmutate' some of the main subroutines, calls to which are used in the 'Then' part of 'If' statements. When this happens, the bodies of these subroutines will replace the 'If' statements, changing the behaviour of the virus significantly, and possibly preventing it from working. I believe that this is a bug.

There is a 25% probability of the virus picking a random variable from its body and changing all occurrences of it to some other randomly-constructed identifier. This can be from two to eleven characters long, and consists of letters only. The variable is the first word to the left of an '=' sign, which is an unreliable way to select it – the argument of a system command (for example, the '.Find' in 'EditFind .Find = Name2$') could be selected. Of course, when this happens, the resulting replicant will no longer work, yet again confirming my strong opinion that most virus writers are sloppy, bad programmers, incapable of producing a bug-free virus.

Vicis also has a 25% chance of mutating a numeric constant. Only integer constants are mutated, and the mutation itself follows one of four procedural options. First, if the constant happens to be divisible by a randomly-selected number between one and ten inclusive, it is replaced with a multiplication which results in the same constant. Thus, the constant '10' might be replaced with the expression '5 * 2'. In the second case, the constant is substituted with an expression of division which evaluates to the constant. For example, the expression '30 / 3' can take the place of the constant '10'. The constant can also be replaced with a subtraction, so that the constant '10' is substituted by the expression '(17 – 7)', for instance. Lastly, the constant can be replaced with an addition – '(4 + 6)' for the constant '10', in this case. Only one of the above four replacements can happen during a single replication of the virus. The second operand of the replacement expression is always a randomly-generated number between one and ten inclusive.

During one in ten replications, Vicis inserts a new line at the beginning of a randomly-selected subroutine. This new, do-nothing line has the form:

```
If <identifier1> = <identifier2> Then
     <identifier3> = <number>
```

where the identifiers and number are randomly generated.

There is a 0.39% possibility that the virus will corrupt itself. This is achieved by removing a random number of lines from a random subroutine of the virus. I have no idea why it was designed this way, since it is extremely likely to produce non-working replicants. Nevertheless, it is obviously intentional (i.e. not a programming error – maybe a design error), despite its very low probability.

Detection

Regardless of the convoluted polymorphic mechanism described above, the Vicis virus can be detected reliably with a simple scan string – no wildcards are necessary. This is due to the fact that all replicants of the virus contain reasonably large chunks of code which are constant, although their locations are not. Furthermore, like all other WordBasic viruses which attempt to be polymorphic, Vicis' macros are not Execute-Only. They are not encrypted even with the trivial encryption algorithm used by *Word* for this purpose. Thus, they can be listed and edited freely.

The command 'MacroCopy "Global:Vicissitator", WindowName$() + ":FileSave"' is present in all working replicants of the virus.

Summary

Like most other polymorphic WordBasic macro viruses, Vicis is too buggy, slow, and obvious when replicating. It has no chance of surviving in the wild or becoming widespread. If a Vicis outbreak occurs, it will be very easy to detect the virus with a simple scan string. The problems posed by its sophisticated, polymorphic mechanism are only faced by those anti-virus products that identify viruses exactly. Still, it is worrisome to think what CyberYoda might be able to achieve with Visual Basic for Applications, the macro language of *Microsoft Word 8*. This is much faster, more powerful, and more suitable for the implementation of polymorphic engines. As usual when I analyse a clever virus, I feel pity for the obviously ingenious mind behind it that has been wasted on creating something so utterly useless and damaging.

Vicis

Aliases: None known.

Type: Polymorphic *Word 6/7* macro infector.

Self-recognition in Documents:
Will not infect if the target already contains a FileSave macro.

Hex Pattern in Files:

```
67C2 806A 1347 6C6F 6261 6C3A
5669 6369 7373 6974 6174 6F72
```

Disinfection: In a clean *Word* environment, use the Organizer to delete FileSave macros from infected documents.

FEATURE 1

A View from the Lab

Peter Morley
Dr Solomon's Software

[*The views expressed here are those of the author, and are not necessarily endorsed by* VB. *Ed.*]

Nearly four years ago, I was invited into the Virus Lab at *S&S* (now *Dr Solomon's*), and have not escaped since. It is a strange and peaceful place, providing a unique vantage point from which to view the computer industry in general, and the machinations of the small group of anti-virus companies in particular. These can be more entertaining than old-fashioned music hall, but music hall jokes were more credible than some of the advertising literature I see!

Some History

In September 1993, there existed over 3000 viruses and variants, and most anti-virus companies could detect nearly all of them. There was a *laissez-faire* attitude to whether it was necessary to deal with every new virus, and most organizations had a 'backlog' of a couple of hundred viruses. As I saw it, we were in the middle of the wolf pack and running hard. The first question which came to mind was 'What do we have to do to get so far ahead that the others can't bite our tails?'

I was told by Alan Solomon that there were fewer than ten people in the world with the knowledge and experience necessary to disassemble new viruses, and write the code to deal with them. We calculated that we needed to process 150 viruses per month as well as new ones, so that by mid-1994 there would be no backlog. At the time, about 50 viruses per month were being processed, with laborious analysis performed on each. To increase from 50 to 150 per month required both more manpower, and totally different work practices. Mid-1994 came and we made it, albeit a month late.

I think Alan was the first anti-virus guru to perceive that processing had to become an efficient, no-frills, production operation. That realization has made him a multi-millionaire. I was joined by Dmitry Gryaznov, already acknowledged as one of the top ten experts, and Duncan Long, whose assembly language skills were adequate to design and write his own Operating System. It was up to me to provide the production operation.

As I write this, in July 1997, new viruses and variants are appearing at over 250 per month. There are occasional gluts, such as the release of the two Ludwig CD-ROMs, and the shenanigans of the past three months. There seems to be a quiet patch each August and September, but do not bank on it this year!

Any organization which cannot process 300 viruses per month in times of stress, has no chance of keeping in the game. We have been joined by Igor Muttik (another of the the top ten), and we *have* processed all the viruses (14,117 as detected in *Dr Solomon's AVTK* v7.75) which have come our way. These are passed freely between members of the anti-virus community, so any processing omissions are *not* due to 'We haven't seen it'. They are due either to a failure to implement the necessary resources and working practices, or to a deliberate policy of only processing a subset.

Anti-virus Organizations

As I see it, anti-virus organizations can now be split into three categories. Category A comprises those which process nearly every virus. I personally know of four such companies – the others are *Sophos*, *Alwil*, and *AVP*. These organizations excel in technical competence, and are pretty good at technical organization. This does not, necessarily, mean their products sell well. I recall the early, derisory efforts of what was then *S&S* in the US market… which are now being rectified. Further, *AVP* seems to have made little effort outside the former USSR. This indicates that the emphasis has been on virus detection rather than on making user interfaces really friendly. *Dr Solomon's* is putting this right, too.

Category B's companies are those which try to process every virus, but fail. They fail because they do not put in place the necessary organization and resources. To them, virus processing is just another part of the programming operation. This has little or no bearing on commercial success. With top-class user interfaces, and excellent marketing to a customer base which cannot adequately test the product, in geographical areas with little competition, commercial success is still virtually assured.

It is possible for organizations to slip from category A to B. This happens gradually, one day at a time. Could a company move back up? Bearing in mind that back in 1993, *every* anti-virus organization was in category B, the answer must be 'yes'! The simple way is to arrange to use the engine, and/or detection database from one of the category A companies. Early in 1997, *McAfee* introduced *VirusScan* v3.0, which suddenly detected more than 1000 additional viruses and variants (I actually tested!). I do believe in fairies, but not in miracles. History shows that there is a rational explanation for them; in this case incorporating another scanning engine helped.

Category C consists of companies which accept that they cannot process every new virus, and which advocate alternative strategies. These include prevention and change detection. Subset processing is a third tactic. All this is as old as the hills. Prevention and change detection are two of the wider facilities also offered by category A and B companies. Both play a part in a comprehensive anti-virus strategy, in addition to the use of scanners.

Change detection, as a main weapon, becomes less attractive when you have to perform it on every macro in your word-processing suite. Also, it is totally ineffective when you have just installed a major new set of software from CD-ROM. You have to wait and see if running the new suite causes executables to change. If it does, you have more work to do. Maybe a lot more work.

The WildList

Before discussing this term (which I hold in some disdain), let me suggest a scenario, and pose a question. Someone telephones our technical support unit, and says 'We've just had a bad outbreak of a virus which adds 1027 bytes to each executable, and I'm sending some samples. None of the present anti-virus scanners, including yours, detects it. It seems to affect boot sectors on hard disks, too. Please ring me as soon as you have looked at the samples.'

All absolutely normal. The virus, in this case, was Junkie, a kid's stuff multi-partite virus, distributed via the Internet and suddenly world-wide. The samples came to me, and I contacted the sender, explaining the virus (and the fact that it infected floppies too), and arranging to send an Extra Driver by return, so he could detect all instances, and repair them. Everyone was happy.

My question is 'Do you want to minimize the number of times the above scenario occurs? Or are you happy to telephone your anti-virus vendor, even when they have had the virus for months, but have not bothered to process it?' Your workload will be lighter if your scanner identifies and repairs the virus, and you will not need to make the call. If your vendor hides behind an 'In the Wild' list, the scenario will occur much more often. Various people have different ideas, but the one most used is attributed to Joe Wells. For a virus to get on this list, two of Joe's 'reporters' must have received samples of it from the field.

Each month I receive about six new viruses from our distributor in India. As far as I'm concerned, they are 'in the wild', but they will not make the official WildList until another vendor finds and reports them. This may be several months later, or never. It seems that India may be a little *too* 'wild'! I also come by forty or more South American viruses every month. I do not get them directly from the 'wild', but I have no doubt many of them are in it. I process them all, but they are not reported to Joe by *Dr Solomon's*. Later on, when they *are* 'in the wild' over here, and Findvirus detects and repairs them, nobody will call, so they may *never* get on the list.

The latest Ludwig CD-ROM contains well over 4000 viruses and is available world-wide to anyone who is prepared to pay. A third version may be imminent, providing several thousand more. Some IT managers bought the second release to test anti-virus products. (This is a misconception; when the list becomes available, it is already several months out of date, and will fall behind by another month, for each month which passes.) Relatively few of the viruses on it are on the WildList, which contains less than 600 of the 1500 viruses which I believe to be truly 'in the wild'. Even 1500 may be an underestimate – it is just 10% of the viruses known to exist.

My conclusion is that the WildList is an excuse often adopted by those vendors who cannot handle the viruses they receive. It is of no value whatever to end-users, or to category A vendors. The WildList has progressed from an indication of what should be processed next to an excuse for falling well behind in the game. Some category A vendors use it for no better reason than that they used to be category B, and are just continuing the habit. I see that as playing second division football, *after* promotion! Some will disagree with my conclusion, but few will dispute that any such list will inevitably be at least three months out of date, or that there are hundreds of field viruses which never get listed, particularly those which do not go memory-resident, and which are easy to clean up.

Opting Out

Macro viruses have provided a heaven-sent opportunity for category B and C vendors to claim that boot-sector and file viruses are now relatively unimportant, and the essential strategy is to protect against *Word* macro viruses which are now becoming prevalent. Try telling that to a German IT manager who has just had an outbreak of the Manzon virus! Of course, since there are fewer than 1200 macro virus variants, these vendors have not fallen too far behind in processing these yet.

The Choice

Some corporate IT managers find it difficult to choose software because they do not have the viruses against which to test. They, like me, may be suspicious of advertising claims (particularly those destined to be drawn to the attention of the ASA), and even more so of the adverts' omissions. If you wish to select one of two or three vendors, and are in the market for a site licence, try the following approach.

Ask each vendor to provide the latest shipping version of the software in question. Suggest they hold a half-day event, at which you can test each of the three products against a virus library to be provided. This should consist solely of those viruses which the vendor has added to his own detection capability in the last six months. If you want to be really fair, remove the last two months. You will not meet the Cascades, Jerusalems and Dark Avengers of yesteryear, but you can expect the samples to include new viruses received from the 'wild' by that vendor. If your vendor is category B or C, you will soon know! Try to handle the folklore as kindly as you can. The procedure I suggest has a major advantage for you – the vendors have nowhere to hide.

FEATURE 2

The Wrong Way

Vesselin Bontchev

I read with considerable disappointment the article *Do You Know the Way to VBA?* in the August issue of *Virus Bulletin*. I've known its author, Nick FitzGerald, as moderator of Virus–L/comp.virus, the mailing list and newsgroup dedicated to discussion of computer viruses, anti-virus products and techniques. I've known him as a competent anti-virus researcher, and it is sad to see his name on a paper that not only defends ethically questionable practices but also contains several serious technical mistakes. I sincerely hope that the position Mr FitzGerald expressed is the result of his not having given the necessary consideration to the material discussed (or his not having enough technical information on the subject) and that, should he do so, his position would change for the better.

I readily admit to being in the group of anti-virus researchers referred to as 'the "thou shall not up-convert" crowd' in Mr FitzGerald's article. A detailed explanation of all the dangers and the futility of doing up-conversions would be rather lengthy – I am preparing a paper on this subject to be presented at an upcoming anti-virus conference. The limited space of these pages allows me only to point out particular technical, logical, and ethical errors in Mr FitzGerald's article. I will do this by quoting each relevant part and pointing out the mistakes in it.

1. *The truth is that some developers have not done this* [the up-conversion]*… However, probably even more surprising is that some developers have avowed that they will not do this!*

I am one of those who have avowed not to create any new viruses via up-conversion. I hardly see how such a fact can be surprising – one would rather expect that the high ethical principles of legitimate anti-virus researchers prevent them being involved in virus creation and distribution. By no means have I avowed not to implement detection of up-converted viruses. I agree that it is a good idea to provide protection against them to my users. However, this must not be done at the cost of creating new viruses.

2. *Macro viruses exist. Their prevalence is increasing rapidly, as are the number of known virus families and variants.* The author then points out that the natural up-conversion of those 'in the wild' is imminent.

None of us who opposes macro virus up-conversion has ever expressed any reluctance to implement detection of viruses that *exist*. As soon as a VBA virus appears as the result of the natural up-conversion of a WordBasic virus, we update our scanners and make them able to detect, recognize, identify and remove the new virus. We do not do it before the virus exists, this being the very basic way scanners are developed. We wait for the viruses to appear, either by themselves or by being created by the virus writers; we do *not* create them ourselves. To do so would be ethically and morally wrong.

3. *To me, that spells 'clear and present threat.'*

Is the threat so clear and especially present? If we look at the July issue of Joe Wells' WildList – a list of viruses confirmed to be 'in the wild' by at least two independent sources, which is often used by anti-virus product testers – we see only four up-converted viruses. If we look at the prevalence tables published in *Virus Bulletin*, there are no such viruses mentioned there. Finally, if we look at the number of VBA viruses produced by natural up-conversion from WordBasic viruses, their number is minuscule compared to the number of known WordBasic viruses.

The exact reasons for this phenomenon are not very clear and need to be researched further. It is possible that only a very few of the known WordBasic viruses are widespread. The others are either 'collection viruses' or those created locally by *Word* corrupting some widespread virus so that it has produced a different set of macros which is, nevertheless, still able to replicate. Such local viruses are quickly exterminated by updated anti-virus products. It may be that *Word 97's* built-in macro virus protection actually works, effectively stopping existing macro viruses – although I, personally, have some doubts in this regard. It is also possible that, after up-conversion, most widespread WordBasic viruses produce non-working VBA code. (For instance, none of the WordBasic viruses which spread in encrypted form is able to work after being up-converted, because *Word 97* does not allow macros from a protected project to be copied elsewhere). Finally, it may be that all these factors form a significant combination, or that there are other, unknown factors. Nevertheless, the fact remains; for whatever reasons, naturally up-converted VBA viruses are few and far between. Therefore, up-conversion presents a danger which is not that clear and far from present.

4. *…adding an up-convert test would add two hours of analysis time per day. In practice it would not, as many (even most) up-converts will not remain viable and thus will require less analysis time.*

In fact, up-converts will require *no* analysis time because their WordBasic originators have already been analysed. The only additional time would be spent on up-conversion itself and on replicating the up-convert.

5. *Some developers not working on up-converted macro viruses argue that opening 'old'* Word *macro viruses in* Word 8 *creates new viruses.*

In fact, I am not aware of any *developer* (that is, somebody actually involved in the creation and maintenance of an anti-virus product), who denies the fact that the procedure described above creates new viruses. The argument usually revolves around the debate whether it is ‘okay’ for the anti-virus people to create these viruses in order to provide protection against them. Only non-technical people who, do not have intimate knowledge of how anti-virus products work and of how WordBasic and the VBA viruses are represented in files, argue these viruses are not different.

6. *It is reasonable – in fact, I encourage anti-virus developers not to develop and distribute new viruses.*

Unfortunately, such a statement is in direct contradiction to the rest of Mr FitzGerald’s paper, which clearly encourages anti-virus developers to create VBA viruses via up-conversion. Furthermore, once they begin claiming detection of the up-converts, some testing organizations will want to test this allegation. The only way to do this is either to have the testers create these viruses themselves (with the very high probability that they goof up, since most of them are much less experienced than the anti-virus researchers), or to obtain samples from some anti-virus producer who has created them (thus, that anti-virus producer will be engaging in the distribution of his own viruses). Regardless of which method is used, most testing organizations provide samples of missed viruses to the anti-virus producer whose scanner has failed to detect them – therefore, again, involving the distribution of viruses created by the anti-virus business. Clearly, this commendable goal (non-creation and non-distribution of viruses by the anti-virus industry) is not achievable if, at the same time, these people are required to perform the up-conversion.

7. *Arguing that making up-converts sets us on the slippery slope of facing such claims is a cop-out – like it or not we are already on that slope.*

This is not only totally unsubstantiated; it is plain ridiculous. The anti-virus industry is not on any ‘slippery slope’ and is certainly not required to engage in the creation and distribution of new viruses. At least, the legitimate sector is not. There always have been, and will be, some ‘black sheep’, but their behaviour should be denounced as highly unethical. The remaining legitimate anti-virus people should not be encouraged to engage in similar activities.

8. *Better to be known for acting impeccably and doing everything one could do to ensure anti-virus consumers had the best possible protection against known, current risks than for denying that the anti-virus industry has a few skeletons in the closet and a few differently-pigmented sheep in the back pasture.*

I am all for acting impeccably and doing everything I can to ensure anti-virus consumers have the best possible protection against known, current risks. Our own macro virus scanner has always been quick to respond to these. In order to keep up-to-date with them, we even provide hourly updates of the scanner’s database of virus definitions. However, we are talking about *known, current risks* here. As pointed out above, the up-converted VBA virus is nowhere near a ‘known, current risk’ and closer to a ‘possibly likely, although still not existing’ one. As soon as an up-converted virus appears naturally, we add proper handling of it to our product. The purpose of a known-virus scanner is, after all, to detect *known* viruses, not imaginary ones, regardless of how real they might seem to the person imagining them.

The situation is muddled further because the up-conversion process is far from unambiguous. So far we have found four *Word 97* settings which result in diverse VBA5 programs being created from one and the same WordBasic program. We are now actively researching the up-conversion process to clarify all such cases of ambiguity. Until this research is completed, it would be premature to claim that a product detects the up-converts which occur naturally in the real world, just because it happens to detect them on its developer’s or some other tester’s system.

As to the ‘skeletons in the closet’ and the ‘differently-pigmented sheep in the back pasture’ – yes, such things must be loudly denounced as highly unethical, and the people who engage in unethical activities rejected by the legitimate anti-virus community. Encouraging activities like theirs is the wrong approach, in my opinion.

9. *More than being a theoretical possibility, however, some old* Word *viruses have been up-converted at real client sites.*

Yes, this is true. As soon as it happens (and we receive a sample), we will implement handling of these particular viruses in our product. *Unless* this is documented as happening, however, the existence of any particular up-convert remains just that – a theoretical possibility. When or if this theoretically possible virus moves into the real world – *then* the known-virus scanners will implement detection of it. This has always been how virus-specific anti-virus products have been developed and there is no need to change this procedure, especially when changing it involves the creation of viruses by anti-virus people.

10. *Some developers probably brand the users of machines where such real-world up-converts have happened as ‘virus creators’ who therefore deserve what they get: the old ‘blame the victim’ approach.*

This statement is slanderous and plain wrong. I challenge Mr FitzGerald to quote any responsible anti-virus person taking such an attitude. It is true that the person who opens a document with a WordBasic virus under *Word 97* causes a new VBA virus to be created (in some cases). However, nobody even thinks of blaming the user for that. If anyone is to blame, it is *Microsoft*, for providing an automatic virus creation tool in their popular office automation package. The user, in most cases, has no way of knowing that the original document is infected, that up-conversion will take place, and that it will result in a new virus being created. Of course, if somebody knows these facts and nevertheless uses *Word 97* to create a new VBA virus by up-conversion, that person should be blamed as an unethical virus creator; one more reason for the legitimate anti-virus people not to get involved in such activities.

11. *An anti-virus developer 'making' an up-converted macro virus so as to add detection and disinfection of it to their product, prior to the virus actually appearing in the wild, does not increase the total threat.*

Unfortunately, the reality contradicts the above claim. I once up-converted a few XM viruses to their X97M counterparts (that was before I was firmly convinced that such up-conversion is a 'Bad Thing').

Later, up-conversions of precisely the same viruses were found at a very popular virus exchange site. It turned out somebody was using our macro virus scanner to maintain his virus collection. As he had seen the X97M viruses on our detection list, he decided that these viruses existed and he wanted them in his collection. The easiest way to get them was by up-converting, as he already had the corresponding XM viruses. Therefore, that I had created these up-converts (in order to implement detection of them), had indirectly caused them to be created by a virus distributor who made them publicly available. One could argue that the two events are not necessarily related. However, then one would have a hard time explaining how the exact same up-converts reported as detectable by our product were found on that virus distribution site, rather than more than a thousand other viruses suitable for up-conversion.

If the above example is not convincing, readers should ask themselves the following question. Suppose that every month an anti-virus producer uses a virus construction kit to create a few dozen new viruses, implements detection of them in their product, and then destroys them, without sending samples to anyone else. They could say that this is in order to provide their customers with protection in case somebody actually uses that construction kit to create those viruses, while in practice they can claim that their product detects more viruses than the competition. Do the readers not think that such behaviour would be, at the very least, unethical? Is such behaviour really so different from creating macro viruses which do not yet exist?

12. *Are W97M/Wazzu.A and WM/Wazzu.A really different viruses? Some of the experts say they are different, some not.*

Apparently, even the experts at *Virus Bulletin* do not agree on this! For example, in the March 1997 issue we read, in the unattributed *The Word of the Day*, that 'The new virus – for a new virus it is, despite the fact that it is a conversion of an old one – has been named W97M/Wazzu.A.'

13. *A reasonable counterclaim would seem to be that relationship between WM/Wazzu.A and W97M/Wazzu.A is similar to that between the EXE-infecting form of a multipartite and its MBR or SYS-file form.*

Such a claim would be far from reasonable. In fact, it is plain ridiculous. One would expect such a claim to be made by an ignorant amateur, not by an experienced anti-virus researcher. It should be crystal clear to the latter that a multi-partite virus carries within itself, programmed, all infection techniques it can use, both for infection of EXE files and MBRs. As opposed to that, there is absolutely no code in a WordMacro virus for handling the infection of *Word 97* documents. In fact, WordMacro viruses are simply incapable of infecting a *Word 97* document. They must be first rewritten in a completely different language, to a completely different structure and representation. In short, a totally different virus has to be made out of the first one. A more correct analogy would be to take the source of a virus written in Pascal, rewriting it in C, and compiling it. Most competent anti-virus researchers have no problem understanding that the virus produced by this procedure is different from the original – that it is a new virus.

14. *According to the CARO standard, if the binary representations of the non-variant parts of the two differ, they are different viruses. Unfortunately, this has little to do with the generally accepted definitions of what a computer virus is. These definitions focus on behaviour, not code representation.*

I was amazed to see such a statement from the moderator of Virus–L/comp.virus. According to that newsgroup's 'Frequently Asked Questions' list, a computer virus is:

> …a self-replicating program containing code that explicitly copies itself and that can 'infect' other programs by modifying them or their environment such that a call to an infected program implies a call to a possibly evolved copy of the virus.

In short, a computer virus is a program that replicates. A program. As a consequence, different self-replicating programs are different viruses. The program's behaviour (besides replicating) is totally irrelevant to whether it is considered to be a computer virus or not. Considered as a program, a W97M virus is totally different from its WM originator. It is different in language, different in contents, different in structure, different in binary representation, and it requires different methods to detect, recognize, identify and remove it. On many occasions, it even behaves differently – although, as noted above, this is irrelevant. A more scientific explanation of why the two viruses are different requires Dr Fred Cohen's formal definition of a computer virus, and space here really does not allow me to present the three to four pages of mathematical formulas needed to clarify the issue.

15. *Laboratory-generated up-converts are thus much like the tens of thousands of polymorphic samples that most researchers generate to test their product's detection of that polymorphic engine.*

The above statement is obviously false. All possible mutations of a polymorphic virus are programmed in that virus' polymorphic engine. They are contained in the body of the virus. This is certainly not the case with up-conversion. As explained above, a WM virus does not carry any code capable of changing it into a virus able to infect *Word 97* documents. An external program is needed to do the up-conversion, a program which is not even intentionally invoked by the virus. A better analogy is that laboratory up-converts are like viruses produced from a single virus source by compiling it with different assemblers. This is an operation which most anti-virus researchers warn against, because it results in the creation of new viruses.

Finally, after I have so much disagreed with the different points of Mr FitzGerald's paper, I would like to finish on a positive note.

16. *I look forward to the results of further research in the area of* Word *macro up-conversions. Once more is known about these specifics, we will be better placed to decide the role of up-converts in formal testing.*

Yes, I completely agree with the above. Further research in the area of *Word* macro up-conversion is necessary and should be pursued actively. If possible, anti-virus products should be made to detect those up-converts which are likely to occur. I myself have already done significant research in this area, and hope to present the results of it in a scientific paper relatively soon. However, we must always keep in mind that all this must *not* be done at the cost of anti-virus people being involved in the creation and distribution of new macro viruses.

Editor's Reply

Virus Bulletin does not have the space to afford a detailed, point-by-point response to Mr Bontchev's objections to my article. In the following I have, wherever possible, reduced my references to the objections simply to the numbered point in Mr Bontchev's article.

It seems that Mr Bontchev and I agree on several points, although these are sometimes suggested as differences. We agree that users of anti-virus software should receive the highest level of protection possible, and that anti-virus developers should pursue providing this. Further, we agree it is wrong for anyone – developer or otherwise – to create *and* distribute new viruses. Mr Bontchev misrepresents my position on this however, as careful reading of my original article and his responses in points 1, 6 and 8 will show. We differ significantly on the issue of whether it can ever be acceptable for anti-virus developers, under carefully-controlled conditions, to make 'new' viruses.

Mr Bontchev is clearly strongly convinced that creating viruses at all is morally reprehensible. So much so in fact, that in places he suggests (1, 2, 7, 10) that people who do can not be considered 'legitimate anti-virus people'. As there seems to be no statute of limitations on this fairly absolute position, one must wonder if Mr Bontchev feels he should no longer be considered part of the legitimate anti-virus industry, as by his own admission he has engaged in this most reprehensible of acts (11). I think Mr Bontchev's position as a leading anti-virus researcher is still assured and hope he stops judging himself, and others, so rashly.

The rest of my response defends my claim that up-converted macro viruses can be considered different representations of the same virus. Acceptance of this removes the basis for Mr Bontchev's moral resistance to making up-converts.

Regarding Mr Bontchev's claims in point 14, I was amazed to see such a statement from such a significant contributor to the Virus-L/comp.virus FAQ, and especially to the definition of the term 'computer virus'. In the quoted definition, notice the emphasis on 'replication', 'copying itself' and 'modifying' its targets or environment so the virus code 'or a modification of it' would be run at the next attempt to execute the original. As I read this, it emphasizes behaviour. In fact, Mr Bontchev is aware that we tried to re-word the definition to avoid using the word 'program' altogether, making the definition more generic, but the resulting definitions were deemed too formal and/or likely to confuse the intended readership.

The claim in 14 that application of Dr Cohen's models and theorems provides a more scientific explanation sounds impressive, but in my own discussions with Dr Cohen, he and I agreed that his formal models support my position equally well. The problem of applying formal mathematical theorems to such issues is that you get the 'right' answers if you make the 'right' simplifications and reductions in deciding which pieces of the puzzle fit into which axiomatically defined 'boxes'. Unfortunately, many of the relevant puzzle pieces cannot be fitted according to the formalisms as they simply ignore the mechanics of doing so. Thus we could reduce the debate to whether and where the pieces fit – that is not the route to progress. I look forward to convincing Mr Bontchev of this during *VB'97*.

My interpretation of Dr Cohen's formalisms also suggests that they support my claims, which Mr Bontchev dismisses, in 13 and 15. Further, the claims in 13 and 15 about all forms of the viruses being present in all replicants are neither required by Dr Cohen's formalisms, nor observed in the current, working CARO classification scheme, where Spanish Telecom is seen as one virus, although the boot-infecting part cannot regenerate the file-infecting part.

Mr Bontchev's insistence that different means different seems odd coming from the person who admitted in personal email that Mac and PC representations of, say, Concept.A, with their different endian-ness, are the same virus. Maybe Mr Bontchev accepts that some difference can be part of 'being the same' after all.

In 11, Mr Bontchev provides perhaps the best pragmatic reason for considering up-converts as different representations of the same virus as their 'parent', rather than as 'new' viruses. What this case shows is not that making up-converts *per se* encourages virus creation, but that giving them new names does. Had he silently added detection of the *Excel 97* representations of these viruses to his product, Mr Bontchev would have improved his clients' protection without fuelling the VX site maintainers who allegedly use his product's virus list as their scorecard.

There are other developers who agree with me – I have email from them – but most are content with improving their products, rather than debating the point.

Nick FitzGerald

PRODUCT REVIEW 1

Vet for Windows NT Server

Martyn Perry

Vet for Windows NT (*VetNT*) v9.44 from *Cybec* supplements the organization's products for DOS, *Windows 3.1x*, *Windows 95*, *NT Workstation* and *NetWare* servers.

The software can be licensed for single or multiple users. If the number of users represents the total number of company PCs, the licence can be extended to cover employees' personal use. The initial purchase price covers quarterly updates for the first year, but a more frequent update option can be purchased separately.

Presentation and Installation

The review package consisted of two diskettes and a product manual. Installation was performed by running SETUP32.EXE from the first diskette. The first thing displayed was the licence agreement, followed by prompts for user and company names. The installer is then presented with a selection of three options. The 'Typical' option has default settings for a typical user, while 'Standard' allows the customization of options and must be selected if the *VET* Scheduler is required. The 'Create Automatic' option creates a master setup from which you can subsequently install to *Windows NT* workstations.

Selecting the 'Typical' option displays the default settings and options, giving the installer the chance to move back to earlier screens and select another path should these not suit. Clicking the Finish button starts the installation proper.

This option results in the creation of a program group that includes *VET Help*, *VetNT AntiVirus*, *Virus Encyclopaedia*, *Web Update* and *Technical Support*. A short-cut for *VET Scheduler* is also included here if the 'Standard' option is chosen at install time.

Although not investigated for this review, the third installation option can be used to make completely automatic installation scripts to assist in unattended roll-outs and the like. The description of this feature suggests a reasonable amount of flexibility is given to the network administrator, including being able to leave some fields empty, thus requiring input during the installation process.

Vet for NT

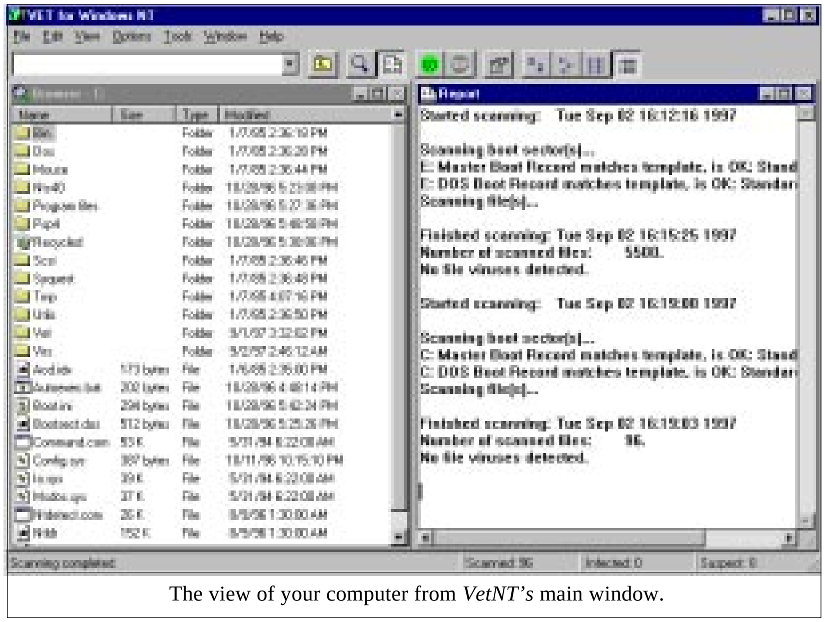
When complete, the setup program recommends restarting *NT* to enable the on-access scanner. Once this is done and *VetNT* is run, the screen displays the single line Type-in Command Box, the Browser Window and the Report Window. The former allows command line options to be entered to change some of *Vet's* operational parameters. This is particularly relevant when running the scheduler, since *VetNT* and any other applications must be run in 'invisible' mode.

The main program options are accessed through the Options/Program dialog. The initial defaults are to do a 'fast' rather than 'full' scan of all sub-folders, skipping renamed files, cleaning infected programs and document files, warning the user of 'suspect' files and recording suspect and infected files in the log. Further, only identified boot viruses will be cleaned, although there are some interesting sounding options for detecting 'invalid' and 'unknown' boot sectors – unfortunately beyond the scope of this review to investigate further.

The Browser Window provides an *Explorer*-style display of drives and files for making scanning selections, and information on scanning activity is displayed in the Report Window. *VetNT* has three scan modes – Immediate, Scheduled and Resident Protection. The Scheduled scan depends on a separate program and is only installed if the 'Standard' installation is selected. The Resident Protection option is activated and its configuration amended from the main *VetNT* program. However, many of its configuration changes only take effect after the next *NT* restart.

Immediate Scan

In this version, the default executable file extensions for scanning were BIN, COM, DLL, DOC, DOT, DRV, EXE, OVL, SYS, and XLS. Additional file selections could be



Name	Size	Type	Modified
Bin		Folder	1/7/98 2:36:10 PM
Dos		Folder	1/7/98 2:36:20 PM
Mouse		Folder	1/7/98 2:36:44 PM
Win40		Folder	10/28/96 5:23:08 PM
Program files		Folder	10/28/96 5:27:36 PM
Pipel		Folder	10/28/96 5:48:58 PM
Recycled		Folder	10/28/96 5:30:00 PM
Scsi		Folder	1/7/98 2:36:46 PM
Sysgen		Folder	1/7/98 2:36:48 PM
Temp		Folder	1/7/98 4:07:16 PM
Utils		Folder	1/7/98 2:36:50 PM
Vet		Folder	9/1/97 3:32:02 PM
Virs		Folder	9/2/97 2:46:12 AM
Acd.idx	173 bytes	File	1/6/98 2:35:00 PM
Autoexec.bat	202 bytes	File	10/28/96 4:48:14 PM
Boot.ini	294 bytes	File	10/28/96 5:42:24 PM
Bootsect.dos	512 bytes	File	10/28/96 5:25:26 PM
Command.com	53 K	File	5/31/94 6:22:08 AM
Config.sys	397 bytes	File	10/11/96 10:15:10 PM
Io.sys	39 K	File	5/31/94 6:22:08 AM
Msdos.sys	37 K	File	5/31/94 6:22:08 AM
Ntdetect.com	26 K	File	8/9/96 1:30:00 AM
Ntldr	152 K	File	8/9/96 1:30:00 AM

```
Started scanning:    Tue Sep 02 16:12:16 1997

Scanning boot sector(s)...
E: Master Boot Record matches template, is OK: Stand
E: DOS Boot Record matches template, is OK: Standar
Scanning file(s)...

Finished scanning: Tue Sep 02 16:15:25 1997
Number of scanned files:       5500.
No file viruses detected.

Started scanning:    Tue Sep 02 16:19:00 1997

Scanning boot sector(s)...
C: Master Boot Record matches template, is OK: Stand
C: DOS Boot Record matches template, is OK: Standar
Scanning file(s)...

Finished scanning: Tue Sep 02 16:19:03 1997
Number of scanned files:       96.
No file viruses detected.
```

The view of your computer from *VetNT's* main window.

made, or all files checked. Interestingly, *Windows* virtual device drivers (VxDs with the usual filename extension of 386) are not on the default list of files to scan.

The scanner has two modes of operation, Full Scan or Fast Scan. The former checks every byte of a file, whereas Fast Scan limits itself to the file's entry point and those areas most vulnerable to viruses. In the event of detecting a virus, *VetNT's* actions can be set depending on the level and location of detection. Aside from invoking a scan from *VetNT's* main program window, immediate, background scans of the currently-selected drive, folder or file can be started in Explorer by right-clicking on the appropriate object and selecting the *Vet* option from the menu.

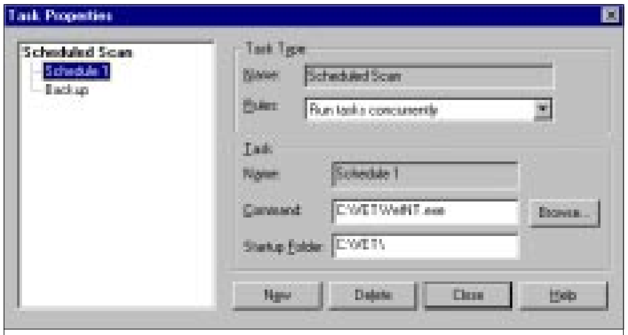
For infected program files, four options are available. 'Report only' performs no further action on the file, whereas 'Clean' allows *VetNT* to attempt disinfection. If it is not possible to do this, for example in the case of an overwriting virus, it will delete the infected file. The 'Rename' option changes the first letter of the file extension to V, and 'Delete' overwrites the contents of the file with 'D' characters and sets the file length to zero.

If there is only a suspicion of infection (the file contains some characteristics of a virus, but not sufficient to make a positive identification), the 'Clean' option is not available and 'Delete' may be used with caution, since there may be a false positive. In the case of infected documents, the only available options are 'Report only' and 'Clean'.

When scanning the test-set, if the files were copied from the CD-ROM with their 'Read Only' attribute still set, the 'Delete' option did not function even though testing as the *NT* Administrator. The files had to be set to 'Read Write' before *VetNT* could delete them. Although this is not the only *Windows NT* scanner with such a problem, this 'functional omission' should be rectified.

Scheduled Scan

The *VET* Scheduler is based on defining events or 'tasks' which are run at particular times for a specific number of times. Each task is given a name, a command line call and a working directory as a Start-up Folder. These commands are not limited to running the scanner, and other tasks, such as making backups, can be included in the Task Group.


Task Properties

Scheduled Scan
- Schedule 1
- Backup

Task Type
Name: Scheduled Scan
Rules: Run tasks concurrently

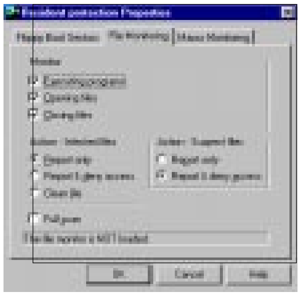
Task
Name: Schedule 1
Command: C:\VET\VetNT.exe
Startup Folder: C:\VET\

How tasks look in the *VET* Scheduler.

Several tasks can be grouped together under a uniquely-named Task Type. These are then subject to rules which allow either just the first task to be run, or all of them, concurrently or sequentially. It can be set to run once only, every minute, hour, day, weekday, week, month or multiples thereof, or to stop after a defined number of repeats. A preview of the defined schedule can also be displayed.

Resident Protection

VetNT's on-access scanner independently monitors floppy boot sectors, program files and 'document' files (those that can contain macro viruses). The same boot sector options as mentioned above are available with the on-access scanner.


Resident protection Properties

Monitor
☑ Executing programs
☑ Opening files
☑ Closing files

Action - Infected files
● Report only
○ Report & deny access
○ Clean file

Action - Suspect files
○ Report only
● Report & deny access

☐ Full scan

File file monitor is NOT loaded

VetNT can monitor the execution of programs as well as the opening and closing of files, separately. Subsequent actions depend on whether a file is positively identified as infected or only suspected of having an infection. In the latter case, the choices are 'Report only' or 'Report and deny access'. If a virus *is* positively identified, cleaning (disinfecting) is an additional option. In the case of Macro monitoring, the same three choices – 'Report only', 'Report and deny access', and 'Clean' are available.

Administration, Logs and Updates

When logging in with Administrator rights, no additional password is required to access scanner administration. As well as the scanner options, there is a tools menu that enables the repair of corrupted boot sectors, and the storing of current boot sector templates. *VetNT* can then determine whether they have subsequently changed. Copies of the templates may be transferred to a floppy disk for reference.

The Report Window displays individual files as they are scanned as well as showing a summary of scanned files, whether suspect or infected. In addition, the scan's start and finish times are reported. The default name for report files is VET_LOG.1 in the installation directory.

Updates are performed using the same procedure as initial installation. The program scans for existing versions of the software and offers to install the upgrade in the same directory as the existing version, if not the default.

Scanning Tests

A diskette consisting of 43 EXE and COM files was used to test the scanner's detection speed. The scan was repeated with the files infected with Natas.4744. The times were as follows: uninfected – 60 seconds; infected – 71 seconds; overhead – 18.33%. On the *VB* Clean test-set, the time taken to scan 5500 clean files was 3 minutes 9 seconds. No false positives were reported for this test-set.

The scanner was checked using *VB's* new test-sets. Tests were run using the default scanner file extensions in Fast Scan mode. The scan action option deleted the infected files, and the residual file count determined the detection rate. In the case of the macro viruses, the count of uninfected files was taken from the scan report.

The Boot Sector test gave 100% detection success and the Polymorphic test only missed two samples of Cryptor.2582. The single sample of W97M/Wazzu.C was missed in the In the Wild File test-set. The macro misses were mainly in the *Excel* samples, and the scanner failed to detect 24 samples from the Standard test-set.

Real-time Scanning Overhead

To determine the impact of the scanner on the workstation, the following test was timed. 200 files of 21.24 MB (EXE and COM files) were copied from one folder to another using XCOPY. The folders used for the source and target were excluded from the virus scan, so as to avoid the risk of a file being scanned while waiting to be copied.

The default setting of Maximum Boost for Foreground Application ensured consistency in all cases. Due to the different processes which occur within the server, the tests were run ten times for each setting and an average taken. The tests were:

- Program not loaded: establishes the baseline time for copying the files.
- Program installed but not scanning; Resident Protection not enabled: tests the impact of the application in a quiescent state.
- Program loaded, Resident Protection enabled; Opening Files and Closing Files both off: tests the impact of having the monitor software loaded but inactive.
- Program loaded**,** Resident Protection enabled; Opening Files on and Closing Files off: tests the impact of the scan when opening files.
- Program loaded, Resident Protection enabled; Opening Files off and Closing Files on: tests the impact of the scan when closing files.
- Program loaded, Resident Protection enabled; Opening Files and Closing Files both on: tests the impact of the scan when opening and closing files.
- Program loaded, Resident Protection enabled; Opening Files on and Closing Files on; Manual scan also included: tests the full impact of the scan when opening and closing files as well as the normal scanning of files.
- Program unloaded: run after the other tests to check how well the server is returned to its former state.

See the table in the summary box for detailed results. The overhead results appear to be a little incongruous. It seems that scanning when opening files is quicker than not scanning at all. This interesting effect is also seen when opening and closing a file as compared to just scanning when closing the file.

Summary

Despite having an installation option for multiple workstation deployment, *VetNT* appears to have no effective support for multi-server establishments. However, the separate scheduler does provide a useful facility for managing multiple, repetitive tasks.

VetNT maintains *Cybec's* tradition of providing one of the fastest scanners available, and it is still combined with good detection performance. The heuristics seem to detect viruses effectively, while avoiding the problems of generating false positives.

Vet for Windows NT Server v9.44

Detection Results

Test-set[1]	Viruses Detected	Score
In the Wild File	548/549	99.8%
Standard	750/774	96.9%
Polymorphic	12998/13000	99.9%
Macro	704/716	98.3%

Overhead of On-access Scanning:

The tests show the time (in seconds) taken to copy 200 COM and EXE files (21.2MB). Each test was repeated ten times, and an average taken.

	Time	Overhead
Baseline	15.2	–
Loaded, disabled	16.0	5.2%
— + enabled, no scanning	16.1	5.9%
— + — + scan on open	15.8	3.9%
— + — + scan on close	22.5	48.0%
— + — + scan on both	17.2	13.2%
— + — + — + on-demand scan	23.1	52.0%
Program unloaded	15.2	0.0%

Technical Details

Product: *Vet for Windows NT Server v9.44.*

Developer/Vendor: *Cybec Pty Ltd*, 2nd Floor, 1601 Malvern Road, Glen Iris, VIC 3146, Australia. Tel +61 3 9825 5600, fax +61 3 9886 0844, email info@vet.com.au, and WWW http://www.cybec.com.au/.

Distributor UK: *Vet Anti-Virus Software Ltd*, 342 Glossop Road, Sheffield, S10 2HW, England. Tel +44 1142 757 501, fax +44 1142 757 508, email Support@vetavs.co.uk.

Price: £250 for a single server with quarterly mail and monthly WWW updates. Monthly mail updates £20 extra. Site licences and volume discounts should be negotiated with the distributor. Subsequent yearly renewals are 50% of RRP.

[1] **Test-sets:** Described in detail in *VB*, September 1997, p.16.

PRODUCT REVIEW 2

F-PROT Professional v3.0 for Windows 95

Dr Keith Jackson

F-PROT was originally developed by a previous Technical Editor of *VB*, and for all I know, he is still working on it. It has been a best-selling scanner for many years (my last review was in February 1994), and is available, amongst other incarnations, in a *Windows 95* version. This review covers that version, and does not look at any of its network-aware features. *Data Fellows* claims that *F-PROT* is the 'Ultimate Protection against Viruses'. Can the reality live up to this brash claim?

Installation

F-PROT was provided for review on one CD-ROM. Installation proved to be very simple. The first task was to choose from installing *F-PROT* for *Windows 95*, DOS or *Windows 3.1*, making a Rescue Disk, and reading the Release Notes. I selected the former.

The InstallShield Wizard then took over (the shields canna take it Cap'n?), very shortly warning me that five DLLs had been 'replaced with older versions by a program you recently ran', and suggested they should be 'repaired'. The problem was that on my test PC, the entire *Windows 95* system is renewed from CD-ROM between product reviews. So what was the program that I 'recently ran'?

Next, I selected which parts of *F-PROT* were to be installed (all of it), and where its files were to be stored. The mandatory pretty bar-graphs whizzed up and down, a 'Licensing Information' screen had to be completed in the usual manner, and that was it. Error boxes appeared every time the system was rebooted. Now, however, there were four consecutive warning boxes babbling on about wrong versions of various system files.

Operation and Tasks

Rather than letting the scanner loose on targets, this version of *F-PROT* centres around 'tasks', providing features roughly halfway between a scanner and a scheduler.

Each task has a unique name, and a set of associated parameters which define when and how it should be executed. By default, tasks are provided which scan drive A, scan all hard disks, scan network drives, and scan a folder. Tasks can be either executed immediately or at their prescribed times, and copied and edited ad infinitum. I soon grew to like this method of working, happy in the knowledge that even if I forgot to do something, *F-PROT* would carry out a scan as and when scheduled.

On installing all of *F-PROT*, *Windows 95* menus make available F-Agent, *F-PROT Professional for Windows*, scan diskette in A, scan hard disks, and scan network. The latter three are tasks – instantiations of the *F-PROT* scanner designed to carry out a particular job. F-Agent runs in the background to control the *F-PROT* tasks.

This leaves *F-PROT Professional for Windows* which on first activation appears only as a toolbar, but this can be easily toggled to show the more familiar type of *Windows* display screen. The Toolbar only contains icons to activate the most frequently-used parts of *F-PROT*.

F-PROT has a few menu options that do not do anything. Thus, it is possible to select the scanning method, but the only available option is 'Secure Scan'. Likewise, English is the only Language option in the reviewed version.

Documentation

A single, 180-page, A5 book was provided, containing lots of graphics, but only a meagre Index. Every time I used the Index, it did not contain a reference to what I wanted to find. For instance, *F-Agent*, *Gatekeeper*, *scan*, *scanning*, *setup*, and *virus* are not featured. The omission of the last two words is particularly damning.

Manuals that try to cover multiple versions of a package always run into problems. This one is no exception. No matter what version of *F-PROT* you are using, large swathes of the manual will not refer to your product. Unfortunately, long tracts of text that are irrelevant to some users would become even more voluminous with the addition of some missing, but much-needed, information.

However, the parts of the documentation which deal with what to do when a virus is found are concise, well-written, and contain just the right amount of technical information. The first subtitle states 'Don't Panic'. Excellent advice. Many times I have heard stories of people who format an infected hard disk in a mad panic, only to find that with most common boot sector viruses this removes everything but the virus! As the *F-PROT* manual says ‘Go get a cup of coffee, the virus will wait.’

Scanning

All scanning detection tests were carried out using a shiny, new virus test-set provided by *Virus Bulletin* – literally shiny, it is on a gold CD-ROM.

F-PROT detected all 549 samples from the In the Wild File test-set. 100% perfect. However, it detected only 685 of the 774 samples in the Standard test-set (just 88%), indicating one file as a ‘suspected infection’, but nothing defined this term. This was a sample of the Peanut virus which *F-PROT* declared to be ‘Possibly a dropper program for a new variant of Aircop.’ As *F-PROT* does not add a list of all scanned files into either its report file or its on-screen results, I cannot comment on the particular types of virus it missed here.

The Macro test-set is a new category for *VB*, emphasizing the recent significance of these viruses. *F-PROT* detected 705 (98%) of the 716 samples.

On the Boot Sector test-set, 88 of the 91 samples were correctly detected (96.7%). This is a surprise result, as usually *F-PROT* hits a perfect score here – it was most unusual to see a product miss Michelangelo.A!

The polymorphic test-set currently contains 13,000 viruses (500 samples of 26). *F-PROT* correctly detected 7,053 samples (54.2%) as infected files. Three files were again detected as being ‘suspected’ infections. Given that there are 500 test samples of each polymorphic virus, picking out just three for special mention seems most odd.

F-PROT performed well when detecting the polymorphic test samples of Arianna.3076, Cordobes.3334, Digi.3547, Girafe:TPE, Coffee_Shop, MTZ.4510, Natas.4744, Neuroquila.A, Nightfall.4559.B, One_Half.3544, Pathogen, SatanBug.5000.A, SMEG_v0.3, and Tequila.A. It fared poorly with three other test polymorphics, detecting fewer than ten of the 500 samples each of Alive.4000, Cryptor.2582 and Uruguay.4. It failed to detect any samples of the remaining nine polymorphic stems. Overall, the polymorphic detection rate could, and should, be a lot better.

The *VB* false positive test-set comprises 5,500 executable files, held on CD-ROM, all of which have been copied from well-known software products. *F-PROT* did not report a single file as infected.

Speed

Using the default settings, *F-PROT* scanned the C: drive of my test PC in 33.5 seconds. For comparison purposes, the DOS versions of *Dr. Solomon’s Anti-Virus Toolkit*, and *SWEEP* from *Sophos*, took 31 and 53 seconds respectively.

F-PROT is no slowcoach, and its hard disk scan time is very good indeed compared to its direct competitors. These figures are actually even better than they at first appear. The scan time quoted for *F-PROT* was taken on a stop-watch, while those of the two competitors were the times reported on-screen, which manufacturers *always* underestimate. Oh yes, they do.

I cannot leave scanning without mentioning that *F-PROT* twice locked up solid during a scan. No warning message, things just froze during the initial ‘Scanning memory… ’ part. No error message appeared, and as Ctrl-Alt-Del did nothing, a power cycle was required to restart the PC.

Memory-resident Software

Gatekeeper, *F-PROT’s* memory-resident component, is configured within *F-PROT* itself. To test its detection capabilities, I copied the file-virus test-sets from CD-ROM to hard disk, and observed which files were spotted as infected. Such files can be left alone, renamed or deleted.

No matter what settings were used, on detecting a virus, Gatekeeper invoked its own controlling software to produce an on-screen report and then caused an error to be reported by the program that initiated the file copy. This methodology is very secure, but it makes Gatekeeper difficult to test.

I used the DOS program XCOPY with the ‘/C’ switch (that means continue copying even if errors are reported), but the Gatekeeper error box still popped up, reporting each infected file to which access had been denied. The ‘/C’ switch did its stuff, however, and files were copied.

Using XCOPY sometimes induced an error message saying ‘A fatal exception 0E has occurred at 0028:C002233D’, and requesting that I choose from ‘Any key to terminate or Ctrl-Alt-Del to reboot’. This happened many times, and often gave different hex addresses. This also seemed to remove Gatekeeper protection, so that whenever I tried to re-enable Gatekeeper, the PC just froze, requiring a power cycle to restart. These problems need fixing.

Memory-resident Detection

Gatekeeper seems to detect viruses by (at least) two methods. Accessing a folder whose sub-directories contained infected files causes a warning to pop up. After clicking away the hundreds of ‘File Overwritten and Deleted’ messages, I found that under these circumstances Gatekeeper had spotted that 158 of the 549 In the Wild File samples (29%), 301 of the 774 Standard samples (39%), and 382 of the 716 Macro samples (53%), were infected with a virus.

However, the number of files XCOPY reported as copied was far less than the difference between the number of files detected plus the total number in the test-set. Huh? I can only conclude that access was being denied to some files without producing a warning message.

Even more confusingly, when the Standard test-set was copied to hard disk using XCOPY, 94 files were reported as being copied. When these files were copied again from one sub-directory to another, more error messages appeared, more acknowledgement mouse clicks were required, and just 89 files were copied. I have no idea why.

Gatekeeper specified 163 files as virus-infected from the In the Wild test-set, but only three files were copied. The rest were ‘magically’ barred. These three were copied again, one more was removed, leaving just two files that Gatekeeper thought were not infected.

In the Macro test-set, 322 files were removed, but just twenty copied. On re-copying them, three more were removed by Gatekeeper, leaving just seventeen files.

Even after multiple copying and file removal, these results are nowhere near as good as the on-demand scanner’s. I returned to *Explorer* to see what had actually been copied, and the DOS results were confirmed. There were 108 files (two from the In the Wild File, 89 from the Standard, and seventeen from the Macro test-sets) that could be copied around with impunity. Even immediately after *F-PROT* was reinstalled, Gatekeeper did not spot any of these files as being virus-infected. Unfortunately, page 41 of the manual states that Gatekeeper ‘Detects the same viruses as does the fully executed scan.’ This claim is simply not true.

Explorer, used by itself, produced very confusing results. Concentrating on the In the Wild test-set, I found when I dragged the ItW folder icon on the *VB* CD-ROM, and dropped it onto a hard disk, just one error message was displayed, relating to the first virus found. *Explorer* then stopped copying. If the same procedure was followed with the infected files dragged as a set, the by now familiar flurry of individual ‘virus found’ error messages came on-screen before the list was displayed. Once again, each message had to be acknowledged with a mouse click before it would go away. However, there were 211 error messages in all, a completely different total from the number produced in a DOS box.

As for XCOPY, this 211 total was reduced when the list was visible in *Explorer*, and the first and last files were marked so that *Windows* selected the entire set. Under these circumstances, another 71 error messages popped up.

The manual states that ‘*F-PROT* Gatekeeper’s functioning is based on Dynamic Virus Protection Technology.’ This is almost a content-free sentence, of no help to users, and none of the last four words actually feature in the Index. If you are wondering what they mean, they are just buzz words, mere marketing hype designed to look important.

The Rest

F-PROT users are well served by modern telecommunications. Not only are Sales and Technical Support email addresses available, but the vendors have a presence on the World Wide Web, alongside the so-called ‘Web Club’ which provides help, including access to the latest information about viruses. (Shades of Blue Peter here for those readers reared on UK television.) On installation, *F-PROT* provides a local copy of its Virus Information database, a voluminous guide to the properties of known viruses. Whenever a task finishes executing, a report file is made available, with details of all infected files. Double clicking on the virus name displays details of its operation.

The log file created by *F-PROT* is maintained in its own internal form and is only formatted for visual inspection when you choose to view it. This obviously saves a large amount of disk space, but it proved impossible to obtain an ASCII text file of the log. The closest I got was to use the ‘Copy’ button which made a Clipboard file copy of a viewable version of the log file.

Conclusions

My introduction stated that *F-PROT* has been a ‘best-selling scanner for many years’. It has also been one of the consistent top-performers. The above results show this good level of performance remains, although the incessant deluge of viruses, particularly polymorphics, is making the game of catch-up harder as time goes by.

The Gatekeeper memory-resident software contained in the review version gave spurious errors when lots of viruses were detected, and did not live up to the claim in the manual that its detection rate was as good as the scanner’s. *F-PROT* is not alone in this respect, but other products do not make the same exaggerated claim. The memory-resident detection rate *per se* was reasonable.

As for keeping up with the assertion that *F-PROT* is the ‘Ultimate Protection against Viruses’, well no, it is not. In reality no single product could be. However, I have no hesitation in recommending the *Windows 95* version of *F-PROT* as a competent anti-virus product – in spite of the marketing hype in the manual!

Technical Details

Product: *F-PROT Professional for Windows 95 v3.0.*

Vendor: *Data Fellows*, Päiväntaite 8, FIN-02210 Espoo, Finland. Tel +358 9 4784 4513, fax +358 9 4784 4599, email info@DataFellows.com, support@DataFellows.com, WWW http//www.DataFellows.com/.

Availability: Shipping now.

Version evaluated: 3.0 (scan engine revision 2.27).

Price: Single user licence, including monthly updates for one year: £180, with quarterly updates: £140. For corporate and site licences contact the vendors.

Hardware used: A 133 MHz Pentium with 16 MB of RAM, a 3.5 inch floppy disk drive, a CD-ROM drive, and a 1.2 GB hard disk divided into drive C (315 MB), and drive D (965 MB). This PC can be configured to operate under *Windows 95*, *Windows 3.11*, *Windows 3.1*, or DOS 6.22.

Viruses used for testing purposes: For a detailed listing of test-sets used for this review see *VB*, September 1997, p.16.

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

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, The Pentagon, Abingdon Science Park,
Abingdon, Oxfordshire, OX14 3YP, England

Tel 01235 555139, International Tel +44 1235 555139
Fax 01235 531889, International Fax +44 1235 531889
Email: editorial@virusbtn.com
World Wide Web: http://www.virusbtn.com/

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

***CompSec 97* will be held in London** from 5–7 November 1997. The conference aims to help highlight the risk to IT systems, assess security shortcomings, and protect against fraud, disaster, and negligence. Information is available from Amy Richardson at *Elsevier Science*; Tel +44 1865 843643, fax +44 1865 843958, or email a.richardson@elsevier.co.uk.

***Integralis* and *Data Fellows* recently announced a strategic partnership**. *Integralis' MIMEsweeper* is to be integrated into *Data Fellows'* new *F-PROT Mail Checker*, providing high-speed, real-time virus scanning for email servers and comprehensive protection against email-borne viruses. Email info@mimesweeper.com, or contact Heidi Alpe at *Integralis;* Tel +44 118 930 6060.

The **International Conference on Forensic Computing** will be held in Brighton from 3–5 December 1997. More than sixteen speakers will cover topics from evidence recovery and analysis techniques to computer forensics methodology and criminal case studies. For further information, contact *The International Journal of Forensic Computing*; Tel +44 1903 209226, or email ijfc@pavilion.co.uk.

***Sophos Plc* is holding two computer virus workshops** this month at its training suite in Abingdon, UK, with an introductory course followed by an advanced one, on 19 and 20 November respectively. A *Practical NetWare Security* course is also being run at the same site on 13 November. More information is available from Karen Richardson; Tel +44 1235 544015, fax +44 1235 55935, or at the company's Web site http://www.sophos.com.

Computer Associate's ***Cheyenne InocuLAN* has been chosen by *Microsoft*** to protect its global network of over 90,000 desktops and servers. Manageability, automated software distribution and ease of signature file updating were among the reasons cited for its selection.

The **13th Annual Computer Security Applications Conference** is to be held from 8–12 December 1997, at the Hyatt Islandia in San Diego, California. Subjects covered include technology applications in several aspects – policy issues and operational requirements for civil and military systems, as well as hardware and software tools and techniques under development. Email Student_chair@acsac.org, or access the Web site at http://www.acsac.org for details.

Reflecting the increasing costs and complexities of distributing multiple-diskette packages of various products, ***Dr Solomon's* announces *Toolkit* updates in CD format**. This is in line with changes made by other anti-virus software companies, and looks set to become standard procedure.

Symantec's artificial intelligence and heuristic technology has detected 80% of new and unknown file-infecting viruses in recent internal tests. Combined with their existing detection methods, the company claims close to 95% detection of new macro viruses. This technology, called **Bloodhound, will be incorporated into the company's *Norton AntiVirus* products from October**.

CyberSoft Inc **has released *Jscan*, a Java applet scanner**. The initial release detects twelve known attacks and variations, as the scanning component is capable of detecting 'both specific and generic variations of known threats'. The heart of *Jscan* is *Jdis*, *CyberSoft's* Java class disassembler, claimed to be up to twenty times as fast as its rivals. *Jdis* is written in ANSI C and can be licensed by developers wishing to add Java class scanning to their products. More details can be found on the WWW at http://www.cyber.com/.

The *Virus Bulletin* offices will be relatively quiet from 30 September until 8 October with the staff away at the *VB'97* Conference and taking a few days well-deserved break afterwards.