ISSN 0956-9979 NOVEMBER 1997

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

IN THIS ISSUE:

• **Troublesome trio**: There are three interesting virus analyses this month: Baboon (as reported on last month's News page), and two Win32 viruses signifying advances in virus techniques for *Windows 95* and/or *NT*. The first of the analyses starts on p.10.

• **Conferencisco:** The seventh annual *VB* conference in San Francisco was better attended than ever before. Read our conference report on p.6.

• **Up to the Mike:** Reformed virus writer Mike Ellison participated in the *VB* conference, and features in our Insight column on p.8.

CONTENTS

EDITORIAL

A Snappy Title…

…and little else! A month bereft of ideas for a catchy editorial. Sad. Like the *Windows 95* comparative that should have been in this issue. MIA. Thwarted at almost every turn, it seemed. I had innumerable small problems with the product testing itself and generating reliable results. Mostly nothing serious in terms of product usability. Mostly the minor peculiarities that seem to go with testing antivirus products against large numbers of viruses. Just much more so than usual.

The sort of situation that *VB*'s readers are (hopefully) unlikely to face in their organizations once they've selected and installed one of these products. Like the scanner that would not produce a report of what files it scanned or what viruses it found, regardless of the configuration options selected. Except it always produced the first few lines of what it should have reported – namely the date and time the scan started and the product name and version. Useful.

> "Virus Bulletin *will be awarding* VB 100% *marks…*"

Or the product that deleted the first few hundred infected files in the total test-set then steadfastly refused to delete any more, regardless of settings or where in the test-set it was directed to start scanning. Despite being completely uninstalled and reinstalled several times. Not to mention reinstalling the operating system almost as many times as well.

And the on-access scanner whose 'quiet operation' mode – especially useful for machines that have to run unattended, according to the manual – always pops up a system-modal dialog box informing that it has detected a virus. But, as I said, I was throwing a huge load of viruses at these products –something they would never have to handle 'in the real world'.

Leaves one wondering what dimension some vendors' quality assurance processes run in though…

This was enough to make a grown editor cry – or at least put off publishing the review until quite sure all the wrinkles and twists are found and ironed out of the results. Thus, the *Windows 95* comparative review that should have featured in this issue will now appear in December.

Despite this hold-up, we are confident of our testing methods. So confident, in fact, that we have a new feature to increase the attractiveness of our comparative reviews for vendors. Starting with the January 1998 DOS comparative, *Virus Bulletin* will be awarding *VB 100%* marks to products having 100% scores on the In the Wild Overall test.

The advantages of this mark over some other well-known product certification schemes are manifold. The associated logo 'earned' by a product scoring 100% In the Wild Overall will clearly state the product, platform and date of award. This allows purchasers to see through marketeering tricks such as using outdated 'awards' in packaging artwork, or using accolades earned by one product line as if they belong to the product in the currently considered package.

Submitting a product for *Virus Bulletin* testing is all but free – in the worst case it runs to the vendor's manufacturing and shipping costs. Some of the more confident vendors send their products as Email attachments! Other much-heralded schemes cost the anti-virus vendors many thousands of dollars *per annum*, and as the livelihood of the testers concerned partially depends upon selling certifications, there has been suspicion that the required standards are set suitably low such that fairly much any product from a vendor who can afford the certification fee will pass.

Related to this, *VB* tests are always run against the most recent WildList at the product submission date. This is a 'tighter' test than any others and thus gives readers of *VB* comparative reviews and furture followers of the *VB 100%* logo a better feel for the products that best keep up to date.

This is the first news of this new scheme, and as such, I'm sure many email messages are being composed to me, as you read these words. The final details regarding *VB 100%* logo entitlement and use will be published in January, along with the initial recipients – the 100% ItW scorers in the January DOS comparative.

NEWS

CounterSign on the Dotted Line

DataFellows and *KAMI*, producers of *F-PROT Professional* and *AVP*, have announced a strategic partnership. Together they have developed *F-Secure Anti-Virus CounterSign Technology* which combines multiple scanning engines into a single framework, using *F-PROT* and *AVP* engines simultaneously. The simple theory behind the idea is that what one scanner misses, the other will be likely to find. The technology also employs 'advanced heuristic analysis to find unknown viruses, while limiting the possibility of false alarms'.

The resulting product, *F-Secure Anti-Virus*, can automatically be installed to multiple platforms from a single workstation, send updates to users, and receive automatic virus reports and copies of infected or suspected files. The even more recent *F-Secure Anti-Virus Macro Control* is, claims *DataFellows*, the first anti-virus product specifically to address the problem of macros by certification. It uses a list of macros approved by the network administrator, so that unknown macros can be 'locked out'. The three-stage procedure involves checking the authentication database first for known macros, then for 'smart templates' or third-party commercial templates, and finally for locally-known macros within the organization. Mikko Hyppönen, the anti-virus product manager for *DataFellows*, is confident; 'We believe that a paradigm shift in anti-virus scanning technology has ocurred and we are the catalyst' ▮

… By any Other Name

In a deal reputedly worth $1.3 billion, two California-based companies, *McAfee Associates* and *Network General* announced a merger on 14 October. Leslie Denend, President and Chief Executive of *Network General* sees it as a marriage made in consumer heaven: 'There is a powerful synergy across our product lines, our markets, and our organizations.' In a stock for stock deal, *McAfee* are offering 0.4167 shares for each *Network General* share, in a procedure estimated to take 90 days to complete.

The *McAfee* camp appears equally enthusiastic, despite the disappearance of the *McAfee* name from the new conglomerate. *Network Associates* is set to become the biggest network security and management software company in the world. Another statistic being bandied about is that it will be the tenth largest independent software company. Bill Larson, *McAfee's* President, CEO and Chairman, retains the latter two titles in the new company, and Denend becomes President of *Network Associates*. It is unclear at this stage whether the well-known range of anti-virus software produced by *McAfee* will be renamed. Talks concerned with the complexities of brand naming and trademarks are under way and a result is expected soon ▮

Prevalence Table – September 1997

Virus	Type	Incidents	Reports
CAP	Macro	52	19.2%
Concept	Macro	26	9.6%
NPad	Macro	16	5.9%
Wazzu	Macro	16	5.9%
AntiExe	Boot	15	5.5%
AntiCMOS	Boot	13	4.8%
Form	Boot	13	4.8%
Parity_Boot	Boot	10	3.7%
Dodgy	Boot	7	2.6%
Junkie	Multi	7	2.6%
Monkey	Boot	7	2.6%
NYB	Boot	7	2.6%
Cebu	Macro	5	1.8%
Laroux	Macro	5	1.8%
Showoff	Macro	5	1.8%
WelcomB	Boot	4	1.5%
Baboon	Boot	3	1.1%
Bleah	Boot	3	1.1%
Feint	Boot	3	1.1%
Johnny	Macro	3	1.1%
Angelina	Boot	2	0.7%
Appder	Macro	2	0.7%
Divina	Macro	2	0.7%
ExeBug	Multi	2	0.7%
Helper	Macro	2	0.7%
Imposter	Macro	2	0.7%
LBB_Stealth	File	2	0.7%
MDMA	Macro	2	0.7%
Natas	Multi	2	0.7%
Quandary	Boot	2	0.7%
Sampo	Boot	2	0.7%
She_Has	Boot	2	0.7%
Stealth_Boot	Boot	2	0.7%
V-Sign	Boot	2	0.7%
Others[1]		23	9.2%
Total		271	100%

[1] The Prevalence Table includes one report of each of the following viruses: Azusa, Colfam, Date, Demon, Diablo, Edwin, Hybrid, Jimi, Kilo, Kompu, Lunch, Michelangelo, Muck, NF, One_Half, Ripper, Sack, Square, Stoned.Spirit, Swlabs, Temple, TrackSwap, and Urkel.

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

AlDith.1502

ER: An appending, 1502-byte virus containing the text '♥♥♥AL-DITH FOREVER♥♥♥'. Infected files have the byte 75h ('u') at offset 0014h.

```
AlDith.1502          8EC0 BB00 00B8 004B 9C2E FF1E 1800 2E8B 166F 00B8 0031 CD21
```

Alla.1325

CDMR: A multi-partite, appending, 1325-byte virus containing the texts 'Wild W0rker /RSA' and 'Alla 1.0'. While infecting the DOS boot sector on a floppy, the virus formats an additional track and stores the original boot sector and the rest of the virus code there. It avoids infecting 360K diskettes. Infected files have their time-stamps set to 62 seconds.

```
Alla.1325            B550 33DB B402 B003 CD13 26C6 062D 0500 BF05 03BE 4C00 FCA5
```

Apparition.700

CR: An appending, 700-byte virus containing the texts '???????COM', '*.COM' and 'THE APPARITION'. Infected files start with the word A0B8h.

```
Apparition.700       E800 005E 81EE EE04 B8AC 0FCD 213D 3535 7503 EB5E 90B8 0000
```

Apparition.1248

CR: A stealth, appending, 1248-byte virus containing the texts 'The Apparition virus, written by **********'. It's a second version of **********.', '????????COM', '*.COM' and 'THE APPARITION'. Infected files start with the word A0B8h.

```
Apparition.1248      E800 005E 81EE EE04 B8AC 0FCD 213D 3535 7503 E9B0 0033 C08E
```

AVCS.2700

CN: An encrypted, appending, 2700-byte direct infector containing the plain-text '[AVCS]' and the encrypted texts '*.com' and 'Demovir for LamerZ'. Infected files have their time-stamps set to 62 seconds.

```
AVCS.2700            B62F 0156 8B96 1702 B971 008B FE84 C7FC AD33 C2AB 3AE4 E2F8
```

Beda

CR: Two variants of a stealth, appending virus. Infected files have their time-stamps set to 23:54:52 (the word BEDAh). The virus' 'Are you there?' call, Int 21h, AX=BEDAh, returns AX=C0FEh.

```
Beda.883             BF00 01F3 A4B8 DABE CD21 3DFE C075 03EB 5E90 B452 CD21 268B
Beda.1301            BF00 01F3 A4B8 DABE CD21 3DFE C075 03EB 7390 B452 CD21 268B
```

Castle.3730

EN: An appending, 3730-byte, direct infector armoured with some anti-debugging tricks. It contains the encrypted texts 'COMMAND.COM', 'CONFIG.SYS' and '????????EXE'. The payload, which triggers in September, overwrites data on drive C.

```
Castle.3730          0692 0E52 468B FCC6 0555 555D 803D 5574 0DBC 920E 86C9 E8ED
```

Cholera.2415

CER: A stealth, appending, 2415-byte virus from Poland, containing the text 'Cholera v3.0 by Dr Fleischman Pozdrowienia dla III a z 3-go LO w Olkuszu 93/94 All rights reserwed (c)1994 02 17'. The virus overwrites 64 bytes of the CMOS data with consecutive values from 0 to 63.

```
Cholera.2415         B8FF FEBA FFFE CD21 3DFF FD74 068D 8E71 03FF D10E 1F0E 07BF
```

DeltaPlus.1328

CER: An encrypted, appending, 1328-byte virus containing the texts 'Delta Plus Virus - 03/97' and 'I love you! Let me guide you through life Let me be your best friend Tell me & Bring me all your problems Let me take care of you And get an eternal life full of joy and peace of mind Jesus Christ'.

```
DeltaPlus.1328       07BE 0700 03E5 8BFE B905 053E 8AA6 0C05 FCAC 32C4 AAE2 FAC3
```

Dialogos.1522

CN: An appending, 1522-byte, direct infector containing the encrypted texts '*.com', 'c:\command.com', 'c:\dos\command.com', 'c:\msdos\command.com', and 'c:\drdos\command.com'. The payload which triggers on the tenth day of any month between June and December, displays the usually encrypted message '1984-1994 10□ Aniversario de DIALOGOS-3 en Radio-3 de RNE. Dedicado a Ramon por estos 10 anos, y por venir a la SALA-4. Buscad la belleza es la unica protesta que merece la pena, en este asqueroso mundo. 10/03/95 Valencia ESPANA'. Infected files end with E9h.

```
Dialogos.1522        479D 75F2 B440 B9F2 05BA 0001 03D6 CD21 B001 8BFE 2885 9404
```

Dieg2.1586

CER: An appending, 1586-byte virus containing the text 'DIEG2'. During August, the virus also intercepts interrupts 1Ch and 28h.

```
Dieg2.1586           B932 0690 BA00 01B4 40CD 2172 2B2E 8B16 8201 2E8B 0E84 01B8
```

ErrorInc.260 **CN:** An appending, 260-byte, direct infector containing the encrypted texts '*.com' and '(c) ERROR Inc. ver 1.0'.

```
ErrorInc.260        E830 008B D581 C207 01B9 0401 B440 CD21 E820 00B4 3ECD 21B4
```

EvilHomer.206 **CN:** An appending, 206-byte, direct infector containing the texts '[EVILH0MeR] SÉpùLτürÆ' and '*.com'.

```
EvilHomer.206       8986 CC01 B440 B9CE 008D 9606 01CD 21B8 0042 9933 C9CD 21B4
```

FalusPotrus.1181 **CR:** An appending, 1181-byte virus containing the text 'FALUS POTRUS Is Fucking You.'. The virus infects only files starting with the byte E9h (near jump). The destructive payload overwrites a random number of sectors on the first physical hard disk with the contents of the Interrupt Vector Table.

```
FalusPotrus.1181    B440 B99D 048B D581 EA0F 01E8 EAFD 72C6 B800 4233 C9BA 0100
```

Hackerz.709 **CR:** A stealth, encrypted, appending, 709-byte virus containing the plain-text string '[HackerZ]' located at the end of infected programs. Infected files have their time-stamps set to 62 seconds.

```
Hackerz.709         3700 663E 8B96 9202 8DB6 0D00 B941 0131 1483 C602 E2F9 C35B
```

Hellis.616 **CR:** An appending, 616-byte virus containing the text 'HELL is dedicated to DR.Farmanesh !!! 29/4/95'. Infected files have their time-stamps set to 62 seconds.

```
Hellis.616          B860 9F8E C026 813E 0000 4845 7448 33DB 8EC3 268B 1E84 002E
```

Lewd.4455 **ER:** An appending, 4455-byte virus containing the texts 'DRWEB.EXEAIDSTEST.EXE' and '◄► 1996 by Lewd-H. ◄►'.

```
Lewd.4455           B8BA DE50 0633 C08E C026 A184 002E A314 0626 A186 002E A316
```

Lion.996 **CN:** An appending, 996-byte direct infector containing the texts 'COMMAND.COM*', '.com' and 'Lion', and the encrypted message 'Kangaroo crossing ERROR at t mod 13 please contact your Dealer'.

```
Lion.996            B440 BE32 058B 1AB9 0700 BA8E 0403 D5CD 21B0 0233 D2E8 5A00
```

MainMan **CN:** Two simple, appending, direct infectors containing the texts '..' and '*.com'.

```
MainMan.200         B640 B9C8 008A E68D 9603 01CD 21B4 3ECD 21B4 3B8D 96BE 01CD
MainMan.213         B640 B9D5 008A E68D 9603 01CD 21B4 3ECD 21B4 3B8D 96CB 01CD
```

MainMan.315 **CN:** An appending, 315-byte, direct infector containing the texts '..', '*.com' and 'infected mainmanIII.1997'. On Sundays, the virus overwrites eleven sectors of the first four sides of every cylinder of the first physical hard disk.

```
MainMan.315         B640 B93B 018A E68D 9603 01CD 21B4 3ECD 21B4 3B8D 9615 02CD
```

Mixx.570 **ER:** An appending, 570-byte virus containing the text 'MiXx' which is located at the end of infected programs. The payload, which triggers on 28 February, overwrites five sectors of track zero (usually unused) on the first hard disk. This can corrupt disks using Big IDE software drivers.

```
Mixx.570            B440 8B1E FA01 B93A 0233 D2CD 2172 5690 9090 8B1E FA01 B800
```

Monster.323 **CN:** An overwriting, 323-byte, direct infector containing the texts '[MONSTER]\', '*.*' and '*.com'. The virus truncates infected files to 323 bytes.

```
Monster.323         BA23 03CD 21B4 40B9 4301 BA00 01CD 215A 59B8 0157 CD21 59E8
```

NRLG.719 **CR:** A doubly-encrypted, appending, 719-byte virus containing the text '[NuKE] N.R.L.G. AZRAEL'. Infected files have their time-stamps set to 60 or 62 seconds.

```
NRLG.719            F615 FF05 F715 8135 CDB1 FE05 812D 19EC F715 F715 802D EE80
```

Pandemonium.1520 **CER:** A stealth, appending, 1520-byte virus containing the texts 'pandemonium by retch', '17/04/96' and 'F-TBARRALHPKCH'.

```
Pandemonium.1520    B440 B9F0 0599 EB30 B000 3DB0 01B4 57E8 ADFB 80FE C8C3 B43F
```

Phile.209 **CN:** A 209-byte overwriter containing the texts '*.C?M', 'ANTI-VIR.DAT', 'Phile_16c' and 'Phile_16'.

```
Phile.209           5152 B440 B9D1 00BA 0001 CD21 B801 575A 59CD 21B8 0143 59BA
```

Redarc **CN:** Two encrypted, appending, direct infectors containing the texts '*.com' and '-=* Red Arc *=-'. Infected files have the byte DDh at offset 0003h.

```
Redarc.623          5555 BB75 3F60 06B8 2135 CD21 268A 07FA 26F6 1726 0107 26F6
Redarc.665          5555 BB24 07BF 7400 B901 012E 311B 471E B860 008E D889 1E07
```

Uncompleted.613 **CR:** An appending, 613-byte virus containing the text '.COM'and 'S&M'. Its 'Are you there?' call, Int 21h AX=FEEFh, BX=EFFEh, returns AX=534Dh ('SM').

```
Uncompleted.613     B8EF FEBB FEEF CD21 3D4D 5375 0332 C0C3 B0FF C3B4 52CD 2126
```

VCC.274 **CN:** An appending, 274-byte direct infector containing the texts 'Test Virus #1 Hacking Hell I-EAS Virus Creation Centre v0.19ß[T1] [HH] [IE-VCC v0.19ß]' and '*.COM'.

```
VCC.274             2BC9 99CD 21B4 40B9 1201 8D96 0601 CD21 B43E CD21
```

VCC.278 **CN:** An appending, 278-byte direct infector containing the texts 'VCC Sample Virus #1 Hacking Hell I-EAS Virus Creation Centre 0.09ß[TV][HH] [IE-VCC v0.09ß]' and '*.COM'.

```
VCC.278             2BC9 99CD 21B4 40B9 1601 8D96 0601 CD21 B43E CD21
```

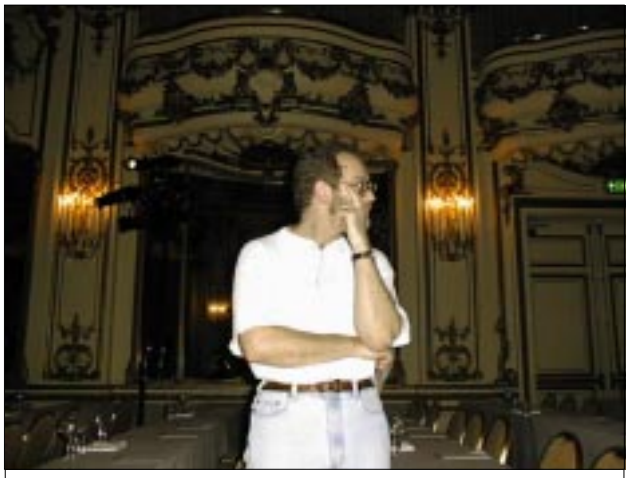
Wangzhen.656 **CER:** An appending, 656-byte virus, containing the texts 'Working Hard For Our CHINA!!! *Wang Zhen Jian*', '*.EXE' and '*.COM'.

```
Wangzhen.656        813E 7204 3412 8CC8 8ED8 754F 803E 0000 FF74 2280 3E00 0000
```

CONFERENCE REPORT

The Full Fairmonty

The luxurious Fairmont Hotel atop Nob Hill in the centre of San Francisco is a sight for anyone's sore eyes. For the *Virus Bulletin* crew, 'fresh' from the Thames Valley, it meant respite for sore something-elses after an airport taxi-ride over the city's extraordinary hills. The hotel provided a stunning backdrop to the seventh annual *Virus Bulletin Conference* – more popular than ever, with over 250 delegates, speakers and exhibitors from Europe, Russia, Australia and of course, the good ol' US of A. With temperatures in the eighties outside, and top-notch air-conditioned facilities inside, the conference swung into action in comfortable style.



Wot… no delegates?

On Wednesday evening, *Symantec* sponsored the welcoming cocktail reception, which, despite a rival bash in the Penthouse, spilled onto the terraces of the Pavilion room as old diehards and fresh faces mingled over champagne and spring rolls. For many, it was the first chance to meet *VB's* new editor, Nick FitzGerald, since his appointment in June. The evening was a great success, cultivating an atmosphere of camaraderie which was characteristic of the conference throughout.

After an all-American breakfast involving careful negotiation of a muffin mountain and vats of fresh OJ, Nick opened the proceedings on Thursday morning with a paper on the two most frequently asked questions he encounters as editor of *VB*. The first, concerning large-scale manageability tests, was quickly put to bed, but he discussed the second question – 'what will be the next big computer security threat? – in more detail.Keynote speaker Paul Ducklin from *Sophos* then took over. His energetic and enthusiastically received talk centred on the relationship between 'us', 'you', and 'them'; anti-virus developers, anti-virus software users and virus writers. He explored the convoluted development of the anti-virus industry, complete with speculations on the motives of virus writers, predictions for product development and caution concerning the volatility of the business.

Delegates split into corporate and technical streams, more often than not shuffling between the Gold and Venetian Rooms to catch parts of both presentations. Thanks to the restrained use of industrial-strength fog horns, most sessions (with one or two notable, but I'm told, not unusual, exceptions) ran to schedule.

Frances Ludgate from *Cybec* opened the corporate session with a lively paper about perceptions of and attitudes to anti-virus evolution and marketing, while *Norman's* Carl Bretteville discussed the possibilities for native *NetWare* viruses. Phil Bancroft and David Aubrey-Jones, both highly respected and common sights at *VB* conferences, rounded off the first morning with talks on the distribution of anti-virus tools and *Office 97's* impact on macro viruses.

DataFellows sponsored a very well-attended and beautifully presented lunch, which was followed by a hugely entertaining paper for the corporate stream from Sarah Gordon and Joe Wells on 'Hoaxes and Hype' – or 'Hypses and Hoax' as Sarah knew she'd end up putting it. *The Imperial Cancer Research Fund's* David Harley followed this tough act with an overview of the *Macintosh* virus situation. Martin Overton of *ChekWare* informed the technical audience of issues with boot viruses and FAT32, prior to Jeffrey Kephart's live demonstration of *IBM's* immune system – a development to keep an eye on.

After tea, where it was loudly and universally noted by both corporate and technical bods that refreshments (ranging from 'cookies' through 'biccies' to 'tucker') were significantly absent, Robert Vibert (*Sensible Security Solutions*) and Bruce Burrell of the *University of Michigan*, discussed virus prevention and cure respectively, ending the day for the corporate stream. *FRISK's* Vesselin Bontchev and *VB's* technical editor Jakub Kaminski, from *Cybec*, closed the technical side.

Once known as Stormbringer, young ex-virus writer Mike Ellison presented the most controversial paper of the conference. At 5.15 pm, the two streams merged to hear his justification for his possible employment within the anti-virus industry. The ensuing Q&A period overran as the audience, split into enthusiastic supporters and unconvinced sceptics, debated the validity of his claim, and the position and value of virus writers in general. An impromptu poll was taken in answer to the question 'If you were negotiating a licence for anti-virus software, and learned that the vendor had an ex-virus writer on the staff, what effect would it have on your decision?' Two people admitted it

would have a negative impact, while thirty to forty thought it would not. Of that number, nearly half felt it may exert a positive influence. The session ended on an energetic if tantalizing note, with a enthusiastic shout of 'you're hired!' from Dmitry Gryaznov of *Dr Solomon's Software*. Watch this space!

Alie Hothersall and 'friends'.

Blocking the door of the Pavilion Room, a sinister, black-clad, 1930s gangster, with an incongruous boyish smile, ensured two hundred and fifty acknowledgements for the Gala Dinner sponsors. Entry to the well-stocked bar was guaranteed with a whispered '*Command Software*'. Suitably mollified, 'Bugsy' proceeded to entertain delegates and partners at their tables with card tricks and conversation. Traditional 'thank-yous' culminated in a hilarious exchange between *VB* ex-editors Richard Ford and Ian ('what I want to know is – why do *Trend* and *McAfee* feel stress balls to be appropriate freebies, particularly?') Whalley as they presented their unsuspecting successor with lavish gifts of software giveaways (oh, and a fairly decent pen [*Thank-you – Ed.*]). Enthusiastic, nay, rowdy appreciation was shown for faultless organization by a far too good-looking crew – conference manager Alie, Petra, John, and 'mike' girls Müsli and Kim.

The prudent omission of candles in close proximity to the dubiously named but delicious vodka beef prevented the resurrection of an old *VB* tradition – trooping out to the tune of a fire alarm. After a hearty spread, gambling commenced with delegates generously unconcerned that their shiny new $500 chips were worth their weight in plastic. Heartfelt thanks are due to Bruce Burrell for his reassuring probability statistics and magic tricks, and to Shane Coursen for subsidizing the editorial assistant even unto dire poverty. There's one born every minute! The partying continued into the small hours, with the impromptu *a cappella* singing becoming increasingly voluminous as its practitioners became increasingly legless.

On Friday at 10 am, avocado-tinged delegates took the scenic route past the muffin mountain and two mugs of the strong black stuff into invigorating papers on hoaxes and the Internet from David Chess and on alternatives to the WildList by David Stang. After topping up at the coffee break, youngbloods Dan Schrader and Carey Nachenberg presented the corporate stream with energetic and informed ideas on anti-virus distribution across intranets and into the 21st Century. The techies were treated to presentations on macro viruses – one from Jimmy Kuo; the other from Allan Dyer and Motoaki Yamamura.

Rejuvenated and refreshed by a superb lunch sponsored by the *NCSA*, the ever-popular Steve White (an original *VB* conference fixture – he's been to every one!) preceded Nick Engleman of *Cybec's* corporate talk on virus education for PC users. In the Venetian Room, Dmitry Gryaznov and Ian Whalley addressed scanning Usenet for viruses and security threats on the Internet. Igor Grebert closed the technical session with a paper on email scanning, while the *NCSA's* Roger Thompson discussed worldwide virus-tracking initiatives with the corporate stream. There was more to come as the speakers' panel session prompted more discussion among a lingering crowd from both streams, eager to prolong the debate. A short slide show presented the images of *VB'97* and this is available for download from http://www.virusbtn.com/VB97/.

Over the honking of the sea-lions on Fisherman's Wharf, the clanging of the cablecars on Powell Street and the swish of limousines at the front step, the sounds emanating from the Fairmont's Gold and Venetian Rooms were innovative, well-researched and superbly delivered to lively and appreciative delegates. Thanks are due to all the speakers and attendees who, after all, make the conference each year. Thanks too to the team who set up and executed the 'do' so apparently effortlessly and so consistently cheerfully.

Thanks to the Fairmont, not least for the prompt arrival of stacks of great 'tucker' at tea-time on Friday, to Ken Bush for his tarot cards, to Kim Ducklin for prudent use of the fog horn in the face of adversity, and to Mikko Hyppönen, for charming Immigration at San Francisco International.

Most of the speakers in a more serious mood. An image map with links to the speakers' biographies can be viewed at http://www.virusbtn.com/VB97/.

INSIGHT

Stormbringer in a Teacup

When news of the rejection of reformed virus writer Mike Ellison's application for a job with a UK anti-virus company filtered through to the offices of *Virus Bulletin*, it seemed like Kismet. With the seventh annual *VB* conference a few weeks away, would the young hopeful put his case to the cream of the anti-virus industry as publicly as he had announced his retirement from virus writing three years before? Ellison duly presented a short paper on justifications for his employment, resembling something between Daniel lecturing to the lions, and a wolf in a lambskin suit. He then fielded a lengthy question and answer session (see the Conference Report, p.6). He confesses to being mildly relieved by the 'fairly decent' treatment he received there:

> …it turned out a bit above my expectation. The only conversation that really annoyed me was a gentleman stating that he wanted to introduce me to his wife, since it was because of me that he never had time to see her as he was analysing viruses. Of course, we got into a discussion of whether his ethics, while he was cracking copyrighted and protected software were better than my ethics, while writing viruses without introducing them into the wild, so at least it was entertaining.

Nevertheless, Ellison left the Fairmont Hotel without a firm job offer, and somewhat disappointed with the rationalizations offered for not employing him:

> I'm rather annoyed that the standing argument against hiring me is that if someone did, they would be immediately attacked by the marketing departments of all the others. This is something I can't deny or dispel – it's true, but it's silly and simply shows the tone of the companies.

Naturally there were those at the conference who were outraged that he was given such a public forum. His motives for speaking – 'I felt I could shake some of the stereotypes that are applied to virus writers as a whole' – may have been *naïve* with hindsight, but he had no illusions about what he was up against:

> I knew there would be a lot of cynicism and criticism, especially from certain people. I did not initially intend to go public. When I applied for a position with *Sophos*, I simply wanted to be honest with them about where I gained my expertise. I could have quite easily gotten a job with an anti-virus company otherwise, but I'd hate to have to worry about some skeleton in the closet.

Rattle, Rattle

By now, everyone knows the reason he stopped writing viruses – 'someone became infected by one of my viruses in the wild. I had never meant for that to happen.' But what drove him in the first place? Ellison was born in Pittsburgh, Pennsylvania on 11 December 1975, and a year later the family moved to Texas, where he has lived ever since. Having started with computers at a young age, and taught himself assembler, analysing any virus he could get his hands on was soon old hat. At the *VB* conference, delegates would accuse Ellison of being disproportionately proud of his programming skills, the very skills which led him to seek fame if not fortune as Stormbringer:

> I got into the VX [Virus eXchange] scene when I was searching for more viruses to analyse – of course, they are the only ones that would give a teenager a bunch to disassemble. I began doing disassemblies and writing little programs such as SelfDisinfectant… to get credits on the VX boards, I re-uploaded my disassemblies. A few of these got published by George Smith, and around this time I finally got the urge to try my hand at writing viruses. My motivation remained purely knowledge until Mark Ludwig's contest – when I began to enjoy the reputation I was getting…

Ellison, a reluctant ambassador, presents an overall picture of a dissatisfied youth culture 'bored' into creating computer viruses. Explaining why certain antiviral packages are targeted over others he said:

> Several people went after *ThunderByte* because it was such an interesting product (especially *TBCLEAN* – Priest, Rinche, and several others exposed some serious flaws in it), and *F-PROT* was similarly harassed because it was seen as the 'leader'. *McAfee* and *Norton* were rarely examined because the technical skill required to escape them was almost nil – hence no challenge and no gratification.

Ellison describes his parents' attitude as encouraging a young mind to discover and explore. His initial interest in computers was supported by both school and family:

> I got into computers at an extremely young age, although I suppose these days it was pretty much average. The first thing I remember doing on computers, besides text games, was 'programming' in LOGO, in second grade in an after school class meant for third and fourth graders (it took a bit to get me in there). Then my father bought a TRS-80 Color Computer for the house and encourage me to learn to program on it

(the operating system was a kind of Basic shell). I enjoyed it considerably, and programmed a lot of silly games and graphics.

When his parents discovered his real 'hobby', they were 'wonderfully supportive' as long as he operated within the code of practice he had already established for himself:

When I told them I was writing viruses, they were strongly against me releasing them in the wild – however, since I was as well, and had made this point, it wasn't a really large deal. I'm not sure if they knew I published the source code until later, but regardless, the point came across: if I wasn't harming people by infecting them with what I created, then I wasn't causing harm.

As his reputation increased, Ellison joined virus writing groups – Phalcon/SKISM and the Trinity. Denying any kind of formal initiation, other than 'proving'oneself inventing 'creative' viruses, and pulling one's weight once established, Ellison is keen to play down the sinister reputation of these groups. He considers the majority of them to be 'just a group of friends with similar interests'. Some, he admits, are political entities with hidden agendas, and others have had to forcibly eject recalcitrant members. John Buchanan, formerly of NuKE, was ousted prior to gaining a reputation as a fraudster on a 'power trip'.

Ellison challenges the sinister labels attached to virus writers in general:

I'm annoyed by the 'criminal' and 'sociopath' labels when they are applied to virus writers as a whole. Sure, some of them are one or both, but definitely not all of them. Nothing I did was illegal (at least in the US), nor was it intended to harm anyone. I was negligent – yes. But I don't think that's enough to be called a sociopath. It if is, I wonder how many teenagers really shouldn't be considered sociopaths.

An element of mystery remains, however: 'We did have our own set of rules; most virus-writing groups do… '. Interestingly, Ellison manifests open disdain for what he calls 'silly macro virus-writing groups', the new kids on the block who recruit through Usenet.

'A Kind of Gratification'

Ellison firmly believes that all virus writers are not the same. Those whom he rather ambiguously calls 'the better ones' still respond to much the same sort of challenge he once called a 'chess game with the AV industry'. Colouring motivations and justifications with youthful confidence, Ellison says he knows what the prevailing attitude in the 'us' and 'them' battle between virus writers and anti-virus companies has always been:

There's a defence there. They say it's secure. I bet it's not – let's prove them wrong and show them we can outsmart them.

It proved harder at the conference to convince the audience of his opinion that most viruses are not intentionally destructive. 'Most contain silly text strings,' he asserts, 'and several have some graphic display that would be funny if the victim wasn't so worried about his or her machine.'

As for himself, Ellison makes no attempt to conceal the petulant, one-up-manship of his past:

Some of my viruses were written specifically with the game in mind. For example, I wrote CoporateLife to specifically avoid blockers and heuristics and to show that I had thought of something 'they' hadn't… Others such as Jump.466, were written specifically to annoy the person(s) with the job of analysing them… Shifting Objective was written partially because I was bored with the standard file/boot viruses, and partially because I knew it would annoy anyone who had to add .OBJ scanning into their scanner.

However, Stormbringer's creations came back to haunt him.

Shutting the Stable Door…

Having disinfected one of his own viruses in response to a panic call from Singapore – 'a nice guy, really, extraordinarily polite for the circumstances' – Ellison found himself chastened and disillusioned with his hobby, and promptly retired in a public apologia posted to the alt.comp.virus newsgroup. Since then, plain old Mike Ellison has been using his skills to good, if bland and unpublicized, effect. Following his experience at the conference, characterized by swings from accusations of terrorism and criminal intent to warmest congratulations and good wishes for the reformee, he has been keeping his cause alive. He admits it is not plain sailing, and that the commonest response still tends to be rather too much in terms of black and white, with Mike Ellison/Stormbringer bound forever as one.

Characteristically, he is undaunted by the split in the anti-virus industry into supporters and detractors of his kind, and singlemindedly optimistic, if cautious, about his future:

I'm currently in the process of coding a few things that may surprise those that criticized my plausibility and programming skills in the AV field, so whether I'm hired by the industry or not, I'm not disappearing from the scene any time soon.

Whether or not Ellison wanted to 'go public' he is making an impact both within and outside the anti-virus industry. In this sense, he deems his appearance at *VB'97* a success:

From a few of the comments made by people after the speech, I think I may have already promoted awareness regarding virus writing and set a bit of an example for others. I intend to continue doing so, and do my best to encourage the new crop of talented, young, and incredibly bored programmers to use their skill in other manners. I have found friends in the AV industry. The question is whether they hire me or not.

VIRUS ANALYSIS 1

Coping with Cabanas

Péter Ször
Data Fellows

1997 seems to be the year of the *Windows 95* virus breakthrough – although, so far, all Win32 viruses have been different in implementation. Virus writers have tried to attack the system in various ways – with some viruses using PE (Portable Executable) infection, or going resident as VxDs (Virtual Device Drivers), while others were able to infect DOS programs too. No Win32 virus to date worked under *NT*. Win32.Cabanas, recently received from its author, changes that.

Cabanas is a per-process resident, anti-debugging, partially packed/encrypted, anti-heuristic, semi-stealth virus. Its author, who also wrote the infamous WM/CAP virus, is a member of the 29A group. Cabanas, like CAP, uses a novel infection mechanism and has what was, initially, a barely comprehensible structure. Usually it is relatively easy to analyse a virus with a disassembler, some DOS viruses taking only a few hours.

This was not the case with Cabanas; it took days to analyse, consuming lots of energy and good hacking utilities. For instance, Cabanas cannot be traced in application level debuggers such as *TD32*; it requires *SoftIce* – 'a debugger on steroids'. This is because typical debuggers focus on the address space of the program and cannot go to the operating system level. Cabanas cannot be loaded into a normal disassembler without it modifying the characteristics of the virus' code section. This trick itself was enough to cause the first headache in my analysis.

Incompatibilities with Windows NT

I have to say that I have a different view of *NT*, in terms of security, now that I understand Cabanas. I drew incorrect conclusions from tests of the first *Windows 95* virus, Boza (see *VB*, February 1996, p.15). Most anti-virus researchers tested Boza on *Windows NT*, which, reassuringly, did not even try to execute the infected image. This led some to think that *NT* had superior virus detection or prevention properties. I patched some Boza-infected files to find out why. The PE format was designed by *Microsoft* for use in all its Win32 systems (*Windows NT*, *Windows 95* and Win32s). However, the implementation of the loader is different from system to system. The *NT* loader simply checks a few more things in PE files than the *Windows 95* one, thus finding Boza-infected files 'suspicious'. One field in the header of Boza's .vlad section is not correctly calculated by its infection routine, but if this was fixed Boza-infected PE files should be able to run under *NT*. However, even if Boza did not have this problem, it would still not be able to replicate under *NT*.

Every *Windows 95* virus has to call two Win32 KERNEL APIs – GetModuleHandleA and GetProcAddress. Since these are in KERNEL32.DLL, it is possible for *Windows 95* viruses to get those functions directly, with a hack. Most *Windows 95* viruses so far have hard-coded pointers to these APIs.

When the linker creates an executable, it assumes that the file will be memory-mapped to a specific location. In the Image File Header of PE files, there is a field called Image Base which holds this address. For executables, this address defaults to 0x400000. *Windows 95's* KERNEL32.DLL has an Image Base address of 0xBFF70000. The two required API addresses will be at fixed offsets from KERNEL32's base address in the same release of *Windows 95*. However, these offsets can be different in other releases, making viruses using these fixed addresses not compatible across *Windows 95* systems. In *NT*, the KERNEL32 Image Base address defaults to 0x77F00000, therefore viruses with a *Windows 95*-specific base address *cannot* work under *NT*.

Furthermore, *NT* does not support VxDs. Thus, viruses like Memorial (see *VB*, September 1997, p.6) cannot operate under *NT*. They would have to include different infection algorithms for *Windows 95* and *NT* in order to succeed on both systems, making them unduly complicated.

If a Win32 virus could overcome these compatibility and implementation problems, it should be able to work equally well on both systems. Such viruses may even have Unicode support, but it would not be mandatory. Win32.Cabanas has all of these features!

The Role of the Import Table

Cabanas relies heavily on the Import Table. In Win32 environments, DLLs are linked through the PE file's Import Table to the applications that use them. The Import Table holds the names of the imported DLLs and the names of the functions imported from them.

The executable code is located in the .text section of PE files (or in the CODE section, as the Borland linker calls it). When an application calls a function from a DLL, it does not call the DLL directly. Instead, the call goes to a JMP DWORD PTR <address> instruction in the executable's .text section. The address is stored in the .idata section (or sometimes in .text). The JMP instruction transfers control to that target address. Thus, the DWORD in the .idata section contains the real address of the API's entry point.

As all calls to a given DLL function are passed through one location, the loader need not patch every instruction. All the PE loader has to do is patch the correct address for each imported function into the list in the .idata section.

Running Infected PE Files

Execution of a Cabanas-infected file starts at the original entry point. Cabanas does not touch the entry point field in the Image File Header, patching the host program at its entry point instead. Leapfrog was the first DOS virus to use this trick. Five bytes at the entry point are replaced with a FAR JMP to the address where the original program ended. You may ask: 'But there can be relocations which may overwrite this location. How can the virus avoid this?' The answer is simple; Cabanas handles the relocation table too.

The first function in Cabanas simply unpacks and decrypts a table of Win32 KERNEL API names. The word 'File' is replaced in the names that would normally contain it. GetProcAddress is not packed at the beginning of the string table, but the next function name is 'encrypted' as 'Ge',t+80h,'AttributesA' or GetFileAttributesA when unpacked. Since Cabanas has Unicode support, the next string is GetFileAttributesW which is described in two bytes: 80h, SizeOfPreviousUnpackedString. The other strings are packed in the same way.

The real problem is that the virus uses Structured Exception Handling (SEH) as an anti-debug function. Not knowing the form of C++'s __try and _except functions in assembly, I ran into this trap several times before it dawned on me – the goal of this function is to set a new SEH FRAME and generate an exception. When execution reaches the instruction which caused the exception, control is redirected to the operating system's Exception Handler (EIP will point into the kernel). This is very annoying and needs *SoftIce* to trace. The operating system's exception handler sets the exception type and returns to the application. As a result, no general protection fault will be displayed and the SEH FRAME will be removed.

When the unpack/decryptor function is ready, the virus calls a routine to find KERNEL32's original Base Address. During infection, the virus searches the Import Table for GetModuleHandleA and GetModuleHandleW. When it finds them, its saves pointers to the DWORDs in the .idata list. If the application does not have either import, the virus uses another, unreliable way to get the address. This is probably the worst bug in the virus. (I should note here that in Win32 environments the Module Handle and Base Address are the same.) When the virus has the Base Address, it calls its own routine to get the function address of GetProcAddress. The first method is based on the search in the Import Table during infection time. In most cases, Win32 applications import the GetProcAddress API, thus the virus should not use a secondary routine to get the same result. If the first method fails, the virus searches for GetProcAddress and GetProcAddressFromExportsTable exports in KERNEL32's .edata section.

Cabanas searches for the GetProcAddress string in the Function Name Table of KERNEL32's Export Table. When it finds the correct string, it gets the entry point from the Function Address Table and returns. This function is one of the most important from the virus' point of view and is compatible with all Win32 systems.

If the entry point of GetProcAddress was returned by the GetProcAddressFromExportsTable function, the virus saves it to use later. If not, the function will be used several times, having been 'secured' with Structured Exception Handling to avoid possible exceptions. The virus can now get the addresses of all the Win32 APIs it needs to use. Cabanas is ready to replicate.

Direct Action Infection

The infection code is surprisingly fast, in spite of the fact it runs through all the files in the *Windows*, *Windows* System and current directories. This is because the virus uses memory-mapped files. The full process takes no more then a few seconds on a 486. First, the virus gets the name of the *Windows* directory with the GetSystemDirectoryA API, then searches it for non-infected executables. This uses the FindFirstFileA and FindNextFileA APIs, searching for non-directory entries and checking file sizes.

Those divisible by 101 are assumed infected. Those larger than 64MB are left alone. Targeted files are opened and mapped using the CreateFileA and CreateFileMappingA APIs. If a file is shorter than 128 bytes, it is closed and infection aborted. Cabanas checks for the 'MZ' marker at the beginning of the image, then repositions to the PE header area. It checks that the executable is for 386+ machines and looks for the internal file type, which must be an executable file, not a DLL.

Next, the virus calculates a special checksum using the checksum field of the PE file's Optional Header and the file-stamp field of the Image File Header. If the file seems to be infected, the virus closes it. If not, Cabanas saves the original file attributes, changing them so it can write to the file. It opens and maps the potential host in write mode and searches for the GetModuleHandleA, GetModuleHandleW and GetProcAddress API imports in the host's Import Table and calculates pointers to the .idata section. Then it calls a routine to patch the virus into the file.

This routine sets the MEM_WRITE flag of the .idata section if it is notalready set, but only if this section is not located in an executable area. This means that there are some extreme cases when this table is part of the .text (CODE) section. The first five bytes at the host's entry point are replaced with a FAR JMP to the end of the host. The infection procedure checks for relocations that may overwrite this FAR JMP. If the relocation table size is non-zero, a routine searches the .reloc area. If a relocation points into the FAR JMP area, its relocation type is cleared so that it will not be used by the loader. This also marks the relocation so Cabanas will be able to find the host later.

A 'parameter block' of information needed to rebuild and start the host application is created, including the original five bytes from the host's entry point and its location. The self-recognition 'checksum' is calculated, as is the new file size. The size of the virus code is around 3000 bytes, but most infected files will grow by a little more than this because Cabanas pads itself to make the infected file size evenly divisible by 101.

The virus does not create a new section header for its code, but modifies the last section header in the file (usually .reloc) to be longer, 'making' enough space for the virus' code and setting the section's characteristics to include the MEM_WRITE flag. This makes infection less risky. The SizeOfImage field in the header is corrected and the file unmapped and closed. Finally, the file is truncated to the previously calculated size and the original time, date and file attributes reset.

Rebuilding the Host and Going Resident

Following the 'seek and infect' phase, Cabanas uses the GetCurrentProcess and WriteProcessMemory functions to write the original five bytes to the host's entry point. After this, it relocates the code area, if necessary, by searching the .reloc section for its specially marked entries. Now the virus goes resident, based on manipulation of the Import Table. With the addresses of imported functions in the host's .idata section, Cabanas need simply replace them with the addresses of its own API handlers.

To achieve this, Cabanas opens and maps the host. It allocates a 12232-byte block, copies itself there, then searches for the names of the functions it hooks: _lopen, CopyFileA, CopyFileW, CreateFileA, CreateFileW, CreateProcessA, CreateProcessW, FindClose, FindFirstFileA, FindFirstFileW, FindNextFileA, FindNextFileW, GetFileAttributesA, GetFileAttributesW, GetProcAddress, MoveFileA, MoveFileExA, MoveFileExW, MoveFileW, OpenFile, SetFileAttrA, and SetFileAttrW. Whenever it finds one, its saves the original address in its own JMP table and replaces the DWORD in the host's .idata section with a pointer to its own function. Finally, the virus closes and unmaps the host, then starts the application by jumping to the original entry point in the .text section.

Some *Windows* programmers may say: 'But this hook mechanism is not efficient enough. Whenever the application does not have imports for some of these APIs, but calls them directly by using GetProcAddress, the virus cannot hook anything other than the GetProcAddress API.' That is the reason that the virus hooks it.

When an infected program calls a Cabanas-hooked API, the virus' handler calls the original GetProcAddress for the address of the requested API. After this, it checks whether the function is a KERNEL32 API, and if it is one that it wants to hook. If so, and it is not yet hooked, the virus returns a new API address pointing into its jump table.

Stealth, *et al*

Cabanas implements semi-stealth: during FindFirstFileA, FindFirstFileW, FindNextFileA, FindNextFileW it checks for already infected programs. If a program is not infected, Cabanas infects it, otherwise it hides the change in file size. Thus, if a scanner checks its size by calling these APIs, it cannot detect the size change and will start scanning if no other checks are made. Anti-virus programs must have robust self-checks. One possible defence against Cabanas' stealth would be to compare the API addresses in your own Import Table with those in the KERNEL32 Export Table.

The virus can see all files accessed on an infected machine and since the *NT* command interpreter (CMD.EXE) uses the Win32 FindFirst/Next APIs during a DIR command, every non-infected file will be infected. The virus will also infect files during every other hooked API request.

Conclusion

Cabanas shows that a virus need not be *NT*-specific to work under *NT*. In fact, a working *NT* virus will more likely not have *Windows 95*-specific functionality. However, I expect virus writers will use knowledge they have gained from *Windows 95* to move to the more robust platform.

The author of Cabanas claims to have written a polymorphic engine, which was not included in this first version – 'One Half is not dead if you understand what I mean'. The next release of Cabanas could have a polymorphic decryptor in the .text section of the infected program. This will make the disinfection of such viruses very complicated in the future.

Win32.Cabanas

Aliases: Cabanas.

Type: Win32 (Windows NT, Windows 95, Win32s) PE infector. Per-process resident, semi-stealth, fast infector.

Self-recognition in Files:
Files with sizes divisible by 101 are assumed to be infected and a special checksum is stored in the file-stamp field of the PE Header (see text).

Self-recognition in Memory:
Not needed.

Hex Pattern in PE Files:

```
AB8B C6AB 6489 2360 8743 FE83
EF97 5857 57AC D2C0 34B5 7920
```

Intercepts: Many Win32 kernel APIs – see text.

Payload: None.

Removal: Recover infected files from backup or replace with originals.

VIRUS ANALYSIS 2

Primate-ive!

Steven Braggs
Sophos Plc

Baboon is, from a technical point of view, an unremarkable boot sector virus. However, it is of interest because it has a highly destructive payload, is in the wild and has actually activated its payload for real.

Purely by coincidence, Baboon first came to my attention on its trigger date. I happened to be analysing a virus from a collection on 11 September. I went through the normal procedure of installing the virus onto a diskette and booting up a test machine from it – I was not altogether surprised when the PC promptly hung. Just another incomplete or non-functional sample, I thought initially. A closer look revealed that the virus paid special attention to 11 September. After rebooting the PC with a clean boot disk, a quick look with *Norton Utilities* showed the extent of damage caused by the virus' payload; the Master Boot Sector had been overwritten with rubbish, and there were two copies of the virus beginning at side 1, cylinder 0, sector 1 (where the DOS boot sector had been).

What was even more unusual than working on Baboon on its trigger date, however, was that later that same day a laptop that had been trashed by this very virus arrived from a customer. Baboon had, in fact, affected about sixty machines at their site. It later transpired that the virus had been proliferated by engineers doing routine service work on the machines. This process required booting the machines from floppy, and write-enabled diskettes had been used. Just the right environment to spread a boot virus!

Hard Drive Infection

Baboon uses typical boot sector infection techniques. On boot-up from an infected floppy, the virus is loaded into memory at location 0:7C00h, as for any boot sector. The first part of the code grabs the Int 13h handler and stores it at the beginning of the virus body in memory, just after the initial jump. Then the virus steals 1KB of memory, calculates a new segment address at the top of conventional memory, and jumps there. This procedure is unremarkable, except that some of the instructions are slightly disguised, possibly in an attempt to avoid heuristic detection.

Once in the new segment, the virus decrements a counter and gets the system date using Int 1Ah. Both the counter and the date can trigger the payload – more of this later. The infection routine is then called, which infects both hard and floppy drives. The hooked Int 13h address is used to read the MBR into memory, immediately after the virus' code. Baboon patches its initial jump into the clean sector and overwrites 365 bytes with its code. Crucially, it leaves the partition information intact. The virus then searches the original MBR for the bootable partition information and uses this to load the active OS boot record to 0:7C00h. Finally, Baboon hooks Int 13h to point to its own handler routine in memory and passes control to the normal OS boot sector.

Booting from an infected floppy does not display the standard message 'Non system disk or disk error' – the boot proceeds from the hard drive. Subsequent boots from the hard drive follow a similar pattern. The virus reinfects the hard drive on every boot so that the counter and the date information are updated.

Floppy Infection

Infection of the floppy drive is through the hooked Int 13h handler. It contains the routine used in the hard drive infection. There is one change, however; the call is made to the byte before the routine. This byte contains either C3h (RET) or 90h (NOP). Hence, infection only takes place when this byte is set to 90h. Initially, it is set to C3h.

When a call to Int 13h is made, the virus first checks whether a hard drive or floppy drive is the subject of the call. Hard drive calls are redirected to the normal handler. There is no attempt at stealth. On floppy drive calls, the virus checks to see if the drive motor is spinning. If so, the virus calls its infection routine. If the motor is not spinning, the virus changes the byte referred to above to 90h, so that it will infect next time.

Most processes that read or write to floppy try at least three times if the initial call fails. This ensures that even a simple DIR command is sufficient to infect a floppy disk. As with MBR infection, the original boot sector is not preserved. The counter, which is decremented every time the PC boots, is transferred to the floppy disk at its current value. This means infections of hard drives from the newly-infected floppy will start nearer to detonation time than the original infection.

Payload

Baboon's payload is activated on one of two events – any call to Int 13h on 11 September, or a call concerning the floppy disk services when the counter has reached zero. The payload is simple, but destructive. The virus reads the DBR into memory immediately behind itself. The partition information stored in the infection routine is used to locate the DBR. It then writes nine sectors beginning at the original location of the DBR with the virus first, then the DBR, then whatever happens to be in memory. To finish, Baboon overwrites the MBR with garbage. Triggering by date will only happen at boot up, as this is when the virus updates its date information.

You might think that you could recover the DBR from the sector next to its original location. In practice, this is not possible. The DBR calls Int 13h several times during the boot process. On the second call, after having already detonated the payload once, the virus reads the contents of what should have been the DBR to immediately behind itself in memory, but this is not the DBR; it is another copy of the virus. The subsequent write to the hard drive then overwrites the preserved copy of the DBR with another copy of the virus. I am not sure whether this is a bug in the code, or whether it was deliberate – it certainly makes recovery from the damage substantially more difficult. The result is the loss of the MBR and the DBR and the first eight sectors of the first copy of the FAT. In return, Baboon gives you two copies of itself.

The 'trigger on count' can only happen on access to a diskette. Most day-to-day floppy access operations cause the same double payload as above. As the counter is a single byte, it will trigger after 255 boots at most. Usually, an infected disk will have come from a machine with an infected hard drive which has been booted up as normal any number of times, so in reality, the fuse will be a lot shorter than 255 boots!

Recovery

Once the payload has triggered, recovery is difficult, but can usually be accomplished successfully. It is a question of using what little information the virus has left behind, together with *Norton Utilities* (or similar), and a little thought. As with all repair or disinfection work this should begin with a boot from a guaranteed clean system disk.

Recovery of the MBR is quite straightforward. The virus stores a complete copy of the partition table in the correct location within its body. Simply copy one of the virus boot sectors from where the DBR or the first sector of the FAT are supposed to be to head 0, cylinder 0, sector 1. Then use FDISK with the /MBR switch to overwrite the sector with the correct code.

Recovering the FAT is trickier. You should be able to find the start of the second copy of the FAT on the hard drive. There should be a sector beginning with the media descriptor byte (F8h) somewhere in the first few hundred disk sectors. This should be the second copy of the FAT. Check that it has not been corrupted by comparing sectors after the eighth with the equivalent sectors in the first FAT. If they match, overwrite the eight sectors following the location of the original DBR with the equivalent sectors from the second FAT. This should restore the FAT.

The most difficult part of recovery is the DBR. The best way to approach this is to cheat slightly. If you can get hold of a clean DBR from a machine using the same version of DOS as the damaged machine, you can use this as a template. Copy the clean sector into the right place on the damaged machine. All you have to do now is to restore the hard disk information to the MBR.

Most of this information is available from *Norton Utilities*. Some of it is available directly – bytes per sector and sectors per track. Other data can be obtained with a little manipulation. For example, you can get sectors per cluster by looking at a cluster – the beginning and ending sector are displayed on the screen.

Other information can be deduced from what the virus has left behind. Total sectors can be obtained from the partition table already recovered. The trickiest part is restoring the sectors per FAT. You know where the first FAT ended – the sector before the second FAT started. Simply subtract one from the logical sector number at the end of the first FAT (to exclude the sector occupied by the DBR) to get sectors per FAT. Once you have restored all the above, you should be able to boot up as normal and recover your data, provided the virus has done no further damage. A free utility that repairs the damage automatically is available from *Sophos Plc*.

Prevention

As with most boot sector viruses, the simple measure of setting the BIOS to default boot from the hard drive will prevent infection by Baboon.

Conclusion

Baboon is a 'traditional' boot sector virus. It infects in a traditional way and spreads very effectively from infected floppy disks. These days, with macro viruses seen as the major threat, it is a sobering thought that a basic, old-fashioned boot sector virus managed to get into the wild and spread with such effect. It is easy to forget that boot viruses used to be the most common form of viral infection, and if effective security measures are not implemented, new ones can still slip through unnoticed and cause the sort of damage described here.

Baboon

Aliases:	None known.
Type:	Boot infector.
Infection:	Hard drive MBR, floppy boot sector.
Hex Pattern:	`8B0E 8801 8B16 8A01 E862 0088 DFB4 03B0 09E8 5900 B403 B001`
Intercepts:	Int 13h.
Payload:	Trashes MBR and first eight sectors of first FAT, and overwrites boot sector.
Trigger:	Date at boot is 11 September or internal boot counter reaches zero.
Removal:	Replace MBR with copy of original. If the payload has triggered, see the text for recovery details.

VIRUS ANALYSIS 3

The *Windows* Virus Drama; Act ii, scene iv

Eugene Kaspersky
KAMI Associates

The scene was set in 1992, when PC viruses first crossed the boundaries of DOS to new horizons. WinVir14 was the first to infect *Windows* executables (NewEXE files –see *VB*, November 1992, p.19). Fortunately, that was only a test virus and was never discovered in the wild. There were no other *Windows* virus outbreaks for more than three years – the first reported incident being that of the Tentacle virus in March 1996 (see *VB*, September 1996, p.11). Just two months earlier, in January of that year, the first *Windows 95* infector appeared. Known around the world, Boza opened the door to *Windows* PE infection. At the end of 1996 came Punch, the first memory-resident *Windows 95* virus. Punch drops a VxD driver that, while loading, hooks the IFS API, then infects PE files as they are opened (see *VB*, April 1997, p.8).

In 1997 the plot thickened with the arrival of encrypted *Windows* viruses like Mad and Memorial (see *VB*, September 1997, p.6), apparently the initial steps to Polymorphism32, and Cabanas, the first Win32 infector to work under *NT* (this issue, p.10). What can we expect in the near future? There are several other known methods of infecting *Windows* and/or *NT*, but not wishing to inspire or encourage virus writers, I shall not go into them. For now, the drama continues to unfold with a new virus, Navrhar, that infects *Word 6/7* documents and *Windows* VxDs.

Infecting *Word* documents without using *Word* itself is no longer a surprise. The first virus known to do so was Anarchy.6093, a multipartite DOS EXE and *Word* document infector (see *VB*, October 1997, p.6). A new chapter in this story is the infection of VxD drivers – that really is new, and until now there were no viruses known even to attempt it.

Carousel of Infection

Navrhar has several stages to its infection procedure. Starting from an infected document or VxD, it takes three steps before again infecting files of the same format as its original host: *Word* document → PE dropper → VxD driver → *Word* document, and so on. When *Word* opens an 'infected' document file, the virus' AutoOpen macro drops and executes its PE part. The dropper looks for certain, standard *Windows* VxDs and infects them. The next time *Windows* loads any of these drivers, the virus goes resident, hooking system calls from which it infects documents as they are opened.

In VxD files and *Word* documents, the virus uses quite a clever method to store its data and code. In both types of files the virus only injects a loader – a small program that loads and executes the main virus code. In infected documents this loader is a short WordBasic macro, while in VxDs it is a small 32-bit program. The main virus code is placed outside the actual bodies of both the *Word* document and the VxD. While infecting, the virus simply appends this code to the end of the file and makes no attempts to link it with the data or code of the host file. It looks much like a code/data overlay, and there are no references to it in the main file except that the virus loader expects it to be there.

When *Windows* loads an infected VxD, it pays no heed to this extra data/code and does not load it into memory. Similarly, when *Word* opens a Navrhar-infected document, it does not load the extra data into memory, in fact, it will cut the data when closing the document. In both cases, the only way for the virus to access its overlay is to open the infected document or VxD as a disk file, find the overlay and read it!

Because the structure and content of the overlay data varies with the type of host, Navrhar increases the length of the objects it infects by different amounts. Documents increase by 17245 bytes, VxDs by 12288 bytes, and the virus dropper, RUNME.EXE, is 16208 bytes long.

Opening an Infected Document

Nowadays, the commonest way of transferring virus infections is through the exchange of infected documents. Navrhar's author depends on this to increase the distribution of the virius by allowing it to travel in readily exchanged documents – afterall, who apart from programmers exchange VxDs in the first place?

In infected documents, Navrhar's macro loader is called AutoOpen, an auto-macro automatically executed by *Word* when the document is opened. Using HeapAlloc, this macro allocates a block of memory, opens the host file, finds the overlay data (main virus code), reads it and drops it to the newly created C:\RUNME.EXE file. It achieves this using standard Win32 API functions such as CreateFileA, SetFilePointer, ReadFile, WriteFile, and CloseHandle. Then it executes the newly created virus dropper with a WordBasic Shell instruction.

RunningMe – Infecting VxD Drivers

When RUNME.EXE is first executed, the virus code checks the *Windows* version, immediately exiting if it is *3.xx* or less. I wonder if that check is really necessary – RUNME.EXE is a 32-bit PE program, unable to run under older versions. The virus locates the *Windows* SYSTEM

subdirectory, looks for and infects, VxD files with these names: EISA, FILESEC, ISAPNP, LOGGER, LPT, LPTENUM, MSMOUSE, MSSP, NWSERVER, NWSP, PARALINK, PCI, SERENUM, SERIAL, SPAP, SPLITTER, UNIMODEM, VFD, VGATEWAY, WSIPX, and WSOCK.

The virus parses and modifies the internal structures of target VxDs, writing its loader code into the middle of the file and appending its main code to the end of the file as extra data. VxD drivers are of LE (Linear Executable) file format. This format is quite complex – more so than DOS EXE or *Windows* NewEXE file formats. LE looks similar to the Portable Executable (PE) format, but it is actually very different.

It would take too many pages to describe in detail what Navrhar does while infecting VxDs. To summarize: it reads the LE Entry Table and Object Tables, looks for the VxD code section and patches it. LE files have a sectional (or paged) structure; if the code or data does not 'fill' its section, the left over space is filled with nulls. Navrhar uses this feature by finding a section with at least 214 bytes 'free', increasing the length of that section and appending its loader code (214 bytes) to it.

To prevent duplicate infection, Navrhar uses fixed date and time stamps – files dated 4 January, 1997, 4:28 am are assumed infected. Further, it only looks for the VxD files listed above, and the RUNME.EXE dropper is not erased after infecting VxDs.

Looking for Documents

When *Windows* loads an infected VxD, the virus loader allocates a block of memory, reads the host file's overlay as data and then executes it as a program (long live *Windows'* kernel protection!). It then installs a hook in the IFS to intercept OpenFile calls. When a file is opened, the virus' handler compares the file's extension with '.DOC', reads the file header and checks it for an OLE2 stamp. It then parses the OLE2 format, creating a new AutoOpen macro at the end of the document, writes its macro loader there, then appends its main code as an overlay. The virus also sets the document's template bit.

The virus does not infect documents with file lengths not aligned to sector boundaries, as infected documents are not aligned and *Word*-made documents are. Navrhar also rejects documents longer than 800KB; its infection routine is not capable of parsing large documents. It also checks an internal identifier, only infecting documents created by a 'PanEuropean' version of *Word*.

When saving an infected document, *Word* should cut the virus overlay code from the end of document, but leave the AutoOpen macro, and in my experiments with *Word 7*, it did. Despite this, documents with DOC file extensions were immediately re-infected. In the case of non-DOC extensions, the documents stayed semi-infected; they had the macro AutoOpen, but they did not have the virus overlay data. This means that the virus is not able to spread itself when such documents are opened.

Navrhar's infection algorithm is not bug-free. If a template already contains a macro, the infection process corrupts the document's internal structure, and the macro stays invisible. Moreover, it is impossible to remove the virus' AutoOpen macro from infected documents using *Word*. If you try this, *Word* crashes, displaying the familiar error message: 'This program has performed an illegal operation and will be shut down.'

Conclusion

Owing to its dependence on VxDs, this virus is not able to infect under *NT*, but while experimenting with it under *Windows 95* no problems were discovered (apart from the *Word* template problem described above). The virus does not manifest itself in any way and does not cause any system error messages when running under *Windows 3.x*.

It seems to me that the VxD infection routine is buggy (it does not parse the LE Object Table correctly), and the virus will corrupt non-standard VxD drivers. However, all the VxDs from the virus' list are 'Navrhar-compatible' and running *Windows 95* with infected VxDs (even doubly-infected ones) caused no problems. Maybe the author was aware of this and thus did not include a general 'infect VxD' function?

I am sure this is not the curtain-call for *Windows* viruses. What will the next act reveal?

Navrhar

Aliases: HZDS.

Type: *Word 6/7* documents and VxD driver infector. Stays resident as a VxD under *Windows 95*.

Self-recognition: VxD files with date and time set to 4 January, 1997, 4:28 am. Document files whose size is not sector-aligned.

Hex Pattern:

```
55E8 0000 0000 5D83 BD82 0000
0000 7572 606A 0068 00A0 0000
080B C074 5C89 8582 0000 00FC
CD20 0000 4000 724D B800 D500
```

Intercepts: Under *Windows 95* – IFS API for Word document infection

Payload: None.

Removal: Under clean system conditions, identify and replace infected documents and VxDs. Delete the dropper C:\RUNME.EXE.

PRODUCT REVIEW

IBM AntiVirus v3.0 Enterprise Edition for NetWare

Martyn Perry

Although products from *IBM* have featured regularly in *Virus Bulletin's* comparative tests, it has been nearly two years since a standalone review. So how does the new *NetWare* version fare?

The program is considered to be in use when it resides in memory or is otherwise stored on a machine. There is provision for portable or home use, provided that the program is not active on both machines at the same time.

Presentation and Installation

The Enterprise Edition supplied for review came on a CD-ROM, which contained *IBM AntiVirus v3.0* for *NetWare* and *NT Server*, plus client versions for DOS, *Windows 3.1x*, *Windows 95*, *NT* and *OS/2*. *IBM* also supplied us with Service Pack 1 (build 559) and signature update 3.0j. This is the configuration tested here. Documentation comprises an *IBM AntiVirus* User's Guide, and an *IBM AntiVirus* Administrator's Guide. The literature is very comprehensive, if a little disjointed for my taste, due to the constant directions to obtain further details on a topic elsewhere in the manual. This means additional referencing in the index. It would perhaps be more useful to indicate cross-references by page number.

Installation requires CLIB 3.12f or later for *NetWare 3.x*, and Administrator workstations must be running *NT v4.0*, *Windows 95* or *OS/2 v3.0*. Further, *Novell's Client32* must be installed on Adminstrator workstations, otherwise stack errors will occur when trying to load the administration program. Manual installation only needs to be conducted on the first server; all subsequent installations can be executed from an Administrator Server.

There are two distinct installation phases. Initially, the necessary server files are copied from the CD-ROM to a server directory (default SYS:\SYSTEM\IBMAVNW) and unzipped using DOS commands. The administration program is then installed. Running the administration program provides the initial options of Install, Create Emergency Disk, and Scan System Without Installing. Having selected the Install option, and chosen between Custom and Express installations, the workstation is scanned for viruses and the required files copied.

When the administration program is first run following installation of the server components, you are asked to define a number of settings. These include the network protocol to be used for communication (IPX or TCP/IP) and the various time-outs, the Response Wait Time, and the Administration Inactivity Period (the period the program can be left unattended before timing out the administration session). Another good option is the Refresh Interval, which determines the time between status updates and helps to reduce traffic on the network. Having configured these settings, the Network Support Module is loaded and the main window displayed.

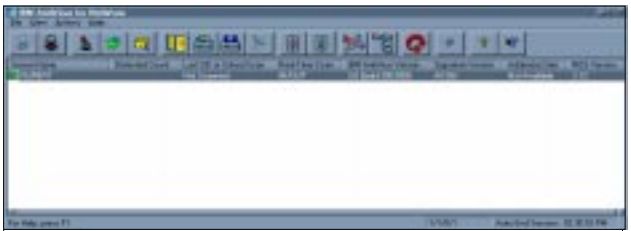
IBM AntiVirus for NetWare

On loading the IBMAVNW.NLM a banner message is displayed, showing that the program is running. There is no facility to control virus scanning from the server console at all – this is all performed from the dministration program on a connected workstation.

The scanner has three modes of operation – Immediate, On-access, and Scheduled. In the first case, the default file extensions for on-demand scanning are EXE, COM, OV?, SYS, BIN, DOT, DOC, SMM, XL?. The on-demand scan had an impressive detection rate, but at a heavy price for scan duration.

The test scan of 15,039 infected files took 4 hours and 33 minutes. This compares unfavourably with typical scan times for recently-reviewed products which may take less than an hour to complete. This is not due to the generation of checksum values for the scanned files, which accounts for the relatively slow initial scan times of *IBM AntiVirus* on other platforms. As the *NetWare* version does not work this way, subsequent scans will take just as long. This could have a real impact if used on a regular basis prior to performing routine checks. In mitigation, the test was run with the evaluation option set to 'on'. This builds the various reports and clearly will generate some overhead as the files get bigger.

In On-access mode, the default file extensions for real-time scanning are the same as for the On-demand scan. The scan selections can be made for incoming file check, outgoing file check, a combination of both or none at all. With a Scheduled scan, up to five distinct schedules are available with different frequencies, start times and durations. Time frequency can be set to None, Once, Weekly, Daily, Saturdays, Sundays, Weekdays, Weekends and At Startup.

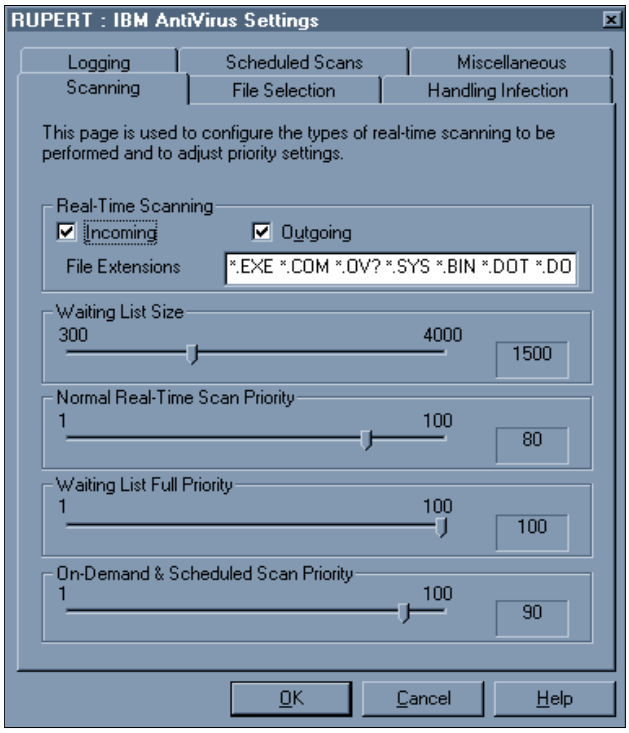


IBM AntiVirus Administrator's main window

In all three modes, the scanner makes no attempt to delete or disinfect infected files immediately. Instead, it records the infections in its log file, providing the option to lock the files from further use or to move them to a 'safe' directory. Once transferred here, the files are effectively 'off-line', giving the administrator the chance to make decisions on how to handle the infections. Having performed the necessary cleaning actions, options are provided for restoring the files to their original locations.

Administration and Reporting Options

A user ID and password are needed to access the main administration options. Each server under the Administrator's control can be viewed in the IBMAV Adminstrator, as can a status summary of each scan mode configuration. If more detail is required, clicking on the server in question brings up a full screen of status detail showing the current activity on that server.



There are a number of log file provisions available. These include current and previous activity logs, which record scanning and virus handling for the current session, and a record of the previous session. A separate log keeps track of software and virus signature distribution.

IBM AntiVirus for NetWare provides an Evaluation option which generates four log files. ODCLEAN.TST and ODINFECT.TST contain all the files checked during On-demand and Scheduled scans, listing clean files and infected files respectively. Separate files (RTCLEAN.TST and RTINFECT.TST) record the real-time scan results. These files are turned off automatically after 24 hours to avoid generating unnecessarily large files. These were used during testing to produce the scan results.

A number of different alerts are stored in a further log file (default AVALERT.LOG). These include reminders about open virus incidents, and server low-memory conditions that fall below the threshold required for *IBM AntiVirus for NetWare.* Should such a low-memory condition arise, virus checking is suspended. Other alert types include incomplete virus checking, scheduled scans not completing within the allocated time, servers without active virus protection, and messages generated when the software is being upgraded or when updated signatures are distributed across servers.

Since all of the administration activities are controlled from the IBMAV Administrator program, no command-line options are available. The administration program has options to configure the System Shield component of the *IBM AntiVirus* client software on workstations attached to the servers under its control.

It is possible to check boot sectors on diskettes loaded on the Administrator machine, and warn the administration program if a virus is found. Further options include loading the workstation scanner into high memory and setting it for an automated, scheduled scan – something that should possibly be made mandatory on any system's anti-virus administrator machine!

Updates

Virus signatures can be updated from a distribution server to other servers also running *IBM AntiVirus for NetWare* and to workstations running the *IBM AntiVirus* client software. In either case, you first obtain the update files from *IBM's* web page and unzip them into the correct server directory. The administration program can then be used to select other servers to send the updates to, and to initiate the update procedure.

If the administration option is selected on your workstations, they will be automatically updated following the server update. Each time the client program is started, it checks for updated virus signature and message files.

Detection Rates

Scanner detection was tested using four of *Virus Bulletin's* virus test-sets – In the Wild File, Standard, Macro and Polymorphic (see the summary box). The tests were conducted using the default scanner file extensions, and undetected viruses identified from the two log files.

The results were impressive, with 100% detection in the In the Wild File, Macro and Standard test-sets. The only failure was missing all 500 samples of the Polymorphic virus Cryptor.2582.

Real-time Scanning Overhead

To determine the impact of the scanner on the server when it is running, the following test was executed. The basis of the test was to time the copying of 63 files of 4,641,722

bytes (EXE files from SYS:PUBLIC) from one server directory to another using *Novell's* NCOPY. Using NCOPY keeps the data transfer within the server itself, minimizing network effects. The directories 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.

Due to the different processes which occur within the server, the tests were run ten times for each setting and an average taken. The default settings were used: Waiting list size – 1500, Normal Real-Time Scan Priority – 80%, Waiting List Full Priority – 100%. The tests were:

- NLM not loaded. This establishes the baseline time for copying the files on the server.
- NLM loaded; In = No, Out = No, Scan = No. This tests the impact of the scanner loaded in its quiescent state with no real-time or immediate scan in progress.
- NLM loaded; In = Yes, Out = No, Scan = No. This shows the on-access overhead when scanning incoming files.
- NLM loaded; In = No, Out = Yes, Scan = No. This shows the on-access overhead when scanning outgoing files.
- NLM loaded; In = Yes, Out = Yes, Scan = No. This shows the on-access overhead of having both read and write scans in effect.
- NLM loaded; In = Yes, Out = Yes, Scan = Yes. This shows the incremental effect of running an immediate scan in addition to the real-time scan.
- NLM unloaded. This is run after the other tests to check how well the server returns to its former state.

The initial impact of loading the scanner software is minimal, but begins to take effect when one of the real-time scans is selected. Oddly, the impact of the write overhead is very much lower than when reading the file. The overall performance, even when the scanner is running, is much lower than many other products. See the summary box for detailed results.

This anomaly is probably due to the disk caching feeding the files, in the sample set, from memory rather than direct from disk. The residual overhead, when *IBM* is unloaded, is minimal and is due to CLIB and Streams NLMs remaining loaded on the server.

Conclusion

I will get the moans out of the way first. Perhaps I am being pedantic, but I feel that the initial Server installation should be carried out within the installation program itself, rather than requiring the user to resort to manual file copying and unzipping. This is reminiscent of software installation from the 1980s and most shareware (and many freeware) products have better, automated installation procedures these days. Moreover, the scan time is too long and it is very awkward ploughing backwards and forwards through the manual (and occasionally *between* manuals) to accomplish a particular task.

Having got that off my chest, I was pleasantly surprised with the move to a *Windows*-based administration package which can be used to control a security domain of servers and workstations. The detection rate is impressive and the product detects the viruses currently in the wild. The philosophy of moving and locking files rather than immediate deletion gives the Administrator the chance to check files. This can be particularly important in widely-dispersed networks, although not having an immediate disinfection option could be disadvantageous in some settings.

In conclusion, *IBM* have a product providing a high level of anti-virus protection for an enterprise network, combined with good administration support.

IBM AntiVirus v3.0 for NetWare

Detection Results

Test-set[1]	Viruses Detected	Score
In the Wild File	549/549	100.0%
Standard	774/774	100.0%
Macro	716/716	100.0%
Polymorphic	12500/13000	96.2%

Overhead of On-access Scanning:

The tests show the time (in seconds) taken to copy 63 EXE files (4.6MB). Each test was repeated ten times, and an average taken.

	Time	Overhead
Baseline	6.2	–
NLM loaded, inactive	6.6	6.5%
— + enabled, scan incoming	6.7	8.1%
— + — + scan outgoing	6.4	3.2%
— + — + scan both	6.6	6.5%
— + — + — + on-demand scan	6.1	-1.6%
NLM unloaded	6.3	1.6%

Technical Details

Product: *IBM AntiVirus Enterprise Edition v3.0.*

Vendor: *IBM Corp, IBM AntiVirus*, MD 227, PO Box 700, Suffern, NY 10901, USA. Tel 800 742 2493, fax 800 494 3045, email ibmav@us.ibm.com, WWW http://www.av.ibm.com/.

Price: *Enterprise Edition* (*Windows NT* – Client and Server, *NetWare*, *Windows 95*, *Windows 3.x*, *OS/2* and DOS) $620 for 25-user version. Signature updates are available from the *IBM AntiVirus* home page.

Hardware Used: Server: *Compaq* Prolinea 590, 80 MB RAM, 2 GB disk, *Netware 3.12*. Workstation: *Compaq* DeskPro XE 466, 16 MB RAM, 207 MB disk, *Windows 95* with *Novell Client32*.

[1]**Test-sets:** see *VB*, September 1997, p.16.

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

The **International Conference on Forensic Computing** will be held in Brighton from 3–5 December 1997. For further information, contact *The International Journal of Forensic Computing*; email ijfc@pavilion.co.uk, or Tel +44 1903 209226.

*Priority Data Systems Ltd***,** Ireland's largest anti-virus supplier, announced the **formation of Network for Information Security in Europe (NISE)** in Copenhagen in September. The network of European companies is to focus on information security solutions for corporate and government clients. Contact Trish Evans; email sales@prioritydata.co.uk, Tel +44 1442 233823.

Peapod Internet* is to distribute *Secure Computing's SmartFilter Internet monitoring and filtering software, incorporating *Wavecrest Computing's ProxyReporter.* In October, Peapod became the UK distributors of ***Trend Micro's InterScan VirusWall v2.0*** for the *Hewlett Packard UNIX* system. Prices start at £1295 for 50 users; for more information contact Chris Durnan, chrisd@peapod.co.uk, Tel +44 181 606 9924.

In late September, ***Symantec* released *Norton AntiVirus 4.0 Deluxe,*** priced £59, shipping for *Windows 95, NT* and *DOS/Windows 3.x* on the same CD. *Norton AntiVirus 4.0 for Windows NT Server*, including AutoProtect, followed in October. For details contact Richard Saunders at *Symantec's* Press Office, UK; Tel +44 1628 592222, email rsaunders@symantec.com.

***AbirNet* and *E92 Plus* have updated *SessionWall-3* in its second release**, to be distributed by *Peapod Internet*. Enhanced view capabilities and features include Java and ActiveX applet detection mechanisms. Model 25, which handles up to 125 sessions, costs £895. Contact Florence Heynard; Tel +44 181 399 3111 for details.

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 several aspects of technology applications in – 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.

Dr Solomon's AntiVirus for Microsoft Exchange was recently announced. Another development at *Dr Solomon's* is the **acquisition of the *Virex* and *netOctopus* lines from *DataWatch Inc.*** *Virex* is a market leader in Macintosh anti-virus; *netOctopus* is a Macintosh network and system admininstration solution. The company claimed record first quarter bookings and earnings for the quarter ended August 1997, virtually doubling operating profit from £1.5 million in the same quarter last year to £3 million. For more information contact Mike Hill at *Dr Solomon's*; Tel +44 1296 318700.

Integralis has extended the capabilities of *MIMEsweeper* in an updated version. ***MIMEsweeper v3.1* includes support for *Lotus Notes***, and runs on *Windows NT 3.51* and *4.0*. The product now intercepts viruses within emails, FTP and HTTP file transfers. For more details email info@mimesweeper.com, or contact Catherine Jamieson; Tel +44 118 930 6060.

Two computer virus workshops are being held this month at the ***Sophos*** training suite in Abingdon in the UK. An introductory course on 19 November will be followed by an advanced session the next day. **A practical *NetWare* Security course**, price £325 +VAT, is also taking place at the same site on 8 January, 1998. More information is available from Karen Richardson; Tel +44 1235 544015, fax +44 1235 55935, or the company's Web site; http://www.sophos.com/.